3. **Alqoritmləşdirmə və proqramlaşdırmanın əsasları**

3.1. Alqoritm və proqram anlayışı
3.2. Alqoritmin xassələri
3.3. Alqoritmin təsvir vasitələri
3.4. Alqoritmin tipləri
3.5. Proqramlaşdırma dilləri

3.1. Alqoritm və proqram anlayışı. Gündəlik həyatımızda hər hansı bir işi icra edərkən bir sıra qayda və qanunları yerinə yetiririk. Bir misala baxaq. Tutaq ki, biz kofe hazırlamaq istəyirik. Onda biz, ardıcıl aşağıdakı işləri yerinə yetirməliyik.

1. Suyu qaynadırıq;
2. Qaynar suyu bir stəkana tökürük;
3. Stəkana lazımı qədər kofe tökürük;
4. Lazımı miqdarda şəkər tozu əlavə edib qarışdırırıq;

Dörd bənddən ibarət bu hərəkətlər ardıcıllığı kofe hazırlamaq alqoritmidir.

Alqoritm – latınca qayda-qanun deməkdir. Alqoritm sözü IX əsrin məşhur özbək riyaziyyatçısı Məhəmməd İbn Musa əl-Xarəzminin (yəni Xarəzmli Musa oğlu Məhəmməd) adının latın hərflərilə olan "alqoritmi" yazılışıyla bağlıdır. Əl-Xarəzminin yazdığı traktatın XII 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 olan əməliyyatları müəyyən edən və onların hansı ardıcıllıqla yerinə yetirilməsini göstərən formal yazılışdır.

Hesablama maşınlarının əsas fərqləndirici xüsusiyyətlərindən biri də onun proqramla idarə olunmasıdır. Yəni, istər sadə, istərsə də mürəkkəb məsələni maşının həll etməsi üçün proqram tərtib edilməlidir.

Proqram - maşının addım-addım yerinə yetirəcəyi təlimatlar və yaxud əmrlər toplusudur. Hər bir proqram tərtib edilərkən müəyyən bir alqoritmdən istifadə edilir. Yəni, proqram hər bir alqoritmi maşının başa düşəcəyi formada ifadə edir. Başqa sözlə *proqram* – maşının girişinə verilən informasiyaları çıxış informasiyalarına çevirən, xüsusi şəkildə tərtib olunmuş sonlu sayda ardıcıl əmrlərdən ibarət alqoritmdir.

3.2. Alqoritmin xassələri. Məsələnin maşında həlli üçün tərtib edilən alqoritm bir çoxşərtləri ödəməlidir. Bu şərtlərə alqoritmin xassələri deyilir. Həmin xassələr aşağıdakılardır:

1. Alqoritm sonlu sayda mərhələdən sonra qurtarmalıdır. Buna, alqoritmin *sonluluq* xassəsi deyilir.
2. Alqoritmin hər bir addımı dəqiq və birqiymətli təyin olunmalıdır. Bu alqoritmin *müəyyənlik* xassəsidir.
3. Alqoritmin müəyyən sayda giriş qiymətləri (məsələnin başlanğıc şərtləri) olmalıdır. Bu şərtlər proqram icra olunmamış və ya olunduqca maşına daxil edilə bilər.
4. Alqoritmin yerinə yetirilməsi nəticəsində giriş qiymətlərindən asılı olan bir və ya bir neçə çıxış qiymətləri alınmalıdır.
5. Alqoritm sadə və səmərəli olmalıdır, yəni alqoritmin nəticəsi (cavabı) mümkün qədər sadə əməliyyatlar vasitəsilə və ən qısa yolla alınmalıdır.
6. Alqoritm ümumi olmalıdır, yəni müəyyən məsələ üçün tərtib olunmuş alqoritm, həmin tipdən (sinifdən) olan bütün məsələlər üçün yararlı olmalıdır. Bu alqoritmin *kütləvilik* xassəsidir

3.3. Alqoritmin təsvir vasitələri. Alqoritmi mümkün qədər əyani şəkildə göstərmək üçün aşağıdakı təsvir vasitələrindən istifadə olunur:

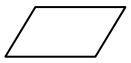
1. **Adi dildə;**
2. **Blok-sxemlə;**
3. **Alqoritmik dildə.**

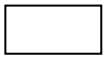
1. Alqoritmin adi dildə təsviri (nəqli). Bu zaman əməliyyatlar, icra olunacaq hərəkətlərin nəqli şəkildə ardıcıl sadalanması kimi verilir. Məsələn, kofenin hazırlanmasını ifadə edən alqoritmin təsviri buna misal ola bilər.

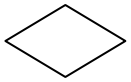
2. Alqoritmin blok-sxem təsviri. Mürəkkəb alqoritmlərin təsviri zamanı blok-sxemlərdən istifadə olunması daha geniş yayılmışdır, çünki bu halda alqoritmin blok-sxem şəklində təsviri daha əyani olur. Bu zaman, adətən alqoritmin bir addımına bir blok uyğun olur. Lakin bir blokda bir neçə eyni tipli mərhələ və ya bir mərhələ bir neçə blokda təsvir oluna bilər. Bloklar standart işarələr şəklində ifadə olunur və bir-birləri ilə şaquli və ya üfüqi xətlərlə birləşdirilir. Birləşdirici xətlərin uclarında istiqaməti göstərən ox işarəsi qoyulur.

Alqoritmin blok-sxem vasitəsilə təsviri zamanı istifadə olunan əsas standart simvollar aşağıdakılardır:

 - proqramın (alqoritmin) başlanğıcı və sonu;

 - giriş-çıxış əməliyyatları;

 - hesablama bloku;

 - keçid (budaqlanma);

 - çapetmə; və s.

3. Alqoritmin proqramlaşdırma dili vasitəsilə təsvir edilməsi (alqoritmik dildə). Alqoritmin proqramlaşdırma dilində təsviri, maşının icra edəcəyi hər bir kiçik əməliyyatın müəyyən əmrlərlə göstərilməsindən ibarətdir. Proqramlaşdırma dili vasitəsilə təsvir çox dəqiq olmalıdır, çünki maşın ancaq ona verilmiş proqramdakı əmrləri icra edə bilir. Çox vaxt proqramı yazmamışdan əvvəl məsələnin həll alqoritminin blok-sxemini qururlar, sonra isə ona uyğun proqram yazılır. Alqoritmin proqramlaşdırma dili vasitəsilə təsviri onun ixtiyari proqramlaşdırma dilində yazılmasının mümkünlüyünü göstərir. Yəni tam kvadrat tənliyin alqoritmini istənilən proqramlaşdırma dili vasitəsilə təsvir edə (proqramlaşdıra) bilərik. Proqramlaşdırma dilləri haqqında bir qədər sonra məlumat veriləcək. Tam kvadrat tənliyin həll alqoritminin (a-A, b-B, c-C olmaqla) FORTRAN dilində yazılışını belə göstərmək olar:

```
PROGRAM TKT
READ (5,1) A,B,C
1 FORMAT (3F8.2)
D=B**2- 4. *A*C
İF (D=O) 2,3,4
2 WRİTE (6,10)
10 FORMAT (2X,"həlli yoxdur")
GO TO 11
3 X= - B/(2. *A)
WRITE (6,20) X
```

```
20 FORMAT (2X,F10.4)
GO TO 11
4 X1= -B+SQRT (D)
X2= -B+SQRT (D)
WRITE (6,30) X1,X2
30 FORMAT (2X, F10.4 , 5X, F10.4)
11 CONTİNUE
END
```

3.4. **Alqoritmin tipləri.** EHM- də müxtəlif tipli məsələləri həll edərkən əsasən üç tipli alqoritmlərdən istifadə olunur: ***xətti (düz), budaqlanan və dövri***.

Xətti alqoritmlər sadə hesablama prosesini ifadə edən bir neçə ardıcıl əməliyyatlardan ibarət olur və onlar yazıldığı ardıcıllıqla da icra olunur.

Misal 1. İkiməchullu birdərəcəli iki tənlik sisteminin həlli alqoritminin blok-sxemini tərtib etməli.

$$\begin{cases} a_1x + b_1y = c_1, \\ a_2x + b_2y = c_2 \end{cases}$$

Kramer qaydasına görə D= $a_1b_2 - a_2b_1$,

$x = \frac{c_1b_2 - c_2b_1}{D}$, $y = \frac{a_1c_2 - a_2c_1}{D}$ olur. (Sxem 1).

Budaqlanma alqoritmlərin tərkibində bir və ya bir neçə məntiq mərhələsi olur. Bu mərhələdə müəyyən kəmiyyətlərin hər hansı bir şərti ödəyib-ödəmədiyi yoxlanılır və ona uyğun olaraq sonrakı gedişin istiqaməti seçilir. Yəni nəzərdə tutulan şərt ödənilirsə, bir istiqamətə, həmin şərt ödənilmirsə, başqa istiqamətə doğru hərəkət edilir. Beləliklə, alqoritmdə budaqlanma baş verir.

Misal 2. $a\ x^2 + bx + c = 0$ tam kvadrat tənliyin həll alqoritminin blok-sxemini qurun. (Sxem 2).

Proqramlaşdırmada tez-tez eyni əməliyyatlar qrupunun çoxlu sayda təkrar olunması lazım gəlir. Bu halda ***dövr*** alqoritmindən istifadə olunur. Dövrlər sadə və mürəkkəb olur. Sadə dövrlü alqoritmin bir dövrü olur. Əgər hər hansı bir alqoritmdə bir neçə daxili dövr iştirak edirsə, onda belədövrlərə mürəkkəb dövr deyilir. Mürəkkəb dövrləri əmələ gətirən sadə dövrlər kəsişə bilməz.

Misal 3. $S = x_1 + x_2 + ... + x_{20}$ cəmini hesablayan alqoritmin blok-sxemini qurmalı. (Sxem3).

Misal 4. y= $\sum_{i=1}^{n} \prod_{j=1}^{m} x_{ij}$ ifadəsini hesablayan alqoritmin blok-sxemini qurmalı. (Sxem 4).

3.5. Proqramlaşdırma dilləri. Dil dedikdə – istənilən işarələr sistemini başa düşəcəyik. İşarə dedikdə isə informasiyanı ötürmək üçün xüsusi seçilmiş bir obyekt nəzərdə tutulacaq. Məsələn: kağızdakı hərf, əl işarəsi, sifətin ifadəsi, çeviricinin vəziyyəti və s. Bu çox ümumi bir tərif olduğundan bura heyvanların təbii dillərini, insanların dillərini də daxil etmək olar. Müxtəlif dillərinolması bir dildən başqa bir dilə tərcümə (translyasiya) problemini yaradır. Belə baxanda, bizim bütün fəaliyyətimiz bu və ya başqa mənada translyasiya ilə bağlıdır. Biz şüurumuzda olan daxili təsəvvürləri nitqə, hərəkətə və s. çeviririk və tərsinə.

Yeni obyekt yaratmaqla, biz onunla bağlı yeni bir dil də yaradırıq. Məsələn, elmin inkişafı xüsusi elmi dilin yaranmasına gətirib çıxartdı (riyaziyyat, fizika və s.).

Bir qayda olaraq, sadə qurğuları, düymələri basmaqla, dəstəkləri hərəkətə gətirməklə idarə

etmək heç də çətin olmur. Amma elə mürəkkəb obyektlər var ki, onları idarə etmək üçün insan ya çətinlik çəkir, ya da ola bilsin ki, heç bacarmır. Bu halda, insanla maşın arasında, bir növ vasitəçi –

tərcüməçiyə ehtiyac olur. Belə qurğulara EHM–ləri misal gətirmək olar. Onları idarə etmək üçün insan üçün sadə olan, onun dilinə yaxın olan bir giriş dilinə ehtiyac olur.

Qeyd edək ki, EHM -lər insan tərəfindən yaradılmış obyektlər sırasında ən çətin idarə olunanlardandır. Bu məsələ o qədər obeyktin mürəkkəbliyindən yox, daha çox onun dilinin (maşın dili – 2-lik kodlar) insan dilindən uzaq olmasıdır. Ona görə də istehsalçıların və proqramçılarin maraqlarını nəzərə alaraq bir ünsiyyət vasitəsinə ehtiyac yaranır.

Proqramlaşdırma dilləri adi dillərdən "sözlərin" (ancaq translyatorun başa düşdüyü) sayına və əmrlərin ciddi yazılış qaydasına görə fərqlənir. EHM-də proqram yazmaq üçün istifadə olunan formallaşmış dillərə *proqramlaşdırma dilləri* deyilir.

Istənilən proqramlaşdırma dilinin əsas elementləri bunlardır: dilin əlifbası, sintaksisi və semantikası.

Dilin əlifbası dedikdə, həmin dildə işlənən bütün simvollar nəzərdə tutulur.

Sintaksis- əlifbada olan simvollardan dilin ayrı-ayrı konstruksiyalarının (komandaların, operatorların) düzəldilməsinin formal qaydalarıdır. Bu qaydalar müxtəlif həll alqoritmlərini proqramlaşdırmağa imkan verir.

Semantika- dilin bu və ya digər sintaksis konstruksiyalarının təsviridir. Məsələn, əgər proqramın bu yerində ***y =a*(b+c)*** ifadəsinin hesablanması yazılıbsa, onda semantika qaydaları maşına "göstərir" ki, əvvəlcə (b+c) cəmini tapsın, sonra həmin cəmi ***a***-ya vursun.

Beləliklə, hər hansı verilənlərin emalı prosesini birbaşa həyata keçirməyə imkan verən proqramlar, dili təyin edən sintaksis qaydalara uyğun olaraq əlifbadakı simvolların birləşməsi nəticəsində və semantika qaydalarını nəzərə almaqla işlənib hazırlanır.

Translyator və kompilyatorlar. Proqramlaşdırma dili vasitəsilə hazır proqram yox, ancaq qurulmuş alqoritmi təsvir edən mətn yaradılır. İnsanın başa düşdüyü dildə olan bu proqram maşının başa düşdüyü dilə çevrilməlidir. Bunun üçün kompüterdə translyatorlar və kompilyatorlar olur.

Proqram ancaq onların translyatorları olan halda icra oluna bilərlər. Translyatordan fərqli olaraq kompilyatorlar exe-faylların yaradılması üçün istifadə olunur ki, onlar da sərbəst icra oluna bilərlər (yəni, proqpamin yazildiği mühitdən (sistemdən) asılı olmadan).

Proqramlaşdırma dillərinin səviyyələri. Müxtəlif tip prosessorlar müxtəlif tip əmrlər sisteminə malikdir. Əgər proqramlaşdırma dili konkret prosessor tipinə yönəlibsə və onun xüsusiyyətlərini nəzərə alırsa, onda ona aşağı səviyyəli proqramlaşdırma dili deyirlər. Assembler aşağı səviyyəli proqramlaşdırma dilidir. Çünki o, bir əmri mnemonika adlanan simvol işarəmələrinin köməyilə ədədlər şəklində yox, maşın kodları şəklində verir. Assemblerin köməyilə çox səmərəli və kompakt proqramlar yaratmaq mümkündür. Assemblerdən adətən, sistem əlavələrin, drayver-proqramların, kompüterin aparat resurslarına müraciət edən proqram modullarının hazırlanması üçün istifadə olunur. Aşağı səviyyəli proqramlaşdırma dillərindən, adətən yüksək səviyyəli peşəkar proqramçılar istifadə edir. Bu dillərdə tutulan proqramlar yaddaşda az yer tutmaqla yanaşı, daha sürətlə icra olunurlar. Yüksək səviyyəli proqramlaşdırma dilləri isə adi dilə daha yaxın və insan üçün daha aydın başa düşüləndir.

Çox yayılmış, bəzi proqramlaşdırma dilləri haqqında məlumat verək.

Fortran - Cim Bekus tərəfindən 1954-cü ildə yaradılmış ilk kompilyasıya olunan proqramlaşdırma dilidir. Bu dildə ilk dəfə olaraq proqramlaşdırmanın ən vacib anlayışları realizə olunmuşdur. Ondan bütün dünyada istifadə olunur. 2000-ci ildə onun yeni F2k versiyası yaradılmışdır. Fortrandan əsasən riyazi, texniki məsələlərin proqramlaşdırılması üçün istifadə olunur.

Cobol - 1960-cı illərdə yaradılıb. Əsasən iqtisadiyyat sahəsində biznes məsələlərinin həlli üçün nəzərdə tutulan kompilyasiya olunan bir dildir. Bu dil özünün "çoxsözlülüyü" ilə fərqlənir. Bəzən onun əmrləri ingilis ifadələrindən heç də fərqlənmir. Bu dildə, hal-hazırda aktiv surətdə istismar olunan çoxlu sayda əlavələr hazırlanmışdır.

Pascal - 1970-ci illərin sonunda Niklaus Virt tərəfindən yaradılmışdır. Ondan böyük layihələrin hazırlanmasında müvəffəqiyyətlə istifadə etmək üçün imkanlar nəzərdə tutulmuş, proqramın strukturuna olan tələblər gücləndirilmişdir.

Basic – Bu dil üçün həm kompilyatorlar, həm də interpretatorlar mövcuddur. İlk dəfə proqramlaşdırmanı öyrənmək məqsədilə 1960-cı illərdə yaradılan bu dil dünyada ən geniş yayılmış dillərdəndir.

C - Bu dil yaranan gündən kütləvi istifadə üçün nəzərdə tutulmamışdır. Sadəcə, assembleri əvəz edən bir proqram kimi planlaşdırılmışdır. Yəni o, həm assembler kimi effektiv və kompakt proqramlar yaratmaq imkanına malik olmaqla yanaşı, həm də konkret tip prosessorlardan asılı olmalı idi.

C++ - C-nin obyektyönümlü genişlənməsi olan bu dil 1980-cı ildə yaradılmışdır. Bu dildə proqramçının məhsuldarlığını kəskin şəkildə artıra biləcək çoxlu sayda imkanlar nəzərdətutulmuşdur.

Java - Bu dil 1990-cı illərin əvvəlində C++ dilinin əsasında yaradılmışdır. O, C++-da olan bütün aşağı səviyyəli imkanları aradan çıxardaraq əlavələrin işlənib hazırlanmasını sadələşdirməyə yönəlmişdir.

Baza verilənlərin proqramlasdırma dilləri. Bu qrup dillər alqoritmik dillərdən həll etdiyi məsələlərə görə fərqlənir. İlk bazalar böyük informasiya massivlərinin emalına və müəyyən əlamətə görə bir qrup informasiyanın seçilməsinə ehtiyac olanda yaradılmışdır. Bunun üçün strukturlaşdırılmış sorğular dili SQL (structured query language) dili yaradılmışdır. O güclü riyazi nəzəriyyəyə əsaslanmaqla verilənlər bazasını effektiv emal etməyə imkan yaradır.

Böyük verilənlər bazalarını idarə etmək, onları effektiv emal etmək üçün VBİS (verilən bazasının idarəetmə sistemi) yaradıldı. Hal-hazırda dünyada 5 aparıcı VBİS istehsalçısını göstərə bilərik: Microsoft (SQL Server), İBM (DB2), Oracle Software AG (Adabas), İnformix və Sybase. Onların məhsulları şəbəkədə minlərlə istifadəçinin eyni zamanda işini dəstəkləyir, verilənlər bazaları isə paylanmış şəkildə bir neçə serverdə saxlanıla bilər.

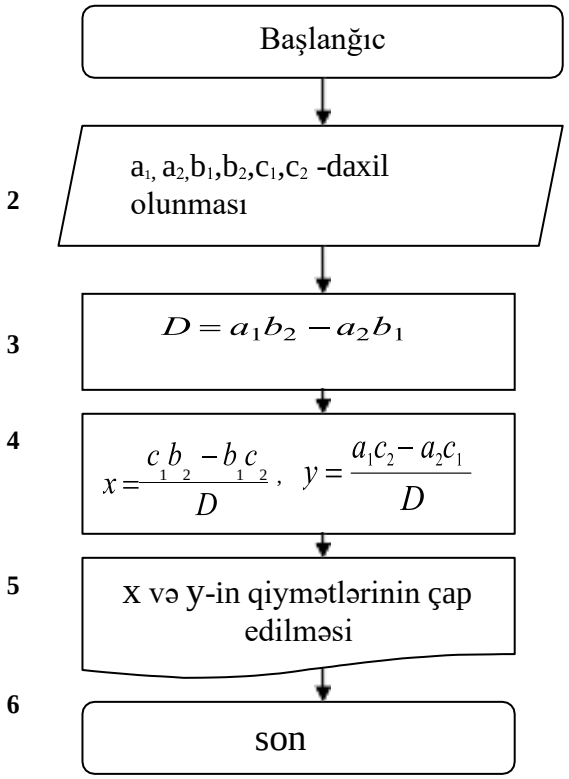
İnternet üçün proqramlasdırma dilləri. HTML (HiperText Markup Language) çox tanınmış, sadə bir dil olmaqla, mətnin formatlaşdırılması, şəkillərin əlavə olunması, şiriflərin və rənglərin verilməsi, istinadların və cədvəllərin tərtib olunması kimi elementar əmrlərə malikdir. Çox WEB-səhifələr HTML- də və onun genişləndirilmiş variantlarında yazılmışdır.

1980-cı illərdə ***PERL*** dili işlənib hazırlandı. O, böyük mətn fayllarının effektiv emal olunması, mətn hesabatlarının qenerasiya olunması və məsələlərin idarə olunması vasitəsi kimi yaradılmışdı. Onun sətrlərlə, massivlərlə işləmək funksiyaları, verilənlərin çevrilməsi vasitələri, sistem informasiya ilə işləmək imkanları da çox genişdir.

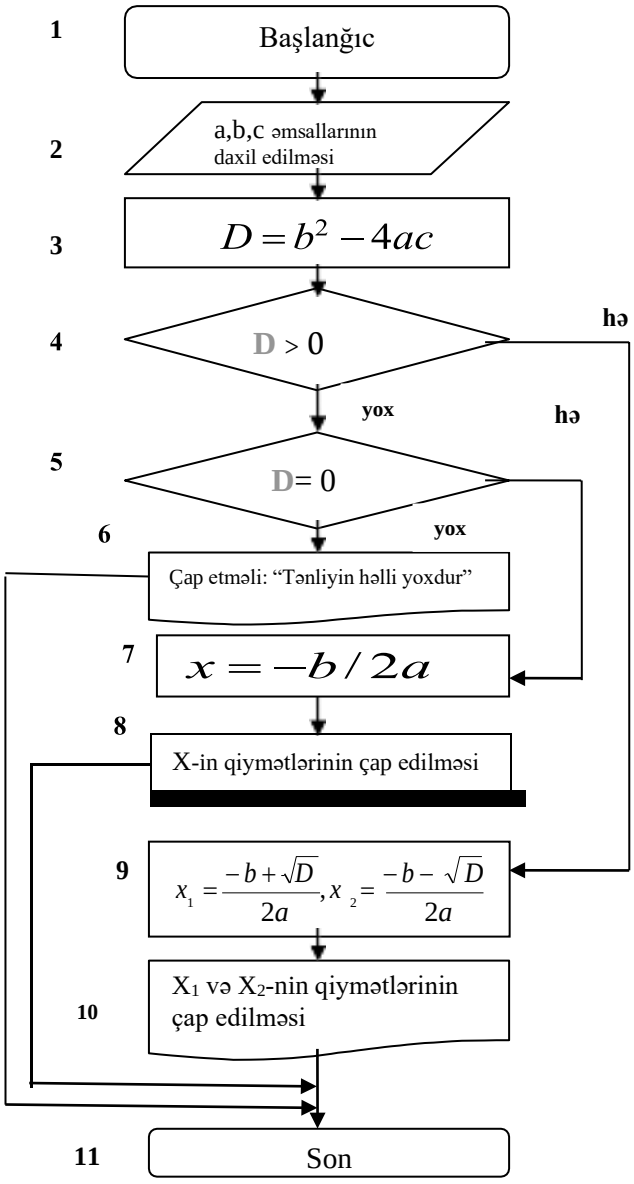
Windows əməliyyatlar sisteminin yaranması ilə bağlı bu mühitdə işləyən yeniproqramlaşdırma dilləri də meydana çıxmışdır. Bunlardan ***Visial Basic, Visial Delphy, Visial C*** və b. göstərmək olar. Bu proqramların köməyilə müxtəlif tip praktik məsələləri həll etmək olar.“Visial” yazısı onu göstərir ki, həmin proqram vasitəsi qrafik istifadəçi interfeysinə malikdir. Yəni Windows sisteminə aid olan bütün imkanlardan həmin proqram vasitəsi istifadə edə bilər.

Ədəbiyyat

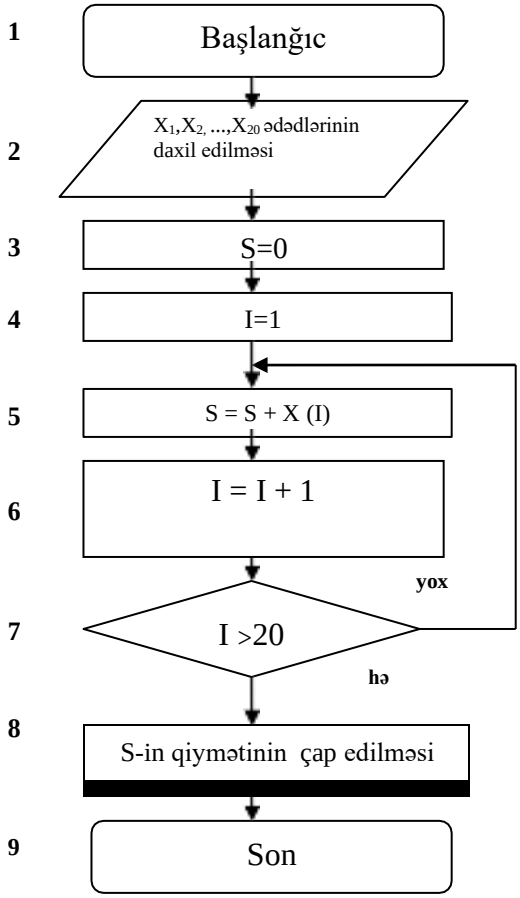
1. Kərimov S.Q., Həbibullayev S.B. İbrahimzadə T. İ. İnformatika – Bakı, 2003.
2. Həsənov G. S. Turbo Pascal və Delphidə proqramlaşdırma - Bakı, ADNA, 2005.
3. Урнов В.А., Климов Д.Ю. Преподавание информатики в компьютерном классе-М: Просвещение, 1990.
4. Кушниренко А.Г. и др. Информатика- М: Дрофо, 1998.
5. Кузнецов А. А. и др. Основы информатика- М: Дрофо, 1998.



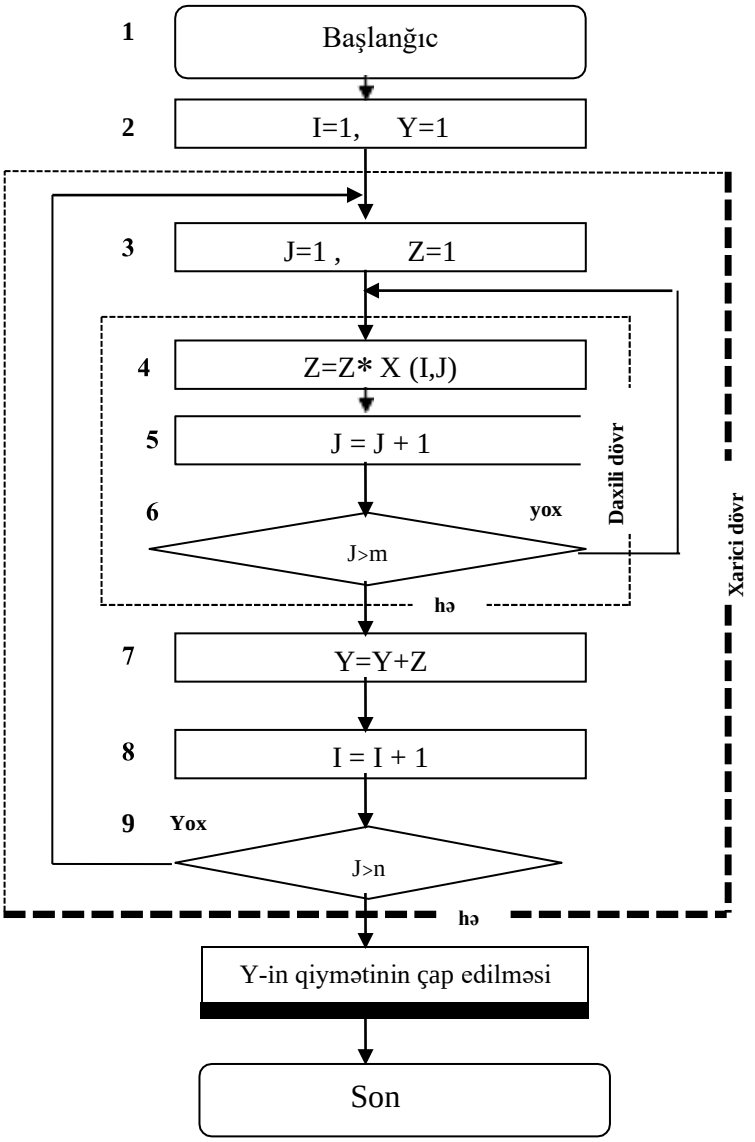
Sxem 1. Xətti strukturlu alqoritmin blok-sxemi. (İkiməchullu birdərəcəli iki tənlik sisteminin həlli alqoritminin blok-sxemi).



Sxem 2. Tam kvadrat tənliyin həll alqoritminin blok-sxemi.



Sxem 3. $S = x_1 + x_2 + ... + x_{20}$ **cəminin hesablanması alqoritminin blok-sxemi.**



Sxem 4. $y = \sum_{i=1}^{n} \prod_{j=1}^{m} x_{ij}$ **ifadəsinin hesablanması alqoritminin blok-sxemi.**