AZƏRBAYCAN RESPUBLİKASI TƏHSİL NAZIRLİYİ
SUMQAYIT DÖVLƏT UNİVERSİTETİNİN NƏZDİNDƏ
SUMQAYIT DÖVLƏT TEXNİKİ KOLLECİ

Alqoritmik dillər fənnindən mühazirələr

Orta İxtisas Təhsil müəssələrində
Fənnin tədrisi üçün nəzərdə tutulub

Sumqayıt- 2020

Mövzular:

1. Alqoritm nədir?.
2. “Alqoritmin icraçısı” və alqoritmin xassələri
3. Həll alqoritiminin yaradılmasının əsas prinsipləri.
4. Sxemlə təsviri.
5. Tipik alqoritmik strukturlar.
6. Proqramlaşdırmanın dilləri.
7. Proqramlaşdırma dillərinin səviyyələri.
8. Stuktur proqramlaşdırma.
9. Turbo Paskalla işləmə əmrləri.
10. Turbo Paskalla dilinin lüğəti.
11. Turbo Paskal dilinin əsas elementləri. İdentifikatorlar.
12. Verilənlərin tipləri təsnifatı.Standart tiplər.
13. Dəyişənlər(həqiqi, məntiqi və bul).
14. Paskal dilində istifadə olunan tiplər.
15. Sabitlər.
16. Hesabi funksiyalar.Tipin çevrilmə funksiyaları.
17. Əməliyyat.Turbo Paskal alqoritmik dilində ifadələr.
18. Proqramın strukturu.
19. Sadə operatorlar. Şərti keçid operatorları.
20. Seçmə operatoru.
21. Parametrli dövr operatorlar.
22. Ön şərtli dövr operatoru.
23. Son şərtli dövr operatoru.

1. Alqoritm nədir?

Alqoritm anlayışı da informasiya anlayışı kimi informatikanın əsasını təşkil edir. **Alqoritm-** qarşıya qoyulan məsələni həll etmək üçün yerinə yetirilməsi vacib olan əməliyyatlar ardıcıllığıdır.Latın dilində mənası qanun deməkdir.Alqoritm 783-850 ci illərdə Xarəzmdə (Özbəkistanda) yaşamış 9-cu əsrin məhşur özbək riyaziyyatçısı Məhəmməd İbn Musa əl- Xarəzminin (yəni Xarəzmli Musa oğlu Məhəmməd) adının latın hərflərilə olan "alqoritmi" yazılışı ilə bağlıdır. Əl-Xarəzminin yazdığı traktatın 12-ci əsrdə latın dilinə tərcümə olunması sayəsində avropalılar mövqeli say sistemi ilə tanış olmuş,onluq say sistemini və onun hesab qaydalarını alqoritm adlandırmışlar. Ümumiyyətlə,alqoritm verilmiş məsələnin həlli üçün lazım formal yazılışdır. İnsan hər gün bu və ya digər qaydalara uyğun hərəkət etmək,müxtəlif təlimatları və göstərişləri terinə yetirmək məcburiyyətində olur.Riyaziyyatın tipik məsələlərini həll etməkdikdə biz,hərəkətlərin ardıcıllığını təsvir edən müəyyən qaydalardan istifadə edirik. **Alqoritm –** sonlu sayda addımlar nəticəsində məsələnin həllini əldə etmək üçün icraçı tərəfindən yerinə yetirilən aydın və dəqiq müəyyən hərəkətlər ardıcıllığıdır. Bu riyazi mənada təyin edilmə demək olmayıb, alqoritm anlayışının intituv təsviridir və onun mahiyyətini açır. Alqoritm anlayışı nəinki riyaziyyatın ,həmçinin də müasir elmin əsas anlayışlarından biridir. Bundan başqa informatika əsrinin gəlməsi ilə alqoritm sivilizasiyasının da əsas amillərindən biri sayıla bilər.

2.Alqoritmin icraçısı və alqoritmin xassələri:

Alqoritmin icraçısı – alqoritm tərəfindən göstərilən hərəkətləri yerinə yetirmək qabiliyyətinə malik olan abstrakt və ya real sistemdir. İcraçını aşağıdakılar xarakterizə edir :

- Mühit;
- Elementar hərəkətlər;
- Əmrlər sistemi;
- İmtinalar.

Mühit (və ya şərait)- bu icraçının "işlədiyi yerdir". Məsələn Robot icraçısı üçün busonsuz xanalar sahəsi ola bilər.Divarlar və rənglənmiş xanalarda mühitin bir hissəsi ola bilər.Onların yerləşməsini və Robotun özünün vəziyyətini **mühitin konkret vəziyyəti** verir.

Əmrlər sistemi. Hər bir icraçı icrasının əmrlər sisteminin verilmiş siyshısının müəyyən sətirdən götürülmüş əmrləri yerinə yetirə bilər. Hər bir əmr üçün **tətbiq olunma şərti** (mühitin hansı vəziyyətdə əmr yerinə yetirilə bilər) və **əmrin yetirilməsinin nəticəsi** təsvir olunmalıdır. Məsələn, Robot üçün "yuxarı" əmri yalnız o vaxt yerinə yetirilə bilər ki,Robotdan yuxarıda divar olmasın. Onun nəticəsində Robot bir xana yuxarı hərəkət edəcəkdir. Əmri çağırdıqdan sonra icraçı uyğun **elementar hərəkətləri** yerinə yetirir. İcraçının işdən imtinaları o vaxt baş verir ki, mühitin vəziyyəti həmin əmrin icrasına imkan vermir. Adətən icraçı alqoritmin məqsədi haqqında heç nə bilmir. O, "nə üçün" və "niyə görə" suallarını vermədən alınmış bütün əmrləri icra edir. İnformatikada universal icraçı rolunda **kompyuter** çıxış edir.

Alqoritm aşağıdakı xassələrə malik olmalıdır:

1. İcraçı üçün **anlamlı** olmalıdır- alqoritmin icraçısı həmin alqoritmin necə yerinə yetirilməsini başa düşməlidir. Başqa sözlə desək,alqoritm və verilənlərin ixtiyari variantına malik olan icraçı bu alqoritmin yerinə yetirilməsi üçün necə hərəkət etmək lazım olduğunu bilməlidir.

2. Diskretlilik -(kəsilmələr,ayrılmalar) –alqoritm,məsələnin həllolunma prosesini ardıcıl olaraq,sadə addımların yerinə yetirilməsi kimi dərk etməlidir.

3. Müəyyənlik- alqoritmin hər bir qaydası dəqiq,bir mənalı olmalı və ixtiyari hərəkətlərə yol verməməlidir. Alqoritmin tərtibi məsələnin həllini ardıcıl yerinə yetirilən mərhələlərə bölmək deməkdir. Bu zaman əvvəlki mərhələlərin nəticələri sonrakı mərhələlərdə istifadə oluna bilər və hər bir mərhələnin məzmunu və mərhələlərin yerinə yetirilmə ardıcıllığı müəyyən olmalıdır.Bu xassə sayəsində alqoritmin yerinə yetirilməsi mexaniki xarakter daşıyır və həll ounan məsələ barəsində əlavə göstərişlər və məlumatlar tələb olunmur.

4. Nəticəvilik- (və ya sonluluq) ondan ibarət olur ki,sonlu sayda addımlar sayəsində alqoritm ya məsələnin həllinə gətirib çıxarmalıdır ,ya sonlu sayda addımlardan sonra həllin əldə edilə bilməməsi səbəbindən dayanmalı və bu barədə uyğun məlumat verməlidir,ya da ki,alqoritmin icrası üçün ayrılmış vaxt ərzində aralıq nəticələri verməklə, icranı davam etdirməlidir.

5. Kütləvilik- o deməkdir ki,məsələnin həll alqoritmi ümumi hal üçün işlənir, yəni,o yalnız ilkin verilənlərlə fərqlənən müəyyən sinif məsələləri üçün tətbiq edilə bilər. Bu halda ilkin verilənlər,alqoritmin tətbiq olunma oblastı adlanan müəyyən bir oblastdan seçilə bilərlər.

3.Həll alqoritminin yaradılmasının əsas prinsipləri

Komputerdə istənilən məsələnin həlli aşağıdakı mərhələlər ardıcıllığı ilə aparılır:

1. Məsələnin qoyuluşu
2. Həll alqoritminin yaradılması
3. Hər hansı proqramlaşdırma dilində ilkin proqramın tərtibi
4. İlkin proqramın maşın dilində tərcüməsi və redaktə edilməsi
5. İşçi proqramın sazlanması
6. İşçi proqramın icraçısı və nəticələrin alınması

Məsələnin qoyuluşu- ilkin verilənlərin siyahısını,onların tipini,dəqiqliyini və ölçülərini, dəyişənlərin dəyişmə hədlərini, başlanğıc və sərhəd şərtlərini,nəticələrin siyahısını,onların tipini,dəqiqliyini və ölçülərini ,məsələnin həllini təmin edən hesabat düsturları və tənlikləri nəzərdə tutur. Alqoritm məsələnin həllini təmin edən formal qaydalar sistemidir. Məsələnin komputerlə həlli baxımından alqoritm axtarılan cavabların alınması üçün məsələnin verilənləri üzərində icra olunan hesabi və məntiqi əməliyyatlar ardıcıllığıdır. Alqoritmin yaradılması prosesində onun təsvirinin müxtəlif üsullarından istifadə olunur:

1. Təbii dil üsulu
2. Sxem üsulu
3. Psevdokod üsulu
4. Stukturoqram üsulu
5. Alqoritmik dillə təsvir üsulu

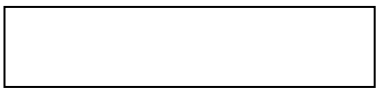
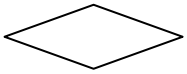
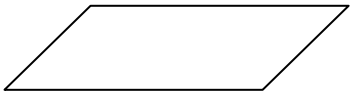
Təbii dil (alqoritmin sözlə təsviri) alqoritmin icrasının insan tərəfindən aparıldığı halda əlverişli ola bilər. Alqoritmin sözlə yazılış üsulu verilənlərin emalının ardıcıl mərhələlərinin təsvirindən ibarətdir.Alqoritm təbii dildə ixtiyari şəkildə yazılır.

Sxem- alqoritmin qrafiki təsviri(əyani təsvir) üsuludur. Alqoritmi qrafik şəkildə təsvir etmək üçün funksional bloklardan istifadə olunur ki, o bloklardan hər biri bir və ya bir neçə hərəkətləri yerinə yetirə bilir.

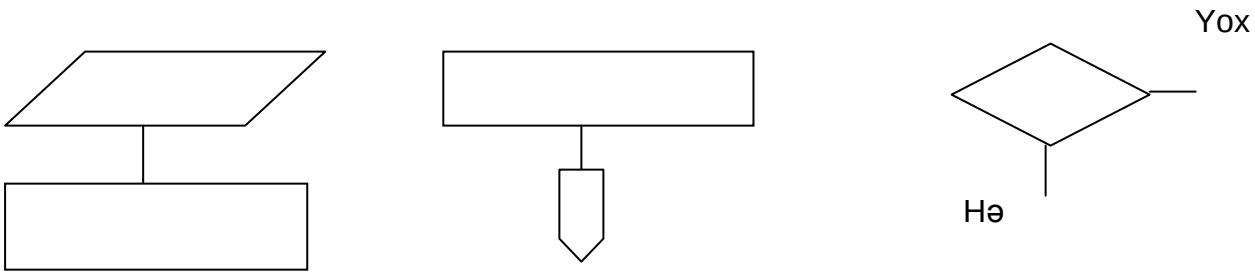
4. Alqoritmin sxemlə təsviri

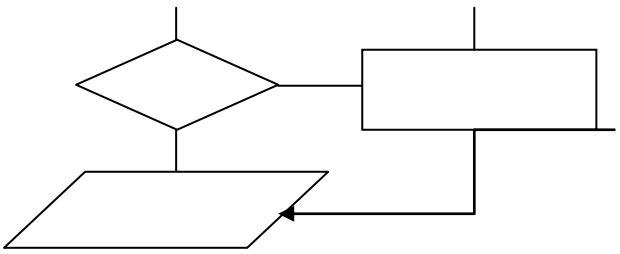
Sxem- alqoritmin qrafik təsviri üsuludur. Alqoritmi qrafiki şəkildə təsvir etmək üçün funksional bloklardan istifadə olunur ki,o bloklardan hər biri bir və ya bir neçə hərəkətləri yerinə yetirə bilər. Bu cür qrafik şəkildə təsvir alqoritmin sxemi

və ya blok-sxem adlanır. Blok-sxemdə hər hərəkət tipinə (ilkin verilənlərin daxil edilməsi,ifadənin qiymətinin hesablanması,şərtin yoxlanılması,hərəkətlərin təkrar olunmasının idarəsi,emalın sonu və.s) **blok simvolu** şəklində təsvir olunan həndəsi fiqura uyğun gəlir. Blok simvolları keçid xətləri ilə birləşdirilir ki, o da hərəkətlərin yerinə yetirilmə ardıcıllığını təyin edir. Blokların həndəsi fiqurla ifadəsi Proqram Sənədlərinin Vahid Sistemli (PSVS) əsasında qəbul olunmuşdur.

Hesab bloku (proses)	
Məntiqi blok (seçmə)	
Daxiletmə-xaricetmə bloku	
Çapetmə bloku	

Bloklar həndəsi fiqur şəklində ifadə olunur və bir-birinə şaquli və yaxud üfüqi xətlərlə birləşdirilir.

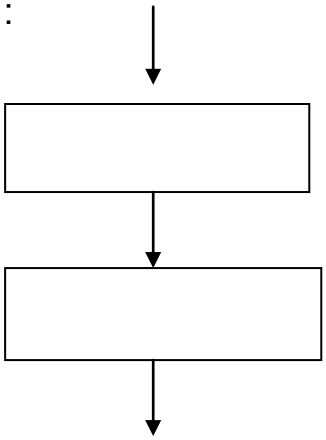




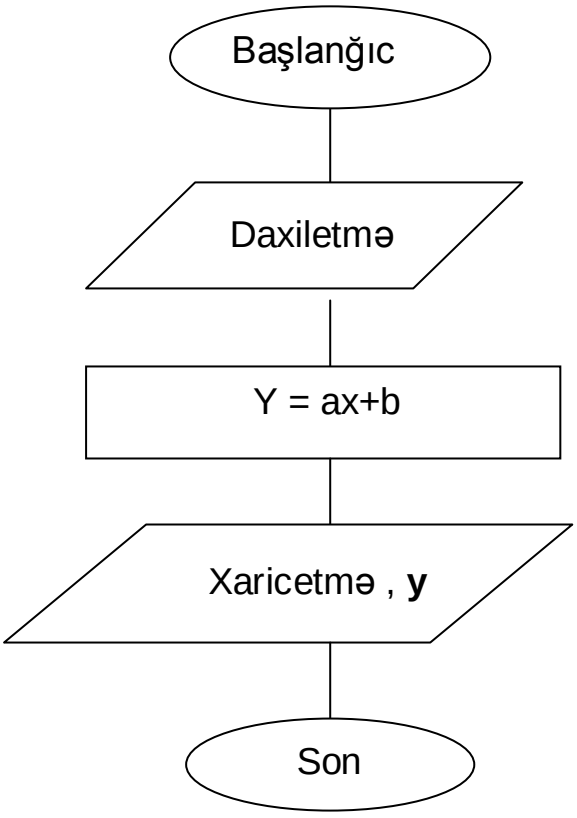
5. Tipik alqoritmik strukturlar

Tipik alqoritmik strukturlar alqoritmlərin və proqramların struktur yolla yaradılmasıda tətbiq olunur. Bu strukturların kombinasiyası ilə alqoritmlər və proqramlar tətbiq edilir. İstənilən hesablama prosesi aşağıdakı tipik (elementar) alqoritmik strukturların kombinasiyasından təşkil olunur : xətti,budaqlanan, dövrü.

Xətti alqoritmik struktur. Tərkibində məntiqi və dövri bloklar olmayan iki və daha çox mərhələlər ardıcıllığından təşkil olunur. Bu strukturu sxematik olaraq aşağıdakı kimi göstərmək olar :



Misal : y= ax+ b xətti tənliyinin alqoritmini tərtib etməli:



6. Proqramlaşdırmanın dilləri

Proqramlaşdırma dillərinin təsnifatının əsas əlamətlərindən biri dilin hasnı proqramlaşdırma üslubuna mənsub olmasıdır.Proqram texnologiyasında əsasən aşağıdakı üslublardan istifadə olunur:

1. prosedur proqlaşdırma
2. məntiqi proqlaşdırma
3. obyektyönlü proqramlaşdırma
4. hadisəyönlü proqramlaşdırma
5. vizual proqramlaşdırm

Proqramlaşdırmada müxtəlif səviyyəli dillərdən istifadə edilir: maşın dilləri, Assembler, yüksək səviyyəli (alqoritmik) dillər.Maşın dili konkret kompüterin əmrlər sistemindən ibarət olub, bilavəsitə həmin maşın tərəfindən həyata keçirilir. Maşın dilində proqram tərtib etdikdə hər şeydən əvvəl dəyişənlər və konstantlar üçün maşının yaddaşında yer ayrılır. Maşın dilində proqram maşın əmrləri ardıcıllığından və dəyişənlər, konstantlar üçün yaddaşda təyin edilmiş müəyyən sahələrdən ibarətdir. Assemberlər səviyyəli dillər konkret kompüterlərin əmrlər sisteminə uyğun gələn maşınyönümlü dillərdir.Buna baxmayaraq, onlar proqramı istifadəçi üçün daha rahat olan formada tərtib etməyə imkan verirlər.Assembler dilinin üstün cəhəti ondadır ki, dildə əmrlərə, konstantlara və dəyişənlərə müəyyən adlar mənsub edilir və bu adlar vasitəsilə onların özlərinə müraciət etmək imkanı yaranır. Assembler dilində kompüterin bütün imkanlarından tam istifadə etməyə imkan verən effektiv proqramlar yazılır.

Çatışmayan cəhət proqramın həddindən artıq təfərrüfatı ilə yazılmasıdır.

Yüksək səviyyəli dillər iki sinfə bölünürlər:

1. problemyönümlü dillər;
2. proseduryönümlü dillər;

Problemyönümlü dillər çox kiçik sinif təşkil edən məsələləri həll etmək üçün təyin edilmişdir Proqramlaşdırma prosesində həll alqoritmi deyil, məsələnin özü təsvir edilir.

Proseduryönümlü dillər məsələnin həll alqoritmi təsvir etmək üçün təyin edilmişdir.Onlar öz növbəsində maşından asılı olan və maşından asılı olmayan alqoritmik dillərə bölünürlər .
Maşından asılı olan yüksək səviyyəli dillər maşının bütün imkanlarından tam istifadə etməyə, aydın və asanlıqla oxunan proqramlar yazmağa imkan verirlər.Bu dillərdə operatorlar və ifadələr istifadəçi üçün daha rahat şəkildə yazırlar.Onlara misal olaraq PL/M dilinin göstərmək olar.Bununla belə bu dillər konkret quruluşdan çox asılıdır və bu səbəbə görə praktikada geniş yayılmışdır.
Maşından asılı olmayan yüksək səviyyəli dillərin yaxud alqoritmik dillərin tərkibində maşından asılı olan operatorlar iştirak etmir .Bu dillərə Alqol, Fortran, Beysik, Fokal,PL/1, Paskal və s. daxildir.Alqoritmik dillərin əsas üstünlüyü proqramçının yüksək əmək məhsuldarlığı, proqramların asanlıqla bir maşından digərinə keçirilməsi, proqramlardan asanlıqla istifadə etmək imkanının olmasıdır.Maşın dilində yazılmış xüsusi proqram-transiyator alqoritmik dildə təsvir edilən alqoritmin simvolik təsvirini emal edir və proqramı avtomatik olaraq maşın dilinə çevirilir.
Perl 80-ci illərdə Larri Uoll tərəfindən işlənmişdir. Bu proqram dilinin böyük həcmli mətn fayllarının effektiv işlənməsində, hesabatların generasiyasında və məsələlərin idarəsində istifadəsi nəzərdə tutulmuşdur.
Perl-dən sətirlərlə, massivlərlə, ayrı-ayrı verilənlərlə, proseslərin idarəsində, system informasiyaları ilə işlərdə istifadə edilir. HTML web səhifələrin hazırlanmasında istifadə edilən populyar dildir.
Assembler-dən başqa, qalan proqramlaşdırma dillərinin hər biri yüksək səviyyəli dil adlandırılır, ancaq bu hələ onların eyni səviyyəli olması demək deyildir. Bir dilin səviyyəsi başqasının səviyyəsindən yuxarı, aşağı ola bilər.
"Yüksək səviyyəli dil" dedikdə, onun insan dilinə nə qədər yaxın olması başa düşülür. Beləliklə, insan üçün daha anlaşıqlı olan və proqramlaşdırma prosesini asanlaşdıran yeni dillər yaradılmağa başladı.
Proqramlaşdırma dillərinin müxtəlif səviyyələri var. Əsasən 5 qrupa ayrılırlar:
1. Çox yüksək səviyyəli dillər və ya vizual dillər: Access,

FoxPro, Paradox, XBase, Visual Basic.
2. Yüksək səviyyəli dillər (bunlara bəzən "alqoritmik dillər" də deyilir): Pascal, Basic, Fortran
3. Orta səviyyəli proqramlaşdırma dilləri: C, C++
4. Aşağı səviyyəli proqramlaşdırma dilləri: Assembly language
5. Maşın dili: Ən aşağı səviyyəli dil olub, 0 və 1-lərdən ibarətdir.
Bundan başqa, proqramlaşdırma dillərini 2 ayrı qrupa da bölmək olar:
1. Prosedur proqramlaşdırma dilləri (Pascal, Basic)
2. Obyekt yönümlü proqramlaşdırma dilləri (C++, Java, Smaltalk)
Hər hansı proqramlaşdırma dilini istifadə etmək üçün isə, bizə ilk növbədə kompilyator lazımdır. Kompilyator olduqdan sonra, biz öz proqramımızı mətn redaktorunda (məsələn, Notepad) yaza bilərik.
Lakin bizə daha çox IDE, yəni proqramlaşdırmanın inteqrallaşmış mühitindən istifadə edirik. Sadə dillə desək, bizə bir mühit verilir, orda komponentlər olur və bunlardan istifadə edərək, biz öz proqramımızı yazırıq.
Və ən əsası, proqramlaşdırmanı öyrənmənin yolu proqram yazmaqdır.

7. Proqramlaşdırma dillərinin səviyyələri

Dilini bilmədiyimiz insana nəyi isə başa salmaq üçün ya jestlərdən, ya da onun başa düşdüyü dildəki sözlərdən istifadə edirik. Kompüterin mərkəzi dili olan prosessorun da öz dili var.
Maşın dili.
Kompüterin bilavasitə "başa düşdüyü" yeganə dil, sadəcə ədədlər yığınından ibarət olan maşın dilidir.
"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 səlıqəsizlik, yazıda yanlışlıq, rəqəmlərin yerinin qarışdırılması gözlənilməz nəticələrə aparıb çıxarırdı. Belə yanlışlığı tapmaq da

asan iş deyildi, çünki proqramçının qarşısında az-çox aydın dildə yazılmış alqoritm deyil, rəqəmlər yığını dururdu.
Assembler: Maşın kodunda proqramlaşdırma "iynəylə yer qazmaq" kimi bir şeydir. Çox sadə hesablama məsələləri maşın kodlarında uzun-uzadı elə mürəkkəb şəkil alırdı ki, kiçik kərpiclərdən böyük bir binanın necə tikilməsini təsəvvür etmək kimi çətin idi.
Assembler 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 proqramçılar tərəfindən qavranılmasını asanlaşdırmağa, səhvlərin sayını azaltmağa, proqramçıların işini yüngülləşdirməyə imkan verdi.
Əlbəttə, Assembler proqramçını işin ən arzu olunmaz hissəsindən - operatorların əl ilə maşın koduna çevrilməsindən azad edir. Amerikalı qadın Qreys Hopper 1952-ci ildə dünyada ilk mnemonik proqramlaşdırma dili olan Assembler dilini yaratdı.
Komandaların simvollar vasitəsilə yazılış sisteminə "mnemonik yazı sistemi" deyilir. O, özündə mnemonik komandalar sistemini, prosedurlar kitabxanasını və proqram mətnlərini maşın koduna çevirmək üçün xüsusi proqramı birləşdirdi. Maşın kodunun alınmasının belə proseduru kompilyasiya (ingiliscə comple - tərtib etmək, yığmaq), onu həyata keçirən proqram isə kompilyator adlanır. Bunları da Qreys Hopper düşünüb tapmışdır.
Ancaq buna baxmayaraq, Assembler dilinin iki böyük nöqsanı var. Birincisi (ola bilsin, siz özünüz artıq sezdiniz), Assembler dilində proqramlaşdırma çox diqqət və səbr 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 İntel 8080 prosessoru üçün yazılmış 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.

8. Stuktur proqramlaşdırma.

Maşın ancaq çox sadə hesabi və məntiqi əməliyyatları yerinə yetirməyə qadirdir. Hətta çox mürəkkəb olmayan məsələlərin həlli üçün yüzlərlə,bəzən minlərlə belə əməliyyatlar tələb olunur. Aydındır ki, bu miqdarda əməliyyatlarla

işləmək insan üçün çox çətindir. Birinci növbədə ona görə ki, insan tez yorulur və nəticədə onun işində çoxlu səhvlərə yol verilir. Bu mənfi cəhət alqoritmik dildə bir neçə maşın əmrlərinin operator adlanan bir əmrlə əvəz edilməsi ilə aradan qaldırılır. Bir neçə maşın əmri elə birləşdirilir ki, o ,təbii dildə bir məqam ifadə etsin.Bunun məqsədi maşınla insan arasında kı ünsiyyəti təbii dilə yaxınlaşdırmaqdır.

Müasir alqoritmik dillərin yaradıcıları maşın əmrini bir operatorla ifadə etməklə iki məqsədi əsas tuturlar; bir tərəfdən daha çox maşın əmrlərini bir operatorda birləşdirmək və beləliklə

operatorların ümumi sayını azaltmaq istəyirlər, digər tərəfdən isə çalışırlar ki, birləşdirmə vaxtı hər bir maşın əmrinin fərdi keyfiyyətlərini saxlasınlar. Müxtəlif alqoritmik dillərin operatorlar çoxluğu bir-birindən fərqlənirlər. Bununla belə onların ümumi xassələri də vardırki,

onları aşağıdakı üç prinsiplə ifadə etmək olar;

1. Hesablanmış qiyməti proqramda təsvir olunmuş dəyişənə mənsub etmək;
2. Prosedura (altproqramı) çağırıb icra etmək;
3. Proqramda göstərilmiş başqa əməliyyatlar ardıcıllığını idarə etmək.

Birinci iki prinsipi yerinə yetirmək üçün lazım olan operatorlar bütün lazım olan operatorlar bütün alqoritmik dillər üçün təxminən eynidir. Üçüncü prinsip isə struktur proqramlaşdırma meydana gəldikdən sonra bütün alqoritmik dillər üçün eyni deyil. Struktur proqramlaşdırmanın ideyası Deykstra və Xoor tərəfindən 60-cı illərdə təklif edilib və həmin ideya proqramlaşdırmanın inkişafında böyük addım hesab edilir.

Struktur proqramlaşdırmanın əsas prinsiplərini nəzərdən keçirək.

Proqram hazırlanıb işə buraxılma prosesi bir neçə ardıcıl mərhələdən ibarətdir;

1. Məsələnin qoyuluşu və texniki tələblərin müəyyənləşdirilməsi;
2. Proqramın layihəsinin hazırlanması;
3. Proqramın alqoritmik dildə tərtibi;
4. Proqramın sazlanması və sınaqdan keçirilməsi;
5. Proqramın işə salınması.

Proqramın işinin səmərəliliyi ayrıca bir və ya bir neçə mərhələdən deyil,göstərilən mərhələlərin hamısından asılıdır. Proqramlaşdırma texnologiyasında bir sıra tövsiyyələr mövcuddur ki, onlar vasitəsilə müəyyən qaydalar əsasında proqram yazıb, hazırlamaq ardıcıllığı müəyyən edilmişdir. Belə ardıcıl struktur proqramlaşdırmanın əsasını ifadə edir. Bu əsasların başlanğıc ideası

” yuxarıdan-aşağı” prinsipi ilə proqramı tərtib etmə,başqa sözlə məsələni addım-addım dəqiqləşdirmədir. Struktur proqramlaşdırmanın əsas tələbləri ondan ibarətdir ki, proqram bir-biri işərisindəolan bloklar çoxluğu kimi tərtib

edilməlidir,blokların yalnız bir girişi və bir çıxışı olmalı və onların icrsı üçün yalnız
Iki idarə əməliyyatından dövrdən və qərar qəbul etmədən istifadə edilməlidir. Bu tələblər daxilində qurulmuş proqram sazlanma ,sınaqdan keçirilmə və buraxılma nöqteyi nəzərdən çətinlik törətmir.

9. Turbo Paskal işləmə əmrləri

Turbo Pascal 7.0 inteqrallaşdırılmış mühitində ən çox istifadə edilən əmrlər
File/New Yeni proqram faylı yaratmaq
File/Open... F3 Mövcud pascal–proqram faylını açmaq
File/Save F2 Proqram mətnini yadda saxlamaq
File/Save As... Proqramı yeni adla yadda saxlamaq
File/Exit Alt+X İnteqrallaşdırılmış mühitdən çıxmaq
Edit/Cut Shift+Del
Seçilmiş proqram fraqmentini kəsib
buferdə yerləşdirmək
Edit/Copy Ctrl+Ins
Seçilmiş proqram fraqmentini buferə
köçürmək
Edit/Paste Shift+Ins
Buferdə olan proqram fraqmentini
kursorla göstərilən mövqeyə yerləşdirmək
Edit/Clear Ctrl+Del Seçilmiş proqram fraqmentini pozmaq
Compile/Make F9
Proqramı sazlamaq (səhvləri aşkar
etmək)
Run/Run Ctrl+F9 Proqramı yerinə yetirmək
Compile/Compile Alt+F9 Proqramı kompilyasiya etmək
Debug/User
screen
Alt+F5
İstifadəçi pəncərəsini ekranda göstərmək
və ya gizlətmək
Run/Trace into F7 Proqramı sətirlərlə yerinə yetirmək
Run/Program
Reset
Ctrl+F2 Proqramın icrasından imtina etmək

Debug/Add
Watch
Ctrl+F7
Dəyişənləri müşahidə pəncərəsinə əlavə
etmək
Ctrl+Break Proqramın icrasını dayandırmaq
F10 Menyu sətrini aktivləşdirmək

10. TURBO PASKAL DILININ LÜĞƏTI

Hər hansı dilin əlifbası sonlu sayda simvollar ardıcıllığıdır. Turbo Pascal dilinin əlifbası aşağıdakı qruplara bölünür.
1. A-dan Z-ə kimi böyük və kiçik latın hərfləri.
2. 0-dan 9-a kimi onluq rəqəmlər.
3. Xüsusi simvollar: . () , # $ ^ @ * _ + " : > < =
4. Boşluq simvolu.

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

= bərabərdir	kiçikdir
<= kiçikdir və ya bərabərdir	<> bərabər deyil
> böyükdür	>= böyükdür və ya bərabərdir

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

11.Turbo Pascal alqoritmik dilin əsas elementləri

Dilin əlifbası. Turbo Pascal dilində proqram müəyyən hərflər, rəqəmlər və

simvollar vasitəsilə yazılır. Bu simvollar dilin əlifbasını təşkil edir. Dilin əlifbası ASCII kodlaşmış simvollar cədvəlinə əsaslanır və dilin baza elementi sayılır. Onun
vasitəsilə dilin digər struktur elementləri – sabitlər, dəyişənlər, ifadələr, identifikatorlar, operatorlar, altproqramlar, modullar və digər obyektlər tərtib olunur. Əlifbanın simvollarını şərti olaraq aşağıdakı qruplara bölmək olar:
1. Latın əlifbasının hərfləri və _ simvolu;

2. Onluq say sisteminin 0-dan 9-a qədər hind-ərəb rəqəmləri;

3.Onun müasir sələfi 1974-cü ildə yaradılmış İcon dilidir.

4. Bu dil XVII əsr böyük fransız alimi Paskalın şərəfinə adlandırılmışdır.

5. 1995-ci ildə dilin yeni versiyası ADA -95 yaradılmışdır.

Xüsusi simvollar və məhdudlaşdırıcılar:
+ - * / = < > > < <=>= := ( ) {} [] ^ # $ (**) , . ; ’

Cüt simvollar

: <> < = > = := (*;*);
Probel və idarəedici simvollar.
Turbo Pascal dilində probel ayırıcı funksiyasını yerinə yetirir. Ardıcıl yazılmış bir neçə probel işarəsini kompilyator bir probel kimi qəbul edir. Idarəedici
simvollar sətir və simvol tipli sabitlərin təsviri üçün istifadə olunur. Sətir və simvol
tipli sabitlərin təsvirində və proqramda yazılmış şərhlərdə həmçinin ASCII kodlaşmış simvollar cədvəlinin kodu 128-255 intervalında dəyişən simvollardan,o
cümlədən psevdoqrafika simvollarından istifadə etmək olar.

İşçi sözlər. İşçi sözlər proqramın operatorların, alt proqramların yazılmasında istifadə olunur və ad kimi istifadə oluna bilməz. Turbo Pascal dilində
aşağıdakı işçi sözlər nəzərdə tutulmuşdur:
and ,array,as,begin,div,do,file, if in,is mod,not,of,var,xor

Sabitlər və dəyişənlər. Hər bir proqram kompyuterdə yerinə yetirilən zaman verilənlər üzərində müəyyən əməliyyatlar aparır. Verilənlər iki yerə - sabitlərə və dəyişənlərə bölünür:
Dəyişənlər proqramın yerinə yetirilməsi prosesində müxtəlif qiymətlər alan

adlı kəmiyyətlərə deyilir. Dəyişənlərdən fərqli olaraq *sabitlər* əvvəlcədən məlum olan və proqramın
yerinə yetiriləcəyi müddətdə dəyişməz qalan kəmiyyətlərdir. Sabit və dəyişənlərə
onların adları ilə müraciət olunur. Sabitlər və dəyişənlər müəyyən tipə malikdir.
Tip verilənin kompyuterin yaddaşındakı ifadə forması ilə təyin edilir və onun qiymətlər oblastını, onunla aparılan əməliyyatlar coxluğunu müəyyən edir.

Sabitlər:

adsız və adlı kəmiyyət kimi proqramda istifadə oluna bilər. Adlı sabitlər və dəyişənlərin tipi mütləq proqramın təsvir bölməsində elan edilməlidir. Bu zaman sabitin tipi aşkar şəkildə proqramlaşdırcı tərəfindən verilə bilər. Əks halda kompilyator sabitin tipini onun qiymətinə əsasən müəyyən edə bilər.

1. Cüt simvollar arasında probel işarəsinin qoyulması yolverilməzdir.
2. Hər bir dəyişən proqram daxilində unikal ada malik olur.

Adsız sabitlər və adlı sabitin aldığı qiymət kimi Turbo Pascal dilində tam, həqiqi, onaltılıq ədədlər, məntiqi
FALSE
(yalan) və ya
TRUE
(doğru) sözləri,

1. simvollar, sətirlər, çoxluq konstruktorları
2. və qeyri-müəyyən göstərici əlaməti olan

istifadə edilə bilər.

İdentifikatorlar. Turbo Pascal dilində proqramlara, sabitlərə, dəyişənlərə, tiplərə, modullara, prosedura və funksiyalara ad vermək üçün identifikatorlardan
istifadə olunur. İdentifikator latın hərflərindən, “_” simvolundan və onluq say sisteminin rəqəmlərdən ibarət simvollar ardıcıllığıdır və onun birinci simvolu mütləq hərf olmalıdır. İdentifikatorun uzunluğu, yəni onu təşkil edən simvolların sayı qeyri-məhdud ola bilər. Lakin, proqram tərtib edərkən nəzərə almaq lazımdır
ki, Turbo Pascal kompilyatoru identifikatorun ilk 63 simvolunu fərqləndirir. Standart adlar və işçi sözlər identifikator olaraq işlədilə bilməz. Standart adlar standart funksiyaların, prosedurların, standart faylların və sabitlərin, tiplərin adları
ola bilər. Turbo Pascal dilində işlədilən standart adlar aşağıdakılardır:
ArcTan False Pi
Assign FilePos Port

Aux FileSize
Pos
AuxInPtr FileChar Pred
AuxOutPtr Flush

12.Verilənlərin tipləri təsnifatı.Standart tiplər.

Verilənlərin tipləri təsnifatı- verilənlərin tipləri dedikdə verilənlərin mümkün qiymətlər çoxluğu və bunlar üzərində əməllər nəzəzrdə tutulur. Paskal dilində verilənlər sadə, struktur,göstərici ,sətir və prosedur tipli olur. Sadə tiplərə nizamlı və həqiqi tiplərə aiddir. Nizamlanan tiplərə tam,məntiqi,simvol,sadalanan və diapozon tipləri aiddir. Struktura tiplərinə massivlər,yazılar,çoxluqlar və fayllar aiddir.
Turbo-paskal dilində verilənlərin aşağıdakı növləri vardır;
1. Tam tiplərə aşağıdakılar aitdir;

İşarəsiz qısa tam (byte) (1b,0+255)
Işarəli qısa tam (shortint)(1b,128+127)
Işarəli tam (word) (2b, 0/65535)
Işarəli tam (integer) (2b, 32768/32767)
Işarəli uzun tam (login) (4b,2147483648/2147483647)

2. Həqiqi tiplərə aşağıdakılar aiddir;

Siqle (4 bayt)-(7-8 rəqəm)
Real (6 bayt)-(11-12 rəqəm)
Double (8 bayt)-(15-16 rəqəm)
Extended (10 bayt)-(19-20 rəqəm)
Comp (8 bayt)-(19-20 rəqəm)

3. Məntiqi və ya bul növü (boolean).
4. Simvol
5. Skalyar və məhdud növlər
6. Sətir(string)
7. Ünvan-göstərici(pointer)
8. Mürəkkəb növlər;

Düzüm (array...of)
Yazı (record)
Çoxluq (set of...)
Obyekt (object)
Fayl (text,file of...)

Müraciət (baza tipi)
Paskal dilində proqram tərtib edərkən identifikatorlardan sabitlərə, dəyişənlərə, prosedura və funksiyalara ad vermək üçün istifadə edilir.İdentifikator hərf və rəqəmlərdən ibarətdir və onun birinci simvolu hərf olmalıdır.Standart adlar və Işçi sözlər identifikator olaraq işlənə bilməz. Standart adlar standart funksiyaların, prosedurların, standart faylların və sabitlərin təyinindən sonra gəlir. Əgər proqramda nişan və sabitlər yoxdursa, onda verilənlərin təsviri proqramda birinci yazılır.

Standart tiplər

Standart tiplərin adları əvvəldən təyin olunmuş identifikatorlardır və proqramın ixtiyari yerində iştirak edə bilər. Bu tiplər standart system modulunda təsvir olunur.
Turbo Pascalda standart tiplərə aşağıdakılar aiddir:
tam tiplər
Tam dəyişən və sabitləri təsvir etmək üçün beş tip mövcuddur ki, onların xarakteristikaları cədvəldə verilmişdir:

Tipin adı	identifikator	ədədin təsvir diapazonu	Yaddaş ölçüsü
İşarəli qısa tam	Shortint	-128.. 127	1 bayt
İşarəli tam	integer	-32768..32767	2 bayt
İşarəli uzun tam	longint	-2147483648.. 2147483647	4 bayt
Işarəsiz qısa tam	Byte	0..255	1 bayt
İşarəsiz tam	Word	0..65535	2 bayt

Həqiqi tiplər. Bu qrupa beş tip daxildir ki, bunlar cədvəldə göstərilmişdir:

Tipin adı	identifikator	ədədin təsvir diapazonu	Mantissadakı rəqəmlərin sayı	Yaddaş ölçüsü

Birqat dəqiqlikli həqiqi	Single	1.5·10-45÷3.4·1038	7..4	4 bayt
Həqiqi	real	2.9·10-39÷1.7·1038	11.. 12	6 bayt
ikiqat dəqiqlikli həqiqi	Double	5.0·10-324÷1.7·10308	15.. 16	8 bayt
Yüksək dəqiqlikli həqiqi	Extended	3.4·10-4932÷1.1·104932	19..20	10 bayt
Tam həqiqi formatda	comp	-263+1.. 263-1 və ya təqribi -9.2·10-18÷9.2·1018	19.. 20	8 bayt

13. Dəyişənlər(Tam,məntiqi,Bul)

Həqiqi dəyişənlər-həqiqi ədədlər mantissası 11 ikilik mənbə olanhəqiqi dəyişənlər 2,9E-39+1,7E+38 qiymətlər alır və bu tipli dəyişənlər yaddaşda 6 bayt yer tuturlar. Həqiqi dəyişənlər üzərində aşağıdakı əməliyyatları aparmaq olar; =; +; -; *; /;
Əgər ifadədə həm həqiqi, həm də tam dəyişən iştirak edirsə,onda tam dəyişən həqiqi dəyişənə çevrilir.
Məntiqi və ya Bul dəyişənlər-Turbo Paskalın 6.0 versiyasına kimi yalnız bir Boolean bul tipi var idi ki,bul-a iki məntiqi qiymət True(doğru) və False(yalan) qiymətlər ala bilər. Turbo Paskalın 7.0 versiyasına daha üç Byte bool,Word Bool, Long Bool Bul tipləri daxil edilib.

Tipin identfikatoru	False-nin qiyməti	True-nun qiyməti	Yaddaş ölçüsü
boolean	0 ədədi	Sıfırdan fərqli	1 bayt
Byte Bool	0 ədədi	ədəd	1 bayt
Word Bool	Hər 2 baytda 0 ədədi		2 bayt
Long Bool	Bütün baytlarda 0 ədədi		4 bayt

Bu tipə malik olan dəyişlər iki qiymət ala bilər; True(doğru) və False(yalan). Yaddasda bir bayt yer tutur.Qeyd etmək lazımdır ki,buradakı yeni bul tipləri,

Windows mühitində proqramların yaradılmasını təmin etmək üçün daxil edilmişdir. False qiymətində 0 True qiymətində isə 0 dan fərqli ixtiyari ədəd uyğundur. Bu qiymətlər üçün aşağıdakı münasibətlər doğrudur. True>False
Bul dəyişənlər üzərində aşağıdakı əməliyyatlar aparmaq olar.
Or- məntiqi toplama əməli
And- məntiqi vurma əməli
Not- inkar
Bul tip dəyişənlər üzərində müqayisə əməliyyatlarını da aparmaq olar. $\leq, \geq, <>, =, >, <$

14.Paskal dilində istifadə olunan tiplər.

Turbo Pascal dilindəki tiplər çoxluğunu iki qrupa bölmək olar;

- Standart tiplər
- İstfadəçi tərəfindən təyin olunan tiplər (iştifadəçi tiplər)

Standart tiplərin adları əvvəldən təyin olunmuş identifikatorlardır və proqramın ixtiyari yerində iştirak edə bilər. Bu tiplər standart System modulunda təsvir olunur.
İstifadəçi tipləri- əlavə abstrakt (sadə və strukturlaşmış) tiplərdir ki, xarakteristikalarını istifadəçi – proqramçı sərbəst təyin edir.
Tiplərin təsvirinin sintaksisi aşağıdakı kimidir;
Tyupe identifikator = tip
Qeyd edək ki,qabaqcadan təyin olunması tələb olunmayan tiplərə tam, həqiqi, məntiqi (Bul),
Char-simvol, string-sətir, Pchar-sətir,Pointer-göstərici tipləri və Text-mətn faylları aiddir.
Digər istifadə olunan bütün verilənlərin tipləri ya tiplərin elan olunma bölməsində təyin olunmalı, ya da dəyişənlərin və ya tipləşdirilmiş sabitlərin elan olunma bölməsində təyin
olunmamalıdır. Standart tiplərdən başqa digər tiplərin təsvirinə dəyişən,sabit və ifadələrin izahından sonra baxılacaq.

15. Sabitlər.

Proqramlaşdırma dillərində əsas anlayışlardan biri sabit anlayışıdır.Proqram yerinə yetirilmə prosesində qiymətini dəyişməyən proqram parametrləri sabitlər

adlanır.Turbo paskal dilində sabitlərin iki növü var: sadə və tipləşdirilmiş. Sadə sabitlərdə tip onun qiyməti ilə təyin olunur. Tipləşdirilmiş sabitlərdə tip hökmən göstərilməlidir.
Sadə sabitlər tam,həqiqi,simvol,məntiqi,sətir(simvol) və çoxluq tipində ola bilər. Sadə sabitlər aşağıdakı kimi təsvir olunur.
Const
Identifikator=sabitin qiyməti;
Tam sabitlər. Tam sabitlərin təsvirində yalnız işarə və rəqəmlərdən istifadə olunur. Onaltılıq say sistemindəki tam ədədin qarşısında S işarəsi qoyulur. Turbo Paskalda əvvəldən təyin edilmiş iki tam sabit mövcutdur:
Maxint=32767;
MaxLongint=2147483647;
Sətir və simvol sabitlər. Bir sətirdə yerləşməklə birqat dırnaqla hüdudlanmış simvollar ardıcıllığı sətir simvolları adlanır. Kampilyator 126 simvoldan çox olmayan sətri qəbul edir. Bir simvoldan ibarət olan sətir simvol sabiti adlanır. Əgər birqat dırnaq içərisində heç bir simvol yoxdursa,onda bu sıfır sətir adlanır. Əgər sətirdə birqat dırnaq işarəsindən istifadə etmək lazımdırsa,onda onu iki dəfə yazmaq lazımdır

16. Hesabi funksiyalar. Tipin çevrilmə funksiyaları

Standart funksiyalar Turbo Paskalda ifadələrdə hazır element kimi istifadə olunan əvvəlcədən hazırlanmış alt proqram- funksiyalar mövcutdur. Bunların sayı artırılmış və standart bir modulda yerləşdirilimişdir. Turbo Paskal proqramında sabit, tip,dəyişən prosedur və funksiyalardan istifadə edərkən ,onların təyin olunduğu modullar təsvir olunmalıdır. İstifadəçi tərəfindən yaradılan modulların və System modulunun təsviri vacib deyil. Digər modullar hökmən təsvir olunmalıdır.

Tipin çevrilmə funksiyaları:

Bu funksiyalar tipin çevrilməsi üçün nəzərdə tutulub. Məs., simvolun ədədə, həqiqi ədədin tama və s. Bunlar aşağıdakı funksiyalardır:

Chr(x) –ASCII kodunun simvola çevrilməsi

Funksiyanın arqumenti 0..255 intervalında olmaqla tam olmalıdır. Nətic bu koda uyğun simvoldur.

Məs., chr(97)-ni nəticəsi ‘a’-dır.

High(x)-kəmiyyətin maksimal qiymətinin tapılması.

Funksiyanın arqumenti sıra, sətir və massiv tipli parametr və ya identifikator ola bilər. Nəticə sıra tipi üçün bu kəmiyyətin maksimal qiyməti, massiv tipi üçün indeksin maksimal qiyməti və sətir tipi üçün sətrin təsvir olunmuş ölçüsü.

Low(x) – kəmiyyətin minimal qiymətinin tapılması.

Funksiyanın arqumenti sıra, sətir və massiv tipli parametr və ya identifikator ola bilər. Nəticə sıra tipi üçün bu kəmiyyətin minimal qiyməti, massiv tipi üçün indeksin minimal qiyməti və sətir tipi üçün isə 0-dır.

Ord(x) –istənilən sıra tipinin tam tipə çevrilməsi.

Funksiyanın arqumenti ixtiyari nizamlı (məntiqi, simvol və sadalanan) tip ola bilər. Nəticə Longint tipinin kəmiyyətidir. Məs., ord(‘a’) nəticəsi 97-dir.

Round(x)- həqiqi ədədin qiymətinin, bu ədədə yaxın olan tamadək yuvarlaqlaşdırılması.

Funksiyanın arqumenti həqiqi, nəticə isə Longint tipində olur.

Trunc(x) – həqiqi ədədin tam hissəsinin tapılması.

Funksiyanın arqumenti həqiqi, nəticə isə Longint tipində olur.

Sıra tipinin kəmiyyətləri üçün funksiyalar. Bu funksiyalar əvvəlki və ya sonrakı elementlərin tapılması, ədədin təkliyinin yoxlanması üçündür. Bu tipə aşağıdakı funksiyalar aiddir:

Odd(x) –x-in təkliyinin yoxlanması

Funksiyanın arqumenti Longint tipində, nəticə isə arqument tək olduqda True, cüt olduqda False olur.

Pred(x) – X-in əvvəlki qiymətinin təyini.

Funksiyanın arqumenti sıra tipin ixtiyari kəmiyyəti, nəticə isə həmin tipin əvvlki qiymətidir. Məs., Pred(2)-nin nəticəsi 1-dir.

Succ(x)-X-in sonrakı qiymətinin təyini.

Funksiyanın arqumenti sıra tipin ixtiyari kəmiyyəti, nəticə isə həmin tipin sonrakı qiymətidir. Məs., Succ(2)-nin nəticəsi 3-dir.

17. Əməliyyatlar.Turbo Paskal alqoritmik dilində ifadələr.

Əməliyyatlar yerinə yetirdikləri əməllərə görə aşağıdakı qruplara bölünür:

1. Hesabi əməliyyatlar:

Unar; +, -

Binar; +, _, * , /, div, mod V

2. Nisbət əməliyyatları:

= , <>, <, > , <= , >=

3. Məntiqi (Bul) əməliyyatları:

Not, and, or, xor

4. İnformasiya bitləri üzrə əməliyyatlar:

Not, and, or,xor, shl, shr

5. Sətir əməliyyatı (konkatenasiya):

6. Çoxluqlar üzrə əməliyyatlar:

_ , * ,in, <=, >=

7. Ünvan əməliyyatı:

8. @

Binar əməliyyatlar arasında əməliyyat simvolu olaraq, iki operatordan ibarət olur.

Hesabi əməliyyatlar. Hesabi əməliyyatlar yalnız həqiqi və tam tipli məsələlərə tətbiq olunur. Bunlar unar və binar əməliyyatlara bölünür.

Unar plyus+ işarəsi tam və ya həqiqi ədədin qarşısında qoyulur və həqiqi qiymətə heç bir təsir etmir.

Unar minus – işarəsi tam və ya həqiqi ədədin qarşısında qoyulur və qiymətin işarəsini dəyişdirir.

Binar hesabi əməliyyatlar və onların işarəsi cədvəldə göstərilib.

Əməl	Əməliyyatlar	Operatorun tipi	Nəticənin tipi
+	Toplama	Tam, həqiqi	Tam, həqiqi
_	Çıxma	Tam, həqiqi	Tam, həqiqi
*	Vurma	Tam, həqiqi	Tam, həqiqi
/	Bölmə	Tam, həqiqi	Həqiqi
Div	Tam bölmə	Tam	Tam
Mod	Bölmədən alınan qalıq	Tam	Tam

Nisbət əməliyyatları. Nisbət əməliyyatlarının nəticəsi **Bul** qiymətlərdir. Sətir qiymətlərinin müqayisəsi ASCII- nin simvollar koduna uyğun olaraq, soldan

sağa simvollar üzrə həyata keçirilir. Göstərici tiplərin müqayisəsində yalnız = və<> əməliyyatlarında istifadə olunur. Nisbət əməliyyatları aşağıdakılardır;

- = bərabərdir
- <> bərabər deyil
- < kiçikdir
- > böyükdür
- <= kiçik bərabərdir
- >= böyük bərabərdir

Məntiqi (Bul) əməiyyatları. Məntiqi əməliyyatlar məntiqi tip kəmiyyətlərə tətbiq
olunur və nəticə də məntiqi tipdə olur. Məntiqi (bul) əməliyyatlar(Bul) cəbrinə əsasən yerinə yetirilir .
Turbo Paskalda məntiqi ifadələrin iki növ hesablanması mövcuddur; tam və qısaldılmış
Tam hesablama onu göstərir ki, bütün ifadənin nəticəsi belə məlum olduqda da hər bir operand hesablanır. Qısaldılmış hesablama bütün ifadənin nəticəsi məlum olana kimi davam edir.
İnformasiya bitləri üzrə əməliyyatlar. Turbo Paskalda bitlər üzrə əməliyyatlarda yalnız tam tipli operandlar ola bilər. Bu əməliyyatlar operand ikilik təsvirində mərtəbələr üzrə yerinə yetirilir.
Not – tam ədədin bütün bitlərinin unar inversiya əməliyyatı
And- iki tam ədədin bitləri üzrə məntiqi və əməliyyatı
Or- iki tam ədədin bitləri üzrə məntiqi və ya əməliyyatı
Xor- iki tam ədədin bitləri üzrə istisna məntiqi və ya əməliyyatı
Shi- A shi B əməliyyatının nəticəsi ,A operandının ikilik təsvirinin B bit qədər sola sürüşdürülməsindən alınan tam ədəddir. Sürüşdürülmə nəticəsində boşalan mərtəbələr sıfırlarla doldurulur.
Shr- A shr B əməliyyatının nəticəsi, Aoperandının ikilik təsvirinin B bit qədər sağa sürüşdürülməsindən alınan tam ədəddir. Sürüşdürülmə nəticəsində boşalan mərtəbələr sıfırlarla doldurulur.
Sətir əməliyyatı(koukatenasiya) Turbo Paskal-da iki sətir və ya simvolların konkatenasiya əməliyyatında (+) simvolundan istifadə olunur. Bu əməliyyatın nəticəsində ikinci operand birinci operandın sonuncu simvolundan başlayaraq, onunla birləşdirilir. Alınan sətir 255 simvoldan çox olmamalıdır. Əgər uzunluq 255-i aşarsa, 255-cidən sonrakı simvollar atılır.
Çoxluqlar üzrə əməliyyatlar. Çoxluqlar üzərində əməliyyatlar, çoxluqlar nəzəriyyəsinin qaydalarına görə aparılır. İki çoxluğun birləşməsi, yəni A+B əməliyyatının nəticəsi,həm A çoxluğunun, həm də B çoxluğunun bütün təkrarlanmayan

elementi özündə saxlayan C çoxludur.
İki çoxluğun fərqi,yəni A-B əməliyyatının nəticəsi, A çoxluğunun B çoxluğuna daxil olmayan elementlərindən ibarət C çoxluğudur.
İki çoxluğun kəsişməsi,yəni A*B əməliyyatının nəticəsi, A və B çoxluqlarını eyni elementlərindən təşkil olunmuş C çoxluğudur.
A və B-nin elementləri eyni olduqda A=B əməliyyatının nəticəsi True və A<>B əməliyyatının nəticəsi False olur.
Əgər A çoxluğu S-nin altçoxluğudursa, onda A<=B əməliyyat nəticəsi True olur.
Əgər A çoxluğu B çoxluğunun bütün elementlərini özündə saxla; onda A>= B əməliyyatının nəticəsi True olur.
Ünvan əməliyyatı. Ünvan əməliyyatı @ işarəsi ilə göstərilir və nəticə kimi operandın göstəricisini verir. Nəticənin tipi nil göstəricisinin tipi ilə uyuşandır . @ əməliyyatında operand kimi dəyişən, prosedur funksiya və metodlardan istifadə etmək olar.
Turbo Paskal alqoritmik dilində ifadələr
İfadə- operandlardan, dairəvi mötərizələrdən və əməl işarələrdən ibarət olub, verilənlərin elementləri üzərində əməllərin yerinə yetirilməsi qaydasını müəyyən edir. İfadənin aşağıdakı növləri var;

- Hesabi ifadələr
- Müqaisə əməliyyatları ilə olan ifadələr
- Məntiqi ifadələr
- Simvol və sətir ifadələr

Hesabi ifadələr tam,həqiqi,məntiqi tiplərə malik dəyişənlərdən və aşağıdakı əməllərdən istifadə etməklə alınır.

- Hesab əməllər; +,-,,*,/
- Münasibt əməlləri; <, >, <=, >=, <>, =

Məntiq əməlləri ; not (məntiqi inkar),and (mənti vurma), or (məntiqi toplama), xor (2 moduluna görə toplama). Bəzi triqonometrik funksiyaları Paskal dilinin standart funksiyaları ilə ifadə etmək üçün aşağıdakı düsturlardan istifadə olunur

tgx = $\frac{sinx}{cosx}$

ctgx = $\frac{cosx}{sinx}$

arcsinx = arctg $\frac{x}{\sqrt{1-x^2}}$

arccosx =arcctg $\frac{\sqrt{1-x^2}}{x}$

secx =$\frac{1}{cosx}$

cos ecx = $\frac{1}{sinx}$

Mötərizələrin və əməliyyat işarələrinin köməyi ilə əlaqələndirilmiş sabitlər, dəyişənlər və standart funksiyalardan ibarət ifadələrin yazılışına nümunə göstərək;
Misal .
Z= $\sin^2 x$ funksiyasının Paskal alqoritmik dilində yazılışı aşağıdakı kimidir;
Z: = sqr (sin (x))

18.Proqramın strukturu

Pascal dilində yazılmış istənilən proqram iki hissədən – *verilənlərin təsviri bölümündən* və *proqramın gövdəsindən* ibarət olur.Proqramın mətnində verilən lərin təsviri proqramın gövdəsindən əvvəl gəlir.
Bu da dilin əsas qaydasından qaynaqlanır.Verilənlər *operatorlar* vasitəsilə emal olunur. Operatorlar barəsində növbəti dərsimizdə daha ətraflı danışacağıq.
Proqramın gövdəsi begin sözü ilə başlanır və operatorlar yığınından ibarət olur. O na 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 çalış-maq lazımdır ki, proqramlar mümkün qədər anlaşıqlı yazılsın.
Şərhlər. Proqram yazarkən siz nə etməli olduğunuzu bilirsiniz. Ancaq müəy-yən müddətdən sonra həmin proqrama qayıtmalı olsanız, qəribə olsa da, görəc ək- siniz ki, çox şeyi unutmusunuz. Buna görə də həm özünüzün xatırlamanız, həm də başqalarının sizin proqramı anlaması üçün proqramın müəyyən yerlərinə şərhlər vermək yaxşı olardı.

Adından da göründüyü kimi, *şərhlər* proqramın mətnini oxuyan şəxs üçün qeyddir. Şərhlərdən proqramın nə məqsədlə yaradıldığı, onun yaradıcısı haqqı nda məlumatı, proqramın son dəyişdirilmə tarixini, proqramdakı dəyişənlərin, funksiyaların təyinatını və s. göstərmək üçün istifadə edilə bilər.
Pascal dilində şərhlər (* və *) simvollarının, yaxud { və } fiqurlu mötə-rizələrin arasında yazılır. Proqram maşın koduna çevrilərkən bu simvollar arası nda yazılanlar nəzərə alınmır.

19. Sadə operatorlar.Şərti keçid opeatorları

Operator qoyulmuş məsələnin həlli üçün özündə verilənlər üzərində bir sıra əməlləri yerinə yetirən alqoritmik dilin əmridir. Proqram mətnində operatorlar sətirlərdə yazılır və hər bir sətir öz aralarında nöqtə vergüllə ayrılırlar. Yəni əmrin eləcə də digər əmrlərin sonu nöqtəli vergül simvolu ilə göstərilir. Əvvəlki bölümdə biz yalnız əmrlərin struktur yazılış qaydaları ilə tanış olduq. Bu bölümdə əmrlərin yerinə yetirdiyi əməliyyatların öyrənilməsi ilə məşğul olacağıq. Delphi mühitində istifadə olunan Objekt Pascal dilinin operatorlarını məntiqi olaraq sadə və strukturlaşdırılmış əmrlər qrupuna ayırmaq olar:

Sadə operatorlar: Tərkibində digər operatorlardan ibarət olmayan operatorlara *sadə operatorlar* deyilir. Sadə operatorlara mənsubetmə, şərtsiz keçid,a boş,a prosedura proqramların çağırılması operatoru aiddir.

Strukturlaşdırılmış operatorlar: Tərkibində digər operatorlardan istifadə edilən operatorlara strukturlaşdırılmış *aoperatorlar* deyilir. Strukturlaşmış operatorlar digər Begin və End ehtiyat sözləri arasında yazılmış əmrlər ardıcıllığından, şərti keçid əmrlərindən, dövrü operatorlardan və qoşma operatorlarından təşkil olunur.
Yəni strukturlaşdırılmış operatorlara budaqlanma (şərti), dövri (təkrar)a və müraciət operatorları aiddir.
Operator: Strukturlaşmış operatorlar bir-birindən nöqtə vergüllə ayrılan və *begin* və *end* ehtiyat sözləri ilə məhdudlaşan ixtiyari sayda istənilən əmrlər qrupundan ibarətdir.

Örnək:

beginaaaa
aaaaa Operator1 ;a
aaaa E;aaaa
aaaaa OperatorN ;

end;

Mənsubetmə əmri:a *Mənsubetmə operatoru* dilin əsas operatorudur. Bu ən sadə mənimsətmə əmridir. Bu əmr sağ hissədə verilmiş ifadənin hesablanmasını təyin edir və ifadənin nəticəsini, dəyişənin qiymətini və ya sabit ədədi başqa bir operatorun sol hissəsində yerləşən dəyişənə mənimsədir. Dəyişən və ifadənin tipi uyğun olmalıdır, məsələn, həqiqi və tam (və ya əks). Fayl tipindən başqa verilənlərin istənilən tipi mənsubetməyə mümkün ola bilər. Burada əmr := işarəsi ilə yazılır. Qeyd etmək lazımdır ki, У := Ф mənsubetmə işarəsi У=У işarəsindən fərqlənir və başqa məna daşıyır. Mənsubetmə işarəsi onu göstərir ki, ifadənin qiyməti əvvəl hesablanır, sonar göstərilən dəyişənə mənsub edilir.a

Dəyişən:= ifadə

Burada,aa **аУ:=Ф** mənsubetmə operatorudur, bərabərlik işarəsi deyildir. Nəzərə almaq lazımdır ki, sağ tərəfdəki ifadənin tipi sol tərəfdəki dəyişənin tipi ilə eyni olmalıdır. Yəni müxtəlif tipləri dəyişənə mənimsətmək olmaz. Belə hallarda bir tipi başqa tipə çevirən operatordan istifadə edərək hər iki tərəfi eyni tipə çevirmək lazımdır. Sonrakı bölümlərdə Delphi mühitində belə çevirmə operatorları haqqında ətraflı verəcəyik.
Dəyişənin adı əvəzinə massivin elementinin və ya yazı sahəsini göstərmək olar. Məsələn, əgər n - ədədi dəyişəndirsə və müəyyən qiymətə malikdirsə, onda düzgün konstruksiya aşağıdakı kimi olacaqdır:

Const
aaaaaaaa aPi=3.14a ;
Var aaax, y :aaaaaaa real;
aa aaaaaa n :aaaaaaaaaaaa integer;
aaaaaaaaa s :aaaaaaaaaaaa string;
Begin
aaaaaaaa n := 17 * n Ц 1;
aaaaaaaaa t := CTarixT + DateToStr (Date);
aaaaaaaa y := -12.3 * sin (pi / 4);
a aaaaaaak := 23.789E+3;
End;

Şərti keçid operatorları verilmiş şərtdən asılı olaraq hər hansı operatorun yaxud operatorlar qrupunun yerinə yetirilməsini təmin edir. Şərti keçid operatoru aşağıdakı yazılış formatına malikdir. If< *şərt* > then < $operator\_1$ > else< $operator\_2$ >;

Burada $< şərt >$ məntiqi ifadədir. Operator yerinə yetirilərkən məntiqi ifadənin qiyməti doğru olarsa,onda $< operator\_1 >$,əks halda $< operator\_2 >$ yerinə yetirilir. Burada $< operator\_1 >$ və $< operator\_2 >$ tək bir operatordan və ya begin və end arasında yerləşən operatorlar qrupundan ibarət ola bilər.

20. Seçmə operatoru.

Seçim və ya variant operatoru şərti keçid operatorunun ümumiləşdirilməsidir. Bu operator yerinə yetirildikdə mümkün variantlardan biri yerinə yetirilir. Operator selektor adlanan ifadədən və seçim üçün istifadə olunan nişanlanmış operatorlar siyahısından ibarətdir. Şərti keçid operatorunda olduğu kimi burada da else işçi sözü işlənə bilər. Seçim və ya variant operatorunun yazılış formatı aşağıdakı kimi olur;

Tam formatda yazılışı

Case $< selektor - ifadə >$ of
<nişan 1> $< operator\ 1; >$
<nişan 2> $< operator\ 2; >$
.....
<nişan n> $< operator\ n; >$
else $< operator\ >$
end;

Qısa formatda yazılışı

Case $< selektor - ifadə >$of
<nişan 1> $< operator\ 1; >$
<nişan 2> $< operator\ 2; >$
.....
<nişan n> $< operator\ n; >$
End;

Burada <nişan 1>, <nişan 2>,.....,<nişan n> selektorun qiymətləridir. Seçim və ya variant operatoru aşağıdakı kimi işləyir. Proqram yerinə yetirildikdə əvvəlcə selektor-ifadənin qiyməti hesablanır,sonra isə selektor Seçim və ya variant operatorları ifadənin qiymətinə bərabər olan nişan müəyyənləşdirilir və ona uyğun olan operator yerinə yetirilir. Selektorun qiymətinə bərabər olan sabit olmadıqda else xidməti sözündən sonra yazılan operator,operatorun qısa yazılışında isə operatordan sonra gələn operator yerinə yetirilir.Qeyd edək ki,həqiqi və sətir tipinə aid olan dəyişənləri selektor olaraq istifadə etmək olmaz.

21.Parametrli dövr operatoru

Dövr operatorları proqramda dövrü alqoritmləri təsvir etmək üçün istifadə olunur. Turbo Pascal dilində üç dövr operatoru vardır.

1. Parametrli dövr operatoru.
2. Sonrakı şərtli dövr operatoru.
3. İlkin şərtli dövr operatoru.

Parametrli dövr operatoru- dövrlərin sayı məlum olduqda parametrli dövr operatorundan, əks halda isə sonrakı şərtli dövr operatorundan və ya ilkin şərtli dövr operatorundan istifadə olunur.

1. for$< dövr\ parametri >:=< n > to\ < m > do < operator >;$

Burada $< n >$ və $< m >$ dövr parametrinin müvafiq olaraq başlanğıc və son qiymətlərini müəyyən edən ifadələrdir. Dövr parametri , $< n >$ və $< m >$ ifadələrinin eyni tipli qiymətlər almalıdır və bu qiymətlər həqiqi tipə aid ola bilməzlər. $< operator >$ dövrün gövdəsi adlanır və sadə-tək bir operatordan və mürəkkəb operatordan –begin və end arasında yerləşən operatorlar qrupundan ibarət ola bilər. Dövr operatoru yerinə yetirildikdə dövrün parametrinin aldığı cari qiymət parametrin son qiymətilə müqayisə olunur. Cari qiymət son qiymətdən kiçik olduqda dövrün gövdəsi yerinə yetirilir,dövr parametrinin qiyməti vahid qədər artır və bu qiymət yenidən parametrin son qiymətilə müqayisə olunur. Bu proses parametrin cari qiymətinin son qiymətdən böyük olana qədər davam edir. Bundan sonra dövr operatorundan sonra yazılan operator yerinə yetirilir.

2. for$< dövr\ parametri >:=< n > downto\ < m > do < operator >;$

Bu yazılışda $< n >$ və $< m >$ dövr parametrinin müvafiq olaraq başlanğıc və son qiymətlərini müəyyən edən adsız sabitlər və ya ifadələrdir,eyni tipli olmalıdır və həqiqi tipə aid ola bilməzlər. $< operator > dövrün\ gövdəsi\ olub,$ sadə-tək bir operatordan və mürəkkəb operatordan begin və end arasında yerləşən operatorlar qrupundan ibarət ola bilər. Dövr operatoru yerinə yetirildikdə dövr parametrinin aldığı cari qiymət parametrin son qiymətilə müqayisə olunur. Cari qiymət son qiymətdən böyük olduqda dövrün gövdəsi yerinə yetirilir,dövr parametrinin qiyməti vahid qədər azalır və bu qiymət yenidən parametrin son qiymətilə müqayisə olunur.Bu proses parametrin cari qiymətinin son qiymətdən kiçik olana qədər davam edir.Bundan sonra dövr operatorundan sonra yazılan operator yerinə yetirilir.

22.Ön şərtli dövr operatoru.

İlkin şərtli dövr operatoru sonrakı şərtli dövr operatoruna oxşar əməliyyatı yerinə yetirir. Fərq yalnız odur ki, dövrün qurtarmasını müəyyən edən şərt dövrün gövdəsindən əvvəl gəlir. Yazılış formatı aşağıdakı kimidir: while do ; Burada məntiqi ifadə, isə sadə və ya mürəkkəb operatordur. Operator yerinə yetirildikdə ilk öncə əvvəl məntiqi ifadənin qiyməti hesablanır. İfadə true qiyməti aldıqda dövrün gövdəsini təşkil edən operatorlar yerinə yetirilir və yenidən məntiqi ifadənin qiyməti hesablanır və ifadə true qiyməti aldıqda proses təkrarlanır. Bu proses məntiqi ifadə false qiyməti alana qədər dövrü olaraq davam edir. Bundan sonra proqramda dövr operatorundan sonra gələn operator yerinə yetirilir. Qeyd edək ki, -in qiyməti yalan (false) olarsa, onda dövrün gövdəsini təşkil edən operatorlar bir dəfə də olsun yerinə yetirilmir. Lakin sonrakı şərtli dövr operatorunda dövrün gövdəsindən asılı olmayaraq ən azı bir dəfə yerinə yetirilir. Sonrakı şərtli dövr operatorunda olduğu kimi, ilkin şərtli dövr operatorunda da gövdəsini təşkil edən operatorlardan biri elə olmalıdır ki, o dövrün qurtarması şərtinə təsir edə bilsin.

23.Son şərtli dövr operatoru.

Sonrakı şərtli dövr operatoru. Sonrakı şərtli dövr operatoru başlıqdan -repeat, dövrün gövdəsindən və dövrün qurtarmasını müəyyən edən şərtdən ibarətdir və aşağıdakı yazılış formatına malikdir: Repeat Until ; Burada şərt məntiqi ifadədir. Operator yerinə yetirilərkən əvvəlcə repeat və until xidməti sözləri arasında olan operatorlar yerinə yetirilir, sonra isə dövrün qurtarması şərti yoxlanılır. Əgər məntiqi ifadənin qiyməti false olarsa, onda dövrün gövdəsini təşkil edin operatorlar təkrar yerinə yetirilir. Əgər məntiqi ifadənin qiyməti true olarsa, onda dövr sona çatır və dövr operatorundan sonrakı operator 361 yerinə yetirilir. Dövrün gövdəsini təşkil edən operatorlardan biri elə olmalıdır ki, o dövrün qurtarması şərtinə təsir edə bilsin. Əks halda, dövretmə sonsuz olaraq davam edə bilər.

İmtahan sualları:

1.Alqoritm nədir?
2.Alqoritmin icraçısı.
3.Alqoritmin xassələri.
4.Diskretlik xassəsi.
5.Müəyyənlik xassəsi.
6.Kütləvilik xassəsi.
7.Sonluluq və nəticəvilik xassəsi.
8.Həll alqoritminin təsvir üsulları.
9.Həll alqoritmi.
10.Alqoritmin təsvir üsulları.
11.Tipik alqoritmik strukturlar.
12.Tipik hesablama proseslərinin alqoritmləşdirilməsi.
13.Alqoritmin sözlə təsvir üsulları.
14.Alqoritmik dillə təsvir üsulu.
15.Sxemlə təsvir üsulu.
16.Budaqlanan alqoritmik strukturlar.
17.Dövrü alqoritmik strukturlar.
18.Arqumenti monoton dəyişən funksiyanın qiymətlər çoxluğunun hesablanması.
19.Sərbəst dəyişən arqumentli dövri hesablama prosesləri.
20.Mürəkkəb dövri prosessor.
21.Proqramlaşdırma dilləri.
22.Alqoritmik təsvir üsulları.
23.Proqramlaşdırmanın dilləri.
24.Proqramlaşdırma dillərinin səviyyələri.
25.Prosedur proqramlaşdırma.
26.Məntiqi proqramlaşdırma.
27.Funksional proqramlaşdırma.
28.Obyekt-yönlü proqramlaşdırma.
29.Hadisə-yonlü proqramlaşdırma.
30.Vizual proqramlaşdırma.
31.Dilin əlifbası və elementləri.
32.Turbo Paskal dilində proqramlaşdırma.
33.Turbo-Paskal işləmə əmrləri.
34.Turbo- Paskalla dilin lüğəti.
35. Turbo-Paskal dilinin elementləri.
36.Turbo-Paskal dilinin əlifbası.

37.İdentifikatorlar.
38. Struktur proqramlaşdırma.
39.Standart tiplər.
40.Tiplərin təsnifatı.
41.Verilənlərin tipi.
42.Tam tiplər.
43.Həqiqi tiplər.
44.Məntiqi(Bul) tip.
45.Simvol tipi.
46.Sətir tipləri.
47.Göstərici tip.
48.Mətn tipi.
49.Dəyişənlər.
50.Sətir və simvol tipli dəyənlər.
51.Sətir tipli dəyişənlər.
52.Simvol tipli dəyişənlər.
53.Sabitlər.
54.Sadə sabitlər.
55.Sətir və simvol tipli sabitlər.
56.Tipləşdirirlmiş sabitlər.
57.Standart funksiyalar.
58.Tipin çevrilmə funksiyaları.
59.Əməliyyatlar.
60.Hesabi əməliyyatlar.
61.Nisbət əməliyyatları.
62.Məntiqi(Bul) əməliyyatlar.
63.İnformasiya bitləri üzrə əməliyyatlar.
64.Sətir əməliyyatı(konkatensiya).
65.Çoxluqlar üzrə əməliyyatlar.
66.Ünvan əməliyyatı.
67.Proqramın strukturu.
68.Proqramın parametrləri.
69.Sadə operatorlar.
70.Operatorlar.
71.Mənsubetmə operatoru.
72.Prosedura müraciət.
73.Şərtsiz keçid operatoru.
74.Baş operator.
75.Seçmə operatoru.

76.Parametrli dövr operatorlar.
77.Ön şərti dövr operatoru.
78.Son şərti dövr operatoru.

ƏDƏBİYYAT:

1. M.Alışov, Ə.Pələngov, Q.Əliyev “İnformatika” Bakı-2005
2. N.Cəfərov, N.Rəhimova“İnformatika” Bakı-2013
3. N.Mahmudov “Fərdi kompüterdə proqramlaşdırma” Bakı-1995