İsmayıl Sadıqov
Ramin Mahmudzadə
Naidə İsayeva

İNFORMATİKA

9

Ümumtəhsil
məktəblərinin
9-cu sinfi üçün dərslik

Azərbaycan Respublikası
Təhsil Nazirliyinin
09.06.2008-ci il tarixli
712 №-li əmri ilə
təsdiq edilmisdir.

2008

Elmi redaktor: Rasim Əliquliyev, AMEA-nın müxbir üzvü, t.e.d., professor

Rəyçilər: Ələkbər Əliyev, t.e.d., professor
Həyat Axundova, Bakı şəhəri, 164 №-li orta məktəbin müəllimi
Valid Məhərrəmov, Bakı şəhəri, Fizika-riyaziyyat və informatika təmayüllü liseyin müəllimi
Səmihə Rüstəmova, Bakı şəhəri, 258 №-li orta məktəbin müəllimi

İnformatika – ümumtəhsil məktəblərinin 9-cu sinfi üçün dərslik.
İ.C.Sadıqov, R.Ə.Mahmudzadə, N.R.İsayeva. Bakı, "Bakınəşr", 2008, 128 səh.

ISBN-978-9952-430-08-8

1

PASCAL PROQRAMLAŞDIRMA DİLİ

1.1. PROQRAM TƏMİNATININ TƏSNİFATI

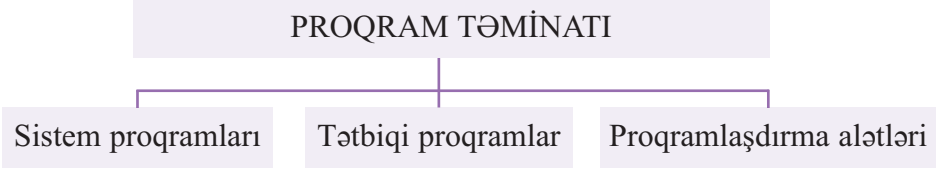
Kompüterin hər hansı bir işi yerinə yetirməsi üçün aparat təminatı ilə yanaşı, ona göstərişlər toplusu, yəni *proqramlar* lazımdır. Klaviaturada klavişin basılmasına, siçanın hərəkətinə, başqa kompüterdən informasiyanın alınmasına və digər hərəkətlərə kompüterin necə reaksiya verməsini məhz proqramlar müəyyən edir. Ekrana görüntünün çıxarılmasını, sənədin printerdə çap olunması üçün hazırlanmasını, kompüterdə musiqinin səsləndirilməsini proqramlar həyata keçirir.

Kompüterin *proqram təminatı* [software] kompüter sisteminin ayrılmaz bir hissəsi olub, kompüterin texniki təminatının məntiqi davamını təşkil edir.

Kompüterin konkret tətbiq sahəsi onun proqram təminatı ilə müəyyən olunur. Kompüterin özlüyündə heç bir "bacarığı" yoxdur. Bütün "bacarıqlar" kompüterdə icra olunan proqramlarda cəmləşdirilib.

Müasir kompüterlərin proqram təminatı oyun proqramlarından tutmuş elmi proqramlara kimi milyonlarla proqramdan ibarətdir.

Kompüterdə olan bütün proqramları şərti olaraq üç sinfə ayırmaq olar: *sistem proqramları, tətbiqi proqramlar, proqramlaşdırma alətləri.*



Sistem proqramları. Sistem proqramları kompüterin resurslarını – mərkəzi prosessoru, yaddaşı, giriş-çıxış qurğularını idarə etmək üçündür. Onlar bütün isti-

fadəçilər üçün nəzərdə tutulmuş proqramlardır. Kompüterin sistem proqramları elə hazırlanır ki, tətbiqi proqramlar səmərəli işləyə bilsin.

Sistem proqramları arasında *əməliyyat sistemləri* xüsusi yer tutur, sistem proqram təminatının əsasını əməliyyat sistemi təşkil edir. O, fərdi kompüterlərin vacib elementlərindən biridir. Əməliyyat sistemi kompüter yandırıldıqda işə düşən, kompüterin bütün hissələrinin tam bir vəhdət halında işləməsini təmin edən və informasiyanı idarə edə bilən proqramlar sistemidir.

Əməliyyat sisteminin köməyilə:

- kompüterlə istifadəçi arasında dialoq yaranır;
- operativ və daimi yaddaş qurğuları işə salınır;
- kompüter idarə olunur;
- istənilən proqram yerinə yetirilməyə başlayır və s.

Vaxtilə IBM PC tipli kompüterlərdə əsasən Microsoft firmasının hazırladığı MS-DOS əməliyyat sistemindən istifadə olunurdu. Bu əməliyyat sistemində işləyən istifadəçi yalnız konkret bir məsələni həll edə bilərdi.

Hazırda fərdi kompüterlərdə *çoxtapşırıqlı* əməliyyat sistemlərindən istifadə olunur – fərdi kompüterlərin yaddaşında eyni zamanda bir neçə proqram və məsələlər olur ki, mikroprosessor kompüterin resurslarını onların arasında bölüşdürür. Belə əməliyyat sistemlərinə misal olaraq OS/2, MacOS, UNIX, Linux, Windows XP, Windows Vista və digər əməliyyat sistemlərini misal göstərmək olar.



Sistem proqramlarının digər vacib hissəsini xidməti proqramlar – *utilitlər* (lat. "*utilitas*" – xeyir, fayda) təşkil edir. Onlar əməliyyat sistemini tamamlayır və onun imkanlarını artırır, həmçinin, müstəqil olaraq bir çox vacib məsələləri də həll edir. Utilitlərin bəzi növləri bunlardır:

- interfeys proqramları;
- antivirus proqramları;
- arxivləşdirmə proqramları;
- proqram örtükləri;
- kompüter qurğularının iş qabiliyyətini yoxlayan proqramlar;
- qurğuların işini idarə edən proqramlar (drayverlər) və s.

Tətbiqi proqramlar. İnsan fəaliyyətinin müxtəlif sahələrinə aid məsələləri həll etmək üçün nəzərdə tutulan proqram təminatına *tətbiqi proqramlar* deyilir.

İndiki zamanda fərdi kompüterlər üçün yüz minlərlə tətbiqi proqramlar işlənib hazırlanmışdır. Onlardan ən çox istifadə olunanlar bunlardır:

- mətn redaktorları (prosessorları);
- cədvəl verilənlərinin emalı proqramları – elektron cədvəllər;
- nəşriyyat sistemləri;
- verilənlər bazasının idarə olunması sistemləri;
- təqdimatların hazırlanması proqramları;
- qrafik redaktorlar;
- verilənlərin statistik təhlili proqramları;
- kompüter oyunları, öyrədici proqramlar və s.

Proqramlaşdırma alətləri. Bu sinfə aid olan proqramlar sistem və tətbiqi proqram təminatını yaratmaq üçün nəzərdə tutulmuşdur.

Proqram təminatının hazırlanması üçün Basic, C++, Pascal və b. proqramlaşdırma dillərindən istifadə olunur. Dünyanın bir çox təhsil müəssisələrində uşaqlara proqramlaşdırmanın əsaslarını öyrətmək üçün LOGO dilindən istifadə olunur.

1. Proqram təminatı neçə sinfə ayrılır və onlar hansılardır?
2. Sistem proqramlarının vəzifəsi nədir?
3. Əməliyyat sistemi nədir?
4. Tətbiqi proqramlar nə üçündür?
5. Proqramlaşdırma alətləri dedikdə nə nəzərdə tutulur?

1.2. PROQRAMLAŞDIRMA DİLLƏRİ

Dilini bilmədiyimiz insana nəyi isə başa salmaq üçün ya jestlərdən, ya da onun başa düşdüyü dildəki sözlərdən istifadə edirik. Kompüterin mərkəzi qurğusu olan prosessorun da öz dili var.

Maşın dili. Kompüterin bilavasitə "başa düşdüyü" yeganə dil – sadəcə, ədədlər yığınından ibarət olan *maşın dilidir* [machine language]. Ədədlərlə işlədiyindən prosessor üçün komandaları kodlaşdırmaq, məsələn, ədədlərlə ifadə etmək lazımdır. Tutaq ki, 1 – toplamanı, 2 – vurmanı, 3 – bölməni (yaxud uyğun olaraq 01, 02, 03) və s. bildirir. Müəyyən əməliyyatı yerinə yetirmək üçün prosessora komandalardan əlavə verilənlər də lazımdır. Sadəlik üçün, tutaq ki, hər hansı uydurma prosessorun komandasının ümumi şəkli belədir:

komandanın kodu	birinci operand	ikinci operand	nəticə yerləşdiriləcək xananın nömrəsi

Proqramlaşdırmada komandanın arqumenti, yəni verilənlər *operand* adlanır.

Hər bir verilən, kompüterin yaddaşının xanalarında yerləşir. Bütün yaddaş xanalara bölünmüşdür və hər xananın öz nömrəsi – *ünvanı* olur. Beləliklə, hər bir operand iki parametrlə – qiyməti və yaddaşda olan yeri ilə təyin olunur.

00xx yazılışı xx ədədinin özünü, 01xx yazılışı isə yaddaşda xx nömrəli xananı göstərir. Bir sözlə, xx verilənin qarşısında 00 olarsa, bu, verilənin özünü, 01 isə yerini göstərir.

Onda 01 və 02 xanalarında yerləşmiş iki ədədin ədədi ortasını tapan proqram aşağıdakı şəkildə olacaq:

```
01    0101  0102  0103
03    0103  0002  0103
```

Birinci sətirdəki komandanı adi dildə ifadə edək: 01 xanasındakı verilənlə 02 komandasındakı veriləni topla və nəticəni 03 xanasına yaz.

İkinci sətirdəki komanda isə aşağıdakı kimi ifadə olunar: 03 xanasındakı veriləni 2-yə böl və nəticəni yenidən 03 xanasına yaz.

> Veriləni yaddaş xanasına yazarkən orada olan əvvəlki informasiya silinir.

Proqramı bir sətirdə yazsaq belə alınar:

```
01010101020103030103000020103
```

Göründüyü kimi, maşın dilində yazılmış belə sadə proqram da açılması lazım olan "sirli bir aləmdir". Uydurma olmayan, gerçək kompüterlərdə isə maşın kodu qat-qat mürəkkəb olur.

"Maşın kodu" deyəndə, *maşın dilində yazılmış proqram* başa düşülür.

Belə kodda proqram tərtib etmək, sonra isə onun düzgünlüyünü yoxlamaq çox çətindir, çünki bunun üçün ya bütün komandaların kod və formatını yadda saxlamaq, ya da hər dəfə xüsusi cədvəllərdən istifadə etmək lazım gəlir.

Kodlaşdırmada azacıq səliqəsizlik, yazıda yanlışlıq, rəqəmlərin yerinin qarışdırılması gözlənilməz nəticələrə aparıb çıxarardı. Belə yanlışlığı tapmaq da asan iş deyildi, çünki proqramçının qarşısında az-çox aydın dildə yazılmış alqoritm deyil, rəqəmlər yığını dururdu.

Proqramçılar öz işlərini yüngülləşdirmək üçün yollar arayırdılar; elə bir alət yaratmaq lazım idi ki, onun köməyilə kiçik detallara vaxt itirmədən proqram yazmaq mümkün olsun, özləri yaradıcılıqla, alqoritmlərin qurulması ilə məşğul olsunlar, qalan işlər isə kompüterə həvalə edilsin.

"İkibaşlı yozum"

Kompüterə komandaları adi ingilis, fransız, türk dillərində, yaxud hər hansı başqa dildə vermək mümkün olsaydı, daha yaxşı olardı. Ancaq, təəssüf ki, maşınlar insanların danışığındakı incəlikləri başa düşə bilmir. İnsanlar danışdıqları zaman öz sözlərinə jestlər və mimikalar əlavə edir, məcazlardan, üstüörtülü ifadələrdən, istehzadan və bəlağətin başqa üsullarından istifadə edirlər ki, bu da onlara bir şey söyləyib, əslində başqa bir şeyi çatdırmağa imkan verir. İnsanlar "ikibaşlı" (çoxanlamlı) sözlər işlədir, deyilmiş sözlərin mənasını dəqiqləşdirən kontekstdən, intonasiyadan və başqa amillərdən istifadə edirlər. Hətta yazışmalarda da fikrin daha düzgün başa düşülməsinə yardım edən eyhamlardan istifadə olunur.

İnsan ağlı təbii dilin "tapmacalarından" baş çıxara bilir, kompüter isə yalnız tam ciddi, riyazi dəqiqliyə malik ünsiyyət sistemini anlayır. Belə sistemdə hər bir simvol, yaxud simvollar qrupu həmişə eyni bir şeyi bildirməli, hər bir cümlə isə hərfiyyən başa düşülməlidir. İstehza, danışıq ifadələri, yaxud dolaşıq eyhamlar, sadəcə, yolverilməzdir.

Assembler. Maşın kodunda proqramlaşdırma "iynəylə yer qazmaq" kimi bir şeydir. Çox sadə hesablama məsələləri (məsələn, ədədin yaddaşdan prosessora yüklənməsi, onun başqa ədədlə toplanması, nəticənin yenidən yaddaşa yazılması) maşın kodlarında uzun-uzadı elə mürəkkəb şəkil alırdı ki, kiçik kərpiclərdən böyük bir binanın necə tikilməsini təsəvvür etmək belə çətin idi.

Assembler dilinin adı ingiliscə *"assemble"* – "toplamaq", "yığmaq" sözündən götürülüb.

Bu vəziyyətdən çıxmaq üçün ilk addım komandaların simvollarla əvəzlənməsi oldu, daha dəqiq desək, komandalar adi sözlərin qısaltmaları ilə göstərildi. İdeya sadə olsa da, onun gerçəkləşdirilməsi yazılmış proqramların digər proq-

ramçılar tərəfindən qavranılmasını asanlaşdırmağa, səhvlərin sayını azaltmağa, proqramçıların işini yüngülləşdirməyə imkan verdi.

Komandaların simvollar vasitəsilə yazılış sisteminə *mnemonik yazı* sistemi deyilir. Belə sistemin köməyilə yazılmış proqram nümunəsinə baxaq.

```
TOP   X1, X2  > X3
BÖL   X3, 2   > X3
```

Bu fraqmentin birinci sətri bildirir ki, yaddaşın 1 və 2 nömrəli Xanalarını TOPlayıb, nəticəni 3 nömrəli Xanaya yerləşdirmək lazımdır. İkinci sətir isə kompüterə yaddaşın 3 nömrəli Xanasını 2-yə BÖLüb, nəticəni yenə də 3 nömrəli xanaya yerləşdirmək göstərişini verir. Göründüyü kimi, belə yazılış yuxarıda verilmiş uyğun maşın kodundakı yazılışdan daha anlaşıqlıdır.

Bu həm insanın, həm də kompüterin başa düşdüyü proqramlaşdırma dilinin yaradılması yolunda ilk addım idi.

İkinci addım prosedurlar kitabxanasının yaradılması, yəni koddan təkrar-təkrar istifadə olunması idi. Ona qədər isə hər bir proqramçı hər dəfə öz "Amerikasını kəşf etməli" idi. Prosedurlar kitabxanaları proqramçılara yeni mürəkkəb proqramları qısa vaxt ərzində keyfiyyətlə tərtib etmək imkanı verirdi.

Bu iki istiqamət yeni proqramlaşdırma dillərinin yaradılması və inkişafı üçün özül oldu.

Əlbəttə, assembler proqramçını işin ən arzuolunmaz hissəsindən – operatorların əl ilə maşın koduna çevrilməsindən azad edir. Ancaq buna baxmayaraq, assembler dilinin iki böyük nöqsanı var. Birinicisini, ola bilsin, siz özünüz artıq sezdiniz: assembler dilində proqramlaşdırma çox diqqət və səbir tələb edən işdir. Mikroprosessorun işini birbaşa idarə edərkən həddən artıq xırda məsələlər nəzərə alınmalıdır.

İkinci çatışmazlıq assembler dilindəki proqramların "daşınabilən" [portable] olmamasıdır. Məsələn, assembler dilində Intel 8080 prosessoru üçün yazılmış

Yunanca "mnemonikon" "yaddasaxlama bacarığı" deməkdir.

Qreys Murrey Hopper
(1906 – 1992)

1952-ci ildə Qreys Hopper [Grace Murray Hopper] (ABŞ) dünyada ilk mnemonik proqramlaşdırma dili olan *assembler dilini* [assembly language] yaratdı. Assemblerə mnemonik komandalar sistemi, prosedurlar kitabxanası və proqramın mətnini maşın koduna çevirən xüsusi proqram daxil idi. Proqramın maşın dilinə çevrilməsi proseduruna *kompilyasiya*, onu gerçəkləşdirən proqrama isə *kompilyator* deyilir (bu termin də Qreys M.Hopperindir).

proqramı Motorolla 6800 prosessorlu kompüterdə işlətmək olmur, proqramı həmin prosessor üçün yenidən yazmaq lazım gəlir. Doğrudur, bu o qədər də çətin məsələ olmasa da, hər halda müəyyən işlər görülməlidir.

1. Maşın dili nədir?
2. Assembler nədir və onun maşın dilindən üstünlüyü nədədir?
3. Assembler dilinin çatışmazlıqları hansılardır?
4. Proqramın "daşınabilənliyi" nə deməkdir?

1.3. YÜKSƏK SƏVİYYƏLİ DİLLƏR

Assembler dili mnemonik olsa da, dövrünün başlıca kompüter istifadəçiləri olan alimləri qane etmədi, çünki, artıq yuxarıda da qeyd olunduğu kimi, maşın koduna yaxın olduğuna görə onu öyrənmək çətindir. Bundan başqa, hər prosessorun öz assembleri olur və deməli, bir neçə maşında işləyən şəxs bəzən "bir neçə ayrı-ayrı assembler dilini" bilməlidir. Nəhayət, assembler dilində proqramlaşdırmaq üçün bütün prosesi ən kiçik detallarınadək təsəvvür etmək, mürəkkəb düsturları ayrı-ayrı əməllərin ardıcıllığı şəklində göstərmək lazımdır. Proqramçı isə beynində daha ümumi kateqoriyalarla işləyir: "düsturu hesablamalı", "ədədi ekrana çıxarmalı", "əməliyyatı 10 dəfə təkrarlamalı" və s.

Assemblerdən başqa, qalan proqramlaşdırma dillərinin hər biri **yüksək səviyyəli dil** adlandırılır, ancaq bu hələ onların eynisəviyyəli olması demək deyildir. Bir dilin səviyyəsi başqasının səviyyəsindən yuxarı, yaxud aşağı ola bilər. "Yüksək səviyyəli dil" dedikdə onun insan dilinə nə qədər yaxın olması başa düşülür.

Yəni, məsələn, klaviaturada aşağıdakı cümləni yığmaqla kompüter sizin göstərişinizi dərhal yerinə yetirir: "Cari il üçün mənfəət və zərəri hesabla, illik hesabatı hazırla və ondan bir neçə nüsxə çap edib, lazım olan yerlərə göndər". Əlbəttə, bugünkü proqramlaşdırma dilləri bu idealdan hələ çox uzaqdır!

Belə problemləri aradan qaldırmaq üçün yeni proqramlaşdırma dilinə ehtiyac vardı. Assembler dilindən fərqli olaraq, həmin dilin yaradılmasından əsas məqsəd onu maşının anlaması deyil, onunla işləməyin insan üçün əlverişliyi idi.

Beləliklə, insan (proqramçı) üçün daha anlaşıqlı olan və proqramlaşdırma prosesini asanlaşdıran yeni dillər yaradılmağa başladı. Proqramlaşdırma dilinin hər bir yaradıcısı insan-maşın əlaqələri haqqında öz təsəvvürlərini gerçəkləşdirdiyindən qısa müddət ərzində yüzlərlə yeni dil meydana çıxdı (1993-cü ilin hesablamalarına görə, 1950-ci ildən həmin vaxta kimi 1000-dən artıq proqram-

laşdırma dili yaradılıb). Təbii, *yüksək səviyyəli* [high-level] dillər adlandırılan dillərin az bir qismi öz yerini tapıb inkişaf etdi və möhkəmləndi. Bunun əksinə olaraq, assembler dili *aşağı səviyyəli* [low-level] dil hesab olunur, çünki o, maşın dilinə daha yaxındır və kompüterin qurğuları ilə işləyir.

Yüksək səviyyəli dillərin öz müsbət və mənfi cəhətləri var. Yüksək səviyyəli dilin assemblerdən başlıca üstünlüyü onun öyrənmək və istifadə üçün çox-çox asan olmasıdır. Yüksək səviyyəli dildə yazılmış proqram assemblerdəkinə nisbətən daha yığcam və anlaşıqlıdır. Onlar, əsasən, daşınabiləndir, yəni müxtəlif prosessorlu kompüterlərdə eyni cür işləyir. Bu isə o deməkdir ki, proqramı yazarkən onun işləyəcəyi kompüterin arxitekturasının incəliklərini öyrənməyə ehtiyac qalmır. Əlbəttə, bu halda hər bir prosessorun öz kompilyatoru olmalıdır və onun yaratdığı icra faylı yalnız həmin prosessor üçün yararlı olacaqdır.

Ən yaxşı proqramlaşdırma dili hansıdır?

Hər bir proqramlaşdırma dilinin öz tərəfdarları və əleyhdarları var.

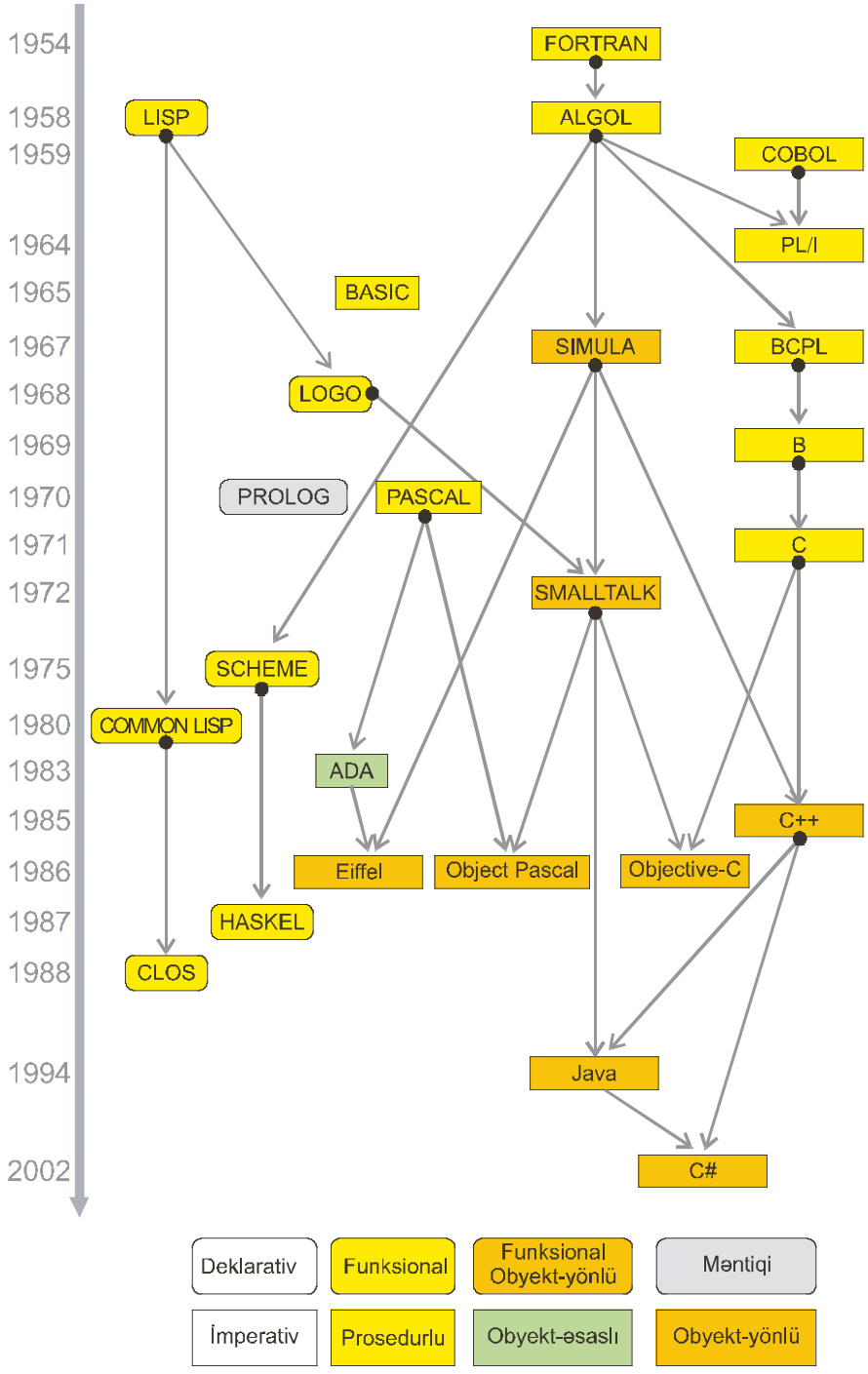
İndi dünyada bir neçə min proqramlaşdırma dili mövcuddur. Bəs bu qədər dilin içərisindən sizə daha münasib dili necə seçməli?

Mühəndislərin, bankirlərin, hərbçilərin qarşısında cürbəcür məsələlər durur. Onlar özləri üçün müxtəlif proqramlaşdırma dilləri seçirlər. Mühəndislər FORTRAN dilinə üstünlük verir, bankirlər çox zaman COBOL (“kobol”) dilindən istifadə edir, hərbçilər isə döyüş əməliyyatlarını planlaşdırmağı və qoşunun idarə olunmasını ADA dilində yazılmış proqramların köməyilə həyata keçirirlər. Süni intellekt sahəsində çalışanlar üçün PROLOG, yaxud LISP dili daha münasibdir. İnternet üçün proqram yazan proqramçılar, adətən, JAVA dilinə üstünlük verirlər.

Bu sadaladığımız proqramlaşdırma dillərinin hamısı xüsusi dillərdir. Bu dillərin hər birində elə operatorlar var ki, onlar vasitəsilə xüsusi məsələləri daha asanlıqla həll etmək olur. Adətən, həmin dilləri iş prosesində ehtiyac olduqca öyrənirlər, belə ki, onları “qabaqcadan” öyrənməyin mənası yoxdur.

Xüsusi dillərdən savayı, universal proqramlaşdırma dilləri də mövcuddur. Onların köməyilə, demək olar ki, istənilən məsələni həll etmək mümkündür. Belə dilləri “qabaqcadan” öyrənmək olar – onlar hər zaman gərəyiniz ola bilər. Belə dillərin üçü daha çox populyardır:

- Basic
- Pascal
- C++



Yüksək səviyyəli dillərin təkamül sxemi

Digər tərəfdən, assemblerdə yazılmış proqram, onun yüksək səviyyəli dildə yazılmış bənzərindən praktik olaraq həmişə səmərəlidir. Yəni yüksək səviyyəli dilin kompilyatorunun yaratdığı icra faylı funksional cəhətdən eyni olan assembler proqramından daha çox yer tutur və yavaş işləyir. Doğrudur, son illər mikroprosessorların inkişafı nəticəsində kompilyatorlar daha optimal kodlar generasiya edir.

Yüksək səviyyəli dillərin köməyilə istənilən sahəni proqramlaşdırmaq olar. Ancaq elə dillər də var ki, onlar xüsusi olaraq müəyyən sahələr üçün nəzərdə tutulub:

- ALGOL – riyazi məsələlər üçün;
- CHILL – telekommunikasiya sistemləri üçün;
- COBOL – iqtisadi məsələlər üçün;
- FORTRAN – riyazi hesablamalar üçün;
- Java – obyektlərlə iş üçün;
- Linda – verilənlərin paralel emalı üçün;
- PostScript – görüntülərin təsviri üçün;
- PROLOG – süni intellekt məsələləri üçün.

1. "Yüksək səviyyəli" və "aşağı səviyyəli" proqramlaşdırma dili nə deməkdir?
2. Yüksək səviyyəli dillərdə proqramlaşdırmanın maşın dilində proqramlaşdırmadan hansı üstünlükləri var?
3. Yüksək səviyyəli dillərin müsbət və mənfi cəhətlərini izah edin.
4. Ən yüksək səviyyəli proqramlaşdırma dilini necə təsəvvür edirsiniz?

1.4. PROQRAMLARIN HAZIRLANMASI

Proqramlaşdırma nəzəriyyəsi ilə praktik proqramlaşdırma paralel inkişaf etmişdir. İnkişafın ilk mərhələsində informasiya emalının riyazi nəzəriyyəsi hazırlanmış və elə həmin dövrdə də proqramların düzgünlüyünü yoxlayan vasitələr və səmərəli *translyatorların* yaradılması prinsipləri işlənib hazırlanmışdır.

O zamanlar proqramçılara çox yüksək ixtisaslı işçilər kimi baxılırdı. Bu, nadir peşələrdən idi, proqramların istehsalı isə hələ kütləvi xarakter daşımırdı.

Kompüterlərin sonrakı inkişafı və yayılması ağırlıq mərkəzini tətbiqi sahəyə keçirdi. Proqramçıların, eləcə də proqramların sayı artaraq, milyonlarla hesablanmağa başladı. Bununla belə, ixtisaslı proqramçıları müəyyənləşdirən bilik səviyyəsi aşağı düşdü.

Getdikcə mürəkkəb proqramların hazırlanmasında daha çox orta səviyyəli proqramçılar iştirak etməyə başladı.

Kiçik və orta həcmli proqramların hazırlanması. Kiçik və orta həcmli (bir neçə min sətirlik) hər hansı bir proqram ayrı-ayrı proqramçılar tərəfindən yaradıla bilər. Onların yaradılması adətən iki mərhələdə baş verir:

Birinci mərhələ *təhlil (analiz)* mərhələsidir. Bu mərhələdə proqramın təyinatı aydınlaşdırılır, alqoritm düşünülüb tapılır, verilənlərin strukturu müəyyənləşdirilir, proqram obyektləri qeyd olunur və onlar arasındakı qarşılıqlı əlaqələr aşkar edilir.

İkinci mərhələ *kodlaşdırmadır*. Bu mərhələdə alqoritm hər hansı proqramlaşdırma dilində yazılır. Kiçik proqramlar üçün bu, maksimal əmək tələb edən əsas mərhələdir. Testləmə və sazlama çox da böyük zəhmət tələb etmir. Belə layihələrin bir proqramçı tərəfindən icrası yarım ilədək çəkə bilər.

Böyük proqramların hazırlanması. Böyük proqramların həcmi milyonlarla sətrə çatır. Onların hazırlanmasına onlarla, bəzən isə yüzlərlə proqramçı bir neçə il birgə əmək sərf edir. Belə proqramların hazırlanması bir neçə ardıcıl mərhələdən ibarətdir.

Proqramların hazırlanma mərhələləri:

1. Məsələnin qoyuluşu və təhlili.
2. Texniki tapşırığın hazırlanması.
3. Layihələndirmə və kodlaşdırma.
4. Testləmə və sazlama.
5. Tətbiq edilmə.
6. Müşayiət.

Birinci mərhələdə *layihəyə olan tələblər təhlil olunur.* Bu, layihənin uğurla sona çatmasını müəyyənləşdirən ən önəmli mərhələdir. İri layihələrin uğursuz-

luğu çox zaman bu mərhələdə buraxılmış səhvlərlə bağlı olur. Tələblərin təhlili gedişində proqramın təyinatı dəqiqləşdirilir, giriş və çıxış verilənləri aydınlaşdırılır. Tələb olunan resurslar və layihənin dəyəri qiymətləndirilir.

Növbəti mərhələ *proqramın spesifikasiyalarının hazırlanmasıdır*. Bu mərhələdə proqramçılar üçün texniki tapşırıqlar formalaşdırılır, iş sənədləri hazırlanır və işin təqvim planı qurulur.

Sonra proqramın *layihələşdirilməsi* və *kodlaşdırılması* işləri başlanır. Böyük layihələrdə buna əsas mərhələ kimi baxılmır.

Kodlaşdırma sona çatdıqda (bəzən isə daha öncədən), proqramın *testlənməsi* və *sazlanması* başlanır. Testləmə mərhələsində proqramın düzgünlüyü, onun işinin səmərəliliyi, korrekt olmayan hərəkətlərə və texniki nasazlıqlara dözümlüyü, kritik rejimlərdə işləmə etibarlılığı yoxlanılır. Müəyyən olunmuş nasazlıqlar proqramçılar tərəfindən dərhal aradan qaldırılır.

İnterpretatorlar. Kompilyatorlar

Yüksək səviyyəli dildə yazılmış proqramın kompüterin mərkəzi prosessoru tərəfindən başa düşülüb icra olunması üçün o, maşın koduna çevrilməlidir. Bu çevrilmə müxtəlif yollarla aparıla bilər:

Birinci yol ilkin proqramın hər bir sətrini maşın koduna *çevirən* (translyasiya edən, ing. *translate*) proqramı başlatmaqdır. Bu proqram bir sətri çevirir və onu yerinə yetirilmək üçün mərkəzi prosessora verir, yalnız bundan sonra növbəti sətrin çevrilməsinə keçir. Belə proqrama *interpretator* (ing. *interpreter*) deyilir.

Bu proqramı iki müxtəlif dilli şəxsin ünsiyyətinə kömək edən tərcüməçi ilə müqayisə etmək olar. Birinci şəxs nəsə deyir, tərcüməçi onun dediklərini tərcümə edir. İkinci şəxs cavab verir və onun cavabı birinci şəxsə tərcümə olunur. Bu proses bütün dialoq müddətində davam edir.

Bu yanaşmanın üstünlüyü onun istifadəçi üçün sadəliyindədir. Proqram yazılıb-başladıldıqdan dərhal sonra kompüterin hər dəfə nə etdiyini görmək olur. Əgər proqramda nəyisə dəyişmək lazımdırsa, bu dəyişiklik edilir və proqram yenidən translyasiya olunur. Ancaq bu yolun bir çatışmazlığı var: proqram tam hazır olduqdan sonra da, hər dəfə icra olunmazdan qabaq onun hər bir sətri təzədən maşın koduna çevrilir və nəticədə proqramın ümumi icra müddəti uzanır.

Başqa bir müqayisə aparaq. Bu kitabı Azərbaycan dilində yazıb sonradan rus dilinə çevirmək üçün naşir tərcüməçi tutur və kitabı bütövlükdə rus dilinə tərcümə etdirir. Translyatorun başqa növü olan *kompilyator* da (ing. *"compile"* – "tərtib etmək", "yığmaq") belə işləyir: proqram yüksək səviyyəli dildə tam yazıldıqdan sonra kompilyator onu oxuyur, proqramı maşın koduna çevirir və ayrı bir faylda saxlayır. Sonradan ilkin koddan asılı olmayaraq, bu fayl istənilən dəfə çalışdırıla bilər. Bu zaman, aydındır ki, yenidən translyasiyaya lüzum qalmır.

Əgər proqram konkret sifarişçinin tələbləri nəzərə alınmaqla yazılmışsa, adətən, proqramın *tətbiq edilmə* mərhələsinə zərurət yaranır. Bu mərhələdə avadanlıq köklənir, əvvəllər istifadə olunmuş proqramlardan verilənlər yeni proqrama keçirilir, proqramla işləyəcək heyət təlim keçir.

Proqramla işin son mərhələsi *müşayiətdir*. Bu mərhələdə istifadəçilərə məsləhətlər verilir, istismar müddətində aşkar olunan xətalar düzəldilir, sifarişçinin istəyinə (xahişinə) görə proqramlarda çox da böyük olmayan dəyişikliklər edilir.

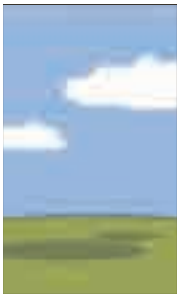

1. Sifarişçi görüləsi işi belə izah edir.


2. Layihə rəhbəri onu belə başa düşür.


3. Analitik işi bu cür layihələndirir.


4. Proqramçı proqramı belə yazır.

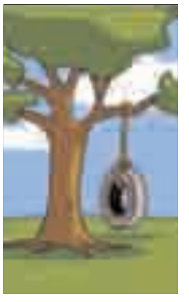

5. Biznes məsləhətçisi bunu belə təsvir edir.


6. Layihə rəsmi olaraq belə sənədləşdirilir.


7. Bu quraşdırılıb.


8. Sifarişçiyə bu işin hesabı təqdim olunur.


9. Layihə belə dəstəklənib.


10. Sifarişçiyə əslində bu lazım idi.

1. Böyük proqramların hazırlanması mərhələləri hansılardır?
2. Translyatorun vəzifəsi nədir?
3. Kompilyatorla interpretatorun fərqi nədədir?

1.5. TURBO PASCAL REDAKTORU

Pascal (“Paskal” kimi oxunur) proqramı hazırda **Turbo Pascal** təki tanınır və istifadə olunur. Bu bölümdə Pascal dilini praktik olaraq Borland International şirkətinin hazırladığı TURBO PASCAL 7.0 vasitəsilə öyrənəcəyik. Turbo Pascalın həm MS-DOS, həm də Windows versiyaları vardır.

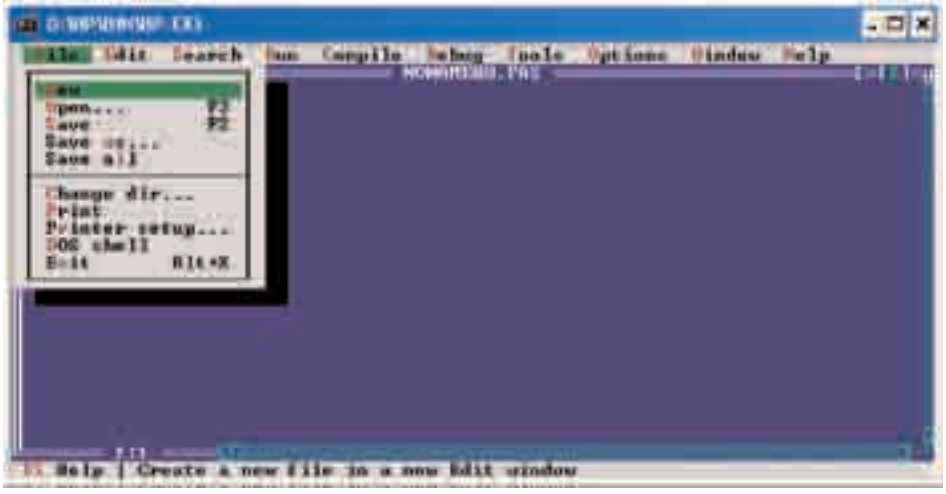
Niklaus Emil Virt (1934)

Pascal proqramlaşdırma dili 1971-ci ildə isveçrəli fizik Niklaus Virt [Nicklaus Wirth] tərəfindən yaradılıb, fransız riyaziyyatçısı və filosofu Blez Paskalın şərəfinə adlandırılıb. Bu dil ən geniş yayılmış proqramlaşdırma dillərindən biridir. Başqa dillərdən proqramların aydın və məntiqi yazılma imkanlarına görə seçilir ki, bu da onu həm yeni başlayanlar, həm də təcrübəli proqramçılar üçün əlverişli edir.

Niklaus Virt MODULA və OBERON dillərinin də yaradıcısıdır.

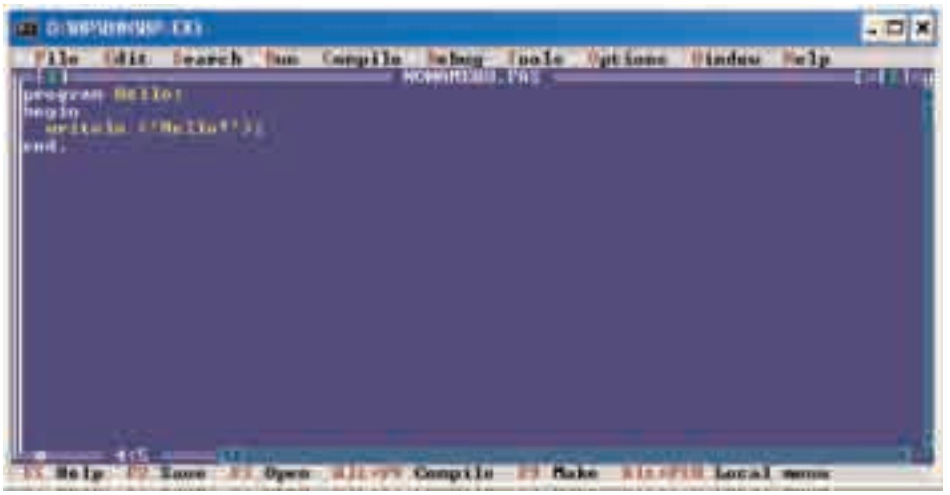
Turbo Pascalın başladılması. Turbo Pascal proqram məhsulu *inteqrasiya olunmuş mühitdir*. Bu o deməkdir ki, siz Turbo Pascal mühitini tərk etmədən proqramı yarada, redaktə, kompilyasiya edə və çalışdıra bilərsiniz.

Turbo Pascal proqramını başlatmaq üçün proqramın quraşdırıldığı qovluqda (adətən, \BP\BIN altqovluğunda) turbo.exe, yaxud bp.exe proqramlarından birini çalışdırmaq lazımdır. Başlatmadan dərhal sonra ekranda şəkildə gördüyünüz pəncərə açılır.



Pəncərənin yuxarısında proqramın baş menyusu (File, Edit, Search və s.) yerləşir. Menyunun hər hansı bəndini seçmək üçün, sadəcə, siçanın göstəricisini həmin bəndin üzərinə aparıb, sol düyməni çıqqıldatmaq kifayətdir. Əgər siz klaviatura ilə işləməyi xoşlayırsınızsa, <Alt> klavişini basılı saxlayıb, menyu bəndinin adında seçdirilmiş hərfə uyğun klavişi basmaqla da həmin bəndi seçmiş olursunuz. Məsələn, File bəndini seçmək üçün <Alt+F> klavişlər kombinasiyasını basmaq lazımdır.

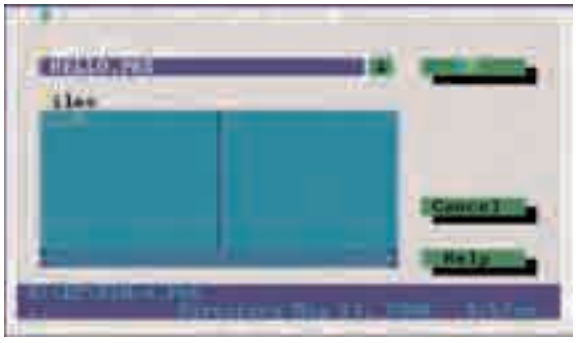
Yeni proqramın yaradılması. Pascal dilində yeni proqram yaratmaq üçün öncə File menyusunda New bəndini seçməklə boş pəncərə açmaq lazımdır. Nəticədə NONAME00.PAS adında boş fayl yaradılacaq. Sonradan yenə fayllar yaradılarkən onlara avtomatik olaraq NONAME01.PAS, NONAME02.PAS və s. adlar veriləcək. Bir şeyi unutmayın ki, yaradılan bu fayllar kompüterin operativ yaddaşında saxlanılır. Sonradan onları diskə yazmaq gərəkdir.



Turbo Pascal redaktoru proqramın mətnini yazmaq və onu redaktə etmək üçün kifayət qədər imkanlara malikdir. Proqramın mətni, adi mətn redaktorunda olduğu kimi, klaviaturadan daxil edilir. Bu zaman proqramın əsas mətni sarı, Pascal dilinin elementləri olan *açar sözlər* isə ağ rənglə işıqlanır (menyunun Options⇨Environment⇨Colors bəndi vasitəsilə uyğun dialoq boksunu açıb, bu rəngləri dəyişə də bilərsiniz). Açar sözlərin bu cür seçdirilməsi proqramın mətninin daxil edilməsi mərhələsində bir sıra səhvləri aradan qaldırmağa imkan verir.

Proqramın mətnini yığıb sona çatdırdıqdan sonra onu diskdə saxlamaq lazımdır. Bunu yalnız işin sonunda deyil, proqramın yaradılması prosesində etmək məsləhət görülür. Proqramı saxlamaq üçün baş menyudan File⇨Save bəndini

seçmək, yaxud <F2> klavişini basmaq lazımdır. Bu zaman ekranda Save File As dialoq boksu açılacaq ki, orada faylın saxlanılacağı yer və faylın adı göstərilməlidir. Susqunluqla təklif olunan NONAME00.PAS, NONAME01.PAS və s. kimi adları proqramın mahiyyətinə uyğun gələn daha anlaşıqlı adlarla əvəzləmək məsləhətdir (məsələn, HELLO.PAS kimi).

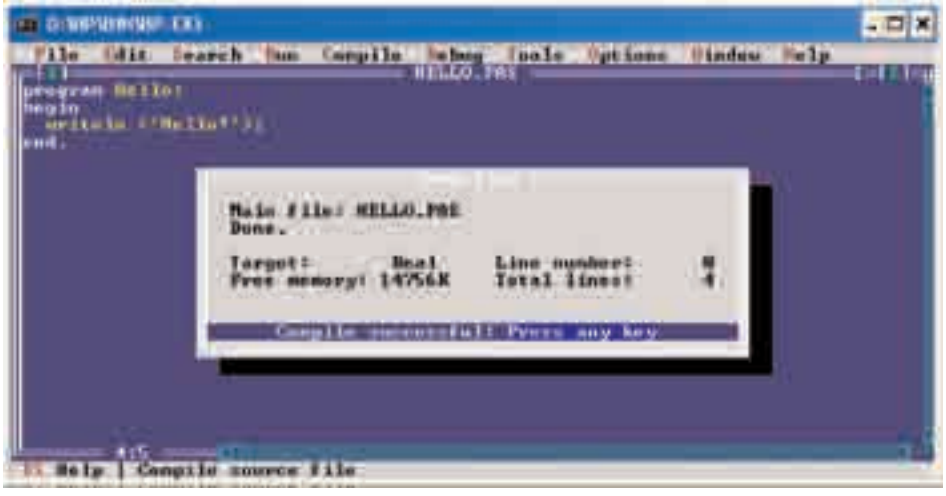


Proqramın kompilyasiyası və başladılması. Proqramın mətninin yığılmasını sona çatdırdıqdan sonra onu kompilyasiya etmək lazımdır. Bunun üçün menyunun Compile bəndi nəzərdə tutulub. O, seçildikdə redaktor pəncərəsindəki proqram kompilyasiya olunmağa başlayır. Əgər proqramın mətnində səhv aşkarlanarsa, bu barədə uyğun məlumat çıxacaq. Bu zaman kursor kompilyasiya prosesinin kəsildiyi yeri göstərəcək. Səhvi tapıb düzəltdikdən sonra yenidən Compile bəndini seçmək lazımdır. Proqramın mətnində səhv yoxdursa, ekrana

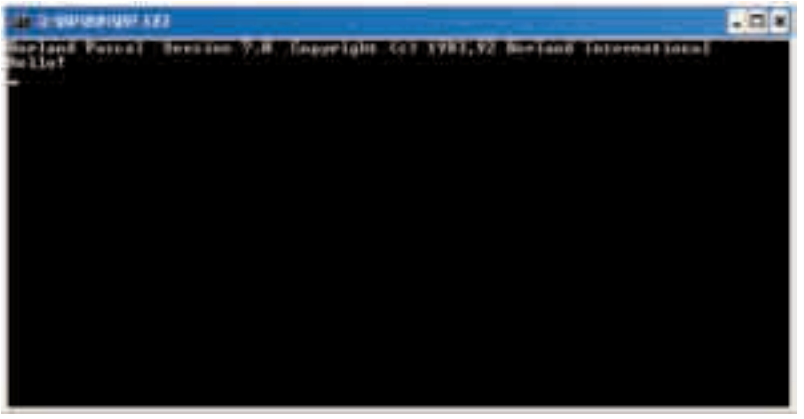
```
Compile successful: Press any key
```

(Kompilyasiya uğurla keçdi: ixtiyari klavişi basın)

məlumatı çıxır. İstənilən klavişi bassanız, redaktor pəncərəsinə dönəcəksiniz.



İndi kompilyasiya olunmuş proqramı başlatmaq üçün menyuda Run bəndini seçmək lazımdır. Bu zaman ani olaraq istifadəçi ekranı görünəcək.



Bu ekrana rahat baxmaq üçün <Alt+F5> klavişlər kombinasiyasından istifadə etmək olar. Açılan istifadəçi ekranı hər hansı klavişi basanadək gözünüzün qabağında olacaq.

Mövcud proqramın açılması. Diskdə saxlanmış faylı redaktor pəncərəsinə yükləmək üçün Open a File dialoq boksu ekrana çağırılmalıdır. Bunun üçün File menyusunun Open bəndini seçmək, yaxud <F3> klavişini basmaq lazımdır. Göstərilən dialoq boksu açıldıqdan sonra sizdən ya lazım olan faylın adını Name sahəsinə yazmaq, ya da həmin faylın adını Files siyahısından seçmək tələb olunur.

Turbo Pascaldan çıxış. Turbo Pascal mühitindən çıxıb əməliyyat sistemi mühitinə qayıtmaq üçün File menyusunun Exit bəndi nəzərdə tutulub. Bu bənd seçilərkən redaktor pəncərəsində saxlanmamış fayl qalarsa, sistem sizə həmin faylı diskdə saxlamaq imkanı verəcək. Bunun üçün ekrana xüsusi dialoq boksu çıxacaq.

İndi <Y> klavişini bassanız, Turbo Pascaldan çıxış zamanı saxlanmamış fayl da saxlanacaq.

1. İnteqrasiya olunmuş mühit nə deməkdir?
2. Proqramda olan səhvləri necə müəyyən etmək lazımdır?
3. Turbo Pascal mühitində proqram icra olunmaq üçün necə başladılır?
4. Yerinə yetirilmədən sonra proqramın nəticələrinə necə baxmaq olar?

1.6. PROQRAMIN ÜMUMİ STRUKTURU

Pascal dilində yazılmış istənilən proqram iki hissədən – *verilənlərin təsviri bölümündən* və *proqramın gövdəsindən* ibarət olur.

Proqramın mətnində verilənlərin təsviri proqramın gövdəsindən əvvəl gəlir. Bu da dilin əsas qaydasından qaynaqlanır.

> Proqramda rast gəlinən istənilən obyekt qabaqcadan təsvir olunmalıdır!

Verilənlər *operatorlar* vasitəsilə emal olunur. Operatorlar barəsində növbəti dərsimizdə daha ətraflı danışacağıq.

Proqramın gövdəsi begin sözü ilə başlanır və operatorlar yığınından ibarət olur. Ona görə də bu hissəyə *operatorlar bölümü* də deyilir. Bu bölüm end açar sözü ilə bitir (sonda nöqtə qoyulur). Pascal dilində proqram aşağıdakı şəkildə olur:

```
program < proqramın adı >;
 < dəyişənlərin təsviri >
begin
 < operatorlar >
end.
```

Proqram istənilən cür sətirlərə bölünə bilər – bununla onun mənası dəyişmir (təkcə sözlərin sətirdən-sətrə keçirilməsinə icazə verilmir). Buna görə də çalışmaq lazımdır ki, proqramlar mümkün qədər anlaşıqlı yazılsın.

Şərhlər. Proqram yazarkən siz nə etməli olduğunuzu bilirsiniz. Ancaq müəyyən müddətdən sonra həmin proqrama qayıtmalı olsanız, qəribə olsa da, görəcəksiniz ki, çox şeyi unutmusunuz. Buna görə də həm özünüzün xatırlamanız, həm də başqalarının sizin proqramı anlaması üçün proqramın müəyyən yerlərinə şərhlər vermək yaxşı olardı.

```
BLAISE PASCAL
BEGIN
(* 1623 – 1662 *)
END.
```

Adından da göründüyü kimi, *şərhlər* proqramın mətnini oxuyan şəxs üçün qeyddir. Şərhlərdən proqramın nə məqsədlə yaradıldığı, onun yaradıcısı haqqın-

da məlumatı, proqramın son dəyişdirilmə tarixini, proqramdakı dəyişənlərin, funksiyaların təyinatını və s. göstərmək üçün istifadə edilə bilər.

Pascal dilində şərhlər (* və *) simvollarının, yaxud { və } fiqurlu mötərizələrin arasında yazılır. Proqram maşın koduna çevrilərkən bu simvollar arasında yazılanlar nəzərə alınmır.

```
program Words
{
 Proqramçı: Alpay Calallı
 Proqramın yaradılma tarixi: 13.05.2008
 Bu proqram mətndə olan sözlərin sayını
 hesablayır.
}
```

Şərhlər həmişə mötərizədə "ulduzlar", yaxud fiqurlu mötərizələr arasında yerləşdirilir:

(* ...*)

{ ...}

```
program Comments;
(* şərhlərin tətbiqinə aid sadə proqram *)
var     a : Integer;
begin (* başlanğıc *)
   a:= 1;
   WriteLn('Bunların hamısı çap olunacaq, a=', a)
    { bu şərhlərin heç birinə lüzum yoxdur }
end.
```

İdentifikatorlar. Proqramlaşdırma dillərində müxtəlif obyektləri, məsələn, dəyişənləri, konstantları, funksiyaları və s. adlandırmaq üçün *identifikatorlardan* istifadə olunur. İdentifikatorlar hərf, rəqəm və bəzi xüsusi simvollardan ibarət olur. İdentifikatorların müxtəlif proqramlaşdırma dillərində yazılış qaydaları fərqli olsa da, onların əsas prinsipləri bunlardır:

1. İdentifikator hərf və rəqəmlərdən ibarət ola bilər. O yalnız hərflə başlanmalıdır.

2. İdentifikatorda boşluq simvolu və durğu işarələri ola bilməz. Bəzi xüsusi işarələrə, məsələn, "_", yaxud "$" işarələrinə icazə verilir.
3. Bütün proqramlaşdırma dillərində dilin operatorlarını yazmaq üçün açar sözlər mövcuddur. İdentifikator heç bir açar sözlə üst-üstə düşməməlidir.
4. İdentifikator həm kiçik, həm də baş hərflərlə yazıla bilər. Yalnız hərflərinin böyük-kiçikliyi ilə bir-birindən fərqlənən identifikatorlara müxtəlif proqramlaşdırma dillərində müxtəlif cür yanaşma var. Məsələn, Bir və bir identifikatorları BASIC və Pascal dillərində eyni, C dilində isə fərqli hesab olunur.

Mümkün identifikatorlara örnək olaraq bunları göstərmək olar:

```
i
a
t0123456789
NoClass
```

Aşağıdakı identifikatorlardan istifadə etmək olmaz:

`1stPlace` – rəqəmlə başlandığına görə;

`one and one` – boşluq simvolları olduğuna görə;

`yes(no)` – mötərizələr olduğuna görə.

İdentifikatorlar iki qrupa ayrılır:

1) standart identifikatorlar;
2) istifadəçinin təyin etdiyi identifikatorlar.

Yuxarıda verilmiş identifikatorlar ikinci qrupa aiddir. Standart identifikatorlara misal olaraq dilin öz prosedurlarını – ReadLn, WriteLn, Real və s. göstərmək olar.

Dəyişənlər. Standart tiplər. Yuxarıda qeyd olunduğu kimi, Pascal proqramında verilənlərin təsvirində məqsəd kompilyatora bütün identifikartorların adları və hər bir identifikatordan necə istifadə olunacağı haqqında məlumat vermək dir. Bundan başqa, bu bölüm proqramın istifadə edəcəyi hər bir yaddaş xanasında hansı verilənlərin yerləşəcəyini kompilyatora bildirir.

Yaddaşda hər hansı qiymətin göstərilməsi həmin qiymətin tipindən asılı olur. Standart Pascal dilində verilənlərin qabaqcadan təyin olunmuş dörd tipi vardır: Integer (tam ədədlər üçün), Real (həqiqi ədədlər üçün), Boolean (məntiqi kəmiyyətlər üçün) və Char (ayrıca simvollar üçün). Turbo Pascalda simvollar

ardıcıllığı ilə işləməyə imkan verən daha bir verilənlər tipi (string) vardır ki, onun haqqında ayrıca dərsdə danışılacaq. Hər bir tip üçün verilənlərin ala biləcəyi qiymətlər üzərində aparıla bilən əməliyyatlar çoxluğu vardır.

Integer. Riyaziyyatda tam ədədlər müsbət, yaxud mənfi ola bilər. Pascal proqramında verilənlərin tam olduğunu göstərmək, ədədləri təqdim etmək üçün Integer tipi nəzərdə tutulmuşdur. Bu tipə aid dəyişən 32768 və 32767 intervalında qiymət ala bilər. Proqramlarda qiyməti qabaqcadan müəyyən olunmuş MaxInt konstantından istifadə etmək olar (MaxInt = 32767). Integer tipinə aid qiymətlərə örnək olaraq

-1050, 425,15, -25 və s.

göstərmək olar.

Tam ədədləri ekrana çıxarmaq, üzərində müxtəlif hesab əməlləri (toplama, çıxma, vurma və bölmə) aparmaq, eləcə də onları müqayisə etmək olar.

Real. Hər bir həqiqi ədəd nöqtə (vergül) ilə ayrılmış tam və kəsr hissədən ibarət olur. Pascal dilində verilənlərin həqiqi olduğunu göstərmək üçün Real tipi nəzərdə tutulub. Bu tipə aid qiymət həmişə rəqəmlə başlanıb, rəqəmlə bitməlidir. Buna görə də -25 kəsrini və 64 tam ədədini Real tipinə aid etmək üçün onlar uyğun olaraq -0.25 və 64.0 kimi göstərilməlidir.

Həqiqi ədədləri oxuyub ekrana çıxarmaq, üzərində müxtəlif hesab əməlləri (toplama, çıxma, vurma və bölmə) aparmaq, eləcə də onları müqayisə etmək olar.

Char. Bu tipə aid qiymətlər ayrıca simvollar – hərflər, rəqəmlər, yaxud xüsusi işarələr ola bilər. Char tipli qiymət bir simvoldan ibarət olur və apostrof alınır:

'A', 'z', '1', ':', '"', ' '

Burada sonuncudan əvvəlki qiymət " (dırnaq) işarəsi, sonuncu isə boşluq simvoludur.

Char tipli qiymətlər üzərində hesab əməllərini yerinə yetirmək olmaz. Başqa sözlə, Pascal dilində '3'+'5' əməli yolverilməzdir. Biz simvolları yalnız müqayisə edə, oxuya və ekrana çıxara bilərik.

Boolean. Başqa tip verilənlərdən fərqli olaraq, Boolean tipli dəyişən yalnız iki qiymət ala bilər: True (doğru) və False (yalan). Proqramlarda bu tipli dəyişənlərdən müəyyən qərar qəbul etmək lazım gəldikdə istifadə etmək olar. Boolean tipli qiymətləri ekrana çıxarmaq olar, ancaq belə

qiymətləri klaviaturadan daxil etmək olmaz. Bu tip dəyişənlər üzərində **not** (deyil), **and** (və) və **or** (və ya) əməllərini yerinə yetirmək olar.

> Dəyişən sözü proqramlaşdırmaya riyaziyyatdan keçib. Riyaziyyatçılar "dəyişən kəmiyyət", "asılı kəmiyyət" anlayışlarından istifadə edirlər. Həmin kəmiyyətlərə funksiya da deyilir. Bundan başqa, riyaziyyatçılar *arqument* adlandırdıqları *sərbəst (asılı olmayan) dəyişən* anlayışından da istifadə edirlər.
>
> $$S = v \cdot t$$
>
> düsturu riyaziyyatda və fizikada onu bildirir ki, S (yol) kəmiyyəti sərbəst v (sürət) və t (zaman) dəyişənlərindən asılıdır.
> Proqramlaşdırma dillərini ilk yaradanlar riyaziyyatçılar olduğundan onlar riyazi terminologiyanı proqramlaşdırmaya da keçirmişlər.

Münasibət əməlləri. Tam ədədlərdən ibarət cütlər üzərində təyin olunmuş və nəticədə məntiqi qiymət verən əməllər də vardır. Həmin əməllər bunlardır:

=	bərabərdir	<>	bərabər deyil (fərqlidir)
<	kiçikdir	>	böyükdür
<=	kiçikdir və ya bərabərdir	>=	böyükdür və ya bərabərdir

Bu əməllər həqiqi ədədlər üzərində də təyin olunub. Simvol qiymətlər üzərində isə yalnız = və <> əməlləri təyin olunmuşdur.

Proqramda dəyişənlərin təsviri hissəsi. Bu bölüm

```
var təsvir1; təsvir2; təsvir3; ...
```

şəklindədir. Burada "**var**" açar sözdür (ingiliscə "**var**iable" – "dəyişən" sözünün qısaltmasıdır), hər bir təsvir isə bir, yaxud bir neçə identifikatordur; identifikatorlar bir-birindən vergüllə ayrılır, onlardan sonra "**:**" (iki nöqtə) və axırda tipin adı gəlir. Təsvir operatoru kompüterə proqramda hansı dəyişənlərin olduğunu və onların tiplərini bildirir. Proqramda olan bütün dəyişənlər *təsvir*, yaxud *bəyan* olunmalıdır (özü də yalnız bir dəfə!).

```
var a, b, c : Real; c : Char;
    n, q1, MaxI : Integer;
        flag : Boolean;
```

İfadələr. Əməllərin öncüllüyü. Proqramların əksəriyyətində hesabi ifadələrsiz keçinmək mümkün deyil. Hesabi ifadələrdə istifadə olunan hesab operatorlarının siyahısı, onların yazılış və hesablanma qaydası aşağıdakı cədvəldə göstərilib.

Hesab operatoru	Operatorun adı	Örnək
+	Toplama	5 + 2 bərabərdir 7 5.0 + 2.0 bərabərdir 7.0
–	Çıxma	5 - 2 bərabərdir 3 5.0 - 2.0 bərabərdir 3.0
*	Vurma	5 * 2 bərabərdir 10 5.0 * 2.0 bərabərdir 10.0
/	Bölmə	5 / 2 bərabərdir 2.5 5.0 / 2.0 bərabərdir 2.5
div	Tam ədədlərin natamam (qalıqsız) qisməti	5 **div** 2 bərabərdir 2
mod	Qalığın hesablanması	5 **mod** 2 bərabərdir 1

Hər bir hesab operatoru konstant, dəyişən, yaxud başqa hesabi ifadə ola biləcək iki operandla işləyir. +, –, * və / operatorlarının operandları Real və Integer tipli ola bilər. Cədvəldən göründüyü kimi, +, –, * operatorlarından istifadə edərkən alınan nəticənin tipi operandların tipi ilə üst-üstə düşür, / operatorunun nəticəsi isə həmişə həqiqi ədəd olur. Sonuncu iki operatorun operandları (**div** və **mod**) isə yalnız tam ədədlər ola bilər.

Yuxarıda sadalanan operatorlardan istifadə etməklə konstant (sabit kəmiyyət) və dəyişənlərdən ifadə qurmaq olar, məsələn:

```
(a + b) / c
(MaxI * n + q1) div (n + q1)
(flag or not(a = b)) and (n <> q1)
```

İfadədə əməllərin yerinə yetirilmə ardıcıllığı mötərizələrlə müəyyənləşir. Bununla yanaşı, riyaziyyatda mövcud olan adi qaydalar da qüvvədədir, məsələn, vurma və bölmə əməlləri toplamadan əvvəl yerinə yetirilir və s.

1. İdentifikator nədir?

2. Bunlardan hansıları identifikator deyil? Səbəbini izah edin.

```
end   ReadLn    program      123XYZ      XYZ123
Y=Z   'Max'     Ay01         Ay_01       1 Ay
```

3. Pascal proqramı hansı əsas hissələrdən ibarət olmalıdır?

4. Dəyişənlərin adlandırılması qaydası necədir?

5. Aşağıdakı şərhlərdə sintaktik səhvləri düzəldin.

```
{ Bu şərhdir *)
{ Bu da {şərhə} oxşayır }
```

1.7. OPERATORLAR

Pascal dilindəki proqramlar dəyişənlərin təsvirindən və onlar üzərində aparılan müxtəlif əməliyyatlardan, başqa sözlə, *operatorlardan* ibarətdir.

Verilənlər operatorlar vasitəsilə emal olunur. Operatorlar iki cür olur: *icra olunmayan* (verilənləri və proqramın strukturunu təsvir etmək üçün) və *icra olunan* (müxtəlif əməliyyatları yerinə yetirmək üçün) operatorlar. Biz təsviretmə operatoru ilə tanış olduq. İndi isə bəzi icra olunan operatorlarla tanış olaq.

Mənimsətmə operatoru. Dəyişənlərə qiymətlər vermək, yaxud onları dəyişdirmək üçün proqramlaşdırma dillərinin hamısında *mənimsətmə operatoru* olur. Həmin operatorun ümumi şəkli belədir:

```
<identifikator> <mənimsətmə işarəsi> <ifadə>
```

Mənimsətmə operatorunun sol tərəfində yeni qiymət alacaq dəyişənin identifikatoru göstərilir. *Mənimsətmə işarəsi* müxtəlif proqramlaşdırma dillərində müxtəlif olur. Məsələn, BASIC və C dillərində mənimsətmə işarəsi adi *bərabərlik işarəsi* (=) kimi olduğu halda, Pascal dilində o, := (iki nöqtə və bərabərdir) simvollar kombinasiyası şəklindədir.

Mənimsətmə operatoruna nümunələr:

x := 5;	*x* dəyişəninə 5 ədədi mənimsədilir;
y := x;	*y* dəyişəninə x-in qiyməti mənimsədilir;
y := x + 10;	*y* dəyişəninin qiyməti x dəyişəninin qiymətindən 10 vahid artıq olur;
x := x - 2;	*x* dəyişəninə özündən 2 vahid az olan qiymət mənimsədilir;
y := y + 1;	*y* dəyişəninə özündən 1 vahid çox olan qiymət mənimsədilir.

```
program Happiness;
var
   I, You, We: Integer;
begin
   I := 1;
   You := 1;
   We := I + You;
end.
```

Giriş və çıxış operatorları. Proqram işləyərkən istifadə olunan bütün verilənlər kompüterin operativ yaddaşında yerləşir. Proqram başqa qaynaqlarda yerləşmiş verilənlərə *xarici verilənlər* kimi baxır. Verilənlərin xarici mənbələrdən alınıb-verilməsi əməliyyatlarına *giriş (daxiletmə)*, yaxud *çıxış əməliyyatları* deyilir.

> Giriş *(daxiletmə)* – verilənlərin xarici mənbədən qəbul edilməsi.
> Çıxış – verilənlərin xarici qəbulediciyə verilməsi.

WriteLn proseduru. İstifadəçi ilə interaktiv qarşılıqlı əlaqədə işləyən proqramlarda çox zaman ekrana çıxarma operatoru tətbiq olunur. Pascal dilində verilənləri ekrana çıxarmaq üçün WriteLn standart prosedurundan istifadə olunur. Ekrana çıxarılası dəyişənlər və ifadələr bu prosedurun parametrləri olur.

```
WriteLn('Cəmi ', a);
```

Bu operator iki elementi – `'Cəmi'` sətrini və `a` dəyişəninin qiymətini ekranda əks etdirir. Onun icrasınadək `a` dəyişəninin qiyməti, məsələn, 2.345 olarsa, ekrana

```
Cəmi  2.3450000000E+00
```

cıxacaq (əgər həqiqi ədədin ekrana çıxarılması zamanı heç bir format göstərilməmişsə, Pascal eksponensional formatdan istifadə edir).

Tutaq ki, proqramda

```
WriteLn('Cəmi ', a);
WriteLn;
WriteLn('Son ');
```

operatorlar ardıcıllığı var. Bu operatorların icrasından sonra ekranda

```
Cəmi  2.3450000000E+00
Son
```

sətirləri əks olunacaq. Gördüyünüz kimi, ikinci operatorda çıxış siyahısı olmadığından ekrana boş sətir çıxarılır.

Beləliklə, WriteLn proseduru çıxış siyahısında verilmiş hər bir dəyişəni, yaxud konstantı əks etdirir, sonra isə kursoru növbəti sətrin başlanğıcına keçirir. Əgər çıxış siyahısında apostrofa alınmış sətir varsa, apostroflar çap olunmur (ekrana çıxarılmır). Çıxış siyahısı boşdursa, WriteLn prosedurunun icrasından sonra kursor, sadəcə olaraq, növbəti sətrin başlanğıcına keçəcək.

Write proseduru. Pascal dilində verilənləri çıxışa vermək üçün daha bir prosedur – Write proseduru nəzərdə tutulub. Bu prosedurun WriteLn proseduru-rundan yeganə fərqi bundadır ki, onun icrasından sonra kursor növbəti sətrin başlanğıcına keçmir. Tutaq ki, proqramda

```
Write('Cəmi ');
WriteLn(a);
```

operatorlar cütü vardır. Bu operatorların icrasının nəticəsi

```
WriteLn('Cəmi ', a);
```

operatorunun icrasının nəticəsi kimi olacaq.

ReadLn proseduru. Verilənləri klaviaturadan daxil etmək üçün Pascal dilində ReadLn proseduru nəzərdə tutulub. Bu prosedurun ümumi forması belədir:

```
ReadLn(giriş siyahısı)
```

ReadLn proseduru proqramın icrası zamanı istifadəçinin klaviaturadan daxil etdiyi verilənləri oxuyub, kompüterin yaddaşına yazır. Giriş siyahısında göstərilmiş hər bir dəyişən üçün istifadəçi bir element daxil etməli, sonra isə <Enter> klavişini basmalıdır. Giriş siyahısında dəyişənlərin adları vergüllə ayrılır. Verilənlərin daxiledilmə ardıcıllığı dəyişənlərin giriş siyahısındakı ardıcıllığına uyğun gəlməlidir. Daxil edilən ədədi verilənləri bir, yaxud bir neçə boşluq simvolu ilə ayırmaq lazımdır. Ədədi verilənlərin daxilində, yaxud onların arasında vergül olmamalıdır.

```
var a, b: integer;

...

Write('Ədədləri daxil edin: ');
ReadLn(a, b);
```

Read proseduru. Verilənləri klaviaturadan daxil etmək üçün daha bir vasitə – Read proseduru da vardır. Read və ReadLn prosedurları arasında əsas fərq ondan ibarətdir ki, Read proseduru verilənlər sətrində olan artıq simvolları oxumur (bu simvollar növbəti Read, yaxud ReadLn proseduru vasitəsilə oxuna bilər). ReadLn proseduru isə, əksinə, daxil edilən sətirdəki bütün simvolları emal edir (yalnız sətrin sonundakı artıq simvolları nəzərə almır).

Formatlama. Yuxarıda qeyd olunduğu kimi, xusüsi göstəriş yoxdursa, Pascal bütün həqiqi ədədləri eksponensional formatda əks etdirir. Bəs verilənləri bizə lazım olan formatda necə göstərmək olar? Öncə Integer tipli dəyişənlər, yaxud qiymətlər üçün formatların necə verilməsinə baxaq. Bununçün dəyişənin adından (yaxud qiymətdən) sonra iki nöqtə (:) və sahənin eni, yəni əks olunası rəqəmlərin sayı göstərilir. Məsələn,

```
Write('a = ', a:1);
WriteLn(' və b = ', b:2);
```

operatorları göstərir ki, a dəyişəninin qiyməti 1 rəqəmlə, b-nin qiyməti isə 2 rəqəmlə əks olunacaq. Məsələn, əgər a dəyişəninin qiyməti 7-yə, b-nin qiyməti isə 8-ə bərabərdirsə, onda ekrana çıxarılacaq sətir belə görünəcək:

```
a = 7 və b =  8
```

Diqqətlə baxsanız, b dəyişəninin qiymətindən qabaqda əlavə boşluq görərsiniz. Bu onunla izah olunur ki, b-nin qiyməti 1 rəqəmli olduğu halda, çıxış formatının 2 rəqəmli olması göstərilib (b:2).

Real tipli dəyişənlər, yaxud qiymətlərin çıxış formatını vermək üçün həm *sahənin enini*, həm də *onluq mərtəbələrin* sayını göstərmək lazımdır. Sahənin ümumi eni yetərincə böyük olmalıdır ki, onluq nöqtənin önündə və arxasında olan mərtəbələrin hamısı yerləşə bilsin. Bu zaman bir yer onluq nöqtə üçün, bir yer də minus işarəsi üçün ayırmaq lazımdır (çünki ədəd mənfi də ola bilər). Məsələn, əgər Real tipli X ədədi -99.9 və 999.9 aralığında qiymətlər ala bilərsə,

```
WriteLn(X :5 :1);
```

operatoru X-in qiymətini bir onluq işarə dəqiqliyilə ekrana çıxaracaq. Aşağıdakı cədvəldə tam və həqiqi ədədlərin formatlanmasına aid nümunələr verilib.

Qiymət	Format	Çıxış
234	:4	234
234	:5	234
234	:1	234
-234	:6	-234
3.14159	:5:2	3.14
3.14159	:5:3	3.142
0.1234	:4:2	0.12
-0.006	:8:3	-0.006
-0.006	:8:5	-0.00600
-0.006	:7:5	-0.00600

Konstantlar. Proqramda dəyişən kəmiyyətlərlə yanaşı sabit kəmiyyətlərdən də (konstantlardan) istifadə olunur. Pascal dilində konstantlar təyin etmək və onlara ad vermək imkanı vardır. Belə olan halda proqramın sonrakı mətnində həmin konstantın əvəzinə verilmiş adı işlətmək olar. Bütün konstantlar proqramın xüsusi bölümündə – *konstantların təsviri bölümündə* sadalanır.

```
const ad1 = qiymət1;
      ad2 = qiymət2;
      .........
      adN = qiymətN;
```

Burada ad1, ad2,... ixtiyari identifikatorlar, qiymət1, qiymət2, ... isə yuxarıda göstərilmiş qaydada yazılmış ədədlər, apostrofa alınmış simvollar, yaxud true, false konstantlarıdır, məsələn:

```
const g = 981E-2;
      atmosfer = 0.76;
      pi = 3.1415926;
```

Ən azı iki səbəbdən konstantların təsvir olunmasının faydası var.

Birincisi, sabit kəmiyyətlər üçün hərfi işarələmədən istifadə olunması fizika və riyaziyyatdan qalma ənənədir. Bu ənənəni saxlamaqla proqramlar daha anlaşıqlı edilir. Konstantlara mənalı ad verilməsi proqrama şərhlər verilməsi üsullarından biridir – "sətrin_uzunluğu" yazısı "60"-a nisbətən daha informativdir.

İkincisi, konstantların təsvir olunması proqramda dəyişiklik edilməsini yüngülləşdirir. Məsələn, çap olunan sətrin uzunluğunu 60 deyil, 40 götürmək üçün konstantların təsviri bölümündə "sətrin_uzunluğu = 60" yazısını "sətrin_uzunluğu = 40" ilə əvəz etmək kifayətdir. Əks halda, proqramda olan bütün 60 ədədlərini axtarıb tapmaq, onların sətrin uzunluğuna aid olduğunu müəyyənləşdirib 40 ilə əvəzləmək lazım gələrdi.

Yeni verilənlər tiplərinin müəyyən olunması. Proqramçı standart tiplərdən başqa, yeni verilənlər tipləri təyin etmək və onlara ad vermək imkanına malikdir. Bundan sonra standart tiplər kimi yeni tiplərdən də istifadə etmək olar. Tiplərin təyin olunması bölümü aşağıdakı şəkildə olur.

```
type  ad1 = təsvir1;
      ad2 = təsvir2;
      .........
      adN = təsvirN;
```

Burada ad1, ad2,... ixtiyari identifikatorlar, təsvir1, təsvir2, ... isə tiplərin təsviridir.

Dəyişənlərin inisiallaşdırılması. Giriş və çıxışı dəyişənlərin inisiallaşdırılması komandası ilə qarışdırmaq olmaz. *Dəyişənin inisiallaşdırılması* dedikdə proqramın icrasından qabaq, yaxud onun işləməsi vaxtı dəyişənlərə başlanğıc qiymətlərin mənimsədilməsi nəzərdə tutulur.

Pascal dilində dəyişəni yalnız onun elan edilməsi zamanı inisiallaşdırmaq olar, məsələn:

```
var i: Integer = 3;
```

Bu operator nə giriş operatorudur, nə də – mənimsətmə.

1. Proqramda konstantlara ad verməyin hansı üstünlükləri var?

2. Tutaq ki,

```
const
   MyPi = 3.14159;
   MaxI = 1000;
var
   X, Y : Real;
   A, B, I : Integer;

   A := 3;
   B := 4;
   Y := -1.0;
```

Aşağıdakı operatorlardan hansılarının düzgün olduğunu göstərin və hər bir yolverilən operatorun qiymətini müəyyənləşdirin.

```
a) I := A mod B
b) I := (990 - MaxI) div A
c) I := B div 0
d) X := A / Y
e) X := MyPi div Y
f) I := (MaxI - 990) mod A
```

3. `WriteLn` proseduru `Write` prosedurundan nə ilə fərqlənir?

4. Aşağıda sözlərlə verilmiş alqoritmin hər məqamına uyğun operator yazın.

1 və 100 aralığında hər hansı ədəd seçin.
Həmin ədədi özünə vurun.
Alınan ədədin üzərinə seçdiyiniz ədədin 4 mislini gəlin.
Nəticənin üzərinə 3 ədədini gəlin.
Alınan nəticəni seçdiyiniz ədəd plyus 3-ə bölün.
Seçdiyiniz ədədi qismətdən çıxın.
Cavabı ekrana çıxarın.

1.8. IF VƏ CASE SEÇİM OPERATORLARI

Alqoritm yerinə yetirilərkən göstərişlər bir-birinin ardınca emal olunur. Ancaq həyatda komandalar ardıcıllığını dəyişmədən həll edilən məsələlərə az hallarda rast gəlinir. Mürəkkəb məsələləri həll etmək üçün öz hərəkətlərini dəyişən çevik alqoritmlər tələb olunur.

Alqoritmlərdə bir neçə mümkün hərəkətdən birinin seçilməsinə *budaqlanma* vasitəsilə nail olunur. Budaqlanma təməl alqoritmik strukturlardan biridir. Budaqlanma bir, yaxud bir neçə şərtin yoxlanmasına əsaslanır və həmin şərtlərin doğruluğundan asılı olaraq müəyyən əməliyyat yerinə yetirilir.

Proqramlaşdırma dillərinin hamısında budaqlanmanı yerinə yetirən xüsusi operatorlar vardır. Belə operatorlara *şərt operatorları* deyilir. Sadə şərt operatoru iki hissədən ibarət olur:

1) şərtin özü;
2) icra olunan operator.

Şərt doğru olduqda icra operatoru yerinə yetirilir, əks halda o, buraxılır.

```
if x < 5 then x := x + 1;
```

Burada **if** (əgər) açar sözü şərt operatorunun başlanğıcını bildirir. Ondan sonra şərt verilir. Daha sonra **then** (onda) açar sözü gəlir. Axırda *icra operatoru* dayanır. Yoxlama zamanı şərt doğru olarsa, bu operator yerinə yetirilir, yalan olarsa, buraxılır. Belə şərt operatoruna *bir alternativli şərt operatoru* da deyilir.

Əgər şərt doğru olduqda bir neçə operatoru yerinə yetirmək lazım gələrsə, *"operator mötərizələrindən"* (begin və end açar sözlərindən) istifadə olunur. Bir neçə icra operatorunun olduğu şərt operatoru aşağıdakı kimi yazılır:

```
if X < 5 then
begin
   X := X + 1;
   Y := Y + 1;
end;
```

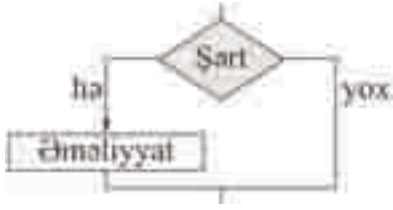
begin və **end** açar sözlərinə *operator mötərizələri* deyilir.

Şərt operatorunun tam forması. Çox zaman şərt doğru olduqda bir, yalan olduqda isə başqa bir əməliyyatın yerinə yetirilməsi tələb olunur. Bu halda şərt operatorunun *tam formasından* istifadə edilir.

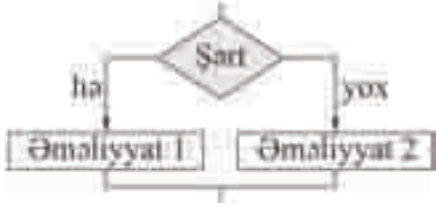
```
if   X < 5 then
     X := X + 1
else
     X := X - 1;
```

Şərt doğru olduqda **then** açar sözündən sonra gələn operator (operatorlar) yerinə yetirilir. Şərt yalan olduqda isə **else** (əks halda) açar sözündən sonra göstərilmiş operator (operatorlar) icra olunur. Belə şərt operatoruna *iki alternativli şərt operatoru* da deyilir.

> Pascal dilində **else** açar sözündən qabaq nöqtəli vergül (;) qoyulmur.



Bir alternativli
şərt operatorunun blok-sxemi



İki alternativli
şərt operatorunun blok-sxemi

Şərt operatorları zənciri. Bir çox alqoritmlərdə bir neçə şərtin yoxlanılması tələb olunur. Bu halda şərt operatorlarının zənciri əmələ gəlir. Məsələn:

```
if X < 5 then
   X := X + 1
else if X < 10 then
   X := X - 1
else if X = 13 or X = 15 then
   X := X * 2
else
   X := 10;
```

Şərt operatorları zəncirinə iç-içə operatorlar qrupu kimi baxmaq olar, ancaq onun yazılışında növbəti **if** açar sözü **else** ilə eyni sətirdə yazılmalıdır, çünki bu halda proqramın strukturu daha sadə və anlaşıqlı olur.

Zəncirdə olan və **else if** komandaları ilə ayrılmış icra operatorlarından yalnız biri yerinə yetirilir.

Seçim operatoru. Şərt operatorları zəncirinin qurulmasında istifadə olunan yoxlamaların hamısında yalnız bir ifadə iştirak edirsə və o tam qiymət alırsa, belə operatorlar qrupunu daha sadə şəkildə yazmaq olar. Bunun üçün *seçim operatoru* nəzərdə tutulub. Pascal dilində seçim operatoru **case of** sətri ilə başlanır. Bu iki sözün arasında yoxlanılası ifadə yazılır. Seçim operatorundan istifadə etməklə yuxarıdakı şərt operatorları zəncirini belə yazmaq olar:

```
case X of
   1..4:   X := X + 1;
   5..9:   X := X - 1;
   13,15: X := X * 2;
else
   X := 10;
end;
```

Yoxlanılan qiymətlər ayrıca ədədlər və intervallar şəklində verilir. İntervalı vermək üçün minimal və maksimal qiymətləri, aralarında iki nöqtə (**..**) olmaqla göstərmək lazımdır. Əgər eyni bir operatoru yoxlanılan ifadənin müxtəlif qiymətləri üçün yerinə yetirmək lazımdırsa, həmin qiymətlər (və intervallar) aralarında vergül qoymaqla yazıla bilər. Qiymətlər siyahısından sonra iki nöqtə

(:) qoyulur, ondan sonra isə yoxlanılan ifadənin qiyməti həmin siyahıdakı qiymətlərin biri ilə üst-üstə düşərsə, növbəti operator icra olunur. Əks halda ifadənin qiyməti növbəti sətirdəki siyahı ilə müqayisə olunur.

Əgər ifadənin qiyməti siyahılardan heç biri ilə üst-üstə düşmürsə, **else** açar sözündən sonra göstərilən operator yerinə yetirilir (**else** bölümü olmaya da bilər). Seçim operatoru həmişə **end** açar sözü ilə bitir.

Əgər eyni qiymət bir neçə siyahıda göstərilibsə, həmin qiymətin birinci rast gəlindiyi siyahıya uyğun operator yerinə yetirilir. Sonra idarəetmə seçim operatorunun ardınca gələn komandaya ötürülür.

1. Budaqlanma nədir və o, proqramlaşdırmada hansı operator vasitəsilə yerinə yetirilir?

2. Aşağıda göstərilmiş proqramın sətirlərinin yerlərini elə dəyişdirin ki, şərt operatoru düzgün alınsın.

```
b := a + 2
b := a - 2;
if a > 2 then
else
```

3. Şərt operatoruna hansı şərti yazmaq lazımdır ki, proqram `c` dəyişəninə `a` və `b` dəyişənlərinin ən kiçiyinin qiymətini mənimsətsin?

```
if ... then
   c := a
else
   c := b;
```

4. Aşağıdakı operator ekrana hansı məlumatı çıxaracaq?

```
if 12 < 12 then
   WriteLn ('Kiçikdir')
else
   WriteLn ('Kiçik deyil');
```

1.9. DÖVRLƏR. WHILE, FOR VƏ REPEAT OPERATORLARI

Məsələlərin həll alqoritmini qurarkən bəzən müəyyən komandalar ardıcıllığını bir neçə dəfə dalbadal yerinə yetirmək lazım gəlir. Əlbəttə, həmin ardıcıllığı tələb olunan qədər yazmaq da olar. Ancaq bu yol çox da əlverişli deyil.

Əgər komandaların və təkrarların sayı çox böyükdürsə, alqoritmin yazılışı çox uzun alınar. Bundan başqa, bir çox alqoritmlərdə təkrarların sayı qabaqcadan məlum olmur və yalnız proqramın gedişi zamanı aydınlaşır. Bu problemi aradan qaldırmaq üçün xüsusi alqoritmik strukturdan – *dövr*, yaxud *təkrardan* istifadə olunur.

Dövr yaratmaq üçün konstruksiyalar bütün proqramlaşdırma dillərində vardır. Dövr üç əsas hissədən ibarət olur:

1. *İnisiallaşdırmada* dövr yerinə yetirilməyə hazırlanır.
2. *Dövrün gövdəsi* təkrar-təkrar icra olunan operatorlar qrupudur.
3. *Sonluq şərti* dövrün gövdəsinin icrasından qabaq yoxlanır və dövrün sona çatmasını yoxlamaq üçün ondan istifadə edilir.

Sayğaclı dövr. Dövrün gövdəsinin neçə dəfə təkrarlanacağı qabaqcadan məlum olduqda dövr operatorunu yazmaq daha asan olur. Bu halda *sayğaclı dövrdən* istifadə olunur.

```
for <dövr dəyişəni> := <dövrün aşağı sərhədi>
    to <dövrün yuxarı sərhədi> do
    <dövrün gövdəsi>
```

Sayğac, yaxud *dövr dəyişəni* xidməti dəyişəndir və o, dövrün yerinə yetirilməsi zamanı avtomatik dəyişilir. Dövrdə birinci operator (ona elə *dövr operatoru* da deyirlər) *dövrün sərhədlərini* göstərir.

Sayğaclı dövrün blok-sxemi

Dövr aşağıdakı qaydada yerinə yetirilir:

1. Dövrün sərhədləri ifadə şəklində verilmişsə, öncə həmin ifadələr hesablanır.
2. Dövr dəyişəninə dövrün aşağı sərhədinin qiyməti mənimsədilir.
3. Dövr dəyişəni dövrün yuxarı sərhədi ilə müqayisə olunur.

4. Dövr dəyişəninin qiyməti yuxarı sərhəddən böyükdürsə, dövrün yerinə yetirilməsi kəsilir.
5. Dövrün gövdəsi icra olunur.
6. Dövr dəyişəninin cari qiymətinə 1 əlavə olunur.
7. Dövrün icrası 3 bəndindən davam etdirilir.

Tutaq ki, ilk on natural ədədin cəmini tapmaq lazımdır. Bunun üçün proqram fraqmentini aşağıdakı kimi yazmaq olar:

```
S := 0;
for I := 1 to 10 do
    S := S + I;
```

Dövrdən kənarda olan birinci operator `S` dəyişəninə başlanğıc qiyməti mənimsətmək üçündür. Dövr operatorunun icrasından öncə çox zaman hazırlıq işləri aparılmalıdır.

Sayğac rolunu `I` dəyişəni oynayır. Dövrün sərhədləri olan 1 və 10 ədədləri dövr operatorunda konstant kimi verilib. Əgər dövrün aşağı sərhədi onun yuxarı sərhədindən böyükdürsə, onda `to` açar sözünün əvəzinə `downto` açar sözündən istifadə olunur. Bu halda hər dəfə dövr icra olunduqda dövr dəyişəninin qiyməti 1 vahid azalır.

> **Sayğaclı dövrün özəllikləri**
>
> Sayğaclı dövr bir neçə özəlliyə malikdir. Dövrün sərhədləri ifadələr şəklində verilmişsə, bu ifadələr *dövrün inisiallaşdırılması* anında hesablanır. Həmin ifadələrə daxil olan dəyişənlərin qiyməti dövrün içərisində dəyişsə belə, bu dövrün gövdəsinin neçə dəfə yerinə yetirilməsinə təsir etmir.
> Dövrün sona çatması şərti dövrün gövdəsi birinci dəfə yerinə yetirilənədək yoxlanılır. Dövrün sərhədləri elə qiymətlər ola bilər ki, dövrün gövdəsi heç bir dəfə də yerinə yetirilməsin.

Şərtli dövr. Şərtli dövrlər dövrün daha ümumi yazılış formasıdır. Belə dövrlərdən adətən dövrün təkrarlanmalarının sayı qabaqcadan məlum olmadıqda istifadə edilir.

Şərtli dövrləri iki növə ayırırlar:

1) Ön şərtli dövrlər,
2) Son şərtli dövrlər.

Ön şərtli dövr belə işləyir:

1. Dövrün təkrarlanma şərti yoxlanılır.
2. Dövrün təkrarlanma şərti ödənmirsə, onun icrası dayandırılır.
3. Şərt ödənmirsə, dövrün gövdəsi yerinə yetirilir.
4. Dövrün icrası 1 bəndindən davam etdirilir.

Ön şərtli dövrün ümumi yazılış forması belədir:

```
while  <şərt> do
       <dövrün gövdəsi>;
```

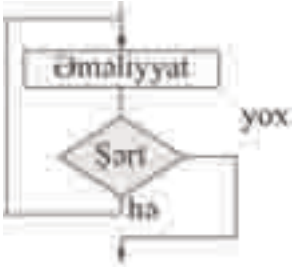
> Ön şərtli dövrün gövdəsi heç bir dəfə də icra olunmaya bilər.

Ön şərtli dövrdən istifadə etməklə ilk yüz natural ədədin kvadratları cəmini hesablayan proqram fraqmentini belə yazmaq olar:

```
S := 0;
I := 1;
while I <= 100 do
  begin
    S := S + I * I;
    I := I + 1;
  end;
```

Son şərtli dövr belə işləyir:

1. Dövrün gövdəsi yerinə yetirilir.
2. Dövrün təkrarlanma şərti yoxlanır.
3. Dövrün təkrarlanma şərti ödənmirsə, onun icrası dayandırılır.
4. Şərt ödənirsə, dövrün icrası 1 bəndindən davam etdirilir.



Son şərtli dövrün blok-sxemi

> Müxtəlif proqramlaşdırma dillərində dövrün sona çatdırılması üçün şərtin ödənilməsi, yaxud ödənilməməsi tələb oluna bilər.

Son şərtli dövr belə yazılır:

```
repeat
     <dövrün gövdəsi>
until <şərt>;
```

Son şərtli dövrün gövdəsi ən azı bir dəfə yerinə yetirilir. Pascal dilində son şərtli dövrün yerinə yetirilməsi şərt ödənildiyi zaman dayandırılır. Proqramçı şərtin ifadəsinin qiymətini müəyyənləşdirən dəyişənləri nəzarətdə saxlamalıdır ki, onlar dövrün gövdəsində dəyişilsin. Əgər dövrün şərti heç dəyişilməzsə, dövr sonsuz davam edə bilər.

İlk yüz natural ədədin kvadratları cəmini hesablayan proqram fraqmentini repeat operatorunun köməyilə belə yazmaq olar:

```
S := 0;
I := 1;
repeat
  S := S + I * I;
  I:= I + 1;
until I > 100;
```

Dövrdən çıxış. Bəzən proqramlarda çox mürəkkəb dövrlərdən istifadə olunur. Belə dövrün bir neçə sonaçatma şərti ola bilər. Bu dövrlərin yaradılmasını sadələşdirmək üçün müasir proqramlaşdırma dillərində *dövrdən çıxma* operatoru nəzərdə tutulub. Dövrdən çıxma operatoru adətən dövrün içərisindəki şərt operatorunun daxilində istifadə olunur. Bu operator yerinə yetirildikdə idarəetmə dərhal dövrdən sonrakı ilk operatora ötürülür. Bir dövrün daxilində bir neçə dövrdən çıxma operatoru yerləşə bilər. Dövrdən çıxma operatoru aşağıdakı kimi yazılır:

```
break;
```

Aşağıdakı misalda klaviaturadan daxil edilən ədədlərin cəmi hesablanır. Daxil edilən ədəd mənfi olduqda, dövr kəsilir.

```
S := 0;
while True do
  begin
    Read(I);
    if I < 0 then break;
    S := S + I;
  end;
```

Bu misalda “sonsuz” dövrdən istifadə olunub, çünki dövrün şərti həmişə doğrudur. Lakin dövrün daxilində çıxış operatoru olduğundan proqram “dövrə düşmür”.

1. **Dövr hansı hissələrdən ibarətdir?**
2. **Sayğaclı, ön şərtli və son şərtli dövrlər Pascal dilində hansı operatorla ifadə olunur?**
3. **Sayğaclı dövrün yerinə yetirilməsini izah edin.**
4. **Aşağıdakı operatorlar ardıcıllığının icrasından sonra `k` dəyişəninin qiymətini müəyyənləşdirin.**

```
var i, k: Integer;
...
k := 0;
for i := 1 to 100 do
    if i mod 2 = 0 then
    k := k + 1;
```

5. **`for` operatorundan istifadə etməklə ilk yüz natural ədədin kvadratları cəmini hesablayan proqram yazın.**
6. **Avtobus biletlərinin nömrələri altırəqəmlidir: 000000-dan 999999-dək. Əgər 1-ci, 3-cü və 5-ci rəqəmlərin cəmi 2-ci, 4-cü və 6-cı rəqəmlərin cəminə bərabərdirsə, bilet "uğurlu" hesab olunur. Bütün "uğurlu" biletləri tapan və çap edən proqram yazın.**
7. **Aşağıdakı operator nəyi yerinə yetirir?**

```
for Ch := 'A' to 'Z' do
    Write (Ch);
```

1.10. MASSİVLƏR

Proqramlarda tez-tez birtipli kəmiyyətlərdən istifadə olunur. Birtipli verilənlərin nömrələnmiş ardıcıllığına *massiv* deyilir. Massivin bir adı olur və həmin ad massivin bütün elementlərinə aid edilir. Massivin konkret elementini onun nömrəsinə görə seçmək olar. Həmin nömrəyə *indeks* deyilir.

Elementlərinin sayı aşkar göstərilmiş massivə *statik massiv* deyilir. Bəzi proqramlaşdırma dillərində *dinamik massivlərdən* istifadə olunur. Belə massivin ölçüsü proqramın icra müddətində dəyişilə bilər. Massiv elan olunan kimi kompüterin yaddaşında onun bütün elementləri üçün yer ayrılır. Ona görə də dinamik massivlər kompüterin yaddaşından daha səmərəli istifadə edir.

Sadə dəyişənlərdən fərqli olaraq, massivlər bütün proqramlaşdırma dillərində qabaqcadan təsvir (elan) olunmalıdır. Massivin təsvirində massivdə olan elementlərin sayı, indekslərin mümkün diapazonu və hər elementin tipi göstərilməlidir.

```
var a: array [1..10] of Integer;
```

array açar sözündən sonra kvadrat mötərizələrdə massiv indekslərinin iki nöqtə (..) ilə ayrılmış diapazonu göstərilir. Sonra of açar sözü və massiv elementlərinin tipi gəlir. Verilmiş misalda massiv 10 tam ədəddən ibarətdir.

Massivin elementinə müraciət. Proqramda massivin elementinə müraciət etmək üçün onun adını və indeksini göstərmək lazımdır. Məsələn, aşağıdakı operator massivin indeksi 7 olan elementinə indeksi 6 olan elementinin qiymətini mənimsədir.

```
a[ 7]  := a[ 6] ;
```

Proqramlarda çox vaxt massiv bir tam kimi emal olunur. Eyni bir əməli ardıcıl olaraq massivin bütün elementlərinə tətbiq etmək üçün *sayğaclı dövrdən* istifadə etmək olar. Bu halda massivin elementinə müraciət edəndə dövr dəyişənindən indeks kimi istifadə olunur. Məsələn, tutaq ki, b massivi 10 elementdən ibarətdir. Aşağıdakı dövr massivin elementlərinin qiymətini sıfra bərabər edir.

```
for  i := 1 to 10 do
     b[ i]  := 0;
```

Belə bir məsələyə baxaq. Tutaq ki, n elementdən ibarət tam ədədlər massivi verilib və massivin ən kiçik elementini tapmaq tələb olunur. Proqramda ən kiçik elementi min, həmin elementin indeksini isə imin dəyişəni ilə işarə etsək, verilən məsələnin həlli üçün proqram fraqmentini aşağıdakı kimi yazmaq olar:

```
imin := 1;
min  := a[ imin] ;
for i := 2 to n do
     if a[ i] < min then begin
        min  := a[ i] ;
        imin := i;
     end;
WriteLn  (min, ' ən kiçik element, ',
          imin, ' həmin elementin massivdə
          indeksidir.' );
```

İkiölçülü massivlər. Birindeksli massivə *birölçülü massiv* deyilir. Bir çox məsələlər üçün belə massivlər yetərli olmur. Proqramlaşdırma dillərinin hamısında bir neçə indeksli *çoxölçülü massivlər* yaratmaq imkanı vardır. Məsələn, *ikiölçülü massivə* bir indeksi sütunun nömrəsini, o biri indeksi isə sətrin nömrəsini göstərən cədvəl kimi baxmaq olar.

İkiölçülü massivi elan etmək üçün onun hər iki indeksinin dəyişmə diapazonunu göstərmək lazımdır. Bu diapazonlar vergüllə ayrılır. Məsələn, şahmat taxtasını təsvir edən tam ədədlər massivini belə elan etmək olar.

```
var t: array [1..8, 1..8] of integer;
```

Pascal dilində ikiölçülü massivə "massivlər massivi" kimi də baxmaq olar. Yuxarıdakı elanetmə tamamilə belə yazılışla ekvivalentdir:

```
var t: array [1..8] of array [1..8] of integer;
```

İkiölçülü massivin elementinə müraciət etmək üçün vergüllə ayırmaqla hər iki indeksi göstərmək lazımdır. Pascal dilində hər iki indeksi ayrıca da göstərmək olar.

Məsələn, şahmat taxtasının e4 xanasına uyğun olan massiv elementinə belə müraciət etmək olar:

```
t[5,4],
```

yaxud

```
t[5][4].
```

İç-içə dövrlər. İkiölçülü massivi emal edərkən hər iki indeksin qiymətini vahid dövr sayğacı əsasında hesablamaqla bir dövrlə də keçinmək olar. Ancaq bu halda proqramı başa düşmək çətinləşir.

Bir alqoritmi ikiölçülü massivin elementlərinə tətbiq etmək üçün adətən iki dövrdən istifadə edilir. Hər dövrün sayğacı uyğun indeks üzrə bütün mümkün qiymətləri alır. Bu zaman bütün massivi əhatə etmək üçün dövrlərdən *biri digərinin içərisində* yerləşməlidir. Belə dövrlərə *iç-içə dövrlər* deyilir. Məsələn, şahmat taxtasının bütün elementlərinə sıfır qiyməti mənimsətmək üçün dövrü aşağıdakı kimi qurmaq olar:

```
for  i := 1 to 8 do
     for j := 1 to 8 do
          t[ i,j]  := 0;
```

Daxili dövr yerinə yetirilərkən xarici dövr sayğacının qiyməti sabit qalır. Sonra o, bir vahid dəyişir və daxili dövr tamamilə yenidən təkrarlanır. Proqramlaşdırmada iç-içə dövrlərdən istifadə təkcə massivlərlə işləmək üçün deyil, başqa işlərdə də çox əlverişli olur.

Tutaq ki, n sətri və m sütunu olan ikiölçülü a massivi verilib. Həmin massivdə, heç olmasa, bir mənfi elementin olub-olmamasını müəyyənləşdirən proqram fraqmentini son şərtli dövrdən istifadə etməklə (həmin dövrün iş prinsipini yada salın!) aşağıdakı kimi yazmaq olar:

```
i := 0;
repeat
    j := 0;
    i := i + 1
    repeat
        j := j + 1;
    until (j = m) or (a[ i,j] < 0);
until (i = n) or (a[ i,j] < 0);
if a[ i,j] < 0 then
    WriteLn('Hə, mənfi element var!')
else
    WriteLn('Mənfi element yoxdur!');
```

1. Massiv nədir və o, Pascal dilində necə elan olunur?
2. Birölçülü a massivində indeksləri k-dan m-dək olan elementlərin maksimumunu hesablayan proqram yazın. Tapılmış qiyməti j dəyişəninə mənimsədin.
3. Verilmiş tam ədədlər massivi 10 elementdən ibarətdir. Onların içərisində iki ədəd bərabərdir. Həmin ədədlərin indeksini müəyyənləşdirən proqram yazın. Tapılmış qiymətləri i və j dəyişənlərinə mənimsədin.
4. Aşağıdakı proqram fraqmentinin icrasından sonra L dəyişəninin qiyməti nə olacaq? Düzgün cavabı seçin.

```
var a: array [1..4, 1..4] of integer;
      L, f, g, v: integer;
...
L := 0;
for g := 1 to 4 do
begin
  v := 0;
  for f := 1 to 4 do
        if a[f,g] < 0 then v:= v + 1;
  L := L + v;
end;
```

a) a massivində müsbət elementlərin sayı

b) a massivində müsbət olmayan elementlərin sayı

c) a massivində mənfi elementlərin sayı

d) a massivinin birinci sətrindəki mənfi elementlərin sayı

e) a massivinin sütunundakı mənfi elementlərin maksimal sayı

1.11. SƏTİRLƏRLƏ İŞ

Proqramlaşdırma dillərinin hamısında *simvollar ardıcıllığı*, başqa sözlə, *sətirlərlə iş* nəzərdə tutulub.

Sətir konstantla, yaxud dəyişənin qiymətilə verilə bilər. Pascal dilində *sətir konstantı* apostrofa alınmış simvollar ardıcıllığı kimi verilir:

```
'Pascal'
'1234'
'Araz çayı'
```

'' ardıcıllığı uzunluğu sıfır olan xüsusi sətir konstantıdır. Belə sətrə *boş sətir* deyilir.

Sətrin maksimal uzunluğu konkret proqramlaşdırma dilindən, yaxud verilmiş dilin konkret translyatorundan asılıdır. Müasir proqramlaşdırma sistemlərində sətirlərin uzunluğuna praktik olaraq həddqoyulmur.

Turbo Pascal dilində sətir dəyişənləri string tipinə malikdir. Məsələn, proqramda

```
var a: string;
```

təsviri a-nın sətir dəyişəni olduğunu bildirir.

Sətirlərlə əməliyyatlar ədədlərlə yerinə yetirilən əməliyyatlardan fərqlənir. Sətirlərin toplanması və ya çıxılmasının, vurulması və ya bölünməsinin elə bir mənası yoxdur. Sətirlər üzərində aparılan əsas əməliyyat *sətirlərin birləşdirilməsi,* yaxud *konkatenasiyasıdır*. Bu əməliyyat nəticəsində ikinci sətir birinci sətrin sonuna birləşdirilir. Konkatenasiya əməli **+** (plyus) simvolu ilə işarə olunur.

```
var a, man: string;
...
man := 'On';
a := man + ' manat';
```

Əməl işarələri eyni olsa da, (**+**) birləşdirmə əməli toplama əməlindən fərqlənir.

Sətirləri birləşdirmək üçün Concat funksiyasından da istifadə olunur. Məsələn,

```
a := Concat(man, ' manat');
```

Əslində sətirlər üzərində əməllər ədədlər üzərindəki əməllərdən daha çoxdur. Ancaq burada hər hansı əməl işarəsindən istifadə o qədər də əlverişli deyil və qeyri-adidir. Ona görə də qalan sətir əməlləri, adətən, *standart funksiyalar* vasitəsilə yerinə yetirilir. Proqramlaşdırma dillərinin əksəriyyətində aşağıdakı əməliyyatlar mümkündür:

- sətrin uzunluğunun müəyyənləşdirilməsi;
- alt sətrin seçilməsi;
- simvolların artırılması, yaxud uzaqlaşdırılması;
- sətirdə simvolun axtarılması;
- sətirdəki simvolların (hərflərin) registrinin dəyişdirilməsi.

Bu göstərilən və bəzi başqa əməliyyatları yerinə yetirmək üçün Turbo Pascal dilində nəzərdə tutulmuş funksiyalarla tanış olaq.

- **Length** (St). Bu funksiya St sətrinin uzunluğunu hesablayır. Sətrin uzunluğu dedikdə həmin sətirdə olan simvolların sayı nəzərdə tutulur. Boş sətrin uzunluğu 0-a bərabərdir. Hər hansı S sətrinin uzunluğunu hesablayıb LenS dəyişəninə mənimsətmək əməlini

```
LenS := Length(S);
```

operatoru kimi yazmaq lazımdır.

```
S := 'Hello';
L := Length(S);
WriteLn(L);          { Çıxışa 5 qiyməti veriləcək }

S := 'Hello Students';
WriteLn(Length(S)); { Çıxışa 14 qiyməti veriləcək }

S := '  ';           { Boş sətir }
WriteLn(Length(S)); { Çıxışa 0 qiyməti veriləcək }
```

- **Copy** (St,Index,Count). Bəzən sətrin müəyyən hissəsini ayırıb götürmək lazım gəlir. Məsələn, əgər Date dəyişəninə mənimsədilmiş '17 may 2008' sətrindəki üç komponenti (gün, ay, il) ayrı-ayrı emal etmək tələb olunursa, bunu Copy funksiyası vasitəsilə etmək mümkündr.
Tutaq ki, proqramda tarix sətri GG AAA İİİİ formatındadır. (Burada GG – ayın gününü (1-2-ci simvollar), AAA – ayın adının qısaldılması (4-6-cı sim-

vollar) və İİİİ – ili (8-11-ci simvollar) göstərir). Əgər Date, Month, Day və Year sətir dəyişənləridirsə, aşağıdakı operator Day dəyişəninə Date sətrinin birinci simvolundan başlayaraq iki ardıcıl simvol (‘17’) mənimsədəcək.

```
Day := Copy (Date, 1, 2);
```

Eyni qayda ilə *ayı* göstərən alt sətir (`'may'`) `Month` dəyişəninə, *ili* bildirən alt sətir (`'2008'`) isə `Year` dəyişəninə mənimsədilir.

```
Month := Copy (Date, 4, 3);
Year  := Copy (Date, 8, 4);
```

- **Insert** (Subst,St,Index). Bu prosedur St sətrinə Index mövqeyindən başlayaraq Subst sətrini artırır. Məsələn,

```
Insert ('p', 'Alay', 3)
```

prosedurunun icrasından sonra

```
'Alay'
```

sətri

```
'Alpay'
```

sətrinə çevriləcək.

- **Delete** (St,Index,Count). Bu prosedur St sətrinin Index mövqeyindən başlayaraq Count sayda simvolu silir. Məsələn, Delete (‘Alqoritm’,1,4) proseduru nun tətbiqi nəticəsində ‘Alqoritm’ sətri ‘ritm’ sətrinə çevriləcək.
- **Pos** (Subst,St). Bu funksiya St sətrində Subst alt sətrini axtarır. Əgər axtarış uğurlu olarsa, funksiya nəticədə alt sətrin aşkarlandığı mövqenin nömrəsini verir. Əks halda, yəni axtarış uğursuz olarsa, funksiyanın nəticəsi 0 (sıfır) olacaq. Nəzərə almaq lazımdır ki, bu funksiya alt sətri birinci aşkarladığı mövqenin nömrəsini çıxışa verir. Başqa sözlə, əgər sətirdə axtarılan alt sətirdə bir neçə dəfə rast gəlinirsə, funksiyanın nəticəsi olaraq alt sətrin ilk aşkarlandığı yerin nömrəsi götürüləcək.

```
S := '1 nömrəli orta məktəb';
S1 := 'orta';
J := Pos(S1, S);
WriteLn(J); { Ekrana 11 ədədi çıxarılacaq }

S1 := 'əsas';
J := Pos(S1, S);
WriteLn(J); { Ekrana 0 ədədi çıxarılacaq }
```

- **Val** (St,X,Code). Turbo Pascalda sətri ədədə çevirən standart funksiya – Val funksiyası vardır. Bu zaman, təbii ki, çevrilən sətir ədədi sətir olmalıdır (yəni, ədəddən ibarət olmalıdır, məsələn, '17.5' , yaxud '1234'). Funksiyanın parametrlərindən St ilkin sətrin özü, X alınan ədədin mənimsədildiyi dəyişən, Code isə çevrilmənin uğurla keçib-keçmədiyini bildirən parametrdir. Belə ki, çevrilmə uğurlu olarsa, Code parametrinin qiyməti 0 olacaq, əks halda həmin parametrə sətirdə səhvin baş verdiyi yerin nömrəsi yazılacaq. Aşağıdakı proqram fraqmenti NumStr dəyişəninə tam ədədi qiymət verir.

```
repeat
    Write ('Tam ədədi daxil edin: ');
    ReadLn (NumStr);
    Val (NumStr, IntNum, Error)
until Error = 0;
```

  Əgər daxil edilən ədəd sətirdirsə, Val proseduru bu sətri IntNum tam ədədinə mənimsədir. Əgər oxunan sətirdə rəqəmdən başqa ayrı simvollar da olarsa, Error dəyişəninin qiyməti sıfırdan fərqli olacaq və beləliklə də dövr təkrarlanacaq.

- **Str** (X,St). Bu prosedur Val prosedurunun əksini edir, yəni ədədi sətrə çevirir. Str (123 :5, NumSt) operatoru NumSt dəyişəninə ' 123' sətrini yazacaq. Burada 5 sətirdəki simvolların sayını göstərir.

İndi göstərilən prosedurların bəzilərinin tətbiq olunduğu aşağıdakı misala baxaq. Burada istifadəçinin göstərdiyi sözə sətirdə neçə dəfə rast gəlindiyi hesablanır.

```
program P;
var
    s, s1: string;
    k, i: Integer;

begin
    Write ('İlkin sətri daxil edin: ');
    ReadLn (s);
    Write ('Axtarılan sözü daxil edin: ');
    ReadLn (s1);

    k := 0;

    while Pos(s1, s) > 0 do
    begin
        k := k + 1;
        Delete(s, Pos(s1, s), Length(s1));
    end;
    WriteLn (k);
end.
```

1. **Sətir nədir və onun üzərində hansı əməliyyatlar aparılır?**
2. **Tutaq ki, Temp1 := 'Abra' və Temp2 := 'kadabra'.**
   Aşağıdakı funksiya və prosedurların nəticəsini müəyyənləşdirin.

   a) `Magic := Concat(Temp1, Temp2)`
   b) `Length(Magic)`
   c) `HisMagic := Copy(Magic, 1, 8)`
   d) `Delete(HisMagic, 4, 3)`
   e) `Insert(Temp1, HisMagic, 3)`
   f) `Pos(Temp2, Magic)`
   g) `Pos(Temp1, Magic)`
   h) `Val('1.234', RealNum, Error)`
   i) `Str(1.234 :3:1, RealStr)`

3. ***Sətrin polindromu*** **onun özündən və tərs yazılışından ibarətdir. İstənilən sətrin polindromunu yaradan funksiya yazın. Yəni bu funksiyanın girişinə `'abc'` verdikdə, çıxışda `'abccba'` alınmalıdır.**
4. **Sətirdə olan saitlərin sayını hesablayan proqram yazın.**

1.12. ALTPROQRAMLAR. FUNKSİYALAR VƏ PROSEDURLAR

İnsan həcmi bir neçə yüz sətir olan alqoritmdən "baş çıxara" bilər. Proqram sətirlərinin sayı artdıqca işin ümumi məntiqi itir. Konkret operatorların yerinə yetirdiyi əməliyyatlar elementar olsa da, onların ümumi məqsədini başa düşmək çətinləşir. Proqramın strukturu və onun yerinə yetirilmə ardıcıllığı aydın olmur. Belə alqoritmi dəyişdirmək, yaxud düzəltmək çox çətin olur.

Bu problemi həll etmək üçün alqoritm sadə əməliyyatları yerinə yetirən ayrı-ayrı alqoritmlərə bölünür. Belə ayrıca alqoritmlərə *yardımçı alqoritmlər* deyilir. Proqramlaşdırma dillərində yardımçı alqoritm termininin yerinə *altproqram* terminindən istifadə olunur. Yardımçı alqoritmə (altproqrama) müraciət etmək üçün onu *çağırmaq* lazımdır.

Adətən orta ölçülü proqramları hər biri çox da çətin olmayan əməliyyatı yerinə yetirən kiçik altproqramlara bölürlər. Yekun alqoritm ayrı-ayrı operatorlardan deyil, hər birinin öz adı olan bitkin kod bloklarından ibarət olur. Bu halda altproqramlara proqramçıların təyin etdiyi yeni operatorlar kimi baxmaq olar.

Standart altproqramlar. Bir çox yardımçı alqoritmlərdən tez-tez və müxtəlif məsələlərdə istifadə edilir. Məsələn, tez-tez tipik riyazi funksiyaları hesablamaq, yaxud sətirlər üzərində standart əməliyyatlar aparmaq tələb olunur. Belə alqoritmləri hər proqramçı özü yazsa idi, bu çox böyük vaxt itkisinə səbəb olardı. Bu problem standart altproqramlar tətbiq etməklə aradan qaldırılır.

Standart altproqramlar adətən proqramlaşdırma dilində deyil, proqramlaşdırma sistemində (mühitində) təyin edilir. Onlar translyatora əlavə edilən altproqramlar kitabxanasına daxil olur. Standart altproqramların geniş kitabxanaları işi əhəmiyyətli dərəcədə yüngülləşdirir.

Yardımçı alqoritmlərin tipləri. Altproqramlar adətən, iki kateqoriyaya bölürlər: *prosedurlar* və *funksiyalar*.

> Prosedur, sadəcə hər hansı operatorlar ardıcıllığını yerinə yetirir.
> Funksiya isə müəyyən qiyməti hesablayır və həmin qiyməti çağıran proqrama (yaxud altproqrama) ötürür (qaytarır).

Bəzi proqramlaşdırma dillərində (məsələn, C-də) altproqramları prosedur və funksiyalara bölmürlər. Onların hamısına funksiya kimi baxılır. Belə dillərdə prosedur heç bir qiymət qaytarmayan funksiyadır.

Altproqramın parametrləri. Altproqramın işinin bir mənası olması üçün o, onu çağıran proqramdan verilənlər almalıdır. Verilənlər altproqrama *parametrlər* şəklində ötürülür. Hər bir altproqram parametr kimi konkret tipli müəyyən verilənlər yığınını almağı gözləyir. Parametrlərə ehtiyacı olmayan altproqramlar da mümkündür.

Parametrlər **Formal parametrlər** **Faktik parametrlər**

Altproqram yaradılarkən onun parametrlərinə ötürülən qiymətlər hələ məlum olmur. Təsvir zamanı altproqramın başlığında *formal parametrlər* göstərilir. Formal parametrlər ötürülən verilənlərin tipini müəyyən edən ixtiyari identifikatorlardır. Onlar yalnız altproqramın yerinə yetirdiyi əməliyyatları təsvir etmək üçün lazımdır.

Altproqram çağırılarkən ona ötürülən *faktik parametrlər* göstərilir. Altproqramın operatorlarının yerinə yetirilməsi zamanı formal parametrləri faktik qiymətlər təmsil edir.

Altproqramın çağırılması. Altproqramı çağıran operatorun növü altproqramın tipindən və konkret proqramlaşdırma dilinin sintaksisindən asılıdır. Altproqramı çağırmaq üçün onun adını göstərmək lazımdır. Ondan sonra mötərizədə faktik parametrlərin siyahısı gəlir. Faktik parametrlərin tipi və onların sayı altproqramda formal parametrlərin təsvirinə uyğun gəlməlidir. Faktik parametrlər təkcə dəyişənlər deyil, həm də konstantlar, yaxud ifadələr ola bilər.

Funksiyanı proqramın istənilən yerindən çağırmaq olar. Aşağıdakı misalda z dəyişəninə katetləri x və y olan düzbucaqlı üçbucağın hipotenuzunun uzunluğu mənimsədilir. Hesablama üçün standart funksiyaya müraciət olunub.

```
z := sqrt(x*x + y*y);
```

Prosedurun çağırılması adətən ayrıca operator kimi göstərilir. Pascal dilində bunun üçün heç bir açar söz tələb olunmur. Məsələn, parametr kimi iki tam ədədi qəbul edən P proseduru belə çağırıla bilər:

```
P(1, 2);
```

Yardımçı alqoritmlərin proqramlaşdırılması. Yardımçı alqoritmlərin təsviri proqramın ilkin mətninə daxil edilir. Proqramlaşdırma dillərinin əksəriyyətində altproqramın ilk çağırılmadan öncə təsvir olunması tələb olunur.

Altproqramın təsviri *başlıqdan, gövdədən* və *sonluqdan* ibarətdir. Başlıq altproqramın adından və formal parametrlərin təsvirindən ibarətdir. Funksiya üçün həm də qaytarılan qiymətin tipi göstərilir:

```
function Square(x: Integer) : Integer;
begin
      Square := x*x;
end;
```

Pascal dilində xüsusi sonluq operatoru olmur. Funksiyanın gövdəsi begin və end operatorlarının arasında yerləşdirilir.

> Funksiyanın qaytardığı qiymət onun adıyla üst-üstə düşən dəyişənə mənimsədilməlidir.

Funksiyanın gövdəsinin daxilində bu dəyişəndən mənimsətmə operatorunun yalnız sol tərəfində istifadə edilə bilər.

N ədədinin faktorialı dedikdə 1-dən n-dək ədədlərin hasili başa düşülür (n! = 1·2·3· ... ·n). Aşağıdakı funksiya ədədin faktorialını hesablayır:

```
function Fact(n : Integer) : Integer;
var
   i : Integer;
   res : Integer;
begin
   res := 1;
   for i := 1 to n do res := res * i;
   Fact := res;
end;
```

Prosedurun təsvirində isə qaytarılan qiyməti hesablamağa lüzum yoxdur. Aşağıda iki tam ədədin cəmini çıxışa verən prosedur təsvir olunub:

```
procedure PrintSum(x, y: Integer);
begin
      WriteLn(x+y);
end;
```

Adətən hesab olunur ki, altproqramın təsviri onun ilk istifadəsindən öndə yerləşdirilməlidir (çünki translyatora belə əlverişlidir). Ancaq bu müəyyən səbəblərdən proqramçı üçün rahat olmaya bilər. Bəzi proqramlaşdırma dilləri altproqramın təsvirini onun birinci istifadəsindən sonra yerləşdirməyə imkan verir, ancaq bu halda altproqramın istifadədən öncə *sadələşdirilmiş* halda elan edilməsi tələb olunur.

Pascal dilində bunun üçün prosedurun başlığı təkrarlanır. Ondan sonra **begin** açar sözünün olmaması bunun *təsvir* deyil, yalnız *elan* olduğunu bildirir.

```
procedure PrintSum(x, y: integer);
```

Elanın qabaqcadan olması altproqramın parametrləri haqqında translyatora məlumat verir. Bu isə altproqramın çağırılmasını düzgün emal etməyə imkan verir.

Turbo Pascalın standart riyazi funksiyaları

Funksiya	Təyinatı	Arqument	Nəticə
`Abs(X)`	X-in mütləq qiyməti	`Real`, yaxud `Integer`	`Real`, yaxud `Integer`
`Cos(X)`	X bucağının kosinusu	`Real`, yaxud `Integer` (radianla)	`Real`
`Exp(X)`	e^X, e = 2.71828...	`Real`, yaxud `Integer`	`Real`
`Ln(X)`	X-in natural loqarifmi, X > 0	`Real`, yaxud `Integer`	`Real`
`Round(X)`	X ədədinə ən yaxın tam ədəd	`Real`	`Integer`
`Sin(X)`	X bucağının sinusu	`Real`, yaxud `Integer` (radianla)	`Real`
`Sqr(X)`	X-in kvadratı	`Real`, yaxud `Integer`	`Real`, yaxud `Integer`
`Sqrt(X)`	X-in müsbət kvadrat kökü, X > 0.0	`Real`, yaxud `Integer`	`Real`
`Trunc(X)`	X-in tam hissəsi	`Real`	`Integer`

Aşağıda verilmiş Artma proqramı a, b, c dəyişənlərinin qiymətlərini elə dəyişir ki, onlar artma sırasıyla düzülür (a ≤ b ≤ c).

```
program Artma;
var
    a, b, c : Integer;
procedure Swap(var x, y : Integer);
var
    t : Integer;
begin
    t := x; x := y; y := t;
end;
begin;
    WriteLn('Üç ədəd daxil edin ');
    ReadLn(a, b, c);
    if a > b then Swap(a, b);
    if b > c then Swap(b, c);
    if a > b then Swap(a, b);
    WriteLn(a:5, b:5, c:5);
    ReadLn;
end.
```

Swap proseduruna diqqət edin. Bu prosedur iki dəyişənin qiymətinin yerini dəyişir. Belə yerdəyişmədən proqramlaşdırmada tez-tez istifadə olunur.

Altproqramın yerinə yetirilməsinin dayandırılması və çağıran proqrama qayıdılması, idarəetmə altproqramın son operatoruna çatanda baş verir. Bu isə həmişə əlverişli olmur. Buna görə də altproqramın yerinə yetirilməsinin yarıda kəsilməsi və idarəetmənin dərhal çağıran proqrama qaytarılması imkanı vardır. Pascal dilində bu məqsədlə exit operatorundan istifadə olunur.

1. Altproqramlardan istifadənin hansı üstünlükləri var?
2. Altproqramların hansı növləri var?
3. Funksiya prosedurdan nə ilə fərqlənir?
4. Klaviaturadan daxil edilən n ədədinin 3-ə bölünüb-bölünmədiyini yoxlayan funksiya yazın.

1.13. FAYLLAR

İndiyədək tanış olduğunuz proqramların hamısı interaktiv proqramlar idi. *İnteraktiv proqramlar* bütün giriş verilənlərini klaviaturadan oxuyur, nəticələri isə ekrana çıxarır. Kiçikhəcmli verilənlərlə işləyərkən interaktiv giriş və çıxış daha yaxşı olur, ancaq bu cür yanaşma böyük həcmli informasiyalarla işləyərkən səmərəsizdir. Bu halda vəziyyətdən çıxmaq məqsədilə giriş və çıxış üçün fayllardan istifadə etmək olar. Pascal iki növ informasiya faylı ilə işləyir: *mətn* və *ikilik* fayllarla.

Mətn faylı diskdə bir ad altında saxlanılan ayrı-ayrı simvollar yığınıdır. Proqramın emal edəcəyi bütün ilkin verilənləri proqramın başladılmasından öncə mətn faylında saxlamaq olar. Bundan sonra proqramı elə dəyişmək lazımdır ki, o özünün istifadə etdiyi verilənləri klaviaturadan deyil, mətn faylından oxusun.

> Mətn faylı bir ad altında saxlanılan ayrı-ayrı simvollar yığınıdır.

Giriş və çıxış faylları. Proqramın giriş verilənlərindən ibarət fayla *giriş faylı* deyilir. Giriş faylı həm mətn faylı, həm də ikilik fayl ola bilər. Giriş faylından istifadənin üstünlüklərindən biri giriş verilənlərinin mətn redaktorunda hazırlana bilməsidir. Bu zaman verilənləri emal edilmək üçün proqrama verməmişdən öncə yoxlamaq və əgər səhvlər olarsa, onları düzəltmək imkanı vardır. İkinci üstünlük ondan ibarətdir ki, giriş verilənlərindən ibarət fayl proqram tərəfindən təkrar-təkrar oxuna bilər. Bu imkan proqramın sazlanmasını asanlaşdırır, çünki proqramınız başladıldıqda ilkin verilənləri hər dəfə giriş faylından oxuya bilər. İnteraktiv proqramda isə siz hər dəfə verilənləri klaviaturadan daxil etməli olacaqdınız.

> Giriş faylı proqramın giriş verilənlərindən ibarət fayldır.

Eyni zamanda proqramı elə dəyişdirmək olar ki, o, hesablanmış nəticələri ekrana çıxarmasın, mətn faylına yazsın. Diskdə olan həmin faylı sonradan çapa vermək, yaxud hətta başqa proqram üçün giriş faylı kimi, istifadə etmək olar. Proqramın işinin nəticələrinin toplandığı fayla *çıxış faylı* deyilir.

> Çıxış faylı proqramın işinin nəticələrinin toplandığı fayldır.

Mətn faylları ilə işləmək üçün beş əməliyyatı yerinə yetirmək lazımdır:

1. **Faylın təsvir edilməsi.** Proqramda olan hər bir obyekt kimi, mətn faylı da istifadədən öncə təsvir olunmalıdır. Fayl var bölümündə təsvir edilir və bunun üçün text açar sözündən istifadə olunur. Məsələn,

```
var f1, f2: text;
```

yazılışı f1 və f2-nin mətn faylı olduğunu bildirir. Bu halda f1 və f2-yə *fayl dəyişənləri* də deyirlər.

2. **Təsvir olunmuş faylın xarici daşıyıcıda olan konkret faylla əlaqələndirilməsi.** Diskdə saxlanılan fayla müraciət etmək üçün proqrama həmin faylın adını və onun hansı qovluqda (kataloqda) saxlandığını bildirmək lazımdır. Başqa sözlə, proqram faylın tam adını bilməlidir. Bunun üçün Pascal dilində **Assign** proseduru nəzərdə tutulub. Məsələn, C diskində ALTAY qovluğunda saxlanılan in.txt faylını f1 fayl dəyişəninə bağlamaq üçün proqramda

```
Assign(f1, 'c:\ALTAY\in.txt');
```

proseduru göstərilməlidir. Əgər yalnız faylın adı göstərilibsə, onda Turbo Pascal nəzərdə tutur ki, fayl proqramın yerləşdiyi qovluqdadır. Məsələn,

```
Assign(f2, 'out.txt');
```

proseduru proqram qovluğunda yerləşən out.txt faylını f2 fayl dəyişəninə bağlayır. **Assign** proseduru faylın gerçək adıyla işləyən yeganə funksiyadır. Qalan funksiyaların hamısı fayl dəyişənindən istifadə edir. Bu səbəbdən proqramçılar f1-ə faylın *daxili adı*, c:\ALTAY\ in.txt-yə isə onun *xarici adı* deyirlər.

3. **Faylın oxuması, yaxud yazılması üçün onun açılması.** Proqramın mətn faylı ilə manipulyasiya edə bilməsi üçün həmin fayl giriş, yaxud çıxışa hazırlanmalıdır, başqa sözlə, fayl açılmalıdır. Mətn faylı eyni zamanda həm giriş, həm də çıxış üçün açıla bilməz. Yəni verilənlər mətn faylından oxunursa, siz nəticələri həmin fayla yaza bilməzsiniz. Mətn faylını açmaq üçün Pascal dilində **Reset** və **Rewrite** prosedurları nəzərdə tutulub.

```
Reset(f1);
```

Bu prosedur f1 fayl dəyişəni ilə əlaqələndirilmiş faylı proqrama daxil edilmək üçün hazırlayır. Bu zaman cari mövqe göstəricisi faylın başlanğıcına keçir. *Cari mövqe göstəricisi* emal olunacaq növbəti simvolu göstərir. **Reset** proseduru faylın oxunmasına başlamazdan əvvəl çağırılmalıdır.

```
Rewrite(f2);
```

Bu prosedur f2 fayl dəyişəni ilə əlaqələndirilmiş faylı proqramdan çıxışa veriləcək verilənləri qəbul etmək üçün hazırlayır. Əgər diskdə göstərilən fayl olmazsa, həmin ad altında yeni boş fayl yaradılacaq. Diskdə göstərilən fayl artıq mövcuddursa, o, boşaldılacaq və cari mövqe göstəricisi faylın başlanğıcına keçəcək.

4. **Verilənlərin fayla yazılması, yaxud fayldan oxunması.** Verilənləri mətn faylından oxumaq üçün **ReadLn** prosedurundan istifadə etmək olar. Bu prosedur klaviaturadan daxil edilən verilənləri necə emal edirsə, mətn faylının hər bir sətri ilə də elə işləyir.

```
ReadLn(f1, n);
```

proseduru f1 giriş faylından verilənləri oxuyur və n dəyişəninə yazır. Faylda iki ardıcıl elementi birdən oxuyub, a və b dəyişənlərinə yazmaq üçün

```
Read(f1, a, b);
```

prosedurundan istifadə etmək olar.
Verilənləri ekrana çıxarmaq üçün **WriteLn** və **Write** prosedurları ilə artıq tanışsınız. Bu prosedurlardan verilənləri mətn faylına yazmaq üçün də istifadə edilir.

```
WriteLn(f2, n);
```

proseduru n dəyişəninin qiymətini f2 faylına yazır. Yeni verilənlər fayla növbəti sətirdən yazılacaq.

```
Write(f2, a, b);
```

proseduru f2 faylına a və b ədədlərini yazır və növbəti yazı həmin sətirdən davam olunacaq.

5. Faylın bağlanması. Giriş, yaxud çıxış faylı ilə işi bitirdikdən sonra diskdəki faylla proqram arasında əlaqəni kəsmək lazımdır. Başqa sözlə, açıq fayllar bağlanmalıdır. Bunun üçün Turbo Pascalda **Close** proseduru nəzərdə tutulub.

```
Close(f1);
Close(f2);
```

Nəzərə alın ki, **Close** (f1, f2); yazılışı düzgün deyil – *hər bir fayl ayrıca bağlanmalıdır!*

İndi fayllarla işləmək üçün iki misala baxaq. Birincidə iki ədəd fayla yazılır, ikinci misalda isə həmin ədədlər fayldan oxunur və ekrana çıxarılır.

```
program P1;
var
  f: text;                    {Fayl elan olunur}
  n: Integer;
begin
  Assign(f,'file1.txt');
                              {f faylı diskdə olan file1.txt
                              faylı ilə əlaqələndirilir}
  Rewrite(f);                 {f faylı yazılmaq üçün açılır}
  n := 7;
  WriteLn(f, n);              {n dəyişənin qiyməti fayla yazılır}
  WriteLn(f, 5);              {5 ədədi fayla yazılır}
  Close(f);                   {Fayl bağlanılır}
end.
```

```
program P2;
var
  f: text;                    {Fayl elan olunur}
  n: Integer;
begin
  Assign(f,'file1.txt');  {f faylı diskdə olan
          file1.txt faylı ilə əlaqələndirilir}
```

```
    Reset(f);              {f faylı oxunmaq üçün açılır}
    ReadLn(f, n);          {Birinci ədəd n dəyişəninə oxunur}

    WriteLn(n);            {n dəyişəninin qiyməti (7 ədədi) ekrana çıxarılır}

    ReadLn(f, n);          {İkinci ədəd n dəyişəninə oxunur}
    WriteLn(n);            {n dəyişəninin qiyməti (5 ədədi) ekrana çıxarılır}

    Close(f);              {Fayl bağlanılır}
end.
```

Sonda fayllarla işləmək üçün tez-tez istifadə olunan bir neçə funksiya ilə də tanış olaq.

Mətn faylına informasiyanın artırılması. Mətn fayllarının sonuna informasiya artırmaq imkanı da nəzərdə tutulub. Bunun üçün faylı **Rewrite** proseduru ilə deyil, **Append** proseduru ilə açmaq lazımdır.

```
Append(<Fayl dəyişəni>);
```

Bu prosedur çağırıldıqdan sonra fayl yazılış üçün açılır, ancaq **Rewrite** prosedurunda olduğu kimi, fayldakı mövcud informasiya silinmir. Cari mövqe göstəricisi faylın sonuna keçir və deməli, yazılan informasiya faylın sonuna əlavə olunur.

Misal olaraq yuxarıdakı P1 proqramı vasitəsilə yaradılan file1.txt faylına yeni yazı əlavə edək:

```
program P3;
var
    f: text;
begin
    Assign(f,'file1.txt');

    Append(f);
    WriteLn(f, 9);

    Close(f);
end.
```

Aşağıdakı cədvəldə fayllarla iş zamanı lazım olan daha bir neçə funksiya haqqında qısa məlumat verilib.

Funksiya	Təyinatı
Eof (<Fayl dəyişəni>)	f fayl dəyişəni ilə bağlı olan fayl üçün End-of-file (faylın sonu) vəziyyətini bildirir: cari mövqe göstərici faylın sonundadırsa, yaxud fayl boşdursa, True, qalan hallarda isə False qiymətini qaytarır.
Erase (<Fayl dəyişəni>)	f fayl dəyişəni ilə bağlı olan xarici faylı silir.
Rename (<Fayl dəyişəni>, <Faylın yeni adı>)	Faylın adı dəyişdirilir.
MkDir (<Qovluğun adı>)	Yeni qovluq yaradır.
RmDir (<Qovluğun adı>)	Qovluğu uzaqlaşdırır. Bu zaman uzaqlaşdırılan qovluğun içərisində altqovluqlar, yaxud fayllar olmamalıdır.

1. Verilənləri fayldan daxil etməyin hansı üstünlükləri var?
2. Turbo Pascalda faylların hansı növləri var?
3. Turbo Pascalda mətn faylları ilə işləmək metodikası necədir və bunun üçün hansı operatorlar nəzərdə tutulub?
4. Fayl dəyişəni nədir?
5. Faylın daxili və xarici adının mahiyyətini izah edin.

1.14. PRAKTİKUM

OPERATORLAR

1. Aşağıdakı komandaların icrasından sonra s kəmiyyəti hansı qiyməti alacaq?

a)
```
s := 7; s := 23;
```
b)
```
s := 1; s := s + 3;
```
c)
```
a := 2; b := 5; b := b - a; s := b + a;
```
d)
```
s := 0; k := 30; d := k - 5; k := 2*d;
s := k - 100;
```

2. Aşağıdakı operatorların icrasından sonra x və y dəyişənləri hansı qiyməti alacaq? Dəyişənlərin qiymətləri yerlərini dəyişdimi?

```
x := 2;
y := 9;
x := y;
y := x;
```

3. Aşağıdakı operatorların icrasından sonra a, b, c dəyişənləri hansı qiymətləri alacaq (a = 1, b = 2, c = 3) ?

```
a := b;
b := c;
c := a;
```

4. Aşağıdakı operatorların icrasından sonra ekrana nə çıxacaq?

```
a := 4;
Write(a);
Write('a');
```

5. Aşağıdakı proqramın icrasından sonra ekrana nə çıxacaq?

```
proqram Task5;
var
    a, b, c : Integer;
begin
    Write(1);
    Write(2, 3);
    WriteLn(4);
    Write(5);
    WriteLn(6, 7);
    WriteLn;
    Write(8);
    ReadLn;
end.
```

6. Klaviaturadan 1, 2, 3 ədədləri daxil edilərsə, aşağıdakı proqramın icrasından sonra ekrana nə çıxacaq?

```
proqram Task6;
var
    a, b, c : Integer;

begin
    WriteLn('Üç tam ədəd daxil edin');
    ReadLn(a, b, a);
    c = a + b;
    Write('a+b=', c);
    ReadLn;
end.
```

7. Hansı ədədləri və hansı ardıcıllıqla daxil etmək lazımdır ki, aşağıdakı operatorların icrasından sonra ekrana 123 çıxsın?

```
Read(a, b, c);
Write(c, b, a);
```

8. Proqramda olan bütün xətaları tapıb düzəldin.

```
proqram Task8;
var
    a; b; c : Integer;

begin
    WriteLn(‘ Ədədi daxil edin ’, a);
    ReadLn(a)
    b = 5;
    c = ab;
    WriteLn(a ‘ * ’ b ‘ = ’, s);
    ReadLn(a);
end.
```

9. Tam ədədin daxil edilməsini istəyən və həmin ədədin kvadratını, kubunu ekrana çıxaran proqram yazın. Proqramın icrasının nəticəsi təxminən belə olmalıdır:

```
Ədədi daxil edin.
4
4**2 = 16
4**3 = 64
```

10. Aşağıdakı proqram istifadəçidən cari aydakı günlərin sayını və bugünkü günü soruşur, sonra isə cari ayın sonuna neçə gün qaldığı haqqında məlumatı ekrana çıxarır. Proqramın sg dəyişəni cari aydakı günlərin sayını, bg dəyişəni bugünkü günü, qg dəyişəni isə ayın sonunadək qalan günlərin sayını göstərir. Proqramın necə icra olunduğunu yoxlayın.

```
proqram Task10;
var
    sg, bg, qg : Integer;

begin
    WriteLn(‘Cari ayda neçə gün var?’);
    ReadLn(sg);
```

Qeyd: 10. – həlli ilə verilmiş tapşırıqlar. Onların həllinə diqqət yetirin və təhlil edin.

```
    WriteLn('Bu gün ayın neçəsidir?');
    ReadLn(bg);
    qg := sg - bg;
    WriteLn('Bu ayın sonuna ', qg, ' gün qalıb');
    ReadLn;
end.
```

11. İstifadəçidən doğulduğu ili və cari ili soruşan, sonra onun yaşını ekrana çıxaran proqram yazın. Proqramın icrasının nəticəsi təxminən belə olmalıdır:

```
Hansı ildə doğulmusan?
1994
İndi neçənci ildir?
2008
Sənin bu il 14 yaşın var.
```

12. x dəyişəninin hansı qiymətlərində aşağıdakı bərabərliklər doğru olacaq?

a) `x div 5 = 8`
b) `50 div x = 7`
c) `50 mod x = 7`
d) `x div 5 = x mod 5`
e) `20 div x = 20 mod x`

13. Tutaq ki, S dəyişənində beşrəqəmli ədəd saxlanılır. a dəyişəni həmin ədəddəki on minliklərin sayını, b minliklərin sayını, c yüzlüklərin sayını, d onluqların sayını, e isə təkliklərin sayını göstərir. Aşağıdakı cədvəlin iki sütunu arasında uyğunluq qurun.

Əməliyyat	Dəyişən
`s div 100 mod 10`	a
`s mod 10`	b
`s div 10 mod 10`	c
`s div 10000`	d
`s mod 100 div 10`	e

14. Dördrəqəmli ədədin rəqəmlərini tapan proqram yazın. Aşağıda istifadəçi ilə kompüter arasındakı dialoq nümunəsi verilib. İstifadəçinin daxil etdiyi verilənlər qalın şriftlə göstərilib.

```
Dördrəqəmli ədəd daxil edin.
4523
Minliklərin sayı 4
Yüzlüklərin sayı 5
Onluqların sayı 2
Təkliklərin sayı 3
```

ŞƏRT

15. Operatorların ardıcıl yerinə yetirilməsi nəticəsində p dəyişəni hansı qiyməti alar?

```
q := -1;
p := 1;
if (p > 0) and (q > 0) then
   p := 2
else
   if (p < 0) and (q < 0) then
      p := 3
   else
      p := 4;
```

16. Operatorların ardıcıl yerinə yetirilməsi nəticəsində p və q dəyişənləri hansı qiyməti alar?

```
q := false;
p := true;
p := p and q;
q := q or false;
q := (not q) or p;
```

17. Operatorlar ardıcıllığı yerinə yetirilərsə, c dəyişəni hansı qiyməti alar?

```
a := 8;
a := a + 2;

b := a - 1;
c := a + b div 2;
```

18. Aşağıdakı proqram fraqmentində olan bütün xətaları tapıb düzəldin.

```
if a >= 10 and a <= 99 then
   WriteLn(a ' ikirəqəmli ədəd');
   WriteLn('onun kvadratı =', sqrt(a));
else (a, ' ikirəqəmli ədəd deyil');
```

19. Klaviaturadan 3, 5, 9 ədədləri daxil olunarsa, aşağıdakı proqramın icrasından sonra v, t, u dəyişənləri hansı qiymətləri alar?

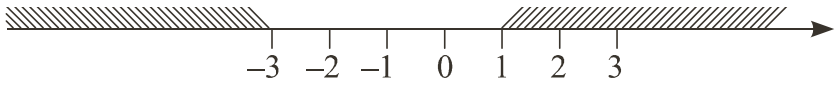
```
proqram Task19;
var
   a, b, c, v, t, u : Integer;

begin
   WriteLn('Üç ədəd daxil edin');
   ReadLn(a, b, c);
   v := 1; t := 0; u := 0;
   if a mod 3 = 0 then begin
      v := v * a;
      t := t + 1;
      u := u + 1;
   end;
   if b mod 3 = 0 then begin
      v := v * b;
      t := t + 1;
      u := u + b;
   end;
   if c mod 3 = 0 then begin
      v := v * c;
```

```
            t := t + 1;
            u := u + c;
        end;
        WriteLn('v=', v, ' t =', t, ' u=', u);
        ReadLn;
    end.
```

20. and, or, not məntiqi əməllərindən istifadə etməklə aşağıdakı şərtləri proqramlaşdırma dilində yazın.

a) x ədədi [-3, 2] parçasında yerləşir;
b) a ədədi ştrixlənmiş sahəyə düşür.



21. $ax + b = 0$ tənliyi verilmişdir. Bu tənliyin həllini tapan, yaxud onun həllinin olmaması haqqında məlumatı ekrana çıxaran proqram yazın.

22. İstifadəçinin yaşına görə onun hansı yaş qrupuna mənsub olduğunu müəyyənləşdirən proqram yazın:

- 13-dək – uşaqlıq
- 14-dən 24-dək – cavanlıq
- 25-dən 59-dək – yetkinlik
- 60-dan çox – qocalıq

23. $ax^2 + bx + c = 0$ kvadrat tənliyi verilmişdir. Aşağıdakı proqram daxil edilən a, b, c qiymətlərinə görə bu tənliyin həllini tapır, yaxud həllin olmaması haqqında məlumat verir. Proqramın icrasını yoxlayın.

```
program KvadratTenlik;
var a, b, c  : Real;
    D : Real;
    x1, x2 : Real;

begin
   Write ('a, b, c əmsallarını daxil edin: ');
```

```
    ReadLn(a, b, c);
    if (a = 0) and (b = 0) and (c = 0)
       then begin
          Write ('Bütün əmsallar 0-ra bərabərdir');
          WriteLn ('x - ixtiyarı ədəddir')
       end
       else
       if (a = 0) and (b <> 0)
         then WriteLn('Tənliyin bir kökü var x=',
                    (-c/b):6:2)
         else
          begin
             D := b*b - 4*a*c;
             if D > 0
                then begin
                  x1:=(-b+sqrt(D))/(2*a);
                  x2:=(-b-sqrt(D))/(2*a);
                  WriteLn('x1=', x1:6:2,'x2=',
                        x2:6:2)
                end
                else
                  if D = 0
                     then begin
                       x1: = -b/(2*a);
                       WriteLn('Köklər eynidir');
                       WriteLn(x1=',x1:6:2,
                              'x2=', x2:6:2);
                     end
                  else WriteLn('Həqiqi kökləri
                              yoxdur');
          end;
end.
```

DÖVRLƏR

24. 1-dən 20-ə kimi natural ədədlərin kvadratlarını çap edin.

25. 4-ə vurma cədvəlini çap edin.

26. 1-dən 100-ə kimi 4-ə tam bölünən natural ədədləri çap edin.

27. n və m natural ədədləri verilmişdir. Vurma əməlindən istifadə etmədən onların hasilini tapan proqram yazın.

28. Verilmiş n ədədinə görə həmin ədədi

```
6
6   6
6   6   6
6   6   6   6
6   6   6   6   6
6   6   6   6   6   6
```

şəklində (nümunə n= 6 halı üçün göstərilib) ekrana çıxaran proqramı aşağıdakı kimi yazmaq olar. Onun necə icra olunduğunu yoxlayın.

```
program Task28;
var i, j, n  : Integer;

begin
   ReadLn(n);
   for i := 1 to n do begin
       for j := 1 to i do Write(n, ' ');
       WriteLn;
   end;
end.
```

29. Tam ədədləri aşağıdakı şəkildə ekrana çıxaran proqram yazın:

```
6   6   6   6   6   6
6   6   6   6   6
6   6   6   6
6   6   6
6   6
6
```

30. Tam ədədləri aşağıdakı şəkildə ekrana çıxaran proqram yazın:

```
0
1   0
2   1   0
3   2   1   0
4   3   2   1   0
5   4   3   2   1   0
```

MASSİVLƏR

31. Operatorlar ardıcıllığı yerinə yetirilərsə, p və q dəyişənləri hansı qiymətləri alar?

```
for i := 1 to 10 do
    for j := 1  to 5 do
      A[i,j]  := i*j;
p := 0;
q := 0;
m := 2;
n := 5;
for k := 1 to 5 do begin
   p: = p + A[m,k];
   q := q + A[n,k];
end;
```

32. Aşağıdakı proqram 10 elementdən ibarət birölçülü X massivinin ilk 5 elementini çıxışa verir. Proqramın necə icra olunduğunu yoxlayın.

```
program Print;
var i : Integer;
    X : array[1..10] of Integer;

begin
```

```
    for i := 1 to 5 do
      Write(X[i], ' ');
    WriteLn;
  end.
```

33. Aşağıdakı proqram birölçülü massivdə müsbət və mənfi elementlərin sayını hesablayıb çıxışa verir. Proqramın necə icra olunduğunu yoxlayın.

```
program Task33;

 const Nmax = 100;
 type  TArr = array[1..Nmax] of integer;
 var   A : Tarr;

 procedure Solve;
   var i, n, p : Integer;
   begin
     p := 0;
     ReadLn(n);
      for i := 1 to n do Read(A[i]);
     for i := 1 to n do
        if A[i] >= 0 then Inc(p);

     WriteLn('Müsbət elementlərin sayı ', p);
     WriteLn('Mənfi elementlərin sayı ', n - p);
   end;

begin
  Solve;
end.
```

34. Verilmiş birölçülü massivdə sonuncu mənfi elementin nömrəsini tapan proqram yazın.

35. Tam ədədlər massivi verilmişdir. Klaviaturadan daxil edilən q və t ədədlərinə görə massivin onlar arasında qalan elementlərinin cəmini tapın.

36. Ölçüsü n x m olan A massivini aşağıdakı qaydada doldurun:

```
1   2    3    4    5    6    7
8   9    10   11   12   13   14
15  16   17   18   19   20   21
22  23   24   25   26   27   28
```

SƏTİRLƏR

37. Aşağıdakı proqram ASCII simvollarının sıra nömrəsini və simvolların özlərini ekrana çıxarır. Proqramdakı k dəyişənindən bir sətirdə 15 simvol vermək üçün sayğac kimi istifadə olunur. Proqramın necə icra olunduğunu yoxlayın.

```
program Task37;

var i, k : Integer;

begin
   WriteLn('Simvolların sıra nömrəsinin –
            i dəyişəninin qiymətinin və simvolla-
            rın özlərinin ekrana çıxarılması');
   k := 0;
   for i := 1 to 255 do begin
     Write(i : 4, ' Simvol ', Chr(i));
     k :=k + 1;
     if k = 15 then begin
        WriteLn;
         k := 0;
     end;
  end;
end.
```

38. Yazı qaydalarına görə mətndə vergüldən sonra həmişə boşluq qoyulur. Aşağıdakı proqram mətndə bu tipli səhvləri tapıb düzəldir. Proqramın necə icra olunduğunu yoxlayın.

```
program Task38;

var i : Integer;
    s : string;

begin
   WriteLn('Mətni daxil edin');
   ReadLn(s);
   i := 1;
    while i < Length(s) do begin
      if (s[ i] = ',') and not(s[ i+1] = ' ')
          then Insert(' ' s, i+1);
      i :=i + 1;
   end;
   WriteLn(s);
   ReadLn;
end.
```

39. Əvvəlki məsələnin proqramını elə dəyişdirin ki, "!", "?", "." simvollarından sonra da uyğun səhvi düzəltsin.

40. Verilmiş sətirdə ən qısa və ən uzun sözü müəyyənləşdirən proqram yazın.

41. Sətirdə müəyyən hərfə (məsələn, "a") neçə dəfə rast gəlindiyini hesablayan proqram yazın.

FAYLLAR

42. Diskdə “test.txt” adı ilə saxlanılan mətn faylı verilib:

```
123 17 25
256 80 5
89 56 234
123 123 123
81 11 11 11
```

Aşağıdakı proqramın icrasından sonra ekrana nə çıxarılacaq?

```
program Task42;
var f : text;
    s : string;
    n, m : Integer;
    c, z : Char;
begin
   Assign(f, 'test.txt');
   Reset(f);
   ReadLn(f, s);
   ReadLn(f, n);
   Read(f, m);
   ReadLn(f, c);
   Read(f, z);
   Close(f);
   WriteLn('s=', s);
   WriteLn('n=', n);
   WriteLn('m=', m);
   WriteLn('c=', c);
   WriteLn('z=', z);
   ReadLn;
end.
```

43. Tutaq ki, f mətn faylı və st sətri verilmişdir. Aşağıdakı proqram s dəyişəninin qiymətini f faylının sətirlərində axtarır və hansı sətirdə tapırsa, həmin sətri yeni g faylına yazır.

```
program Task43;
var f, g : text;
    s, st : string;
begin
WriteLn('Sətri daxil edin ');
ReadLn(s);

Assign(f, 'test.txt');
Assign(g, 'test2.txt');
Reset(f);
Rewrite(g);
while not eof(f) do begin
   ReadLn(f, st);
   if pos(s, st) <> 0 then WriteLn(g, st);
end;
close(f);
close(g);
ReadLn;
end.
```

44. Mətn faylı verilib. Bu faylın ən qısa sətirlərini yeni fayla yazın.

45. Verilmiş mətn faylında ən uzun sətri müəyyənləşdirən proqram yazın.

46. Şagirdlərin siyahısından ibarət mətn faylı verilib. Hər sətirdə bir şagirdin soyadı və adı yazılıb. Bu faylı oxuyub ekrana çıxaran proqram yazın.

2

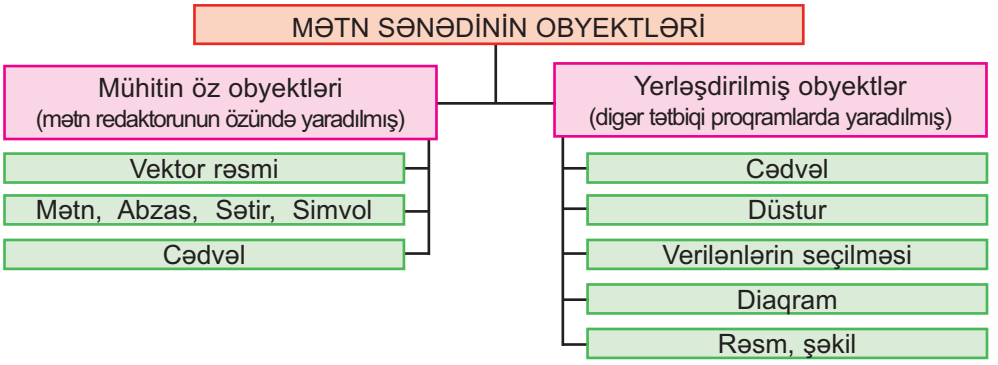
ELEKTRON SƏNƏD

2.1. MƏTN SƏNƏDİ VƏ ONUN OBYEKTLƏRİ

Kompüterdən ən geniş istifadə olunan sahələrdən biri mətnlərin hazırlanmasıdır. Mətnlər hazırlanarkən çox zaman bir neçə dəfə dəyişdirilir. Kağızda edilən hər hansı dəyişikliyin izi qalır və buna görə də istənilən kompüterdə mətnlərlə işləmək üçün xüsusi proqramlar – *mətn redaktorları* quraşdırılıb.

Mətn redaktorunda hazırlanmış istənilən mətnə, ona daxil edilmiş mətn olmayan materiallarla birlikdə *sənəd* [document] deyilir.

Mətn prosessorunda işə başlamamışdan öncə onun istifadə etdiyi obyektlər haqqında müəyyən təsəvvürlərə malik olmaq lazımdır. Bununçün mətn sənədi obyektlərinin təsnifatı ilə tanış olaq.



Sxemdən göründüyü kimi, mətn sənədinin obyektləri iki səviyyə üzrə qruplaşır. Birinci səviyyədə obyektlər yaradıldığı mühitə görə iki qrupa bölünür: *mühitin öz obyektləri* və *yerləşdirilmiş obyektlər*.

Mühitin obyektləri sənədin elə obyektləridir ki, onların yaradılması, redaktəsi və formatlanması üçün başqa bir proqram çağırmağa lüzum qalmır.

Yerləşdirilmiş obyektlər elə obyektlərə deyilir ki, onlar ilkin olaraq başqa proqram mühitində yaradılır.

Yerləşdirilmiş mühitlə bağlı belə bir bənzətməyə baxaq. Mənzil şəhər sakininin yaşayış mühitinin bir hissəsidir. Burada olan hər şey – mebel və qab-qacaq, geyim və ayaqqabı, işıqlandırma və məişət cihazları onun üçün nəzərdə tutulub. Şəhərlər meydana gəlməmişdən qabaq insan, əsasən, canlı təbiətlə əhatə olunmuşdu. Şəhərlilərin çoxu heyvan və bitki aləmi ilə ünsiyyətdə olmadığına görə darıxır. Onlar öz mənzillərində hər hansı bitki, yaxud balıq saxlamaq istəyirlər. Ancaq çiçək döşəmədə, yaxud masanın üstündə bitə bilməz. Normal həyat üçün ona alışdığı mühit – torpaq gərəkdir. Çiçəyin mənzildə yetişdirilməsi üçün dibçək əldə etmək, onu torpaqla doldurmaq və çiçəyi ora əkmək lazımdır. Beləliklə, bir mühitin (şəhər mənzili) daxilində obyektin (çiçəyin) mövcudluğunu təmin edən başqa mühitin fraqmenti (torpaq) meydana çıxır.

Təsnifatın ikinci səviyyəsinin əsasını obyektin növü təşkil edir. Mətn redaktorunda yaradılan obyektləri gözdən keçirək. Öncə "mətn" alt-sinfinin obyektlərinə baxaq. Mətndə aşağıdakı obyektlər seçdirilə bilər: simvol, söz, sətir, abzas, səhifə.

Simvol obyektinin əsas parametrləri bunlardır: *yazılış şəkli, keql, rəng.*

Simvolun yazılış şəkli. Dörd əsas yazılış şəklindən istifadə olunur:

- Normal
- *Kursiv* (Italic)
- **Qalın** (Bold)
- ***Qalın kursiv*** (Bold Italic)

Mətndə kursivdən *yumşaq vurğulama* vasitəsi kimi istifadə olunur. Oxucunun diqqətini nəyəsə cəlb etmək üçün, yaxud nəyəsə xüsusi münasibət bildirmək üçün kursiv şəkli daha münasibdir.

Qalın yazılış şəklindən *güclü vurğulama* vasitəsi kimi, məsələn, dönməz mövqeyi, yaxud möhkəm iradəni bildirmək üçün istifadə olunur. Eləcə də, bu şəkildən sənədin tərtibatı zamanı başlıqlarda istifadə olunur.

UNICODE

Kompüterlər yalnız ədədlərlə işləyə bilir və onların yaddaşda hərfləri və ya başqa simvolları saxlaya bilməsi üçün həmin simvolların hər birinə uyğun bir ədəd qoyulmalıdır. Əvvəllər simvolların belə kodlaşdırılması üçün yüzlərlə müxtəlif sxem var idi. Ancaq onların heç biri lazımi simvolların hamısını göstərə bilmirdi. Hətta bir dil üçün belə bütün hərfləri, punktuasiya işarələrini və texniki simvolları əhatə edən vahid kodlaşdırma sistemi yox idi.

Buna görə də planetimizdəki mövcud əlifbaları əhatə etmək üçün yeni kodlaşdırma sxeminin işlənib hazırlanmasına ehtiyac yarandı. Nəticədə, *Unicode* ("yunikod" kimi tələffüz olunur) meydana çıxdı.

Unicode mürəkkəb obyektdir. Onun uzun sürən işlənib hazırlanma prosesində dünyanın hər yerindən dilçilər və kompüter mütəxəssisləri iştirak etmişlər. Unicode yanaşmasının mahiyyəti ondan ibarətdir ki, hər bir simvol 16 bitlik ədədlə göstərilir. Bu isə o deməkdir ki, hər bir simvolu iki baytla vermək olar. Bu qayda ilə 65 536 işarə və ya simvol kodlaşdırmaq olur. Unicode kodlaşdırmasında platformadan, proqramdan, dildən asılı olmayaraq, istənilən simvola unikal kod mənimsədilir.

Keql sriftin ölçüsüdür. Ənənəvi olaraq keql *punktla (pt)* ölçülür. Bir punkt 0.35 mm-ə bərabərdir. Bir sıra standart keqllər vardır. Onlardan bir neçəsi aşağıda göstərilib:

12 Keql şriftin ölşüsüdür
18 Keql şriftin ölşüsüdür
24 Keql şriftin ölşüsüdür
36 Keql şriftin ölşüsüdür
48 Keql şriftin ölşüsüdür
60 Keql şriftin ölşüsüdür
72 Keql şriftin ölşüsüdür

Rəng. Tətbiqi proqram mühitlərinin obyektlərinin əksəriyyətinin rəngi olur. Simvol, xətt və s. kimi bircins strukturlu obyektlər yalnız bir rənglə boyanır. Mürəkkəb strukturlu obyektlərdə isə (məsələn, avtofiqur, xana, sahə və s.) xəttin (sərhədin) və fonun rəngləri fərqləndirilir. 

Simvol bu əsas parametrlərlə yanaşı, effekt, sürüşmə və kerninq kimi parametrləri ilə də xarakterizə olunur.

Effekt simvolun ekranda və sənəddə necə görünəcəyini müəyyənləşdirir. Ən çox istifadə olunan effektlər cədvəldə göstərilib.

Effekt	
Effekt	
Effekt	
Effekt	
Effekt	
EFFEKT	
EFFEKT	

Sürüşmə simvolun sətrin təməl xəttinə görə yerini müəyyən edir. İki növ sürüşmə olur: *aşağı* və *yuxarı*.

Təməl xəttə nəzərən yuxarı sürüşmə

Təməl xəttə nəzərən aşağı sürüşmə

Kerninq simvollararası intervaldır. Bu parametr üç qiymət alır: normal, genişlənmiş, sıxılmış.

İnterval	Sıxılmış [Condensed]
İnterval	Normal [Normal]
İnterval	Genişlənmiş [Expanded]

Adətən, kerninqin normal qiyməti verilmiş şriftin şəklinə görə avtomatik müəyyən olunur. Kerninqin qiymətini dəyişdirmək olar.

Defis və tire

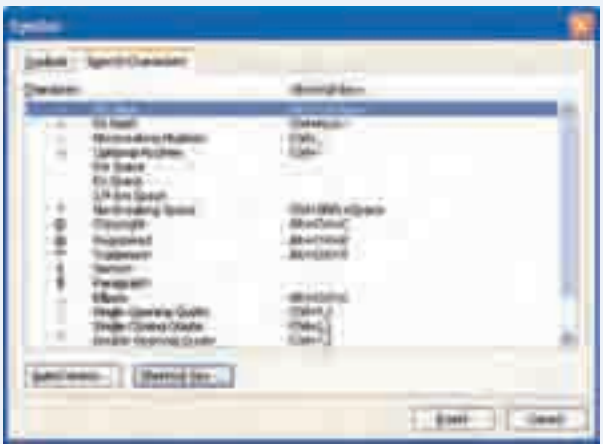
Bəzi simvolların mətndə xüsusi təyinatı olur. Həmin simvollardan ikisi – *defis* və *tire* ilə yaxından tanış olaq.

Defis sözün iki hissəsini, yaxud bir söz kimi oxunan iki sözü bir-birinə bağlamaq üçün simvoldur. Defislə qonşu hərflər arasında boşluq qoyulmur. Əgər defislə yazılan söz bir sətrə yerləşmirsə, mətn prosessoru həmin sözü defisə görə iki hissəyə böləcək. Bu halda oxunma zamanı defis sətirdən sətrə keçid kimi qəbul olunacaq. Defislə ayrılan illər də (məsələn, 1918-1920) müxtəlif sətirlərdə olduqda pis qəbul ediləcək. Bu arzuolunmaz "keçiddən" qaçmaq üçün mətn prosessorunda xüsusi simvoldan – *kəsilməz defisdən* [nonbreaking hyphen] istifadə olunur.

Tire cümlənin iki hissəsini bir-birindən ayıran işarədir. Defisdən fərqli olaraq, tire qonşu sözlərdən *mütləq* boşluqla ayrılır. Çap məhsullarında defis qısa, tire isə uzun üfüqi cizgi ilə işarə olunur. Lakin kompüter klaviaturasında həm minus işarəsi, həm defis, həm də tire kimi istifadə oluna bilən yalnız bir klaviş var. Buna görə də tireni yazmaq üçün xüsusi simvoldan – *uzun tiredən* [em dash] istifadə olunur.

Çalışma

1. Hər hansı mətn sənədini açın.
2. Mətni oxuyun. Defis və tire simvollarından düzgün istifadə olunub-olunmadığını müəyyənləşdirin.
3. Aşkarlanan yanlışlıqları düzəldin.
4. "Uzun tire" və "kəsilməz defis" simvollarını qoymaq üçün Insert⇨Symbol komandasını yerinə yetirin və lazım olan simvolları Special Characters bölümündə tapın.



1. Mətnin hansı obyektləri var?
2. Simvol obyekti nə ilə xarakterizə olunur?
3. Simvolların hansı yazılış şəkli var?
4. Effekt nədir? Onun mahiyyətini nümunələr üzərində izah edin.
5. Sürüşmə və kerninq nədir? Onun mahiyyətini nümunələr üzərində izah edin.
6. Mətn redaktorunda hər hansı mətn yazın və dərsdə öyrəndiyiniz parametrləri ayrı-ayrı simvollara tətbiq edin.

2.2. MƏTN SƏNƏDİNİN HAZIRLANMASI

Kompüterdə sənədin hazırlanması bir neçə mərhələdən keçir:

mətnin daxil edilməsi (yığılması); redaktə olunması; formatlanması; çap edilməsi.

Bu mərhələlərlə tanışlığa keçməmişdən öncə mətn sənədinin bir neçə obyekti – *söz, sətir* və *abzasla* qısaca tanış olaq.

Simvollardan sözlər əmələ gəlir.

Söz "boşluq" simvolunun olmadığı simvollar (hərflər, rəqəmlər, xüsusi işarələr) ardıcıllığıdır. Sözün yeganə "müstəqil" parametri simvolların sayıdır.

Sətir. Sətrlər sözlərdən təşkil olunur. Sətirdə sözlər bir-birindən boşluq simvolu ilə ayrılır. Boşluqdan başqa, sözdən sonra yalnız vergül, nöqtə, nöqtəli vergül, iki nöqtə, mötərizə kimi durğu işarələri gələ bilər. Bu işarələr bilavasitə sözdən sonra qoyulur, başqa sözlə, həmin işarələrlə söz arasında boşluq olmamalıdır. Söz obyektinin bütün parametrləri sətir obyektinə də aiddir. Sətir obyektinin yeganə müstəqil parametri *sözlərin sayıdır*. Formatlama zamanı simvollar və sözlərin tabe olduqları qaydalar sətirlərə də aid olur.

Abzas. Sətir obyektlərindən abzas obyektləri yaranır. Abzasın daxil edilməsi həmişə <Enter> klavişinin basılması ilə sona çatır. Abzasın sonunu bildirən işarə ¶ simvoludur. Adi rejimdə bu simvol ekranda görünmür.

Mətnin daxil edilməsi. Mətnin daxil edilməsi (yığılması), adətən, klaviatura vasitəsilə həyata keçirilir. Burada kağız rolunu monitorun ekranı oynayır. Növbəti simvolun yerini ekranda yanıb-sönən şaquli cizgi – *kursor* göstərir.

Mətn yığarkən aşağıdakı qaydalara əməl edin:

1. Tiredən başqa, bütün durğu işarələrini bilavasitə sözün axırıncı hərfindən sonra qoyun. İstənilən durğu işarəsindən sonra boşluq (<Spacebar>) klavişini basın. Tirenin hər iki tərəfində boşluq saxlayın.
2. Mətni yığarkən buraxılmış səhvləri düzəltmək olar. Kursorun sağında olan yanlış simvolu uzaqlaşdırmaq üçün <Delete> klavişindən, solundakı üçünsə <Backspace> klavişindən istifadə edin.
3. Kompüterdə mətn yığarkən sətrin sonunu nəzarətdə saxlamayın: kursor sətrin sonuna çatan kimi öz-özünə növbəti sətrin başlanğıcına keçəcək.
4. Yeni abzasa keçmək üçün <Enter> klavişini basın.

Mətnin redaktəsi. Redaktə sənədin məzmununun dəyişdirilməsidir. Redaktəyə aşağıdakı əməliyyatlar aiddir:

- mətnin yığılması;
- səhvlərin düzəldilməsi;

• mətn hissələrinin üzünün köçürülməsi, yerinin dəyişdirilməsi, silinməsi;
• mətnə şəkillərin, cədvəllərin və başqa obyektlərin əlavə olunması.

Mətnin formatlanması. Mətndə olan informasiyanın oxucuya tez və asan çatdırılması üçün onun xarici görünüşünün önəmi böyükdür. Oxucunun diqqətini mühüm informasiyaya cəlb etmək üçün ayrı-ayrı sözləri, yaxud ifadələri cürbəcür seçdirmək olar.

> Formatlama mətnin oxunmasını asanlaşdırmaq məqsədilə sənədin və onun ayrı-ayrı hissələrinin xarici görünüşünün dəyişdirilməsidir.

Formatlama sözü "forma" sözündən əmələ gəlib, formatlamaq – nəyəsə forma vermək deməkdir. Formatlama əməliyyatına mətnin seçdirilməsi ilə bağlı müxtəlif üsullar aiddir:

• simvolların xassələrinin dəyişdirilməsi;
• abzasların xassələrinin dəyişdirilməsi;
• başlıqların və altbaşlıqların tərtibatı;
• mətnin siyahıya çevrilməsi;
• mətnin cədvəl şəklinə çevrilməsi;
• kolontitulların, səhifələrin nömrələrinin qoyulması və s.

Mətnin hər hansı fraqmentində simvolların xassələrini dəyişdirmək üçün həmin fraqmenti seçdirmək, sonra isə alətlər zolağındakı düymələrin, yaxud Font dialoq boksunun köməyilə onların parametrlərini dəyişdirmək lazımdır.

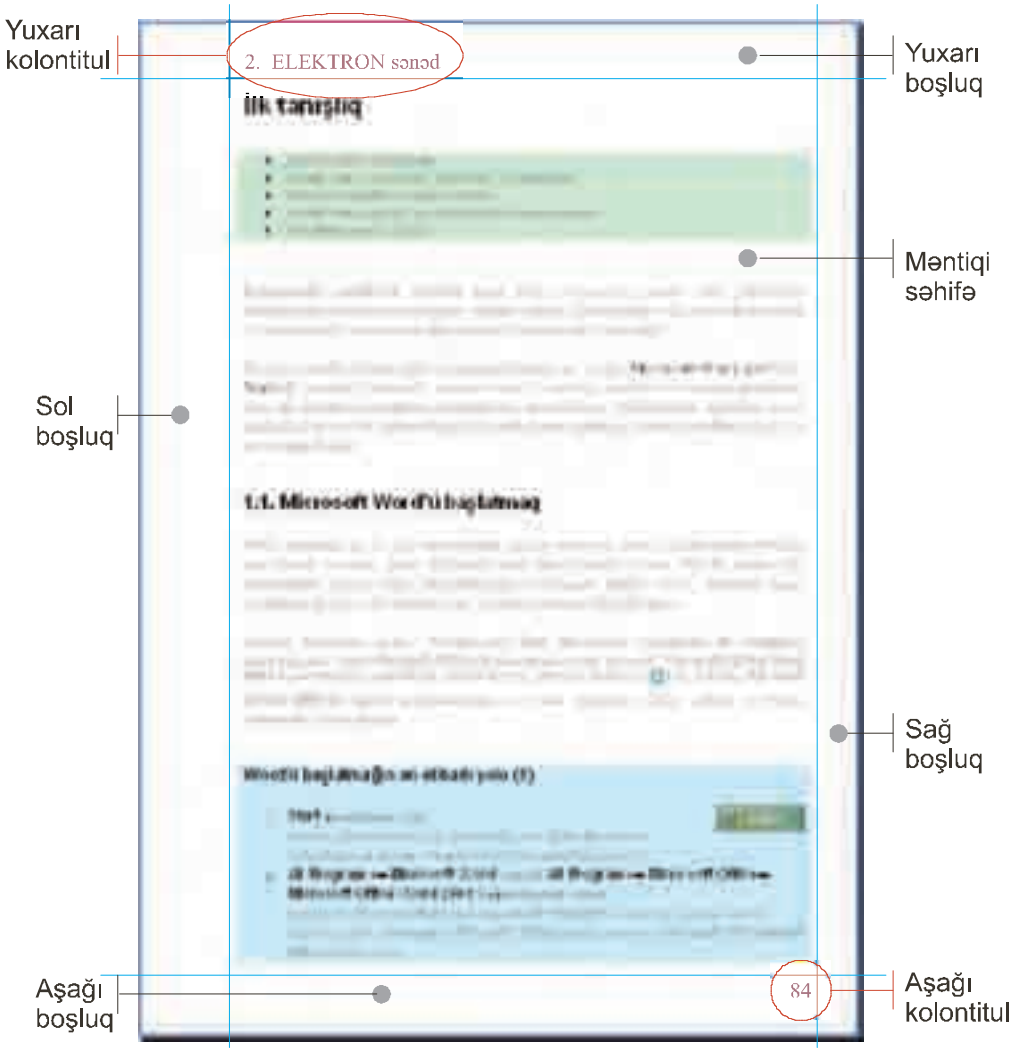
Çalışma

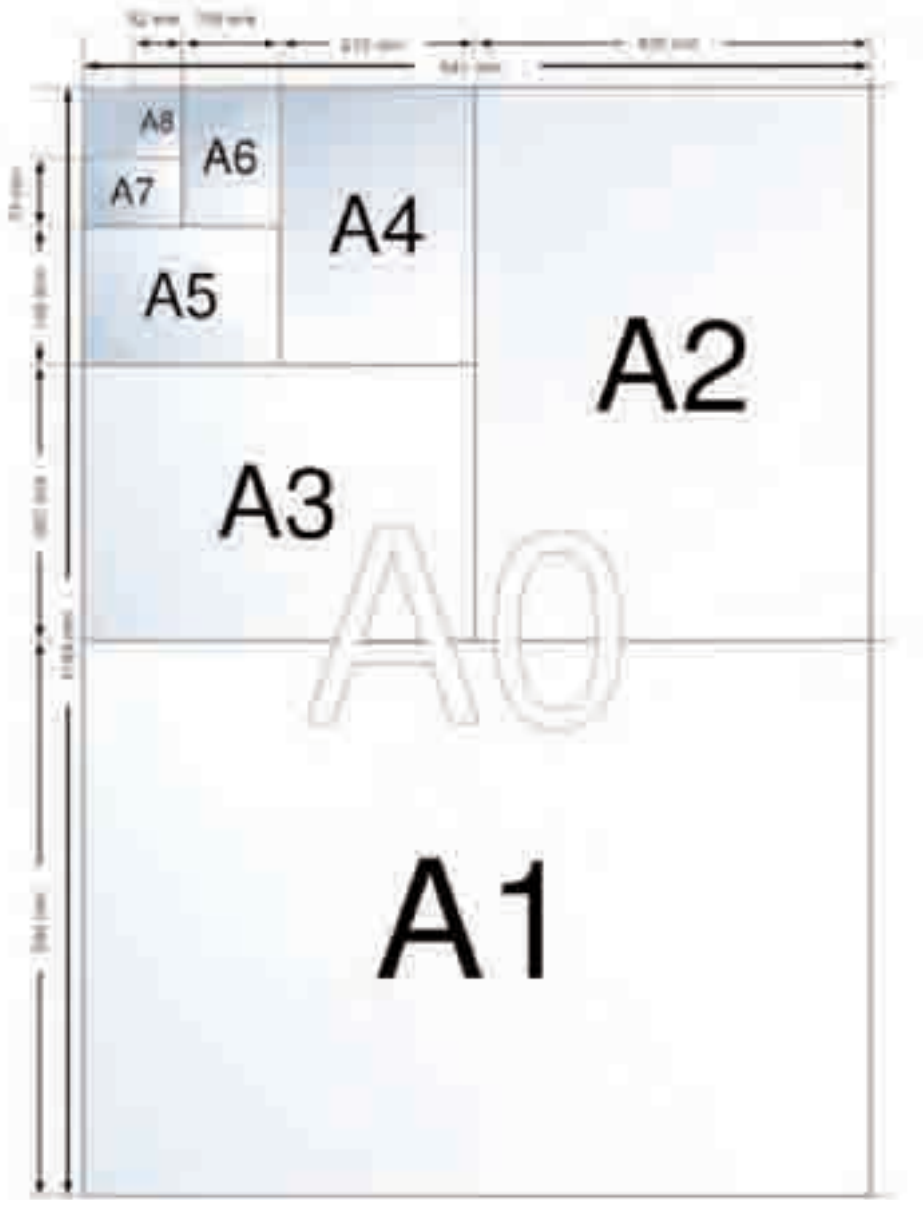
1. Hər hansı mətn sənədini açın.
2. Mətnin bir fraqmentini seçdirin və şriftini belə dəyişdirin:
   Courier New, qalın, 16 pt, qırmızı, hamısı baş hərflər.
3. Mətnin başqa bir fraqmentini seçdirin və şriftini belə dəyişdirin:
   Tahoma, kursiv, 10 pt, üstündən xətt çəkilmiş.
4. Mətnin daha bir fraqmentini seçdirin və şriftini belə dəyişdirin:
   Arial, 10 pt, gizli [hidden].
5. Çap olunmayan simvolların göstərilməsi rejiminə keçin. Bununçün standart alətlər zolağında ¶ düyməsini çıqqıldadın. Üçüncü fraqmentin necə əks olunmasına diqqət yetirin.
   Sənəddə çap olunmayan hansı simvolları görürsünüz?

6. <Ctrl> klavişindən istifadə etməklə mətnin başqa iki fraqmentini seçdirin. Həmin fraqmentlər üçün şrifti belə dəyişdirin: Comic Sans MS, 20 pt, yaşıl, konturlu.

Hər sənəd faylda saxlanıla, ekranda göstərilə, kağızda çap oluna bilər. Özü də sənəd ekranda necə görünürsə, kağızda (*fiziki səhifədə*) da elə çap olunacaq.

Fiziki səhifədə həmişə sənədin obyektlərini yerləşdirmək üçün sahə ayrılır. Bu sahə *məntiqi səhifə* adlanır.





Çap vərəqinin ölçüləri millimetrlə ölçülür. Azərbaycanda təməl ölçü olaraq, keçmiş Sovet İttifaqında olduğu kimi, ölçüsü 841×1189 mm olan və formatı **A0** adını almış vərəq götürülür. Ondan alınan daha kiçik formatlar **A1**, **A2**, ... **A8** adlanır və hər bir format özündən əvvəlkini yarıya bölməklə alınır.

Müasir kargüzarlıqda **A4** (210 mm × 297 mm) əsas format kimi qəbul olunub. Xüsusi hallarda geniş cədvəlləri vermək üçün **A3** (297 mm × 420 mm) formatlı vərəqlərdən istifadə olunur.

İstənilən tətbiqi proqramda sənədin səhifəsinin formatlanması, adətən, aşağıdakı parametrlərin quraşdırılmasını nəzərdə tutur:

səhifənin (vərəqin) ölçüsü;
səhifənin istiqaməti;
boş sahələr;
kolontitullar.

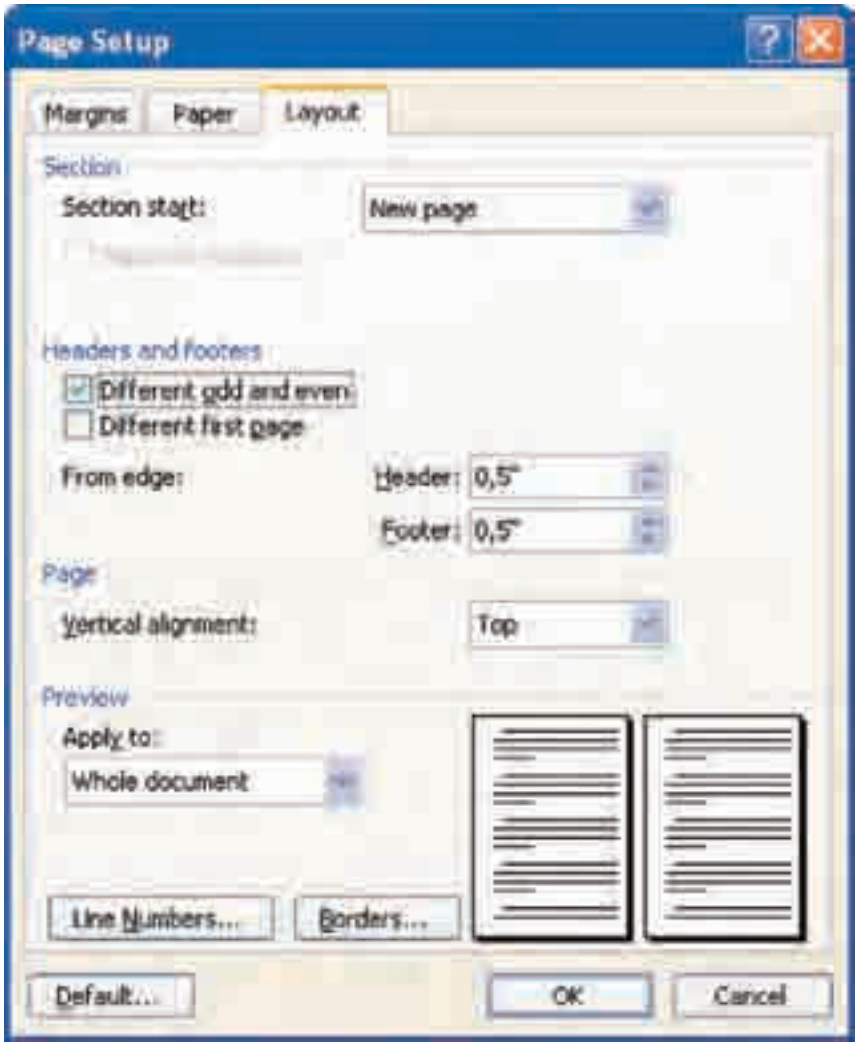
Səhifənin istiqaməti kağız vərəqin fəzada vəziyyətini təyin edir. İki cür istiqamət var: kitab və albom (portret və mənzərə).

Sahələr əlavə informasiyalar (haşiyələr, kolontitullar və s.) yerləşdirmək üçün fiziki səhifənin hissələridir. Adətən, sol və sağ sahələr boş qalır, onlar doldurulmur. Yuxarı və aşağı sahələrdə kolontitullar yerləşir.

Kolontitul (fransısca "colonne" – sütun və latınca "titulus" – başlıq, yazı) səhifənin yuxarı və aşağı sahələrində yerləşdirilən xidməti informasiyadır. Məsələn, bu, müəllifin adı, əsərin, bölmənin, fəslin, paraqrafın adı, tarix, səhifənin nömrəsi və s. ola bilər. Belə elementlər böyük həcmli sənədlərlə işi asanlaşdırır və çap olunandan sonra kitabın rahat oxunmasını təmin edir.

Çalışma

1. Hər hansı mətn sənədini açın.
2. File⇨Page Setup menyu komandasını yerinə yetirin.
3. Layout səhifəsində Different odd and even yoxlama boksunu qeyd edin. Bu onu göstərir ki, tək və cüt nömrəli səhifələrin kolontitulları fərqli olacaq.



4. View⇨Header and footer menyu komandasını yerinə yetirin.
5. Tək nömrəli səhifənin yuxarı kolontitulunun sahəsini tapın və orada **Maket 1** yazın.
6. Cüt nömrəli səhifənin aşağı kolontitulunun sahəsində <Tab> klavişini basın və sətrin ortasında öz soyadınızı yazın.
7. Kolontitulla iş panelini qapadın. Sənədə baxın və kolontitullarda təkrarlanan mətni tapın.

1. Kompüterdə sənədin hazırlanması hansı mərhələlərdən keçir?
2. Mətnin redaktəsi nə deməkdir və ona hansı əməliyyatlar aiddir?
3. Mətnin formatlanması nə deməkdir və ona hansı əməliyyatlar aiddir?
4. Məntiqi səhifə nədir və o, hansı hissələrdən ibarətdir?
5. Mətn redaktorunda hər hansı mətn yazın və onu formatlayın.

3

CƏDVƏL PROSESSORU

3.1. CƏDVƏL PROSESSORUNUN TƏYİNATI

Verilənləri normal qavramaq üçün tez-tez cədvəllərdən istifadə olunur. Cədvəl formasında verilmiş informasiyanı əks etdirmək və emal etmək üçün nəzərdə tutulmuş tətbiqi proqramlara *elektron cədvəllər* deyilir. Bəzən *cədvəl prosessoru* terminindən də istifadə olunur. Elektron cədvəlin iş sahəsi quruluş baxımından şahmat taxtasını xatırladır. O, hər birinin öz adı olan sətir və sütunlardan ibarətdir.

> Cədvəl prosessoru kompüterdə cədvəl formasında – elektron cədvəl şəklində verilmiş informasiyalarla işləmək üçün nəzərdə tutulub.

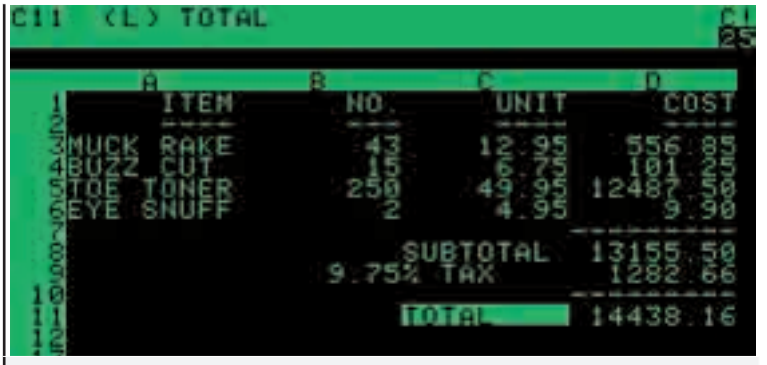
Cədvəl prosessorunun işinin nəticəsi *cədvəl*, yaxud *diaqram* şəklində olan sənəddir. Məsələn, cədvəl prosessorunda sinif jurnalını aparmaq olar. Müəllimlər orada şagirdlərin qiymətlərini yaza bilər, hazır düsturlar isə hər şagird üçün orta balı, sinfin hər fənn üzrə ümumi nailiyyətlərini hesablamağa imkan verər. Hər dəfə müəllim yeni qiymət yazdıqda cədvəl prosessoru bütün nəticələri avtomatik olaraq yenidən hesablayacaq.

Cədvəl prosessorunun xarakterik özəlliyi ondadır ki, orada verilənlər və hesablamaların nəticələri cədvəl formasında verilir. Əyanilik üçün həmin verilənləri diaqram kimi qrafik şəkildə də göstərmək olar.

Öz kağız sələfi ilə müqayisədə cədvəl prosessoru istifadəçiyə işləmək üçün daha çox imkanlar verir.

Cədvəlin xanalarına təkcə ədədlər, tarixlər və mətnlər deyil, həm də məntiqi ifadələr, funksiyalar və düsturlar yazmaq olar. Düsturlar ilkin verilənlərin dəyişilməsi zamanı ani olaraq yenidən hesablama aparmağa və uyğun xanaya yeni nəticəni yazmağa imkan verir.

İlk cədvəl redaktoru 1979-cu ildə yaradılmış **VisiCalc** olmuşdur. Sonradan bazara bu sinifdən olan çoxlu məhsullar çıxarılmışdır: **SuperCalc, Microsoft MultiPlan, Quattro Pro, Lotus 1-2-3, Microsoft Excel, OpenOffice.org Calc.**



VisiCalc – ilk elektron cədvəl proqramı

Elektron cədvəl ideyasını ilk dəfə amerikalı alim **Riçard Mattessiç** 1961-ci ildə "Budgeting Models and System Simulation" adı altında çap etdirdiyi araşdırmasında formalaşdırıb. Konsepsiya 1970-ci ildə **Pardo** və **Landau** tərəfindən genişləndirilib.
1979-cu ildə **Den Briklin** (1951) əfsanəvi VisiCalc proqramını yaratmaqla (**Bob Frenkston**la (1949) birlikdə) elektron cədvəl proqramlarının əsasını qoydu. Apple II kompüteri üçün hazırlanmış bu cədvəl redaktoru fərdi kompüterləri texnofillər üçün ekzotik oyuncaqdan biznes üçün kütləvi alətə çevirdi.

Hazırda cədvəl strukturuna malik sənədlərlə işləmək üçün ən populyar vasitələrdən biri Microsoft Office paketinə daxil olan **Microsoft Excel** proqramıdır.

Excel 2003 proqramının başladılması. Kompüterdə quraşdırılmış digər proqramlar kimi, Excel 2003 proqramını da başlatmağın ən asan yolu Windows sisteminin Start menyusundan istifadə etməkdir. Bunun üçün aşağıdakı addımlar atılmalıdır:

1. Windows sisteminin Start menyusunu açmaq üçün tapşırıqlar zolağındakı Start düyməsini çıqqıldadın.
2. Start menyusunun yuxarı hissəsindəki All Programs elementini seçin.
3. Açılan yeni menyudan Microsoft Excel 2003 bəndini seçin.

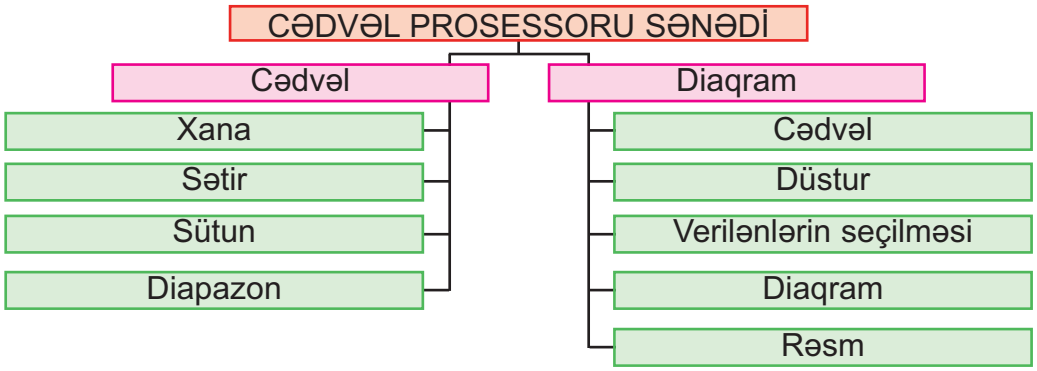
Axırıncı bənddən sonra Excel 2003 proqramı yüklənməyə başlayacaq, ekrana proqramın baş pəncərəsi çıxacaq və boş iş kitabı açılacaq.

Hər bir Excel faylına *iş kitabı* [workbook], yaxud, sadəcə, *kitab* deyilir. İş kitabı bir neçə *iş vərəqindən* [worksheet] ibarətdir.

Burada nə qədər xana var?

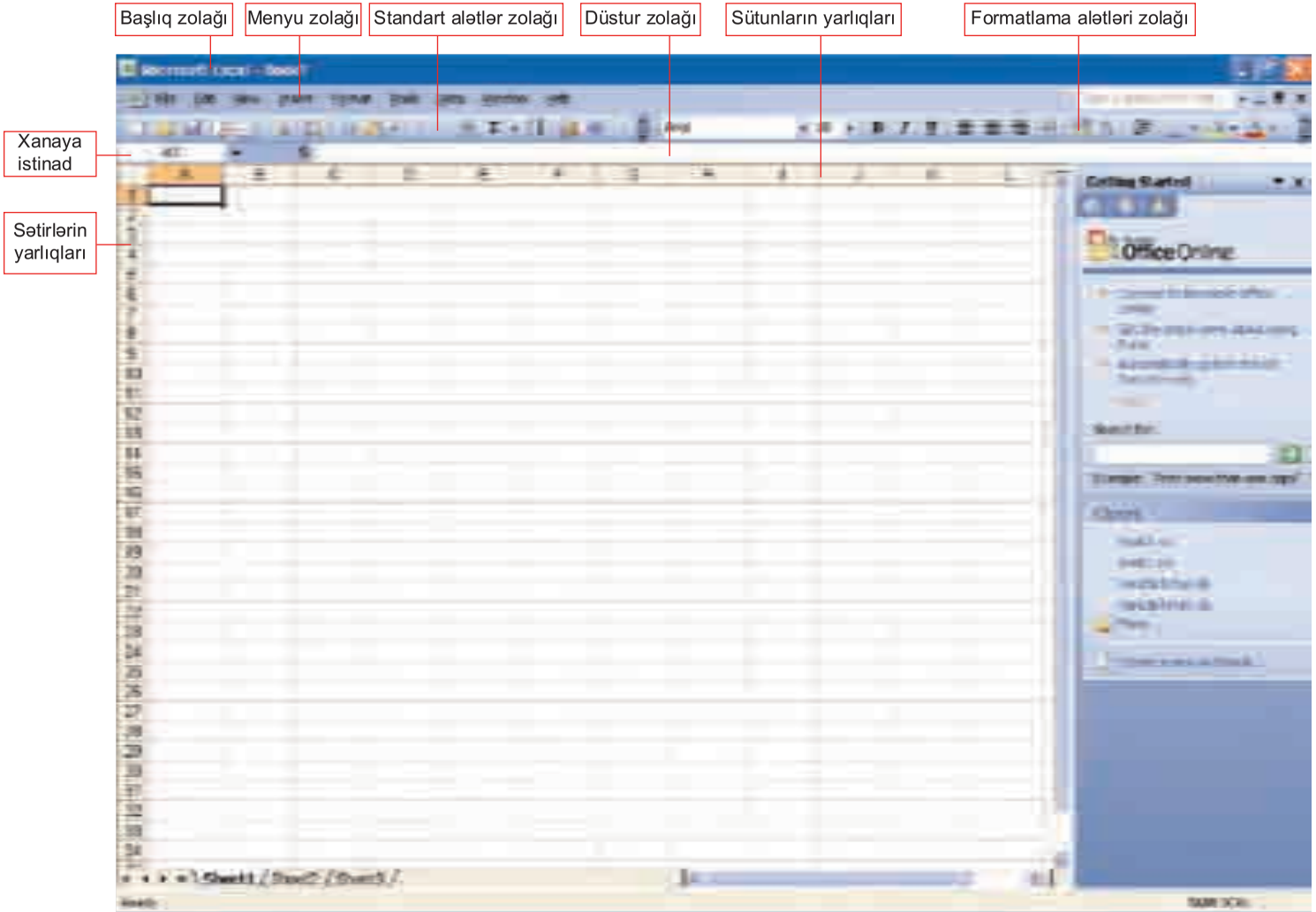
İş cədvəlinin hər vərəqində milyon xana olduğunu desək, heç də yanılmarıq. Doğrudan da, vərəqdə 256 sütun və 65 536 sətir var. Bu iki ədədi bir-birinə vursaq, 16 777 216 alarıq. Deməli, hər iş vərəqində 16 milyondan çox boş xana olur. Bu da sizə bəs etməsə, nəzərə alın ki, hər yeni açılan iş kitabında 3 iş vərəqi olur. Başqa sözlə, iş kitabında 50 331 648 xana sizin ixtiyarınızdadır və həmin kitaba özünüz də yeni vərəqlər artıra bilərsiniz.

Cədvəl prosessoru sənədinin obyektləri. Cədvəl sənədinin iki növü var: *cədvəl* və *diaqram*. Diaqram köməkçi sənəd olub, cədvəlsiz mövcud olmur. Aşağıdakı sxemdə cədvəl sənədi obyektlərinin təsnifatı verilib.



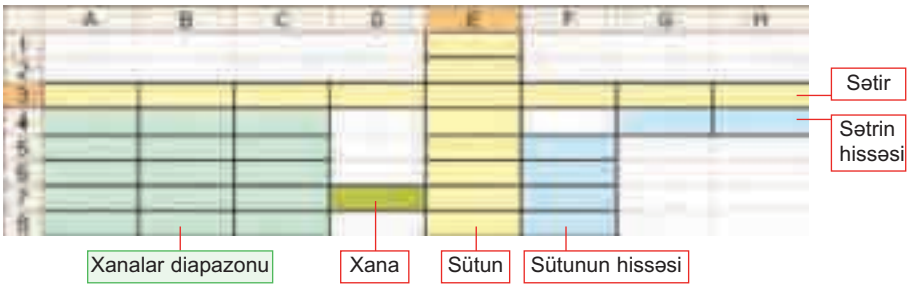
Təsnifatın birinci səviyyəsi sənədin növünə aiddir. İkinci səviyyədə isə cədvəli və diaqramı təşkil edən əsas obyektlər göstərilib. Növbəti dərslərdə bu obyektlər və onlarla necə işləmək haqqında ətraflı danışılacaq.

1. **Cədvəl prosessoru nədir?**
2. **Elektron cədvəlin adi cədvəldən hansı üstünlükləri var?**
3. **Excel cədvəl prosessoru necə başladılır?**
4. **İlk elektron cədvəl proqramı nə vaxt yaradılıb və necə adlanırdı?**
5. **İş kitabı nədir?**
6. **Excel cədvəl prosessorunu yuxarıda göstərilən üsulla başladın. Onu daha hansı yollarla başlatmaq olar?**


Başlıq zolağı
Menyu zolağı
Standart alətlər zolağı
Düstur zolağı
Sütunların yarlıqları
Formatlama alətləri zolağı
Xanaya istinad
Sətirlərin yarlıqları

3.2. ELEKTRON CƏDVƏLİN OBYEKTLƏRİ

Cədvəl mürəkkəb obyekt olub, bir neçə elementar obyektdən – sətirlər, sütunlar, xanalar, xanalar diapazonlarından ibarətdir. Hər bir elementar obyektin, elektron cədvəli hazırlayanlar tərəfindən müəyyənləşdirilmiş adı olur.



Xana elektron cədvəlin sətir və sütununun kəsişməsində yerləşən elementar obyektdir.

Sətir cədvəlin üfüqi istiqamətdə eyni səviyyədə yerləşmiş bütün xanalarıdır. Sətirlərin yarlıqları (başlıqları) 1-dən başlayaraq tam ədədlər şəklində göstərilir.

Sütun cədvəlin şaquli istiqamətdə eyni səviyyədə yerləşmiş bütün xanalarıdır. Sütunların yarlıqları (başlıqları) latın əlifbasının hərfləri ilə verilir: öncə **A**-dan **Z**-dək, sonra isə **AA**-dan **AZ**-yə, **BA**-dan **BZ**-yə və s. Axırıncı sütunun adı **IV** olacaq.

Xanalar diapazonu düzbucaqlı sahə əmələ gətirən qonşu xanalar çoxluğudur; o, bir sətir və ya bir sütundan, sətrin və ya sütunun hissələrindən ibarət ola bilər. Bir xananı da diapazon hesab etmək olar.

Xananın ünvanı onun cədvəldəki yerinə görə müəyyən olunur və kəsişməsində yerləşdiyi sətir və sütunun başlıqlarından ibarət olur. Öncə sütunun başlığı, sonra isə sətrin nömrəsi yazılır, məsələn, **A3, D6, AB19**.

Xanalar diapazonu onu əmələ gətirən xanaların birincisi və axırıncısının ünvanlarını göstərməklə verilir və ünvanlar biri-birindən iki nöqtə (**:**) ilə ayrılır. Məsələn, yuxarıdakı şəkildə seçdirilmiş xanalar diapazonunun ünvanı **A4:C8** olacaq.

Verilənlərin cədvəldə əsas saxlanma yeri xanalardır. Xanaya verilənləri daxil etmək üçün öncə həmin xananı seçdirmək lazımdır. Seçdirilmiş xana qalın tünd

çərçivəyə alınır. Xananı seçdirmək üçün istər siçandan, istərsə də klaviaturadan istifadə etmək olar.

Seçdirilmiş xanaya *aktiv xana* [active cell] deyilir. Aktiv xananın adı iş vərəqinin yuxarısında sol tərəfdə görünür. Əgər siz xanalar qrupunu seçdirmisinizsə, ilk seçdiyiniz xana aktiv xana kimi müəyyən olunur.

Elektron cədvəl obyektlərinin seçdirilmə üsulları

- Xananı seçdirmək üçün onu siçan vasitəsilə çıqqıldatmaq, yaxud ox klavişləri vasitəsilə kursoru həmin xanaya aparmaq lazımdır.
- Bütün sütunu seçdirmək üçün onun başlığını çıqqıldatmaq gərəkdir.
- Bütün sətri seçdirmək üçün onun başlığını çıqqıldatmaq gərəkdir.
- Xanalar diapazonunu bir neçə yolla seçdirmək mümkündür:
  - siçan vasitəsilə – siçanın sol düyməsini diapazonun başlanğıc xanasında basıb saxlamaqla son xanayadak hərəkət etdirməklə;
  - <Shift> klavişini basıb saxlamaqla kursorun idarə olunması klavişləri vasitəsilə;
  - aralarında iki nöqtə qoymaqla diapazonun başlanğıc və son xanalarının ünvanlarını klaviatura vasitəsilə yazmaqla.

Klaviaturadan daxil edilən verilənlər aktiv xanaya yazılır. Xanaya verilənləri daxil etmək üçün aşağıdakıları yerinə yetirin:

1. Bir xananı seçdirmək üçün ya göstəricini həmin xanaya tuşlayıb siçanın sol düyməsini çıqqıldadın, ya da ox klavişlərindən istifadə edin.
2. Ədədi (məsələn, 19, yaxud 12.3), mətni (məsələn, Şagirdlərin sayı), yaxud düsturu daxil edin.

Xanaya verilənləri daxil etdikdən sonra başqa xananı seçdirmək üçün siz müəyyən klavişləri, yaxud klavişlər kombinasiyasını basa bilərsiniz.

<Enter> . . . . . Cari xananın altındakı xananı seçdirir.
<Tab> . . . . . Cari xananın sağındakı xananı seçdirir.
<Shift+Enter> . . . . Cari xananın üstündəki xananı seçdirir.
<Shift+Tab> . . . . . Cari xananın solundakı xananı seçdirir.

Əgər siz qiyməti **A1** xanasına daxil edib, <Enter> klavişini bassanız, Microsoft Excel onun altındakı xananı, yəni **A2** xanasını seçdirəcək.

Əgər siz qiyməti **A2** xanasına daxil edib, <Tab> klavişini bassanız, Microsoft Excel onun sağındakı xananı, yəni **B2** xanasını seçdirəcək.

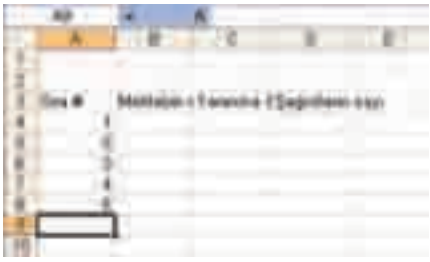
Çalışma

1.  Alətlər zolağındakı New Blank Document düyməsini çıqqıldadın. Book2 adlı yeni iş kitabı açılacaq.
2. **A3** xanasını seçdirin.
3. Klaviaturada **Sıra** yazın. Diqqət edin ki, sizin yığdığınız mətn xanada və düstur zolağında görünür.
4. <Tab> klavişini basın. Daxiletmə nöqtəsi üçüncü sətrin növbəti xanasına (**B3**) keçəcək. **Təhsil qurumu** yazın.
5. <Tab> klavişini basın və **Yaradıldığı il** yazın. Yenə <Tab> klavişini basın və **Şagirdlərin sayı** yazın. <Enter> klavişini basın. Daxiletmə nöqtəsi dördüncü sətrin birinci xanasına (**A4**) keçəcək.
6. **A3** xanasını çıqqıldadın. **Sıra** yazısı həm xanada, həm də düstur zolağında görünəcək. Siçanın göstəricisini düstur zolağında **Sıra** sözünün sonuna aparın və siçanın sol düyməsini çıqqıldadın.

7. Boşluq (Spacebar) klavişini basın və # yazın. <Enter> klavişini (yaxud yaşıl qeyd işarəsini) basın. Düstur zolağında və **A3** xanasında dəyişiklik olacaq.
8. **B3** xanasına keçin. Orada **Təhsil qurumu** yazısı görünür. **Məktəbin adı** yazın və <Enter> klavişini basın. Xananın içindəki yazı sizin yeni daxil etdiyiniz yazı ilə əvəzlənəcək.
9. **C3** xanasını çıqqıldadın. Orada **Yaradıldığı il** yazılıb. <F2> klavişini basın. Diqqət edin ki, daxiletmə nöqtəsi indi xanada mətnin sonuna keçəcək.
10. <Backspace> klavişindən istifadə edib mövcud mətni silin və **Yaranma ili** yazın. <Enter> klavişini basın. Xananın içindəki yazı sizin yeni daxil etdiyiniz yazı ilə əvəzlənəcək və altdakı **C4** xanası aktivləşəcək.

11. **A4** xanasına keçmək üçün iki dəfə <sola ox> klavişini basın. **1** yazıb, <Enter> klavişini basın. Altdakı **A5** xanası aktivləşəcək.
12. Hər nömrədən sonra <Enter> klavişini basmaqla aşağıdakı ədədləri yazın.

5
3
4
5



13. **A** sütununu seçdirmək üçün uyğun hərfi çıqqıldadın. Diqqət edin ki, sütunun birinci xanasını çıxmaqla qalan xanalar seçdiriləcək (tutqunlaşacaq), çünki birinci xana indi aktiv xanadır.
14. 5-ci sətri seçdirin. **2** yazın və diqqət edin ki, sizin yeni daxil etdiyiniz yazı **A5** xanasındakı yazını əvəzləyəcək.
15. İş vərəqinə **Məktəblər** adını verməklə yazıb saxlayın.

1. Elektron cədvəlin obyektlərini sadalayın.
2. Xanalar, sətirlər və sütunlar necə tanınır?
3. Xanalar diapazonunun ünvanı necə verilir?
4. Aktiv xana nədir?
5. Aşağıdakı nümunəyə uyğun olaraq sinfiniz haqqında cədvəl qurun.

	A	B	C	D
1	Gəncə şəhəri 19 nömrəli məktəbin 9A sinfi			
2				
3	Sıra #	Soyadı	Adı	Atasının adı
4	1	Abbasov	Sənan	Eldar
5	2	Abdullayeva	Aysel	İbrahim
6	3	Cavadzadə	Murad	Vaqif
7	4	Cəfərli	Orxan	Araz
8	5	Hüseynli	Sevinc	Yaşar
9	6			
10	7			
11	8			
12	9			
13	10			
14	11			

6. İstədiyiniz mövzuda elektron cədvəl yaradın.

3.3. ELEKTRON CƏDVƏL VERİLƏNLƏRİ. DÜSTURLAR

Cədvəl prosessorlarında müxtəlif tipli verilənlərlə – simvollarla (mətnlərlə), ədədlərlə, məntiqi və tarixi göstərən verilənlərlə işləmək imkanı vardır. Bu və ya digər xana üzərində hansı əməliyyatların aparıla bilməsi həmin xanadakı verilənin tipindən asılı olur. Tutaq ki, xanada 28051918 rəqəmləri yazılıb. Bəs cədvəl prosessoru onları necə qəbul edəcək? Əgər *mətn formatı* qoyulmuşsa, onda həmin rəqəmlər 2, 8, 0, 5, 1, 9, 1, 8 simvollarının ardıcıllığı kimi «başa düşüləcək». Əgər *ədədi format* qoyulmuşsa, onda elektron cədvəl xanada yazılanı ədəd kimi qəbul edəcək. Xanaya *tarix formatı* təyin olunmuşsa, xanadakı rəqəmlər 28 may 1918 tarixi kimi qəbul ediləcək.

Mətn verilənlər tipi. Mətn verilənləri müəyyən simvollar yığınıdır. Onun birinci simvolu hərf, dırnaq işarəsi, apostrof, yaxud boşluq simvoludursa, eləcə də simvolların içərisində həm rəqəmlər, həm də hərflər vardırsa, onda belə yazı mətn kimi qəbul edilir.

Simvol verilənləri üzərində əməliyyatlar mətn prosessorunun obyektləri üzərindəki əməliyyatlar kimidir.

Mətn (simvol) verilənlərinə örnəklər:

Dərs cədvəli
“236
9 A sinfi
001

Ədədi verilənlər tipi. Ədədi verilənlər onluq nöqtə ilə ayrıla bilən rəqəmlər ardıcıllığıdır. O, rəqəm, ədədin işarəsi (+ və ya –), yaxud onluq nöqtə ilə başlana bilər. Elektron cədvəldə ədədi verilənlər üzərində cürbəcür riyazi əməliyyatlar aparmaq olar.

Ədədi verilənlərə örnəklər:

232.5
–13.7
.546
+100

Unutmayın ki, cədvəl xanasında saxlanılan rəqəmlər ardıcıllığı dırnaq işarəsilə başlanırsa, ədəd kimi görünməsinə baxmayaraq, onlardan hesablamalarda istifadə etmək olmaz. İstənilən mətn verilənləri həmişə sıfır kimi qəbul olunur.

Xanada olan verilənlərin mətn, yaxud ədəd olduğunu belə müəyyənləşdirmək olar: əgər yığımdan sonra xanadakı qiyməti cədvəl prosessoru mətn kimi qəbul edirsə, onda daxiletmənin sonunda (<Enter> klavişi basıldıqdan sonra) onlar avtomatik olaraq xananın sol qırağına nəzərən düzləndirilir. Ədəd kimi qəbul edilən verilənlər isə daxiletmədən sonra xananın sağ qırağına nəzərən düzləndirilir.

Mətn	12.5
.0123	0.0123
\456	-456

Məntiqi verilənlər tipi. Məntiqi verilənlərdən məntiqi düsturlarda və funksiyalarda istifadə olunur. Bu tip verilənlər xanada belə əks olunur: əgər sıfırdan fərqli istənilən ədəd (tam, yaxud kəsr) daxil edilirsə, onda <Enter> klavişini basdıqdan sonra xanada True [doğru] qiyməti çıxacaq. Sıfır uyğun xanada False [yalan] kimi əks olunacaq.

Verilənlərin belə göstərilməsi məntiq cəbrində istifadə olunan *məntiqi dəyişən* anlayışı ilə bağlıdır.

Tarix verilənlər tipi. Bu verilən tipindən tarixə ədəd əlavə edilməsi, iki tarixin fərqinin hesablanması, yaxud tarixin geriyə və ya irəliyə yenidən sayılması kimi funksiyaların yerinə yetirilməsi zamanı istifadə olunur. Ədədlərin tarixə çevrilməsi verilmiş formatdan asılı olaraq avtomatik baş verir. Cədvəl prosessoru daxil edilən ədədləri tarix kimi bir neçə formada göstərməyə imkan verir. Məsələn,

28 aprel 2008

Aprel 2008

Aprel

28.04.2008

04.2008

28 aprel

Düsturlar. Elektron cədvəl, ilk növbədə, hesablamaları avtomatlaşdırmaq üçün nəzərdə tutulub. Bu məqsədlə cədvəlin xanasına *düsturlar* daxil edilir.

> İstənilən düsturun daxil edilməsi bərabərlik işarəsi (=) ilə başlanır. Bu işarə olmazsa, daxil edilən düstur mətn kimi qəbul ediləcək.

Düsturda ədədi verilənlər, əməl işarələri, müxtəlif funksiyalar, eləcə də cədvəl obyektlərinin ünvanları ola bilər. Xanaların ünvanları olan düsturları riyazi tənliklərlə müqayisə etmək olar: orada xanaların ünvanlarının yerinə dəyişənlərdən istifadə olunur.

Düsturlarda istifadə olunan ünvanlara *istinadlar* deyilir. İstinadlar elektron cədvəlin ixtiyari xanalarını bir-birilə əlaqələndirməyə və cədvəl verilənlərinin lazım olan emalını həyata keçirməyə imkan verir.

> İstinad düsturun yazılışında istifadə olunan obyektin (xananın, sətrin, sütunun, xanalar diapazonunun) ünvanıdır.

Düsturlar hesab və məntiq əməlləri ilə birləşdirilmiş *operandlardan* ibarətdir. Konkret qiymət, istinad, yaxud funksiya operand ola bilər.

Düsturlar iki cür olur: *cəbri və məntiqi.*

Cəbri düsturlar. Cəbri düsturlarda hesab əməllərindən («+» toplama, «–» çıxma, «*» vurma, «/» bölmə, «^» qüvvətə yüksəltmə) istifadə olunur. Düsturlarla hesablamalar zamanı riyaziyyatda hesab əməllərinin yerinə yetirilmə ardıcıllığına burada da əməl olunur: öncə qüvvətə yüksəltmə, sonra vurma və bölmə, nəhayət, toplama və çıxma əməlləri yerinə yetirilir. Eyni səviyyəli əməllər (məsələn, vurma və bölmə) soldan sağa yerinə yetirilir. Hesab əməllərinin yerinə yetirilmə ardıcıllığını dəyişmək üçün mötərizələrdən istifadə olunur. Mötərizənin içərisində olan operandlar üzərində əməllər birinci növbədə yerinə yetirilir.

Cəbri düsturlar vasitəsilə aparılmış hesablamaların nəticələri ədədlərdir. Hər dəfə düstura daxil olan operandları dəyişdirdikdə nəticə yenidən hesablanır və uyğun xanada əks olunur.

Məntiqi düsturlar. Məntiqi düstur şərtdən ibarət olur və onun doğru, yaxud yalan olması müəyyənləşdirilir. Doğru ifadəyə «true» («doğru», 1), yalan ifadəyə isə «false» («yalan», 0) qiyməti mənimsədilir.

Eynitipli düsturlar. Elektron cədvəllə işləyərkən çox zaman müəyyən diapazonda olan xanaları eyni struktura malik olan, ancaq dəyişənləri müxtəlif qiymətlər alan düsturlarla doldurmaq lazım gəlir. Bu o deməkdir ki, onlar yalnız istinadlara görə fərqlənir. Belə düsturlara *eynitipli düsturlar* deyilir.

> Eynitipli (oxşar) düsturlar elə düsturlardır ki, onlar eyni struktura (quruluşa) malikdir və yalnız konkret istinadlara görə fərqlənir.

Eynitipli düsturların daxil edilməsini asanlaşdırmaq və tezləşdirmək üçün belə bir üsuldan istifadə olunur: ***düstur yalnız bir xanaya daxil edilir, sonra isə o, başqa xanalara köçürülür.***

Eynitipli düsturlara nümunələr:

=A1+5	=A1*5	=A1+B3	=A1*B3	=(A1+B3)*D2
=A2+5	=B1*5	=A2+B4	=B1*C3	=(C3+D5)*F4
=A3+5	=C1*5	=A3+B5	=C1*D3	=(D4+E6)*G5
=A4+5	=D1*5	=A4+B6	=D1*E3	=(B4+C6)*E5

Mütləq, nisbi və qarışıq ünvanlama. Eynitipli düsturlarda ən müxtəlif istinadlardan istifadə oluna bilər. Elə eynitipli düsturlar ola bilir ki, bir düsturdan

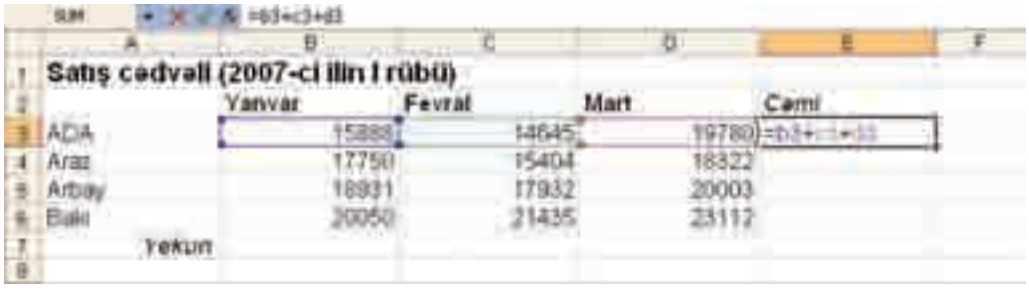
başqasına keçid zamanı istinadların müəyyən bir qismi qanunauyğun şəkildə dəyişilir, digər qismi isə bütün düsturlar üçün dəyişilməz qalır.

Düsturu cədvəlin başqa yerinə köçürərkən ona daxil olan istinadların avtomatik dəyişilməsi üsulunu müəyyənləşdirmək lazımdır. Bunun üçün nisbi, mütləq və qarışıq istinadlardan istifadə olunur.

- Düsturda *nisbi istinaddan* o halda istifadə olunur ki, o, köçürmə zamanı dəyişilməli olsun.
  Nisbi istinad adi formada yazılır, məsələn, **F3,** yaxud **E7**. Köçürmədən sonra düsturun köçürüldüyü bütün xanalarda həm sütunun hərfi, həm də sətrin nömrəsi köçürülməyə uyğun olaraq dəyişilir.
- Düsturda *mütləq istinaddan* o halda istifadə olunur ki, köçürmə zamanı onun hər iki hissəsi – sütunun hərfi və sətrin nömrəsi dəyişilməsin. Mütləq istinadda sütunun hərfinin və sətrin nömrəsinin qabağında $ simvolu qoyulur, məsələn, **F3,** yaxud **E7**. Köçürmə zamanı düstur bütün xanalarda dəyişilməz yerləşdiriləcək.
- Düsturda *qarışıq istinaddan* o halda istifadə olunur ki, köçürmə zamanı onun iki hissəsindən biri – ya sütunun hərfi, ya da sətrin nömrəsi dəyişilsin. Bu zaman $ simvolu istinadın yalnız dəyişilməz qalacaq hissəsinin qabağına qoyulur. Məsələn, **$F3,** yaxud **E$7**.

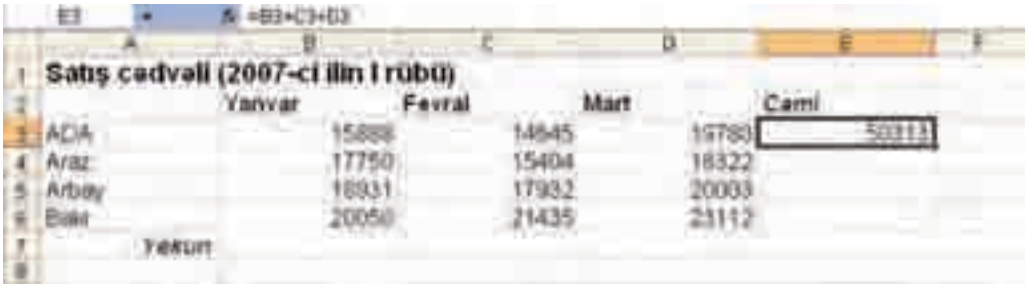
Çalışma

1. Cədvəl prosessorunu başladın.
2. Yeni sənəd (iş kitabı) yaradın.
3. Cədvəli nümunəyə uyğun doldurun.



4. **E3** xanasını seçdirin. Klaviaturadan **=b3+c3+d3** daxil edin. Diqqət edin ki, yazdığınız düstur həm xanada, həm də düstur zolağında (Formula Bar) görünür.

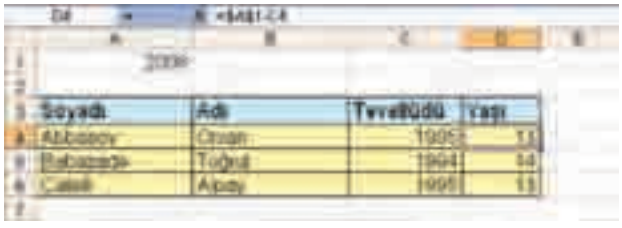
5. <Enter> klavişini basın, yaxud düstur zolağındakı Enter düyməsini çıqqıldadın. Düsturun nəticəsi, 50313 ədədi **E3** xanasında görünəcək.
6. **E3** xanasını çıqqıldadın. Diqqət edin ki, düstur hələ də düstur zolağında görünür.



7. **E4** xanasını çıqqıldadın. Klaviaturadan = daxil edin. Sonra **B4** xanasını çıqqıldadın. Diqqət edin ki, **B4** xanasına istinad = işarəsindən sonra həm düstur zolağında, həm də **E4** xanasında görünür.
8. Klaviaturadan + daxil edin və sonra **C4** xanasını çıqqıldadın. + daxil edin və **D4** xanasını çıqqıldadın. Həm xanada, həm də düstur zolağında **=B4+C4+D4** düsturu görünəcək.
9. <Enter> klavişini basın, yaxud düstur zolağındakı Enter düyməsini çıqqıldadın. Düsturun nəticəsi, 51476 ədədi **E4** xanasında görünəcək.
10. **B4** xanasını qoşa çıqqıldadın. Mövcud məbləği **16750** edin. Sonra <Enter> klavişini basın. Diqqət edin ki, **E4** xanasındakı nəticə dəyişərək, 50476 olacaq.
11. **E5** xanasını seçdirin. Klaviaturadan **=b4+c4+d4** düsturunu daxil edin və sonra <Enter> klavişini basın. **E5** xanasında 50476 nəticəsi görünəcək. Diqqət edin ki, görünən məbləğ **E4** xanasındakı ilə eynidir, çünki hər iki xana eyni bir düsturla hesablanır.
12. **E5** xanasını qoşa çıqqıldadın və düsturdakı **B4** istinadını **b5** ilə əvəz edin. <Enter> klavişini basın.
13. **E5** xanasını çıqqıldadın, <F2> klavişini basın və düsturdakı ikinci istinadı **C4**-dən **c5**-ə dəyişin.
14. Düstur zolağını çıqqıldadıb üçüncü istinadı (**D4**) **d5** ilə əvəzləyin. Sonra <Enter> klavişini basın. **E5** xanasında 56866 nəticəsi əks olunacaq.
15. **E6** xanasını çıqqıldadın və klaviaturadan = daxil edin. Sonra **B6** xanasını çıqqıldadın, + daxil edin, **C6** xanasını çıqqıldadın, + daxil edin və **D6** xanasını çıqqıldadın. <Enter> klavişini basın və **E6** xanasında görünən ədədə baxın. Siz 64597 nəticəsini görəcəksiniz.

1. **Cədvəl prosessorunda istifadə olunan verilənlər tiplərinin adlarını və onların özəlliklərini deyin.**
2. **Elektron cədvəldə istinad nədir və o, ünvandan nə ilə fərqlənir?**
3. **Düsturlarda nisbi ünvanlama nədir? Misallar göstərin.**
4. **Düsturlarda mütləq ünvanlama nədir və onlar necə işarə olunur? Misallar göstərin.**
5. **Düsturların köçürülmə qaydasını söyləyin.**
6. **A2 xanasına x dəyişəninin, B2 xanasına y dəyişəninin, C2 xanasına z dəyişəninin qiymətilərini daxil edin. (x+y)*2+3*z riyazi ifadəsinin qiymətini elektron cədvəlin D2 xanasında alın. Bunun üçün D2 xanasında həmin düsturu yazın. x, y, z dəyişənlərinə müxtəlif qiymətlər verməklə D2 xanasında qiymətin dəyişilməsini izləyin.**

7. **Aşağıdakı nümunəyə uyğun cədvəl yaradın. Düsturdan istifadə etməklə şagirdlərin yaşını hesablayın.**



Soyadı	Adı	TəvəllüdG	Yaşı
Abbasov	Orxan	1995	11
Babazadə	Toğrul	1994	12
Cabbar	Aydın	1995	11

8. **A2 xanasında avtomobilin sürətini, B2 xanasında onun yola sərf etdiyi zamanı göstərin. C2 xanasında düstur vasitəsilə avtomobilin getdiyi yolu hesablayın ($s=v*t$). B2 xanasının qiymətini 2, 3, 4 dəfə artırın. C2 xanasının qiyməti necə dəyişir?**
9. **Aşağıdakı elektron cədvəli yaradın. B3, B7, C3, D7 xanalarında ədədlər yerləşmişdir. C3 xanasına =4*B3, D3 xanasına =B3*B3, E7 xanasına =B7+C7+D7, F7 xanasına =B7*C7/2 düsturlarını yazın. B3, B7, C7, D7 xanalarındakı ədədləri dəyişdirin və düsturlarla hesablanan qiymətlərin dəyişilməsini izləyin.**

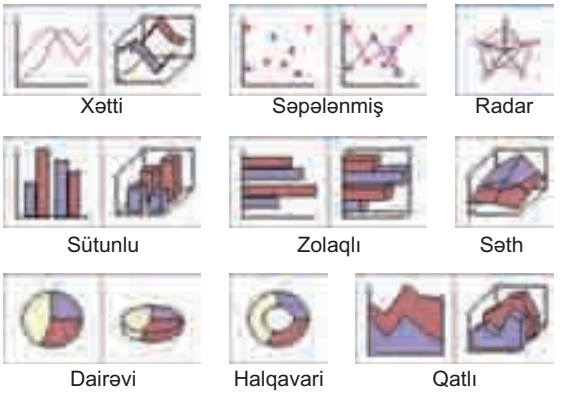
10. **Alman alimi Q.Farengeyt 1724-cü ildə onun adı ilə bağlı temperatur şkalasını təklif etmişdir. Farengeyt temperatur şkalası ilə Selsi temperatur şkalası arasında münasibət belədir: $1^{\circ}C=\frac{5}{9}(1^{\circ}F-32)$. Farengeyt şkalası ilə ölçülmüş temperaturu Selsi şkalasına uyğun temperatura çevirmək üçün cədvəl qurun.**

3.4. DİAQRAMIN YARADILMASI VƏ REDAKTƏSİ

Elektron cədvəlin başqa bir obyekti diaqramdır. *Diaqram* verilənləri qrafik formada təsvir etmək üçün nəzərdə tutulub. Bir sətirdə, yaxud bir sütunda yerləşmiş verilənlərə *sıra* deyilir. Diaqram qurarkən öncə sıraları seçdirmək, sonra isə diaqramın tipini seçmək lazımdır. Hər bir diaqram bir neçə parametrlə xarakterizə olunur: *ad, tip, sahə, yerləşdirmə.*

Ad. Hər bir diaqrama ad verilir və o, həmin adla elektron cədvəlin tərkibinə daxil olur.

Tip. Cədvəl prosessorunda çeşidli diaqramlar qurmaq mümkündür. Aşağıda əsas diaqram tipləri göstərilib.



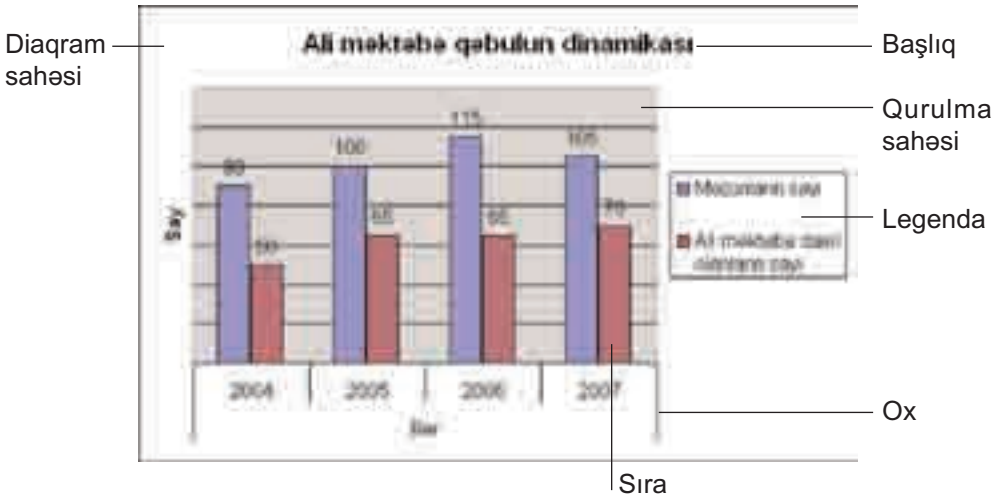
Bunlardan bir neçəsi ilə tanış olaq.

- *Xətti diaqramlar* [line chart] sizə müxtəlif fənlərdən tanışdır. Onlara *qrafik* də deyilir. Bir sxemdə bir neçə qrafik yerləşdirmək mümkündür ki, onların da hər biri öz verilənlər sırasına uyğun olacaq.
- *Sütunlu diaqram* [column chart], yaxud *histoqram* da bir neçə verilənlər sırası üçün qurula bilər. Hər bir sütunun hündürlüyü uyğun xanadakı qiymətdən asılıdır.
- *Səth diaqramı* [surface chart] yalnız bir neçə sıra üçün qurulur və çoxqatlı rəngarəng səthlər qrupundan ibarət olur. Hər bir səth bir verilənlər sırasına uyğundur.
- *Dairəvi diaqramdan* [pie chart] bir sırada yerləşmiş qiymətləri əks etdirmək üçün istifadə olunur. Belə diaqramda hər bir bölmə (sektor) uyğun qiymətin bütün qiymətlər cəmində nisbi payını (faizlə ifadə olunmuş) əks etdirir.

Sahə. Diaqramın çəkiləcəyi yeri əhatə edir.

Yerləşdirmə. Diaqram cədvəlin olduğu vərəqdə və ya ayrıca vərəqdə də yerləşdirilə bilər.

Diaqramın obyektləri. Diaqram özü mürəkkəb obyekt olub, aşağıdakı elementar obyektlərdən təşkil olunur: *sıra, ox, başlıq, legenda, qurulma sahəsi.*



Sıra. Diaqram istər bir sıraya, istərsə də bir neçə sıraya görə qurula bilər. Seçdirilmiş xanalar diapazonu üçün diaqramın qurulması bir neçə verilənlər sırası üzrə aparılır. Bu halda sıra kimi seçdirilmiş diapazonun uyğun sətri, yaxud sütunu götürülür.

Ox. Diaqramın hər bir oxu aşağıdakı parametrlərlə xarakterizə olunur: *görünüş, şkala, şrift, ədəd.*

- *Görünüş* oxun ekranda xarici görünüşünü əks etdirir.
- *Şkala* minimal və maksimal qiymətləri, əsas və aralıq bölgülərin qiymətini, başqa oxlarla kəsişmə nöqtəsini müəyyən edir.
- *Ədəd* diapazonda olan verilənlərin tipinə uyğun olaraq şkalanın formatını müəyyən edir.

Başlıq. Başlığı istifadəçi müəyyənləşdirir və o, adətən, diaqramın üstündə yerləşdirilir.

Legenda. Diaqrama legenda – sıraların adlarının, başqa sözlə, dəyişənlərin işarələmələrinin siyahısını da əlavə etmək olar.

Qurulma sahəsi. Oxlarla əhatə olunmuş bu sahə verilənlər sırasını yerləşdirmək üçün nəzərdə tutulub. Nəticələrin təhlilini əlverişli etmək üçün qurulma sahəsinə tor da əlavə etmək olar.

Diaqram obyektini seçdirmək üçün qoşa çıqqıltı vasitəsilə diaqramı seçmək, sonra isə lazım olan obyekti çıqqıldatmaq lazımdır.

Belə bir cədvəl yaradaq:

	A	B
1	2007-ci ildə dünyada yaşayan azərbaycanlıların sayı, milyon nəfər	
2	İran	30,1
3	Azərbaycan	8,6
4	Türkiyə	2,5
5	Rusiya	2,3
6	MDB ölkələri	2,3
7	ABŞ	1,2
8	Avropa ölkələri	2,6

 Bu cədvələ uyğun diaqram qurmaq üçün öncə qrafik formada göstəriləsi xanaları (A2:B8) seçdirmək, sonra isə standart alətlər zolağındakı Chart Wizard düyməsini çıqqıldatmaq, yaxud Insert⇨Chart menyu komandasını seçmək lazımdır. Nəticədə uyğun dialoq boksu açılacaq.

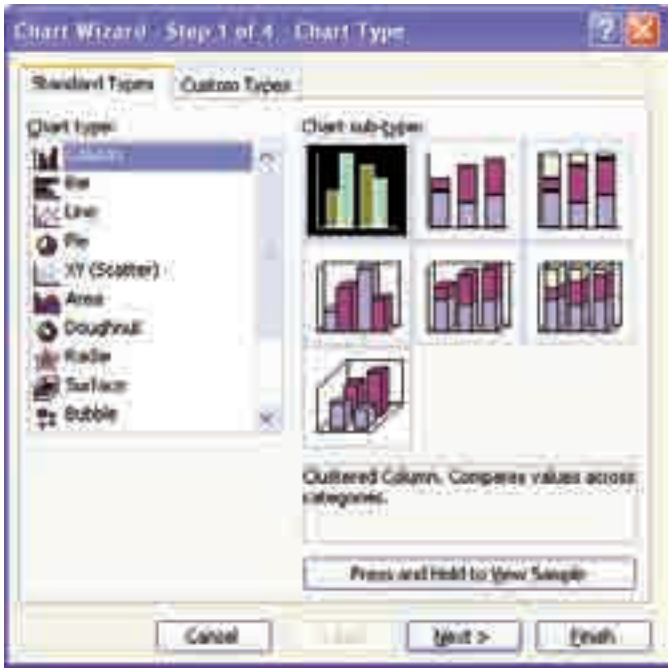


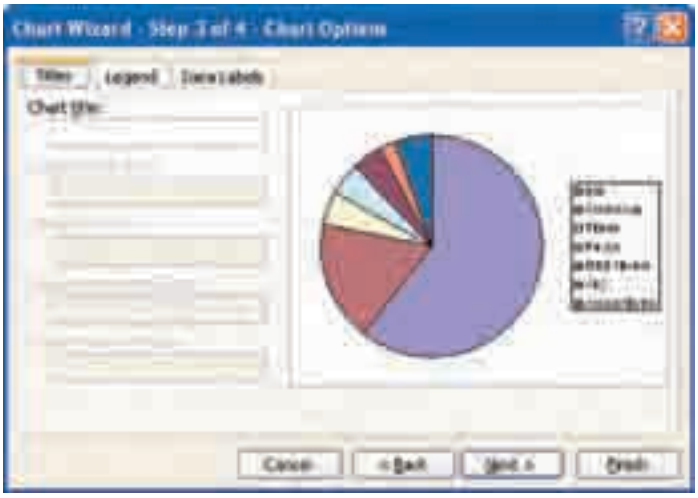
Chart Wizard dialoq boksunun birinci səhifəsində yaratmaq istədiyiniz diaqramın tipi və növü seçilir. Diaqramın tipini seçdikdə Chart sub-type sahəsində həmin tipə uyğun diaqram növləri görünür və onlardan biri susqunluqla seçdirilmiş olur. Dairəvi (Pie) diaqram tipini seçək və növbəti səhifəyə keçmək üçün Next düyməsini çıqqıldadaq.

Bu səhifədə əmin olmaq lazımdır ki, Data range sahəsində diaqramın yaradılması üçün istifadə olunan xanalara istinadlar göstərilib. Sonra Next düyməsini çıqqıldadaq.

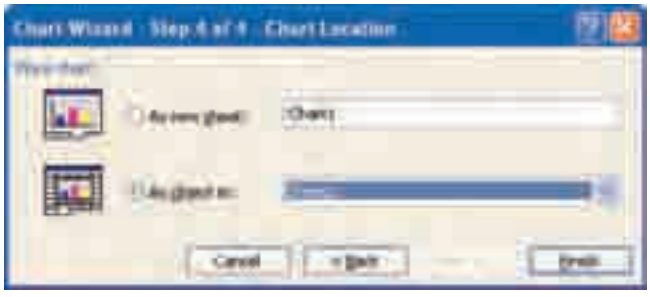
Bu səhifədə diaqramın xarici görünüşü müəyyənləşdirilir. Məsələn, Chart title sahəsində diaqramın başlığı **(2007-ci ildə dünyada yaşayan azərbaycanlıların sayı, mln.)** göstərilir.

Əgər diaqramda verilənlərin qiymətlərinin əks olunmasını istəyiriksə, Data Labels səhifəsində Value yoxlama boksunu qeyd etmək lazımdır. Diaqramın xarici görünüşü ilə işi bitirdikdən sonra yenə də Next düyməsini çıqqıldadırıq. Chart Wizard dialoq boksunun sonuncu səhifəsi açılır.

Bu səhifədə diaqramın yeni vərəqdə (As new sheet), yaxud mövcud iş vərəqində (As object in) yaradılması müəyyənləşdirilir. Finish düyməsini çıqqıldatdıqda diaqram seçilmiş yerdə görünəcək.



Başqa bir misala baxaq. Tutaq ki, $y = x^2 - 7x + 10$ funksiyasının qrafikini [-8; 8] parçasında qurmaq lazımdır. Bununçün öncə funksiyanın qiymətlərindən ibarət aşağıdakı cədvəli yaradaq. Arqumentin dəyişmə addımını 1-ə bərabər götürək.



Diaqramı (qrafiki, **ş.3.1.**) qurmaq üçün cədvəlin **A4:B20** xanalar diapazonunu seçdirək və Chart Wizard dialoq boksunu çağıraq. Diaqramın tipi olaraq səpələnmiş tipini, növü olaraq isə

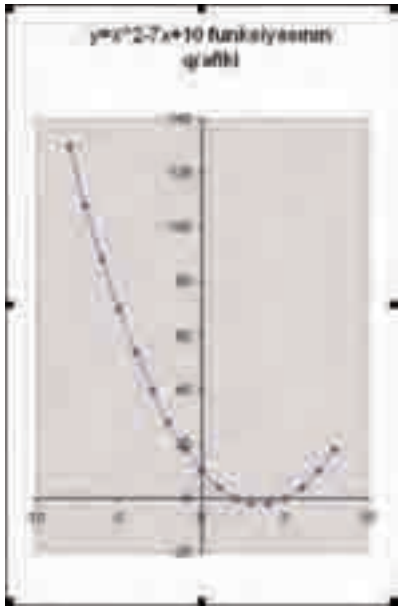
 növünü, yaxud  növünü seçək.

Sonra Next düyməsini çıqqıldadıb, növbəti səhifəyə keçək və orada diaqramın başlığını göstərək. GridLines bölümündə bütün yoxlama bokslarındakı

qeydləri silək və Finish düyməsini çıqqıldadaq. Nəticədə aşağıdakı şəkildə göstərilən qrafik (**ş.3.2.**) alınacaq.

	A	B
1	x0	-8
2	addım	1
3	x	y=x^2 - 7x + 10
4	-8	130
5	-7	108
6	-6	88
7	-5	70
8	-4	54
9	-3	40
10	-2	28
11	-1	18
12	0	10
13	1	4
14	2	0
15	3	-2
16	4	-2
17	5	0
18	6	4
19	7	10
20	8	18

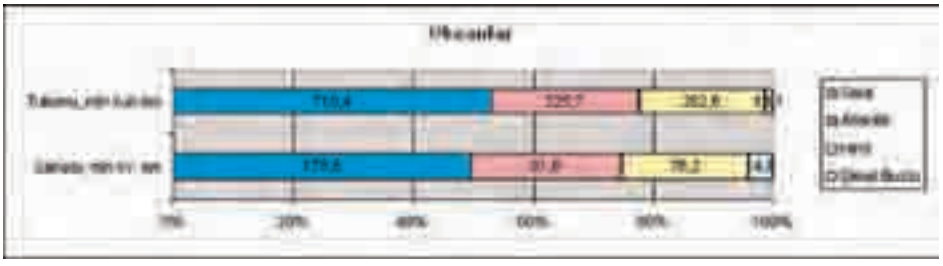
ş.3.1.



ş.3.2.

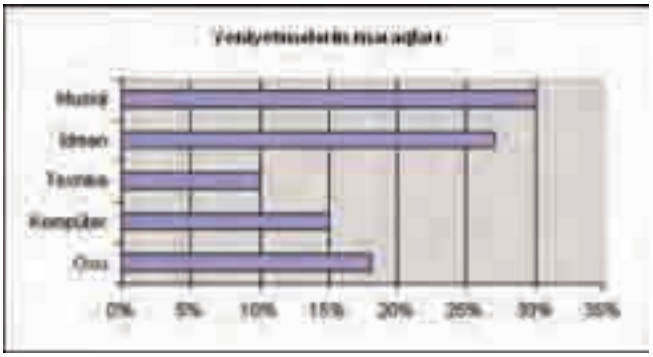
1. Diaqramlar hansı elementar obyektlərdən ibarətdir?
2. Cədvəl prosessorundakı əsas diaqram tiplərinin adlarını deyin.
3. İstənilən diaqramı xarakterizə edən parametrlər hansılardır?
4. Aşağıdakı cədvəli və ona uyğun diaqramı qurun.

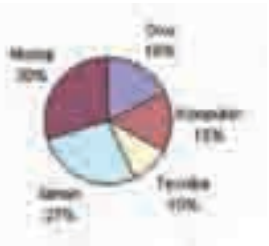
	A	B	C
1		Ölkələr	
2	Adı	Sahəsi, min kv. km	Tullumu, min kub km
3	Sakit	178,6	710,4
4	Atlantik	91,6	329,7
5	Hind	76,2	282,6
6	Şimal Buzlu	14,8	18,1
7			



5. Diaqramlara baxıb aşağıdakı suallara cavab verin:

- **Diaqramda hansı informasiya təqdim olunub? Bunu necə təyin etdiniz?**
- **Bu informasiyanı cədvəl şəklində necə vermək olar?**
- **Yeniyetmələri daha çox nə cəlb edir?**





3.5. CƏDVƏL SƏNƏDİNİN FORMATLANMASI

Cədvəl sənədinin formatlanması dedikdə, istər sənədin özünün, istərsə də onun obyektlərinin təqdimolunma formasının dəyişdirilməsi ilə bağlı bir sıra əməliyyatlar başa düşülür. Mətn prosessorunda qəbul olunmuş adi üsullarla yanaşı, elektron cədvəl obyektlərinin formatlanması üçün özəl üsullar da vardır:

- xanalarda verilənlər müxtəlif formatlarda göstərilə bilər;
- verilənlərin saxlandığı sütunun eni, yaxud sətrin hündürlüyü dəyişdirilə bilər;
- elektron cədvəlin istənilən obyekti çərçivəyə alına, yaxud xüsusi naxışla seçdirilə bilər.

Cədvəl sənədinin istənilən obyektinin formatlanması menyunun Format bölümünün komandalarının köməyilə yerinə yetirilir. İndisə ayrı-ayrılıqda xananın, sətrin və sütunun formatlanması ilə tanış olaq.

Xananın formatı aşağıdakı parametrlərlə xarakterizə olunur: *ədəd, düzləndirmə, şrift, sərhəd, görünüş, qoruma*. *Ədəd,* xanada saxlanılan verilənlərin tipini və həmin ədədin təqdimolunma formatını müəyyənləşdirir. *Düzləndirmə* və *şriftdən* başqa proqram mühitlərində olduğu kimi istifadə edilir. *Sərhəd,* xananın xarici çərçivəsini (tipini, qalınlığını, xəttin ştrixini) müəyyən edir. *Görünüş,* xananın fonunun rəngini və naxışını təyin edir. *Qoruma,* verilən-

lərin xanada qorunma səviyyəsini müəyyənləşdirir. Məsələn, xananı elə qorumaq olar ki, oradakı verilənləri dəyişmək mümkün olmasın, yaxud düstur gizlədilsin.

Sətrin formatı sətrin hündürlüyünü tənzimləməyə və onun cədvəldə əks olunmasını idarə etməyə imkan verir. Sətrin hündürlüyü ya avtomatik olaraq, ya da «əl ilə» müəyyən olunur. Avtomatik tənzimləmə zamanı sətrin hündürlüyü üçün elə qiymət seçilir ki, sətirdəki bütün verilənlər normal görünsün.

Sütunun formatı sütunun enini tənzimləməyə və onun cədvəldə əks olunmasını idarə etməyə imkan verir. Sütunun enini həm avtomatik olaraq, həm də «əl ilə» tənzimləmək olar. Avtomatik tənzimləmə zamanı sütunun eni üçün elə qiymət seçilir ki, sütundakı bütün verilənlər bir sətrə yerləşsin.

Sətir və sütunların əks etdirilməsi. Cədvəldəki istənilən sətri, yaxud sütunu gizlətmək olar. Bu o zaman gərəkli olur ki, həmin sətirlərdən və sütunlardan aralıq hesablamaların nəticələrini saxlamaq üçün istifadə olunur. Sonradan gizli sətir və sütunları ekranda göstərmək də olar.

Verilənlərin formatı. Ədədi verilənləri göstərmək üçün müxtəlif formatlardan istifadə olunur: ümumi, ədədi, faiz, pul, eksponensial və s.

- *Ümumi formatdan* susqunluqla istifadə olunur və həmin formatda daxil edilən istənilən verilənlər (mətnlər, ədədlər, tarixlər və s.) avtomatik olaraq tanınır və formatlanır.
- *Ədədi formatda* istifadəçi xanalardakı ədədlərin onluq işarələrinin sayını istədiyi kimi verə bilər. Məsələn, onluq işarələrin sayı üç olan format seçilibsə, xanaya daxil edilən 19 ədədi 19.000 kimi, 0.12345 ədədi isə 0.123 kimi yazılacaq.
- *Faiz formatı* ədədi verilənləri % işarəsilə faiz formasında göstərməyə imkan verir. Məsələn, dəqiqlik bir onluq işarəyə qədər qoyulubsa, 0.257 ədədini xanaya daxil etdikdə 25.7% qiyməti, 257 ədədini daxil etdikdə isə 25700.0% qiyməti əks olunacaq.
- *Pul formatı* ədədləri elə əks etdirməyə imkan verir ki, mərtəbələr üç-üç qruplaşdırılaraq boşluq simvolu ilə ayrılır, sonuncu onluq işarədən sonra isə pul vahidi göstərilir.
- *Eksponensial (elmi) formatda* verilən ədədlər iki hissədən ibarət olur: mantissa və ədədin tərtibindən. Məsələn, əgər dəqiqlik onluq nöqtədən sonra iki işarədirsə, onda 12345 ədədi eksponensial formatda 1.23E+04 şəklində olacaq (burada 1.23 mantissa, E+04 yazısı isə 10-un 4-cü qüvvətini bildirir).

Sətrin hündürlüyünün və sütunun eninin dəyişdirilməsi. Daxil edilən ədədin uzunluğu xananın (sütunun) enindən böyükdürsə, xanada ədədin yerinə «diyez» işarələri (######) görünəcək. Əgər daxil edilən ədəd mətndirsə, onda ekranda mətnin yalnız bir hissəsi (xananın eninə yerləşən) görünəcək. Xanada görünən simvolların sayı təkcə xananın enindən deyil, həm də seçilmiş şriftdən və onun ölçüsündən asılı olur.

Bu sadalanan hallar baş verdikdə sütunun enini artırmaq zərurəti yaranır. Bunu iki yolla etmək olar:

- Format⇨Column⇨AutoFit⇨Selection menyu komandasını seçmək, yaxud Format⇨Column⇨Width komandasını seçmək və açılan dialoq boksunda lazım olan qiyməti göstərmək;
- Siçanın göstəricisini iki sütunun adlarının arasında yerləşdirmək (göstərici formasını dəyişəcək) və onun sol düyməsini basılı saxlamaqla sütunun sərhədini lazım olan istiqamətə hərəkət etdirmək.

Sətrin hündürlüyünün dəyişdirilməsi də eyni qaydada yerinə yetirilir.

Excel cədvəl proqramında obyektlərin fonunu rəngləmək, yaxud fona hər hansı naxış salmaq imkanı da vardır.

Çalışma – 1

1. Cədvəl prosessorunu başladın və yuxarıdakı nümunəyə uyğun olaraq **Məktəblər.xls** faylını yaradın.
2. **B4**-dən **B8**-dək xanaları doldurun.
3. Siçanın göstəricisini **B** sütununun başlığının sağ tərəfindəki sərhəd xəttinin üzərinə aparın. Göstərici ikibaşlı ox şəklini aldıqda sərhəd xəttini tutub sağ tərəfə dartmaqla sütunu genişləndirin.
4. **B** sütununu seçdirmək üçün onun başlığını çıqqıldadın.
5. Format menyusunu açın və Column, sonra isə AutoFit Selection bəndini seçin. Excel avtomatik olaraq sütunu elə genişləndirəcək ki, həmin sütunun xanalarındakı yazılar tam görünsün.
6. **3** sətrini seçdirmək üçün onun başlığını çıqqıldadın.
7. Format menyusunu açın və Row, sonra isə Height bəndini seçin. Açılan dialoq boksunda hündürlüyü **25** edin və OK düyməsini çıqqıldadın.
8. **3** sətrinin o biri sətirlərdən iki dəfə hündür olduğunu görəcəksiniz.
9. İş vərəqinin hər hansı yerini çıqqıldatmaqla sətrin seçdirməsini ləğv edin.

Dəyişiklikləri yazıb saxlayın və faylı qapadın.

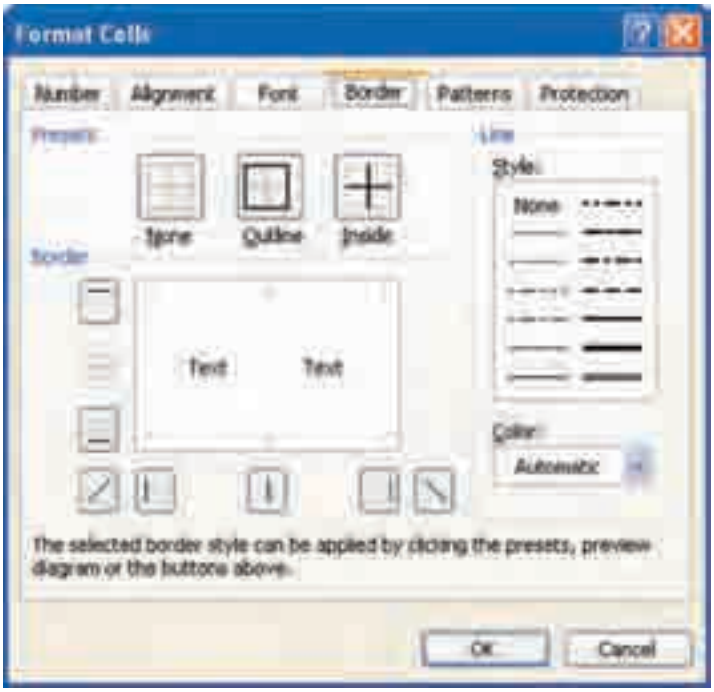
.

Çalışma – 2

1. Excel proqramında aşağıdakı cədvələ uyğun öz dərs cədvəlinizi yaradın.

	A	B	C	D	E	F	G	H
1	DƏRS CƏDVƏLİ							
2								
3		bazar ertəsi	çərşənbə axşamı	çərşənbə	cümə axşamı	cümə	şənbə	
4	1	cəbr	ana dili	cəbr	tarix	cəbr	həndəsə	
5	2	ədəbiyyat	cəbr	tarix	konstitusiya	fizika	biologiya	
6	3	coğrafiya	həndəsə	ana dili	həndəsə	ana dili	xarici dil	
7	4	biologiya	ədəbiyyat	kimya	idman	xarici dil	kimya	
8	5	rəsmixett	tarix	informatika	iqtisadiyyat	tarix	idman	
9	6	xarici dil	fizika	fizika		rus dili		
10								

2. Cədvəlin adını (**DƏRS CƏDVƏLİ**) onun üstündə ortadan yerləşdirmək üçün **A1:G1** xanalar diapazonunu seçdirin. Format⇨Cells menyu komandasını seçin.
3. Açılan dialoq boksunda Alignment bölməsinə keçin. Merge cells (Xanaların birləşdirilməsi) parametrini qeyd etdikdə həmin xanalar birləşərək tək **A1** xanasını əmələ gətirəcək.
4. Xananın içində yazıları ortadan yerləşdirmək (mərkəzə düzləndirmək) üçün Text alignment bölümündəki Horizontal siyahısından Center sətrini seçin.
5. Cədvəldə olan sütunların və sətirlərin ölçülərini tənzimləyin.
6. **B3:G3** xanalar diapazonunu seçdirin. Format Cells dialoq boksunun Patterns bölməsinə keçin. Oradan istədiyiniz rəngi (məsələn, mavi) seçin. OK düyməsini çıqqıldatdıqda seçdirdiyiniz xanalar həmin rəngə boyanacaq və xanalar arasındakı xətlər itəcək.
7. Xanaların ətrafında çərçivələr qoymaq üçün həmin xanaları seçdirin və dialoq boksunun Borders bölməsinə keçin.



Outline-ni seçərsinizsə, seçdirdiyiniz sahənin yalnız qıraq (kontur) xətləri çəkiləcək, daxili xətlər isə göstərilməyəcək. Inside-ni seçərsinizsə, yalnız daxili xətlər görünəcəkdir. Həm daxili, həm də kontur xətlərinin olmasını istəyirsinizsə, hər iki halı seçin.

8. Cədvəlin **B4:G9** xanalarını sarı rənglə boyayın.

DƏRS CƏDVƏLİ

	bazar ertəsi	çərşənbə axşamı	çərşənbə	cümə axşamı	cümə	şənbə
1	cəbr	ana dili	cəbr	tarix	cəbr	həndəsə
2	ədəbiyyat	cəbr	tarix	konstitusiya	fizika	biologiya
3	coğrafiya	həndəsə	ana dili	həndəsə	ana dili	xarici dil
4	biologiya	ədəbiyyat	kimya	idman	xarici dil	kimya
5	təsviri	tarix	informatika	iqtisadiyyat	tarix	idman
6	xarici dil	fizika	fizika		rus dili	

1. Sənədin formatlanması nə deməkdir?
2. Xananın formatlanması hansı parametrlərlə xarakterizə olunur?
3. Formatlanmadan istifadə edib aşağıdakı elektron cədvəli yaradın.

QİYMƏT CƏDVƏLİ

	cəbr	həndəsə	ana dili	ədəbiyyat	informatika	biologiya	coğrafiya	fizika	tarix	kimya	təsviri
1 yarımil	5	4	4	5	5	4	4	5	5	4	5
2 yarımil	5	5	5	5	5	4	4	4	5	4	5
İllik	5	5	5	5	5	4	4	4	5	4	5

4

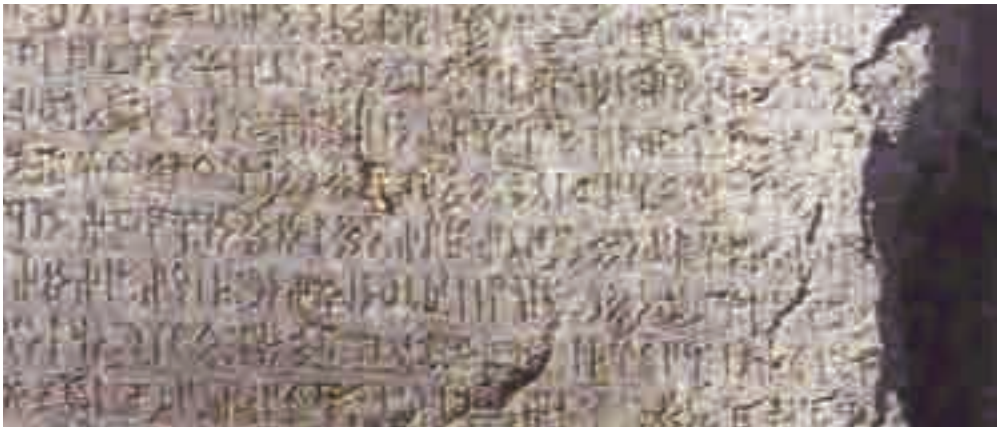
İNFORMASİYA CƏMİYYƏTİ

4.1. CƏMİYYƏTİN İNFORMASİYALAŞDIRILMASI

Bəşəriyyət yarandığı gündən əvvəlcə maddələrə, sonra enerjiyə və nəhayət, informasiyaya sahib olmaq uğrunda çarpışmışdır. Sivilizasiyanın ilk dövrlərində insana elementar bilik və bacarıqlar kifayət edirdisə, informasiyanın tədricən çoxalması nəticəsində insan şəxsi biliklərinin az olmasını hiss etməyə başladı. İnformasiyanı düzgün emal etmək və lazımi qərarlar qəbul etmək üçün insandan malik olduğu bilik və təcrübələri ümumiləşdirmək tələb olunurdu. Ona görə də o, müxtəlif qurğular düzəltməyə başladı. İnformasiyanın emalı üçün nəzərdə tutulan üsul və vasitələr meydana gəldi və onlar cəmiyyətdə ciddi dəyişikliklərə – informasiya inqilablarına gətirib çıxardı.

Bəşəriyyət, onun inkişafına daha çox təsir edən və *informasiya inqilabı* adlanan dörd mərhələdən keçmişdir.

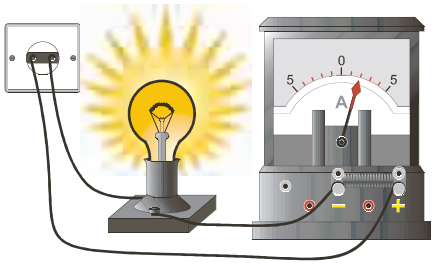
Birinci mərhələdə yazının meydana gəlməsi nəticəsində bilikləri saxlayaraq, gələcək nəsillərə ötürmək imkanı yarandı.



İkinci mərhələ (XVI əsrin ortaları) kitab çapının ixtirası ilə bağlıdır. Bu inqilab nəticəsində informasiyanın saxlanılmasının tamamilə yeni bir üsulu yarandı. İnsan informasiyanı saxlamaq, sistemləşdirmək və yaymaq üçün yeni bir vasitə əldə etdi.

Bu vasitə mənəvi və mədəni dəyərləri böyük kütlələrə çatdırmağa imkanı verdi.

Üçüncü mərhələ (XIX əsrin sonu) elektrik cərəyanının kəşfi ilə bağlıdır. Teleqraf, telefon və radio yarandı. Onlar informasiyanı ixtiyari həcmdə və yüksək sürətlə ötürməyə və qəbul edib saxlamağa imkan verdi. İnformasiya-kommunikasiya vasitələri yarandı.



Dördüncü mərhələdə (XX əsrin 70-ci illəri) mikroprosessor texnologiyası ixtira olundu və fərdi kompüterlər meydana çıxdı.

İnformasiyanın elektrik və mexaniki vasitələri keçmişdə qaldı. Onları elektron vasitələr əvəz etdi. Bu vasitələr maşın və cihazları daha kiçik ölçüdə istehsal etməyə və proqramla idarə olunan qurğular yaratmağa imkan verdi. Dördüncü inqilabın baş verməsinin əsas səbəbi ötən əsrin 40-cı illərində elektron hesablama maşınlarının yaranması oldu.

Dördüncü informasiya inqilabı bəşəriyyətin *sənaye cəmiyyətindən informasiya cəmiyyətinə* keçməsinə təkan verdi. Bu, bir vaxt bəşəriyyətin aqrar cəmiyyətdən sənaye cəmiyyətinə keçidinə bənzəyirdi.

> İnformasiya cəmiyyəti elə bir cəmiyyətdir ki, orada cəmiyyət üzvlərinin əksəriyyəti informasiyanın istehsalı, saxlanılması, emalı və onun istifadəsilə məşğul olur.

İnformasiya cəmiyyətinin bəzi xarakterik cəhətləri:

1. *İnformasiyanın həcmi artdıqca onun emalı və saxlanması üçün insan xüsusi texniki vasitələrdən istifadə etməli olacaqdır.*

   İnformasiya cəmiyyətində insan və kollektivlər hər hansı bir qərarı qəbul edərkən müəyyən informasiya toplamalı, onu emal və təhlil etməlidir. İnformasiyanın həcmi o dərəcədə artmışdır ki, insan özü onu emal etməyə qadir deyil. O, bu işə xüsusi texniki vasitələri cəlb edir.

2. *Kompüterlərdən istifadə qaçılmaz olacaqdır.* İnformasiya cəmiyyətində kompüterlərdən istifadə olunması labüddür. Bu, etibatlı informasiya mənbələrindən istifadə etməyə imkan verir, faydasız işi azaldır, optimal qərarların qəbul olunmasını sürətləndirir və informasiyanın emalını avtomatlaşdırır.

3. *Cəmiyyətin hərəkətverici qüvvəsi informasiya məhsulunun istehsalı olacaqdır.* XX əsrin ikinci yarısında insanların müəyyən bir qisminin maddi məhsul istehsalından informasiya sahəsinə keçməsi baş verdi. Bilavasitə maddi sərvətlər istehsal etməyən əhalinin yeni sosial təbəqəsi yarandı. Bu insanlar (müəllimlər, bank işçiləri, proqramçılar və s.) informasiyanı emal etməklə

məşğul olurdular. Yeni cəmiyyətdə isə maddi nemətlər daha çox «informasiya tutumlu» olacaq. Onun dəyəri isə innovasiyalardan, dizayn həllərindən, marketinq keyfiyyətindən asılı olacaq.

4. *İnformasiya cəmiyyətində istehsal olunacaq məhsul bilik və intellekt olacaqdır ki, bu da ümumiyyətlə, əqli əməyin payını artıracaqdır.* İntellektual əməklə bağlı sənət sahələrini seçən insanların sayı artacaqdır.
5. *Dəyərlər dəyişərək, yeni həyat tərzi formalaşacaq, asudə vaxtda məşğuliyyətlər dəyişəcəkdir.*
   Artıq indi kompüter oyunları insanın boş vaxtının əsas hissəsini tutur. Bu oyunlar uzaq məsafədə yerləşən bir neçə oyunçunu özündə birləşdirən şəbəkə sisteminə transformasiya olunur. İnternetdə vaxt keçirənlərin sayı artır. Onlar tədris saytlarına və virtual muzeylərə səyahət edirlər, lazımi ədəbiyyatı tapıb oxuyurlar və s. İnternetdə gap xidmətləri [chat] və ICQ xidməti istifadəçilər tərəfindən çox bəyənilir. Bu xidmətlər vasitəsilə uzaq məsafələrdə olan insanlar real zaman rejimində bir-birlərilə əlaqə saxlayırlar.
6. *Kompüter texnikası, kompüter şəbəkələri, informasiya texnologiyaları inkişaf edəcək.*
   İnternet şəbəkəsi ayda 10-15% artacaq və onun istifadəçilərinin sayı yüz mil-yonlarla ölçüləcək. Özündə müxtəlif qurğuların funksiyalarını cəmləşdirən (kompüter, televizor, radio, telefon və s.) müasir multimedia sistemlərinin istifadəsi informasiya texnologiyalarının universallaşmasına gətirib çıxaracaqdır. İnformasiyanı saxlayan qurğular isə bapbalaca – ovucun içində yerləşən ölçüdə olacaq. Bu qurğular daxilində həcmi bir neçə ensiklopediyaya bərabər olan şəxsi universal soraqçalar da yerləşdiriləcəkdir. Bu qurğuları şəbəkəyə qoşub operativ informasiya, məsələn, hava haqqında, yollardakı tıxaclar barədə məlumatlar almaq mümkün olacaqdır.
7. *Evlərdə cürbəcür elektron cihazlar və kompüterləşdirilmiş qurğular olacaqdır.*
   Mənzillər naqillər sisteminin əvəzinə bir cərəyan və bir də informasiya kabeli ilə təchiz olunacaqdır. İnformasiya kabeli rabitə, televiziya kanallarını və İnternetə çıxışı özündə birləşdirəcəkdir. Xüsusi elektron blok bütün məişət avadanlıqlarına və yaşayış sistemlərinə nəzarət edəcək, binalar "ağıllı" olacaqdır. "Ağıllı binalar"la yanaşı "ağıllı avtomobil"lər də yaranacaqdır. Onlarda avtomobilin texniki hissəsinə cavab verən kompüterdən başqa, şəhər informasiya xidmətlərinə qoşulan sistem də işləyəcək. Belə avtomobil "ağıllı ev"lə əlaqədə olacaq və hətta evi idarə də edə biləcək.
8. *Enerji və maddi məhsullar istehsalını maşınlar təmin edəcək, insan isə, əsas etibarilə, informasiyanın emalı ilə məşğul olacaq. İstehsalatda insanların sayı azalacaq, onların yerini robotlar və manipulyatorlar tutacaqdır.*

9. *Təhsil sahəsində fasiləsiz təhsil sistemi yaradılacaq*. İnsanlar zamanla ayaqlaşmaq üçün, sənəti dəyişmək və cəmiyyətdə layiqli yer tutmaq üçün ömrü boyu oxumaq imkanı qazanacaqlar.
10. *Uşaqlar kompüter proqramları və telekommunikasiyalar vasitəsilə evdə təhsil ala biləcəklər.* Bununla əlaqədar tədris prosesində təlimin formaları də-yişəcək və təlimin tərbiyəvi aspektlərilə bağlı problemlər yarana biləcəkdir.
11. *İnformasiya xidmətləri bazarı yaradılaraq inkişaf etdiriləcək.* İnformasiya məhsul və xidmət növünə çeviriləcək. Bu məhsulu adi əmtəə kimi alıb-satmaq mümkün olacaqdır.

Sənaye cəmiyyətindən informasiya cəmiyyətinə keçmək üçün informasiya böhranı şəraiti əmələ gəlməli idi. Bu şərait XX əsrdə yarandı. İnsanın üzərinə sel kimi gələn böyük həcmdə informasiya axınında istiqamət götürmək qeyri-mümkün oldu. Artıq və lazımsız olan çoxlu miqdarda informasiya yarandı. İnformasiya cəmiyyətinə keçid isə müxtəlif sahələrdə informasiyanın ötürülməsi və onun emalı üçün müasir vasitələrdən istifadə edilməsi ilə başlandı. Bu proses *informasiyalaşdırma* adlanır.

Cəmiyyətin informasiyalaşdırma prosesi sənaye cəmiyyətinin informasiya cəmiyətinə keçməsini təmin edir.

Cəmiyyətin informasiyalaşdırılması prosesi cəmiyyətin hər bir üzvünə öz tələbatına uyğun informasiya almaq imkanı verir.

Son dövrlərə kimi «informasiyalaşdırma» sözü əvəzinə «kompüterləşdirmə» sözündən istifadə olunurdu. Lakin «kompüterləşdirmə» sözünün mənası kompüter texnikasının inkişafı və tətbiqi deməkdir. Cəmiyyətin informasiyalaşdırılması isə kompüterləşdirməyə nisbətən daha geniş anlayışdır. Belə ki, bu gün texniki vasitələr deyil, sosial-texniki prosesin məqsəd və mahiyyəti daha önəmlidir. Kompüterləşdirmə isə informasiyalaşdırma prosesinin bir hissəsidir, onun texniki bazasıdır.

İnformasiya mədəniyyəti.

İnsan mədəniyyəti

- bilik, bacarıq və peşəkar vərdişlər ilə;
- intellektual, estetik və mənəvi inkişaf səviyyəsinə görə;
- insanlarla ünsiyyətin forma və üsulları ilə müəyyən olunur.

İnsanın şəxsi mədəniyyəti

- əqli inkişaf səviyyəsinə görə;
- peşəkarlıq və yaradıcılıq fəaliyyətinin xarakterinə görə müəyyən olunur.

Bu o deməkdir ki, insan nə qədər öz zehni qabiliyyətlərini inkişaf etdirirsə, nə qədər düşünür, fikirləşirsə, bir o qədər öz şəxsi mədəni səviyyəsini yüksəltmiş olur. Ona görə də elm və incəsənətlə məşğul olan insanın mədəni səviyyəsi daha yüksək olmalıdır. İnformasiya cəmiyyətinə keçərkən insanın ümumi mədəniyyətinə daha bir kateqoriya əlavə olunur – *informasiya mədəniyyəti.*

> İnformasiya mədəniyyəti informasiya ilə məqsədyönlü işləmək və onun qəbul edilməsi, emalı və ötürülməsi üçün kompüter texnikasından, müasir texniki vasitə və üsullardan istifadə etmək bacarığıdır.

İnsanın informasiya mədəniyyəti dedikdə aşağıdakılar başa düşülür:

– telefondan tutmuş fərdi kompüter və kompüter şəbəkələrinə kimi müxtəlif texniki cihazlardan istifadə etmək vərdişlərinə yiyələnməsi;
– informasiya texnologiyalarına yiyələnmək qabiliyyətinin olması;
– dövri mətbuat və elektron kommunikasiyalarından informasiya əldə etmək bacarığının olması;
– informasiyanı aydın şəkildə təqdim etmək və ondan maksimum səmərə ilə istifadə etmək bacarığının olması;
– informasiyanın emalının müxtəlif üsullarını bilməsi;
– müxtəlif informasiya növlərilə işləmək bacarığının olması.

İnformasiya resursları. Hər bir dövlət, cəmiyyət, şirkət və adi insan özünün həyat fəaliyyəti üçün zəruri olan resurslara malikdir.

> Resurs müxtəlif vəsaitlərin ehtiyatı, mənbəyi deməkdir.

Müasir cəmiyyətdə maddi, xammal, enerji, əmək və maliyyə resursları ilə yanaşı, *informasiya resursları* da mövcuddur.

> İnformasiya resurslarına elmi-texniki biliklər, ədəbiyyat və incəsənət əsərləri, ictimai-dövlət əhəmiyyətli digər informasiyalar daxildir.

İnformasiya resurslarından başqa, istənilən resurs işləndikcə yox olur. Məsələn, yanacaq yanıb qurtarır, maliyyə vəsaiti xərclənir və s. İnformasiya resursu isə «bitib tükənmir», ondan coxlu sayda istifadə etmək mümkündür.

1. Bəşər tarixində hansı informasiya inqilabları olmuşdur?
2. İnformasiya cəmiyyətini xarakterizə edən cəhətlər hansılardır?
3. Yaşadığımız cəmiyyət informasiya cəmiyyəti adlana bilərmi?
4. İnformasiya mədəniyyəti nədir?
5. İnformasiya resurslarına nə daxildir və onların başqa resurslardan fərqi nədədir?

4.2. KOMPÜTER TEXNİKASININ TƏTBİQ SAHƏLƏRİ

İlk elektron hesablama maşınları təxminən 60 il bundan öncə meydana çıxıb. Ötən dövr ərzində kompüterlərin və onlar üçün proqramların istehsalı texnologiyası böyük inkişaf yolu keçib. Şübhəsiz, kompüterlər bütövlükdə yaşadığımız cəmiyyəti dəyişdirib. Maliyyə, kargüzarlıq, sənaye, elm, səhiyyə, təhsil və başqa sahələri bu gün kompüterlərsiz təsəvvür etmək qeyri-mümkündür. İndi informasiyanın emal olunduğu hər yerdə kompüterlərdən istifadə olunur.

Kompüter təhsildə. Yüksək ixtisaslı mütəxəssislərin hazırlanması uzun və mürəkkəb prosesdir. Orta məktəbdə, sonra isə ali məktəbdə təhsil insan həyatının önəmli bir hissəsini təşkil edir. Bununla belə, müasir informasiya cəmiyyətində biliklər çox sürətlə köhnəlir. Hər hansı professional işi yerinə yetirmək bacarığına malik olmaq`üçün insan öz təhsil dairəsini daim genişləndirməlidir.

İnformasiya cəmiyyətində «necə»ni bilmək, «nə»yi bilməkdən daha vacibdir. Buna görə də hazırda orta və ali məktəbin əsas vəzifəsi mümkün qədər çox bilik vermək deyil, həmin bilikləri müstəqil əldə etmək yollarını öyrətməkdir. Bu isə təhsildə müasir informasiya və kommunikasiya texnologiyalarından istifadə etmədən mümkün deyil.

İnformasiya və kommunikasiya texnologiyalarının təhsildə tətbiqinin ən vacib istiqamətlərindən biri kompüterlərin multimedia imkanlarından istifadə etməkdir. Multimedia vasitələrindən istifadə olunması əyaniliyi gücləndirir, informasiyanın məntiqi və obrazlı mənimsənilməsi üsullarını birləşdirir ki, bu da təlim prosesini fəallaşdırır. Multimedia texnologiyalarının interaktivliyi şəxsiyyət yönümlü təlim modellərini gerçəkləşdirmək üçün geniş imkanlar verir.

İnformasiya və kommunikasiya texnologiyaları distant təhsili təşkil etməyə imkan verir. Distant təhsildə (məsafədən təhsildə) müəllim və öyrənən məna baxımından ayrıdırlar və dərs prosesi telekommunikasiya vasitəsilə, başlıca olaraq İnternet şəbəkəsi vasitəsilə həyata keçirilir.

Distant təhsil sayəsində bir çox insanlar (iş və ailə qayğıları ilə yüklənmiş yaşlılar, kənd yerlərində, yaxud kiçik şəhərlərdə yaşayan gənclər) evdə öz təhsillərini artırmaq imkanı əldə edirlər.

Kompüter elmi tədqiqatlarda. Hazırda bir çox bilik sahələrində elmi tədqiqatlar alimlər, mühəndislər və konstruktorlardan ibarət böyük kollektivlər tərəfindən çox mürəkkəb və bahalı avadanlıq vasitəsilə həyata keçirilir. Elmi araşdırmaların səmərəliliyi önəmli dərəcədə kompüter texnikasından istifadə

səviyyəsindən asılı olur. Kompüterlərdən eksperimentlərin idarə olunmasında, hesabatların və sənədlərin hazırlanmasında, eksperimentlərin nəticələrinin təhlilində və s. istifadə olunur. Nəticədə:

- tədqiqatların aparılması müddəti bir neçə dəfə qısalır;
- nəticələrin dəqiqliyi və doğruluğu artır;
- eksperimentin gedişinə nəzarət güclənir;
- nəzarət edilən parametrlərin sayının artması və verilənlərin daha dəqiq emalı nəticəsində eksperimentin keyfiyyəti və informativliyi yüksəlir;
- eksperimentlərin nəticələri operativ surətdə daha əlverişli formada (məsələn, qrafik) verilir.

Kompüter səhiyyədə. Müasir cəmiyyətdə kompüterlərin səhiyyədə rolu günbəgün artır. Həkimlər kompüterlərdən bir çox məqsədlər üçün istifadə edirlər. Onlardan bir neçəsini sadalayaq:

1. Kompüter texnikasından diaqnozların qoyulmasında, müayinələrin aparılması və profilaktik yoxlamalarda geniş istifadə olunur.
2. Kompüter şəbəkələrinin köməyilə transplantasiya əməliyyatını gözləyən xəstələr üçün donor orqanları haqqında məlumatlar göndərilir.
3. Tibbi verilənlər bankı tibb işçilərinə son elmi və praktik nailiyyətlər haqqında məlumatlı olmağa kömək edir.
4. Kompüterlər havanın çirkliliyinin əhali arasında xəstəliklərin yayılmasına necə təsir etməsini müəyyənləşdirməyə kömək edir.
5. Kompüter texnikasından tibb işçilərinə praktik vərdişlərin aşılanmasında istifadə edilir. Bu zaman kompüter təcili tibbi yardıma ehtiyacı olan xəstə rolunda çıxış edir. Öyrənən, kompüterin verdiyi simptomlar əsasında müalicə kursunu müəyyənləşdirməlidir. Səhv olan kimi kompüter bu barədə xəbərdarlıq edir.
6. Kompüterlərdən epidemiyaların yayılma sürətini göstərən xəritələrin hazırlanması üçün istifadə edilir.
7. Kompüterin yaddaşında xəstəlik tarixçələri saxlanılır. Bu da həkimlərə yazı işlərinə deyil, xəstələrin özlərinə daha çox vaxt ayırmaq imkanı verir.

Kompüter ticarətdə. Supermarketlərin çoxunda hər malın üzərində *barkod* – müxtəlif qalınlıqda qara cizgilərdən ibarət etiket olur. Ödəmə zamanı barkod skanerdən (barkod oxucusundan) keçirilir. Bu skanerə bağlı kompüter barkodu qiymət siyahısında tapır və nəticəni kassa aparatına göndərir.

Barkod oxucusundan istifadə edən mağazalarda, yaxud anbarlarda hər bir mal haqqında informasiya *verilənlər bazasında* saxlanılır.

> **Barkod** mal və onun istehsalçısı haqqında informasiyanı özündə saxlayır.

Ən geniş yayılmış barkodlar 13 mərtəbəli Avropa kodu EAN-13 (European Article Numbering) və ABŞ ilə Kanadada tətbiq olunan UPC kodudur.

EAN-13 kodunun strukturu

1. Ölkənin kodu
2. İstehsalçının kodu
3. Malın kodu
4. Nəzarət rəqəmi
5. Lisenziya əsasında hazırlanmış malın işarəsi

Ölkə və istehsalçının kodlarındakı rəqəmlərin sayı dəyişə bilər. Bu sistemdə Azərbaycanın kodu 476-dır.

İnformasiyalaşdırmanın ən mühüm istiqamətlərindən biri pul – kredit və maliyyə sahələrində *elektron pula* keçiddir.

Nağdsız ticarət. Getdikcə daha çox ticarət müəssisələrində Ödənişlər *kredit kartları* vasitəsilə aparılır. Kredit kartı olan alıcının bankdakı hesabından lazım olan məbləğ avtomatik silinərək, mağazanın bank hesabına köçürülür.

Nağdsız ticarət sistemi POS (Points of Sale System) aşağıdakı funksiyaları yerinə yetirir:

- kredit kartlarının verifikasiyası, yəni onların həqiqiliyinin təsdiqi;
- alıcının hesabından pulun silinməsi;
- pulun satıcının hesabına köçürülməsi.

Kredit kartlarına məlumatlar maqnit yazısı üsulu ilə vurulur. Kredit kartında informasiyanı saxlamaq üçün maqnit kartından istifadə olunur.

Maqnit kartına aşağıdakı informasiyalar yerləşdirilir:

- şəxsi hesabın nömrəsi;
- bankın adı;
- ölkə;
- müştərinin ödəniş qabiliyyətinin kateqoriyası;
- verilmiş kreditin miqdarı və s.

Bankomatlar. *Bank avtomatları – bankomatlar* (ATM – Automated Teller Machine), yaxud *nağd ödəmə maşınlarının* şəbəkəsi, ən geniş şəbəkələrdən biridir. Dünyadakı minlərlə bankın kompüterləri bir-birinə bağlanır. Bankomatlar banklar tərəfindən qabaqcadan kredit kartları verilmiş müştəriləri üçün quraşdırılır.

Bunun da nəticəsində siz xarici ölkədə olarkən Azərbaycanda yerləşən bank tərəfindən verilmiş kredit kartından istifadə edərək, bankomatdan pul götürə bilərsiniz.

Bankomatda iş prinsipi belədir:

1. Müştəri kartını avtomata yerləşdirir və avtomatın klavişlərindən istifadə edərək, öz şifrəsini yığır.
2. Kartın üzərindəki maqnit zolağında hesab sahibinin adı, hesab nömrəsi və onun bağlı olduğu şəbəkə haqqında məlumatlar avtomat tərəfindən oxunur.

3. Avtomat (daha doğrusu, avtomatdakı kompüter) telefon xətləri vasitəsilə minlərlə bankın məlumatının saxlanıldığı mərkəzi kompüterə məlumat göndərir.
4. Mərkəzi kompüter hesabı yoxlayır və pulu vermək, yaxud istəyi rədd etmək haqqında bankomata məlumat göndərir.

Kompüter kənd təsərrüfatında. Kompüteri olan fermer asanca və tez bir zamanda əkin üçün tələb olunan toxum və gübrələrin miqdarını hesablaya bilər. Kompüter sistemləri növbəli əkini planlaşdıra, suvarma qrafikini hesablaya, malqaraya yemin verilməsini idarə edə və başqa faydalı işlər görə bilər.
Kənd təsərrüfatında texnoloji inqilab gözlərimiz önündə baş verir: kompüterlər və fərdi mikrosensorlar hər bir heyvanın və bitkinin vəziyyətinə nəzarət etməyə imkan verir. Bu, maddi və insan resurslarına qənaət edir, insanın həyat səviyyəsini yaxşılaşdırır.

1. Təhsildə İKT-nin tətbiqini necə təsəvvür edirsiniz?
2. Distant təhsil nədir?
3. Səhiyyədə kompüterlərdən hansı məqsədlər üçün istifadə olunur?
4. Barkod nədir və onun strukturu necədir?
5. Bankomatla iş prinsipini izah edin.
6. Kredit kartı nədir və onun quruluşu necədir?

PROQRAM

İnformatika – IX sinif

(həftədə 1 saat, cəmi 32 saat)

I PASCAL PROQRAMLAŞDIRMA DİLİ (21 saat)

Proqram təminatının təsnifatı. Proqramlaşdırma dilləri. Yüksək səviyyəli dillər. Proqramların hazırlanması. Turbo Pascal redaktoru. Pascal-proqramın ümumi strukturu. Operatorlar. Praktikum. Şərti yoxlayan operatorlar. Praktikum. Dövr operatorları. Praktikum. Massivlər. Praktikum. Sətirlərlə iş. Praktikum. Altproqramlar. Fayllar. Praktikum.

II ELEKTRON SƏNƏD (4 saat)

Mətn sənədi və onun obyektləri. Mətn sənədin hazırlanması. Sənədin redaktəsi. Sənədin formatlanması.

III CƏDVƏL PROSESSORU (5 saat)

Cədvəl prosessorunun təyinatı. Elektron cədvəlin obyektləri. Elektron cədvəl verilənləri. Düsturlar. Diaqramın yaradılması və redaktəsi. Cədvəl sənədinin formatlanması.

IV İNFORMASİYA CƏMİYYƏTİ (2 saat)

Cəmiyyətin informasiyalaşdırılması. Kompüter texnikasının tətbiq sahələri.

MÜNDƏRİCAT

İnformatika – ümumtəhsil məktəblərinin 9-cu sinfi üçün dərslik.
İsmayıl Calal oğlu Sadıqov
Ramin Əli Nazim oğlu Mahmudzadə
Naidə Rizvan qızı İsayeva

Bakı, “Bakınəşr”, 2008. 128 səh.

Format $70 \times 100^{1}/_{16}$. Ofset kağızı №1. Fiziki çap vərəqi 8.
Çapa imzalanmışdır 29.07.2008. Tiraj 155 000. Pulsuz.