Computer Science

Computer[kəm'pjuter] Science['saiəns], the scientific study of the possible uses of computer.

Computer Science, the study of computers, including their design, operation, and use in processing information. Computer science combines both theoretical and practical aspects of engineering, electronics, information theory, mathematics, logic, and human behaviour.

Computer science is a combination of theory, engineering, and experimentation. In some cases, a computer scientist develops a theory, then engineers a combination of computer hardware and software based on that theory, and experimentally tests it. An example of such a theory-driven approach is the development of new software engineering tools that are then evaluated in actual use. In other cases, experimentation may result in new theory, such as the discovery that an artificial neural network exhibits behavior similar to neurons in the brain, leading to a new theory in neurophysiology. Experimentation and the traditional scientific method are thus key parts of computer science.

***

(Перевод Google)

Компьютерная наука научное исследование возможных вариантов использования компьютера.

Компьютерные науки, изучение компьютеров, в том числе их проектирования, эксплуатации и использования при обработке информации. Информатика сочетает в себе как теоретические, так и практические аспекты, электроники, теории информации, математики, логики и человеческого поведения.

Информатика представляет собой сочетание теории, инженерии и экспериментов. В некоторых случаях, ученый развивает теорию, то инженеры комбинацией компьютерного оборудования и программного обеспечения на основе этой теории, так и экспериментально проверяет его. Примером такого подхода теории управляемой является разработка новых технических средств, программного обеспечения, которые затем оцениваются в реальных условиях эксплуатации. В других случаях, экспериментирование может привести к новой теории, таким как открытие того, что искусственную нейронную сеть проявляет поведение, подобную нейронов в головном мозге, что приводит к новой теории в нейрофизиологии. Экспериментирование и традиционный научный метод, таким образом, ключевыми элементами компьютерной науки.

Perfect Posture

Small assaults on your muscles and tendons can take a big toll over time. How to stay limber :

- **Sit at arm's length from monitor;**
- **Screen should be low enough so you don't have to flex your neck;**
- **Minimize reflection on screen;**
- **Keep elbows above keyboard;**
- **Keep wrists, hands, thighs parallel to floor.**

Computer measurement[‘meʒmənt] units

Baud	**max. number of state transitions per second**	**2400-56,600**
Bps	**Bits per second**	**Modems speed (14,400 – 56,000)**
Cps	**Characters per second**	**Printers speed (25- …)**
Dpi	**Dots per inch**	**Printer resolution (300-1200)**
MHz	**Megahertz (millions of cycles per second)**	**CPU (Bas)speeds (12-1000 (GHz))**
MIPS	**Million Instructions Per Second**	**speed with which computer programs are ran**
ms	**Milliseconds**	**The average access time of HDD – the speed at which a hard drive finds data (9 - 14)**
pin		**A stiff prong in an electronic connector(port) (9 or 25)**
pixel	**Smallest element on the screen**	**Resolution of display, number of horizontal and vertical dots (640*480 ÷ 1,280*1,024)**
Ppm	**Pages per minute**	**Printers speed (4-20)**
RPM	**revolution per minute**	**speed of rotation(newer HD rotate at 7200 RPM)**
W	**Watts (joule per second) the rate at which electrical power being consumed**	**Speakers build - in amplifier (Watts=volts * amperes)**

3D	TREE-DIMENTIONAL GRAPHICS
A.D.	Anno Domini (since Christ was born)
a.k.a.	Also known as
AI	ARTIFICAL INTELIGENCE
AIFF	Audio Interchange File Format.
AMI	American Megatrends, Inc.
ANSI	American National Standards Institute
APL	A Programming Language
app	APPLICATION PROGRAM
ASCII	American Standards Code for Information Interchange
AT	Advanced Technology
ATM	1.Adobe Type Manager. 2.asynchronous transfer mode
AVI	Audio-Video Interleave
AWT	Abstract Window Toolkit (in Java)
B.C.	Before Christ
BBS	Bulletin Board System
BCC	Blind Carbon Copy.
BIOS	Basic Input Output System.
BMP	Bitmap
Bps	Bits per second
CAD	Computer-Aided Design
Cc	Carbon Copy
CGA	Color Graphics Adapter
CGM	Computer Graphics Metafile
CIO	Chief Information Officer
CORBA	Common Object Request Broker Architecture
DAC	DIGITAL-TO-ANALOG CONVERTER
DARPA	Defense Advanced Research Projects Agency
DIMM	dual inline memory module
DNS	Domain Name Server.
EDORAM	Extended dada out random access memory
EGA	Enhanced Graphics Adapter
EOF	End of fail
EOL	End of line
FAT	File Allocation Table
FAQ	Frequently asked questions
FTP	File Transfer Protocol.
GIF	Graphics Interchange Format

Numeric Prefixes

- **Atto 10^{-18} =0,000000000000000001**

Nonillion 10^{30} =1000000000000000000000000000000

Prefix	Abbreviation	Pronounce	Multiplier	Prefix	Abbreviation	Pronounce	Multiplier
Atto	**A**	**At-to**	10^{-18}	**Deca**	**Da**	**Dek-a**	10^{1}
Femto	**F**	**Fem-to**	10^{-15}	**Hecta**	**H**	**Hek-to**	10^{2}
Pico	**P**	**Pe-ko**	10^{-12}	**Kilo**	**K**	**Kil-o**	10^{3}
Nano	**N**	**Nan-o**	10^{-9}	**Mega**	**M**	**Meg-a**	10^{6}
Micro	**µ**	**Mi-kro**	10^{-6}	**Giga**	**G**	**Ji-ga**	10^{9}
Milli	**M**	**Mil-l**	10^{-3}	**Tera**	**T**	**Ter-a**	10^{12}
Centi	**C**	**Sent-ti**	10^{-2}	**Peta**	**P**	**Pe-ta**	10^{15}
Deci	**D**	**Des-l**	10^{-1}	**Exa**	**E**	**Ex-a**	10^{18}
				Zetta	**Z**	**sextillion**	10^{21}
				Yotta	**Y**	**septillion**	10^{24}
						octillion	10^{27}
						nonillion	10^{30}

Multiplier İn PC	Prefix	Abbreviation
2^{10}	Kilo(byte)	K(b)
2^{20}	Mega	M
2^{30}	Giga	G
2^{40}	Tera	T
2^{50}	Peta	P

2400 BC
The abacus, the first known calculator, was invented in Babylonia

300 BC
Pingala invented the binary number system

1492
Da Vinci depicts flying machines, the first mechanical calculator and one of the first programmable robots

60AD
Heron of Alexandria invents machines which follow a series of instructions

1642
Blaise Pascal invents the "Pascaline" a mechanical adding machine

1822
The Analytical Engine was invented by Charles Babbage

1835
Samuel Morse invents Morse code

1880
Alexander Graham Bell invents a telephone called the Photophone

1895
Radio signals invented by Guglielmo Marconi

1924
Electromechanical television system was invented by John Logie Baird

1937
Alan Turing develops the concept of a theoretical computing machine

1912
Alan Turing born 23rd June 1912

1911
IBM is formed on June 15, 1911

IBM

1939
William Hewlett and David Packard start Hewlett Packard

1950
The first electronic computer is created in Japan by Hideo Yamachito.

1953
The IBM 701 becomes available and a total of 19 are sold to the scientific community

1943
Alan Turing develops the the code-breaking machine Colossus

1949
Claude Shannon builds the first machine that plays chess

1961
General Motors puts the first industrial robot, Unimate, to work in a New Jersey factory

GM

1972
Atari releases Pong, the first commercial video game

1971
E-mail was invented by Ray Tomlinson

1964
IBM introduces the first word processor

1975
Microsoft founded by Bill Gates and Paul Allen

Microsoft

1970
Intel introduces the world's first available dynamic RAM chip and the first microprocessor, the Intel 4004

1985
The Nintendo Entertainment System makes its debut

1976
Apple Computers founded by Steve Wozniak and Steve Jobs

1979
Over half a million computers are in use in the United States

1982
The Commodore 64 becomes the best-selling computer of all time

1977
The Apple II, the first personal computer with colour graphics, is demonstrated

1991
The World Wide Web is launched to the public on August 6, 1991

2004
Facebook launched

2010
Apple launches the first iPad

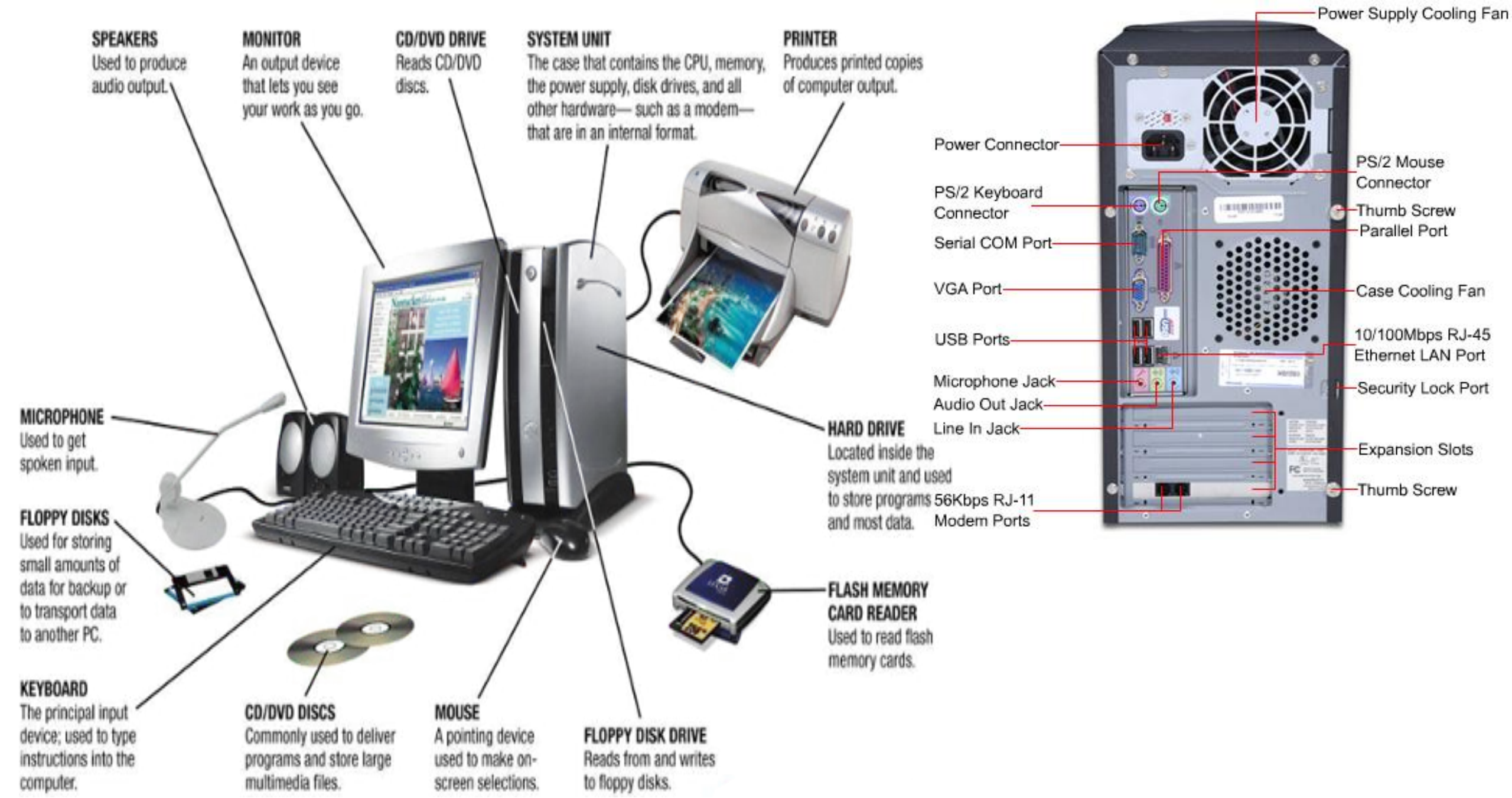
2012
Google Chrome has 28% worldwide usage share of web browsers

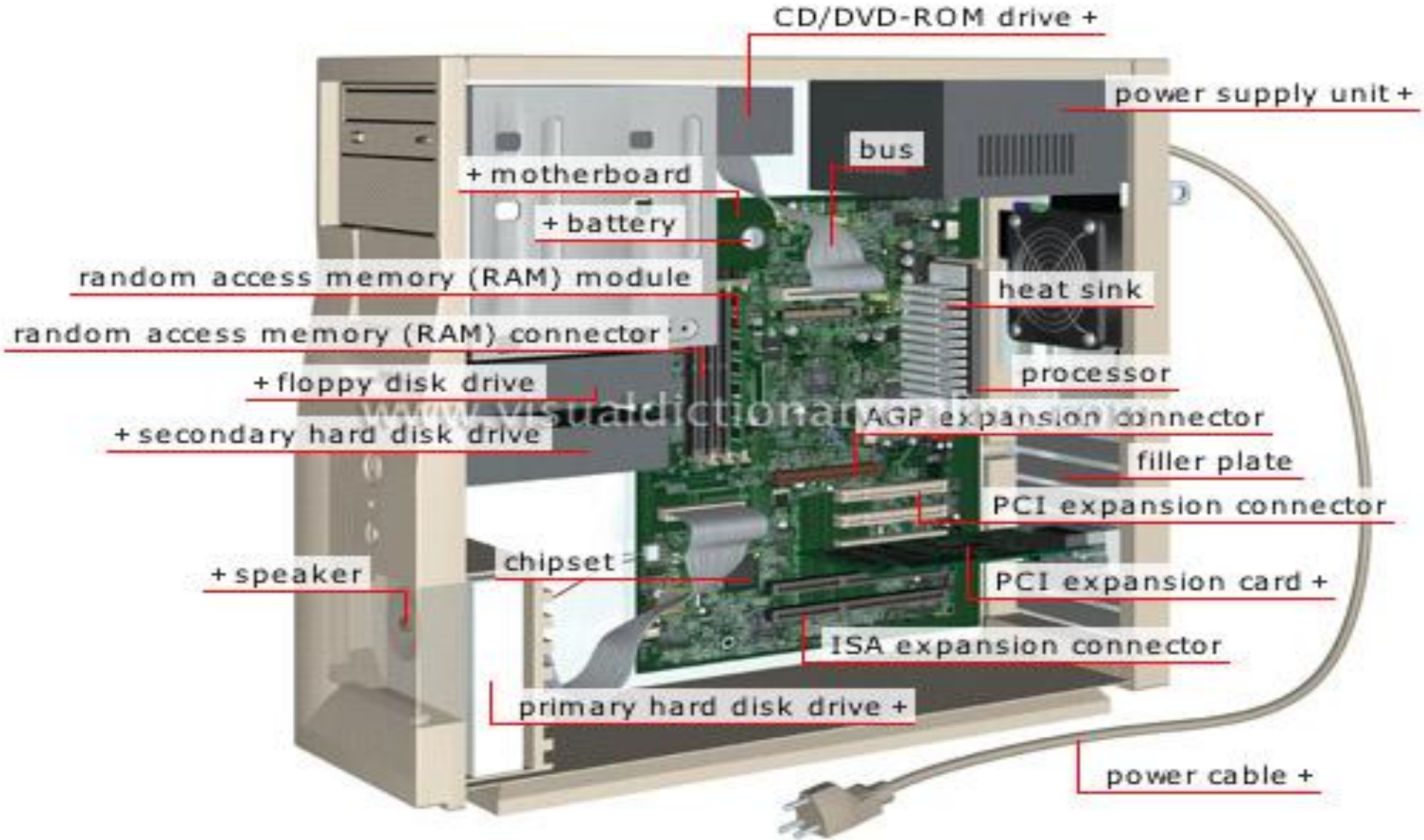
2006
Twitter launched

2002
Approximately 1 billion PCs sold to date

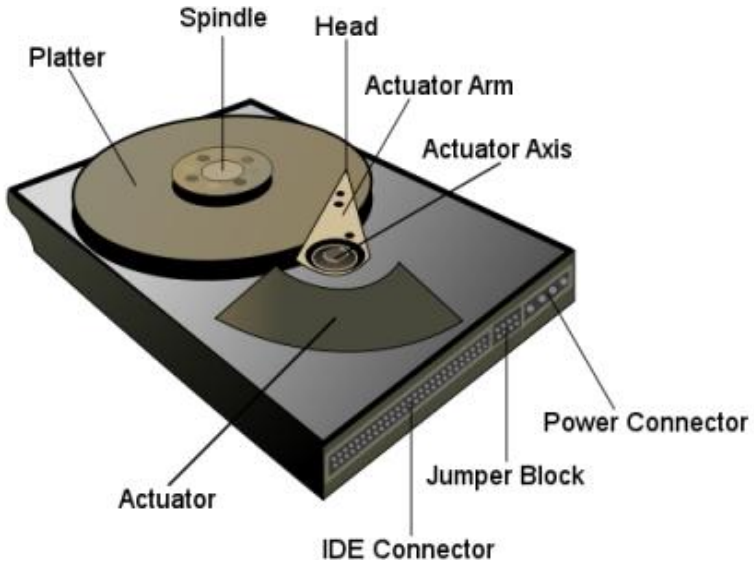
1998
Google founded by Sergey Brin and Larry Page

Google


SPEAKERS
Used to produce audio output.
MONITOR
An output device that lets you see your work as you go.
CD/DVD DRIVE
Reads CD/DVD discs.
SYSTEM UNIT
The case that contains the CPU, memory, the power supply, disk drives, and all other hardware— such as a modem— that are in an internal format.
PRINTER
Produces printed copies of computer output.
MICROPHONE
Used to get spoken input.
FLOPPY DISKS
Used for storing small amounts of data for backup or to transport data to another PC.
KEYBOARD
The principal input device; used to type instructions into the computer.
CD/DVD DISCS
Commonly used to deliver programs and store large multimedia files.
MOUSE
A pointing device used to make on-screen selections.
FLOPPY DISK DRIVE
Reads from and writes to floppy disks.
HARD DRIVE
Located inside the system unit and used to store programs and most data.
FLASH MEMORY CARD READER
Used to read flash memory cards.
Power Supply Cooling Fan
Power Connector
PS/2 Mouse Connector
PS/2 Keyboard Connector
Thumb Screw
Parallel Port
Serial COM Port
VGA Port
Case Cooling Fan
USB Ports
10/100Mbps RJ-45 Ethernet LAN Port
Microphone Jack
Audio Out Jack
Line In Jack
Security Lock Port
Expansion Slots
Thumb Screw
56Kbps RJ-11 Modem Ports


CD/DVD-ROM drive +
power supply unit +
bus
+ motherboard
+ battery
random access memory (RAM) module
heat sink
random access memory (RAM) connector
processor
+ floppy disk drive
AGP expansion connector
+ secondary hard disk drive
filler plate
PCI expansion connector
chipset
+ speaker
PCI expansion card +
ISA expansion connector
primary hard disk drive +
power cable +


Ports
Optical Audio "Toslink"
USB A
Firewire 4 pin iLink
Firewire 400 1394a
Firewire 800/1600 1394b/c
Ethernet 8P8C
Modem RJ-11
Apple Desktop Bus - ADB
Mac Serial
PS/2
USB A 3.0
DE-9F
DB-25 Serial/Com Port
DE-9 Serial RS232
e-SATA
Centronics Parallel 36pin
Centronics SCSI 50pin
AT Keyboard
50 pin SCSI 2
Surround sound
Composite Audio/Video
S-Video
Component Video
F-Connector RF/CCDAE
Parallel Port/SCSI 3/DB-25F
Mini DisplayPort
Mini-DVI
Mini-VGA
HDMI
Micro-DVI
DisplayPort
DVI Video
IF YOU ENJOYED THIS IMAGE AND/OR HAVE A LARGE PENIS, CHECK OUT MEMEDIX.COM

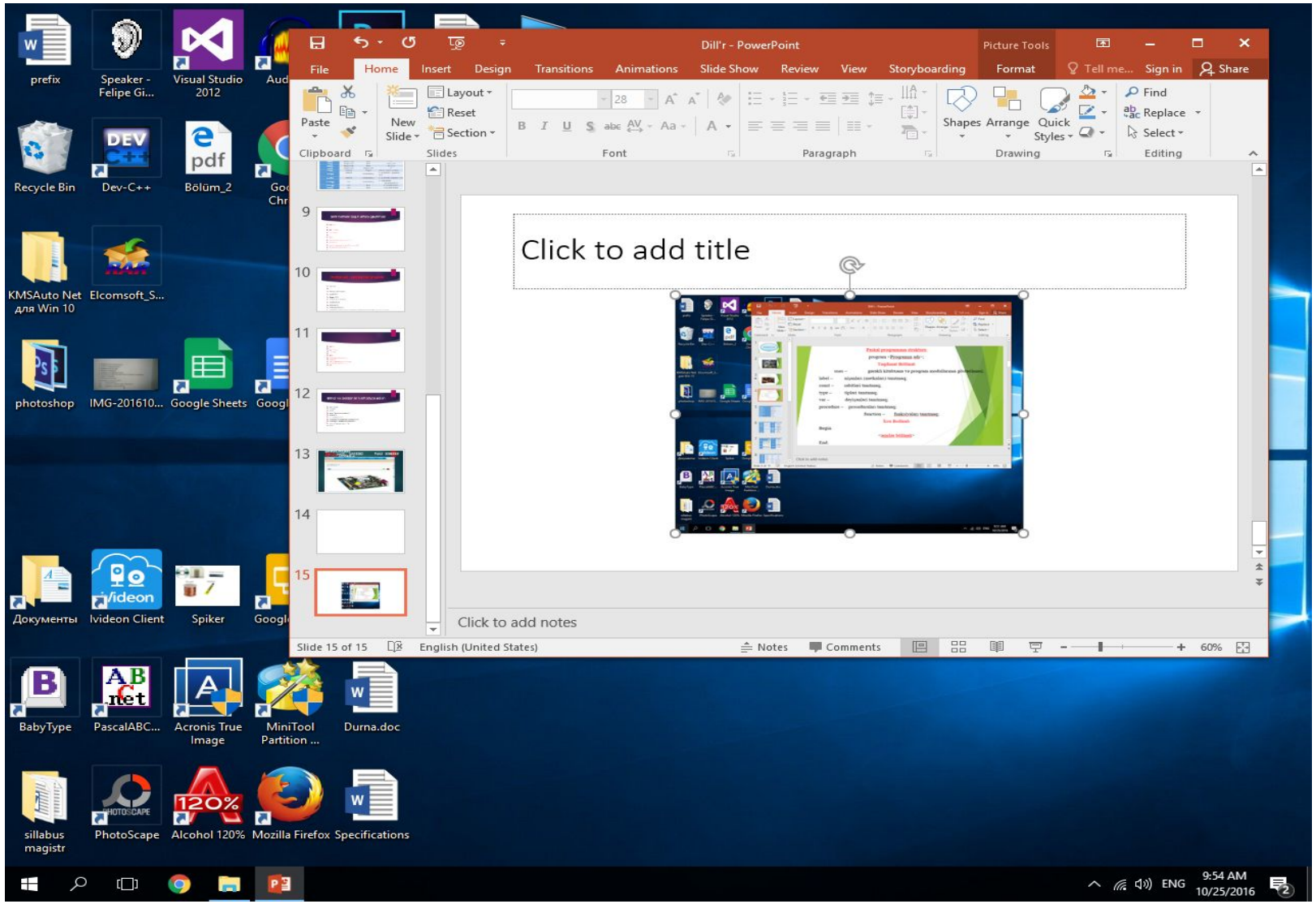

Platter
Spindle
Head
Actuator Arm
Actuator Axis
Power Connector
Jumper Block
Actuator
IDE Connector

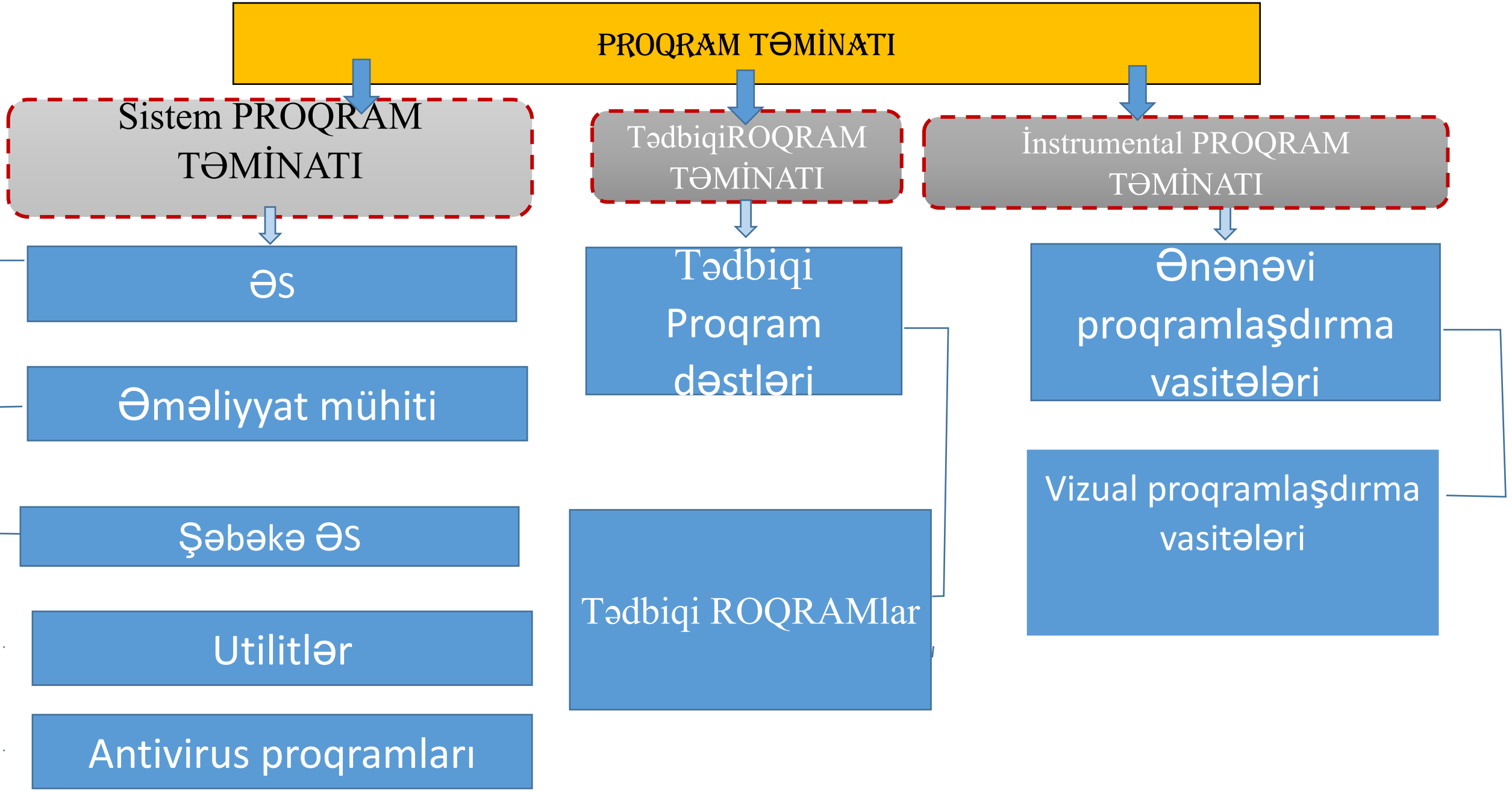
MOTHERBOARD

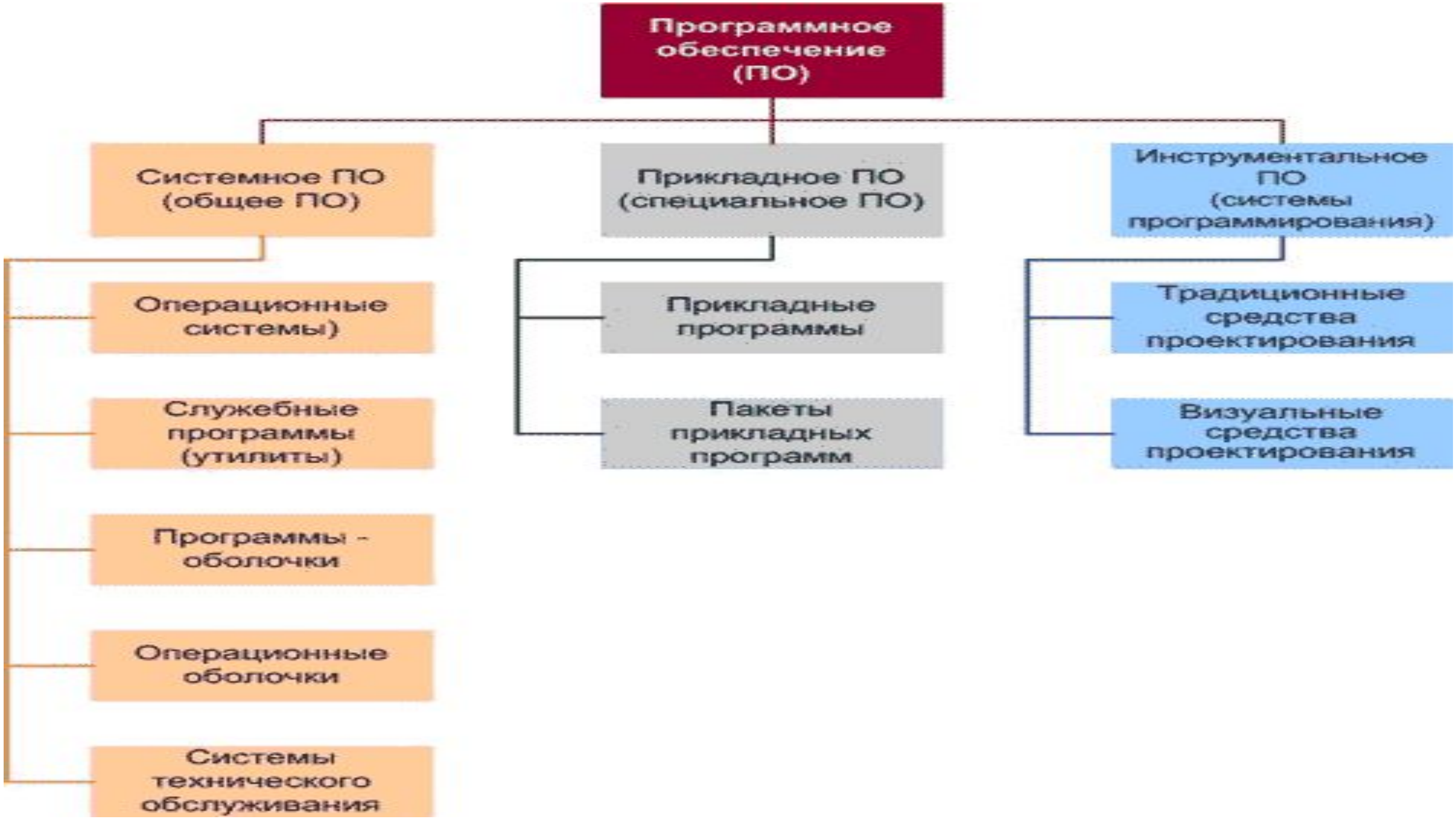
ANA LOVRO


Multimedia Navigation Buttons
Function Keys
Calculator Button
Sleep Button
Web Navigation Buttons
Number Lock LED
Caps Lock LED
Scroll Lock LED
Scroll Wheel
Buttons
Cut Button
Paste Button
Numeric Keypad
Editing/Navigation Keys

Print Screen düyməsinin funksiyası:
Cari ekranın təsvirini Clipboard-a, bufer yaddaşa yerləşdirir
Scrool Lock – skrolling (ekranı fırlatmaq) prosesini dayandırır




PROQRAM TƏMİNATI
Sistem PROQRAM TƏMİNATI
TədbiqiROQRAM TƏMİNATI
İnstrumental PROQRAM TƏMİNATI
ƏS
Əməliyyat mühiti
Şəbəkə ƏS
Utilitlər
Antivirus proqramları
Tədbiqi Proqram dəstləri
Tədbiqi ROQRAMlar
Ənənəvi proqramlaşdırma vasitələri
Vizual proqramlaşdırma vasitələri


Программное обеспечение (ПО)
Системное ПО (общее ПО)
Прикладное ПО (специальное ПО)
Инструментальное ПО (системы программирования)
Операционные системы)
Служебные программы (утилиты)
Программы - оболочки
Операционные оболочки
Системы технического обслуживания
Прикладные программы
Пакеты прикладных программ
Традиционные средства проектирования
Визуальные средства проектирования


SİSTEM PROQRAM TƏMİNATI
Baza PROQRAM TƏMİNATI
Xidməti (servis) PROQRAM TƏMİNATI
ƏS
Əməliyyat mühiti
Şəbəkə ƏS
Diaqnostika proqramları
Antivirus proqramları
Qurğulara xidmət proqramları
Drayverlər
Arxivləşdirmə proqramları
Şəbəkə xidməti proqramları

Əməliyyat Sistemləri

DOS

DOS sözü ingiliscə Disk Operating System olub Azərbaycan dilinə tərcüməsi “Disk əməliyyat sistemi”. DOS komputerlər üçün nəzərdə tutulmuş kiçik və sadə bir əməliyyat sistemi növü olub,əsas vəzifəsi disket və sabit disk kimi saxlama mühitlərinin idarə edilməsidir.Komputerlərin digər funksiyaları:qrafika,səs,yazma,şəbəkədə gəzinmə,yaddaşa nəzarət,çoxlu istifadəçi və çoxlu iş xüsusiyyətləri DOS tərəfindən yerinə yetirilə bilir.DOS sistemləri 90-cı illərin ortalarına qədər demək olar ki,hər personal komputerdə öz vəzifəsini yerinə yetirmişdir.Bu gün computer dünyasının çox sahəsində qrafik əməliyyat sistemlərindən istifadə olunsa da,DOS sadə və kiçik olmaq üstünlüyü ilə müxtəlif əmrlər və nəzarət etmə sistemlərində öz həyatına davam etməkdədir.DOS-un tarixi komputerlərin tarixi ilə başlayır.DOS-da qrafik bir istifadəçi pəncərəsi olmadığından hər bir əməliyyat əmrlərlə və az sayda olan parametrlərlə həyata keçirilir.

Windows XP

Windows XP Maykrosoft şirkətinin personal komputerlər və server istemləri üçün Windows NT-nin kernel sistemi üzərində qurulmuş olan 6-cı əsas distributividir.Windows XP 25 oktyabr 2001-ci il tarixində satışa təqdim edilmişdir.XP adı ingilis sözü olan “experience” sözündən gəlir və Azərbaycan dilində mənası “təcrübə” deməkdir.Windows XP hal-hazırda dünyada ən çox istifadə edilən əməliyyat sistemidir.Windows XP əvvəlki distributivlərindən fərqli olaraq tamamilə 32 bitlik Windows NT və Windows 2000 kernel sistemi üzərində qurulmuşdur.Onu da qeyd etmək lazımdır ki,Windows əməliyyat sisteminin əvvəlki versiyaları aşağıdakılardır:

- Windows 1.0
- Windows 2.0
- Windows 3.0
- Windows 95
- Windows 98
- Windows NT

Bu kernel system 16 və 32 bitlik tətbiqləri işlədə bilir və göy ekran səhvlərini azaldır.Windows əməliyyat sistemi ailəsinin Cairo,Nashville,Neptune,Odyssey kimi versiyaları satışa çıxarılmayaraq ləğv edilmişdir.Maykrosoft şirkəti Windows XP-in satışını 30 iyun 2008-ci il tarixindən etibarən dayandırmışdır:amma bəzi mini noutbuklarda hələ də qurulu olaraq XP gəlməkdədir.Windows XP Service Pack 2-nin də satışı 2010-cu ildə bitmişdir.Windows XP Service Pack 3 isə 2014-cü ilə qədər satılacaqdır.

Macintosh

Macintosh qısaca Mac olaraq tanınır və adını Macintosh alma növündən alır.Mac personal komputerlər istehsal edən Apple Computer İnc. bir məhsulu olan əməliyyat sistemidir.Macintosh-un istehsalına 1984-cü ildə başlanılmışdır.Onu da qeyd edək ki,Apple şirkəti siçan və qrafik interfeysdən ilk dəfə istifadə edən şirkətlərdəndir.Macintosh-dan əvvəl Apple şirkətinin Lisa,Apple II,Apple III kimi sistemləri olsada 1986-cı ildə onların fəaliyyəti dayandırılmışdır və bütün məhsullar

UNİX

UNİX 1969-cu ildə Ken Tompson və Denis Riçi tərəfindən Bell Laboratori- yalarında yaradılmış olan çox istifadəçili və çox vəzifəli quruluşu dəstəkləyən bir əməliyyat sistemidir.UNİX törəməli əməliyyat sistemləri çox prosessorlu bahalı komputerlərdən bir prosessorlu sadə ev komputerlərinə qədər bir çox cihaz üzərində işləyə bilən test edilmiş strukturu ilə çox prosessorlu serverlərdə adətən standart halına gəlmişdir.UNİX-in əsası 1965-ci ildə MİT,AT&T Bell Labs və GE-nin birlikdə yaratdıqları MULTİCS (Multiplexed Operating and Computing System) layihəsi ilə atılmışdır.MULTİCS layihəsinin əsas hədəfi çox istifadəçili komputer sistemlərinə icazə verərək eyni vaxtlı məlumat paylaşmasını təmin edə bilməkdi.1969-c ildə layihə qarışıq bir şəkil almağa başlamış və AT&T Bell Labs layihədən çəkilmişdir.Bell şirkətində araşdırmaçı olaraq işləyən Ken Tompson MULTİCS proqramını stimulyasiya edən bir fayl sistemini kodlamaqla UNİX-in ilk distributivi olan UNİCS-i (Uniplexed Operating and Computing) meydana çıxardı.Bu versiya Assembler ilə yazılmışdı.1973-cü ildə Ken Tompson C dilinin yaradıcısı olan Denis Riçi ilə birlikdə əməliyyat sisteminin kernel sistemini C dili ilə təkrar kodlayaraq UNİX 5.0 versiyasını yaratdılar.Beləliklə də,UNİX C dilinin sayəsində müxtəlif avadanlıqlara uyğun olaraq təkrar dəyişdirilə bilən kodlardan ibarət bir əməliyyat sisteminə çevrilmişdir.

Linux

Linux UNİX-ə texniki mənada bənzəyən redaktə qabiliyyətli müstəqil bir əməliyyat sistemidir.Kernelin kodları GNU Ümumi Cəmiyyət Lisenziyası ərçivəsində müstəqil olaraq dəyişdirilə istifadə edilə bilər.Linux hər hansı bir komputer sistemində problemsiz işləmə qabiliyyətinə malikdir. Çox geniş bir təchizat dəstəyinə malik olan Linux netbuk,noutbuk,server komputerləri,is stansiyaları,ağıllı telefon,stolüstü komputerlər kimi hər bir platformada tam bir uyğunlaşma içərisində işləmək qabiliyyətinə malikdir.Linux adətən server və iş stansiyalarında istifadə olunsa da onu şəxsi komputerlərdə də istifadə edənlər çoxluq təşkil edir.Təbii ki,bu halda redaktə edilən kodların və azad proqram anlayışının təsiri böyükdür.Windows XP və Mac OS sistemlərinə nəzərən daha rahat bir struktura malik olan Linux xüsusilə server satışlarında rəqibləri ilə müqayisədə daha üstündür.Mint,Ubuntu,openSUSE,Pardus,Mandriva kimi son versiyaları ilə Linux əməliyyat sisteminin istifadəçi faizi gündən-günə artmaqdadır.

Sun OS

Sun OS əməliyyat sistemi əvvəllər iş stansiyaları və server komputer sistemləri üçün Sun Microsystems şirkəti t ərəfindən istehsal olunan UNİX əməliyyat sisteminin bir versiyası idi.Sun OS adı adətən Sun OS əməliyyat sisteminin 1.0 və 4.1.4 versiyalarına istinad edilir.Bu versiyalar açıq redaktə edilə bilinən kodlu idi.Sun OS sistemi 5-ci versiyasından başlayaraq Solaris brendi altında satışa çıxarılmağa başlanıldı.Sun OS 1 və 2 versiyası Sun-2 sistemindən,Sun OS 3 versiyası Sun-2 və 3 sistemindən,Sun OS 4 versiyası isə Sun-3,Sun-386i,Sun-4 sistemindən daha doğrusu arxitekturalarından istifadə edir.Bu əməliyyat sisteminə yeni SPARC prosessor faylı əlavə edildikdən sonra birinci Sun-4m arxitekturalı multiprosessorlu komputerlər (SRARCserver 600MP seriyası) meydana gəlməyə başladı.Sun OS 4-ün axırıncı versiyası 4.1.4 olmuşdur və 1994-cü ildə buraxılmışdır.2003-cü ildən bəri isə Solaris şirkətinin ən məşhur məhsulu Sun OS 5 əməliyyat sistemidir və bu sistem sadəcə olaraq Solaris adı ilə dünyada tanınır.

Palm OS

Garnet OS kimi də tanınan Palm mobil əməliyyat sistemi 1996-cı ildə PDA-lar üçün Palm İnc şirkəti tərəfindən istehsal edilməyə başlanılmışdır.Palm OS əllə toxunula bilən (touchscreen) qrafik istifadəçi interfeysi ilə asan i stifadə olunması üçün istehsal edilmişdir.Bu,şəxsi məlumatların idarə edilməsi üçün əsas proqramlar dəsti ilə təmin edilir.Palm əməliyyat sisteminin son versiyaları smartfonlarda istifadə üçün nəzərdə tutulmuşdur.2007-ci ildə Palm əməliyyat sistemi ticarət markası aldıqdan sonra Garnet OS deyə yenidən adlandırıldı.Garnet OS sisteminin varisi olan ACCESS 2009-cu ildən Access Linux Platform deyə tanınmağa başladı və gələcəkdə istehsal olunacaq cihazlar üçün Palm Os ilə Web OS sistemləri arasında keçidlər etməyə başladı.Palm OS rəsmi olaraq Cef Hokinsin rəhbərliyi altında Palm Computing şirkəti tərəfindən istehsal olunur.Palm Computing şirkəti sonralar 3Com-a çevrilən US Robotics Corp tərəfindən Palm İnc-ə çevrildi.

PROGRAM LANGUAGES

PROQRAM DILLƏRI

Kompyüterdə ixtiyari problemi həll etmək üçün proqramlaşdırma dillərindən istifadə edilir. Məsələ həll edərkən əvvəlcə yerinə yetiriləcək əməliyyatların *alqoritmi* tərtib edilir. Proqramlaşdırmada məsələni alqoritmləşdirməkdən qabaq aşağıdakı addımlar yerinə yetirilməlidir:

- **Məsələnin riyazi qoyuluşu:**
- **Nə verilir, ilkin verilənlərin sadalanması;**
- **Nə tələb olunur, nəticələrin sadalanması ;**
- **İlkin verilənlərin məhdudiyyət şərtləri.**
- **Riyazi model: nəticələri almaq üçün lazım olan bütün qayda və qanunlar.**
- **Həll metodu: riyazi modelin optimal istifadə olunması.**

Alqoritm

Latınca qayda-qanun deməkdir.

Alqoritm 783- 850-ci illərdə Xorezmdə yaşamış IX əsr məşhur riyaziyyatçı Məhəmməd İbn Musa əl-Xarəzminin (yəni Xarəzmli Musa oğlu Məhəmməd) adının latın hərflərilə olan 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.

1 Alqoritmin xassələri

2 Alqoritmin təsvir üsulları

3 Alqoritmin tipləri

Alqoritmin növləri

Xətti alqoritm

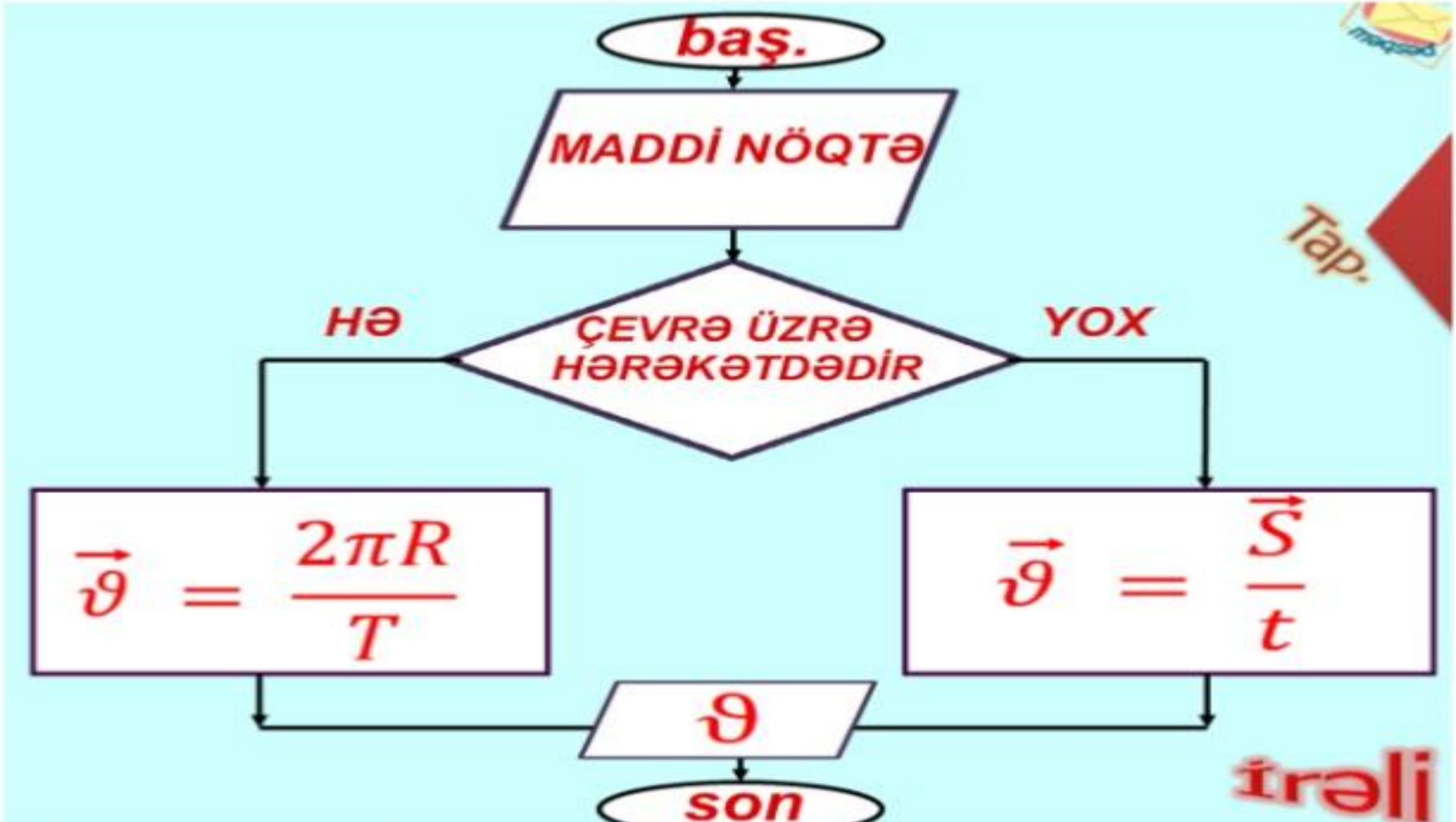
Alqoritmin bütün mərhələləri verildiyi ardıcıllıqla yerinə yetirilir

Budaqlanan alqoritm

Alqoritm həll variantlarından birini seçməyə imkan verir

Dövri alqoritm

Alqoritmdə müəyyən addımlar dəfələrlə təkrarlanır


baş.
MADDİ NÖQTƏ
HƏ
ÇEVRƏ ÜZRƏ HƏRƏKƏTDƏDİR
YOX
$\vec{\vartheta} = \frac{2\pi R}{T}$
$\vec{\vartheta} = \frac{\vec{S}}{t}$
ϑ
son
Tap.
irəli

Alqoritmin xassələri

Məsələnin maşında həlli üçün tərtib edilən alqoritm bir çox şərtləri ödəməlidir. Bu şərtlərə alqoritmin xassələri deyilir. Həmin xassələr aşağıdakılardır:

Alqoritm sonlu sayda mərhələdən sonra qurtarmalıdır. Buna, alqoritmin **sonluluq (nəciləlilik)** xassəsi deyilir.

Alqoritmin hər bir addımı dəqiq və birqiymətli təyin olunmalıdır. Bu alqoritmin **müəyyənlik** xassəsidir.

Alqoritmin müəyyən sayda giriş qiymətləri (məsələnin başlanğıc şərtləri) olmalıdır. Bu şərtlər proqram icra olunmamış və ya olunduqca maşına daxil edilə bilər.
Alqoritmin yerinə yetirilməsi nəticəsində giriş qiymətlərindən asılı olan bir və ya bir neçə çıxış qiymətləri alınmalıdır.

Alqoritm sadə və səmərəli olmalıdır, yəni alqoritmin nəticəsi (cavabı) mümkün qədər sadə əməliyyatlar vasitəsilə və ən qısa yolla alınmalıdır.
Alqoritm ümumi olmalıdır, yəni müəyyən məsələ üçün tərtib olunmuş alqoritm, həmin tipdən (sinifdən) olan bütün məsələlər üçün yararlı olmalıdır. Bu alqoritmin **kütləvilik** xassəsidir
Riyaziyyatda və informatikada məsələnin həllinin alqoritmi yerinə yetirilibsə, məsələ qismən həll edilmiş sayılır.

Düzgün tətbiq edilən alqoritm aşağıdakı əsas xüsusiyyətlərə malik olmalıdır:

- *Nəticəlilik;*
- *Müəyyənlik;*
- *Diskretlilik;*
- *Kütləvilik.*

Nəticəlilik əməliyyatların sonlu sayının mümkünlüyünü göstərir ki, onların yerinə yetirilməsi axtarılan nəticəyə gətirib çıxara bilsin.

Müəyyənlik

istifadəçidən və tətbiq edilən texniki vasitələrdən asılı olmayaraq alınan nəticələrin üst-üstə düşdüyünü göstərir.

Diskretlilik

alqoritmin təsvir etdiyi proses ayrı-ayrı addımlar ardıcıllığına bölünməli və bu bölgü elə olmalıdır ki, alqoritmin yazılışı bir-birindən dəqiq ayrılmış göstərişlər şəklində olsun. Məsələn, "yolun kənarı ilə gedin" və yaxud "kənar şəxslərin daxil olması qadağandır" kimi göstərişləri kəsilməz xarakterli olduqları üçün alqoritm deyil, lakin resept əsasında dərman hazırlamaq və yaxud bankomatdan pul çıxartmaq və bu kimi qaydalar diskret xarakterli olduqları üçün alqoritmdir.

Kütləvilik ilkin məlumatların konkret qiymətləriylə fərqlənən eyni tipli məsələlərin bütöv sinifinə alqoritmin tətbiq edilmə imkanlarını nəzərdə tutur. Tənlik Pr-ı ixtiyari əmsallar üçün korrekt cavab verməlidir.

Alqoritmin təsvir üsulları

- **Mətn şəklində (adi dildə);**
- **Qrafik blok-sxem;**
- **Psevdokod**
- **Cədvəl;**
- **Proqram (alqoritmik dil).**

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.

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.
Alqoritm ayrı-ayrı ədədlərlə yox, verilmiş hər hansı obyektlərlə işləyir.

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.

Budaqlanan 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, məsələnin həli prosesi bir istiqamətə, həmin şərt ödənilmirsə, başqa istiqamətə doğru hərəkət edilir. Beləliklə, alqoritmdə budaqlanma baş verir.

Dövri alqoritm. 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 belə dövrlərə mürəkkəb dövr deyilir. Mürəkkəb dövrləri əmələ gətirən sadə dövrlər kəsişə bilməz.

Kompilyasiya və interpretasiya

Hesablama maşınlarının əsas fərqləndirici xüsusiyyəti onun *proqramla* idarə olunmasıdır. Yəni, istər sadə, istərsə də mürəkkəb məsələni maşının həll etməsi üçün proqram tərtib edilməlidir
əməliyyatlar hər-hansı alqoritm (*proqramlaşdırma*) dilində *əmrlər* şəklində yazılır. Tərtib olunmuş proqram *translyator* proqramlar vasitəsilə *maşın koduna* çevrilir və *işləndikdən* sonra yerinə yetirilir.

Translyasiyanın iki qaydası var: interpretasiya və kompilyasiya. İnterpretasiya şifahi tərcüməyə oxşayır. Giriş proqramının hər bir təlimatı tərcümə olununur və yerinə yetirilir. Bu qaydada təkrar təlimatlar hər dəfə kodlaşdırılır. Kompilyasiya isə yazılı tərcüməyə bənzəyir. Proqram yerinə yetirilməzdən qabaq proqramın bütün tərcüməsi yığılır. İnterpretasiya böyük çevikliyə malik olmaqla asan realizə olunur. Kompilyasiya isə daha effektif proqram yaradır.

Proqramlaşdırmanın əsas obyekti dəyişəndir. Məsələn, x adlı dəyişənə 25 qiymətinin mənimsənilməsini belə müəyyən etmək olar:
x : = 25 yazılır və *x = 25* olur.
Proqramlaşmanın bütün dilləri verilənlərin aşağıda göstərilən ***tipləri*** ilə işləməyə imkan verilir:

- **Tam ədədlər;**
- **Məntiqi dəyişənlər;**
- **Həqiqi ədədlər;**
- **Simvollar;**
- **Mətn tipli dəyişənlər;**
- **Birtipli verilənlər cədvəli;**
- **Fayllar.**

Proqram dillərinin nəsilləri

Nəsillər	Proqram dilləri	Xarakteristika
I	Maşın dilləri	Konkret maşın üçün, mənimsənməsi çətin, arxitekturanı bilmək tələbli
II	Assemblerlər, makroassemblerlər	Maşından asılı olsa da istifadəsi rahat
III	Yüksək səviyyəli Proqram dilləri	Öyrənilməsi asan, mobil, insana yönəli
IV	Q/Prosedur, Obyektyönümlü, Sorğular dilləri, Parallel	q/professianollara hesablanıb. Parallel arxitektura üçündür
V	Süni intellekt, ekspert sistemləri və biliklər bazası, təbii dillər	Dilli interfeysə, maşının artan intellektinə yönəli

Proqram dillərinin sinifləşdirilməsi

Faktor	Xarakteristika	Qruplar	Nümunələr
Proqram dillərinin səviyyəsi	Proqram dillərinin PK arxitekturasına yaxınlıq dərəcəsi	Aşağı	Avtokod, Assembler
		Yüksək	Fortran, Pascal, ADA, Basic, C və s.
		Çox yüksək	Setl
Proqram dillərinin İxtisaslaşdırılması	Potensial və ya real tətbiq sahəsi	Ümumi təyinatlı (universal)	Fortran (mühəndis hesablamaları), Cobol (İqtisadi məsələlər), Refal, Lisp(simvol emalı), Modula, ADA(real zaman miqyasında proqramlaşdırma)
		İxtisaslaşdırılmış	
Alqoritmlik (Prosedurluq)	Alqoritmin xırdalıqlarından mücərrədləşməyə doğrudur.Алгоритмичность тем выше, чем точнее приходится планировать порядок выполняемых действий	Prosedur	Assembler, Fortran, Basic, Pascal, ADA
		Q/Prosedur	Prolog, Langin

Source code . Mənbə kod - Başlanğıc proqram – proqramın mətni

Adətən, aşağıdakı bəndlərdən ibarət olur: (ilk dəfə Cobol dili üçün təyin olunub):

İdentifikasiya- proqramın adı, eləcə də proqramçı və istifadəçi üçün əlavə məlumat saxlanan hissə;

Əlaqələr- əgər proqram digərindən çağırılıbsa və deməli başqa proqramdan nəticələri almalıdırsa mübadilə olunan verilənlərin (proqramın parametrləri adlanır) təsvir olunduğu proqram fraqmentidir. фрагмент текста, описывающий внешние переменные, передаваемые вызывающей программой (если таковая имеется), т.е. ту часть исходных данных, которая обязательно поступает на вход программы при ее запуске. Эти переменные часто называют параметрами программы;

Avadanlıq(mühit) – compüterin proqramın icra olunması baxımından əhəmiyyətli olan prosesor göstəriciləri, tipi, əməli və xarici yaddaşa tələblər.

Məsələ ondadır ki, hətta eyni tipli compüterlər arasında maşın komandaları arasında fırqlaır ola bilər.

Verilənlər- proqramda istifadə olunan verilənlərin tipləri, adları (deklorasiya, elan; təsvir). Verilənlərin tiplərinin göstərilməsi əməliyyat gedişində verilənlər arasındakı uyğunsuzluqları və münkün olmayan çevirmələri üzə çıxarmağa imkan verir.

prosedurlar- proqramın əsas hissəsidir ki, verilənlərin emalı proseslərinin təsvirindən ibarətdir. Prosedurların elementləri proqram dilinin operatorlar və standart funksiyalarıdır.

Bu bəndlər müxtəlif dillər üçün fərqli ola bilər

PROQRAMLAŞDIRMA DİLLƏRİ

- **Ada** - Ada proqramlaşdırma dilinin yaranma tarixi 1974-1980-ci illərə təsadüf edir.
- **Assembler**– bu proqram vasitəsilə effektiv və kompakt proqramlar yaradılır.
- **Basic**– Bu dil üçün kompilyator və interpretatorlar mövcuddur. 60-cı illərdə yaradılmışdır və öyrənilməsi sadədir.
- **C** – Bell laboratoriyasında yaradılmışdır və assembler dilini əvəz etmək üçün nəzərdə tutulmuşdur. Assemblerdən fərqli olaraq konkret tip prosessordan asılı deyil.
- **C++** – 1980-cı ildə Byörn Straustrup tərəfindən yaradılmışdır. Proqramın imkanları proqramçının işinin məhsuldarlığını artırmış olur.

- **LISP** - List Processing (siyahıların emalı) sözlərindən götürülmüşdür. 1958-ci ildə yaradılmışdır
- **Lua**- 1993-cü ildə yaradılmış açıq qaynaq kodlu dildir.
- **Objective-C** - yuxarı səviyyəli, Obyekt yönümlü proqramlaşdırma dilidir.
- **Pascal**– 70-ci illərdə Niklaus Birt tərəfindən yaradılıb, Alqol dilinə daha çox oxşayır.
- **Perl** - 1987-ci ildə Lary Wall tərəfindən yaradılmış dinamik proqramlaşdırma dilidir.
- **PHP**- (Hypertext Preprocessor) dinamik veb səhifələr yaratmaq üçün 1995-ci ildə yaradılıb.
- **Prolog** - Proloq məntiqi proqramlaşdırma dilidir ki Süni intellekt və Hesablamalı dilçilik ilə əlaqədardır.
- **Python** - 1991-ci ildə Guido van Rossum tərəfindən yaradılmışdır. Dilinin sintaksisi çox aydın və anlaşıqlıdır.
- **Ruby** - Yukihiro Matsumoto tərəfindən Perl, Smalltalk və Eiffel dillərindən

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

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

01 0101 0102 0103
03 0103 0002 0103

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 veri

ilə təyin olunur

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. Maşın proqramının strukturu qəti müəyyən edilmədiyindən, dəyişənlər, konstantlar və əmrlər proqramda ixtiyari ardıcıllıqda yerləşirlər. Yaddaşın əmrlər, konstantlar və dəyişənlər yerləşən oyuqları arasında heç bir fərq yoxdur. Belə ki, əmr üçün ayrılmış yaddaş elementi dəyişən yaxud konustant üçün də istifadə olunan bilən maşın əmrləri vasitələ çox sadə əməllər həyata keçirilir. Misal üçün yaddaşınbir oyuqda yerləşən informasiyanı digərinə keçiriməli;iki oyuqun daxilində yerləşən kəmiyyətləri toplamalı və s. Bununla belə, maşın əmrləri vasitəsilə proqram tərtib etmək böyük əmək sərfi tələb edir.Maşın dilində tərtib edilmiş proqram sazlamaq da mürəkkəb məsələdir, çünki proqrama yalız bir əmirin əlavə edilməsi belə yaddaşda çox sayda ünvanın dəyişməsinə səbəb ola bilər.Maşın dili kompüterin qurğularrından tam istifadə etməyə imkan verir. Onun vasitəsilə kifayət qədər effektiv olan proqram tərtib etmək mümkündür.
Bu dil proqramçıdan yüksık ixtisaslılıq tələb edir.
Bnunla belə, bu dil əsasında yüksək əmək məhsuldarlığına nail olmaq olduqca çətindir.

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. Bundan əlavə, dilə bir neçə proqram vahid proqram şəklində birləşdirməyə və səhvlərə nəzarət etməyə imkan verən vasitələr də daxil edilmişdi. Assembler dilində kompüterin bütün imkanlarından tam istifadə etməyə imkan verən effektiv proqramlar yazılır.

Çatışmayan cəhət proqramın həddindən artıq təfərrüfatı ilə yazılmasıdır.

ASSEMBLERİN YARANMASI

1952-ci ildə amerikalı qadın **Qreys Hopper** dünyada ilk mnemonik proqramlaşdırma dili olan Assembler dilini yaratdı. Onun adı "assemble" ingilis sözündən götürülmüşdür, "yığmaq", "quraşdırmaq" mənasını verir. O, özündə mnemonik komandalar sistemini , prosedurlar kitabxanasını və proqram mətnlərini maşın koduna çevirmək üçün xüsusi proqramı birləşdirirdi MAşın kodunun alınmasının belə proseduru kompilyasiya (ing. *compile* – "tərtib etmək", "yığmaq"), onu həyata keçirən proqram isə **kompilyator** adlanır. Bunları da Qreys

Assembler dilinin *sistem proqramlaşdırmaya* aid olan hissəsi isə, həm mürəkkəbliyinə, həm də həcminə görə istifadəçi proqramlaşdırmadan qat-qat böyükdür. İşin problem tərəfi odur ki, sistem proqram kodları istifadəçi proqramları kimi istədiyimiz vaxt kompilyasiya və icra edə bilmərik. Sistem proqramları kodu əməliyyatlar sisteminin proqram kodudur və yalnız bir dəfə kompüter yüklənəndə yüklənir və kompüterin bütün fəaliyyəti boyu icra olunur, kompüteri, onun resurslarını, avadanlıqlarını, fiziki yaddaşı, istifadəçi proqramlarını və s. idarə edir.
Sistem proqram kodlarının işinə misal olaraq, prosessorun rejimlərini dəyişmək, kompüterin portlarına məlumat yazmaq (oxumaq), avadanlıqları sistemə tanıdıb konfiqurasiya etmək, avadanlıqların işini idarə etmək, kəsilmələrə cavab vermək və s. göstərmək olar.

Assembler

— aşağı səviyyəli proqramlaşdirma dili olub maşın kimandalarının işləmək üçün rahatnaşdırılmış yazı formasından ibarətdir.
Əmrləri CPU komandalarına uyğundur,
Rahat simvolik yazı forması var
(мнемокод) команд и их аргументов. Также язык ассемблера обеспечивает базовые программные абстракции: связывание частей программы и данных через метки с символьными именами и директивы.
Verilənlər bloklarını proqrama daxil etmək imkanı var(описанные явно или считанные из файла); повторить определённый фрагмент указанное число раз; компилировать фрагмент по условию; задавать адрес исполнения фрагмента, менять значения меток в процессе компиляции; использовать макроопределения с параметрами и др.
Каждая модель процессора, в принципе, имеет свой набор команд и соответствующий ему язык (или диалект) ассемблера.

Üstünlük və zəif cəhətləri

•İzafi kodun minimal olması(использование меньшего количества команд и обращений в память). Как следствие — большая скорость и меньший размер программы
•большие объемы кода, большое число дополнительных мелких задач
•плохая читабельность кода, трудность поддержки (отладка, добавление возможностей)
•трудность реализации парадигм программирования и любых других сколько-нибудь сложных конвенций, сложность совместной разработки
•меньшее количество доступных библиотек, их малая совместимость
•непосредственный доступ к аппаратуре: портам ввода-вывода, особым регистрам процессора
•возможность написания самомодифицирующегося кода (т.е. метапрограммирования, причем без необходимости программного интерпретатора)
•максимальная «подгонка» для нужной платформы (использование специальных инструкций, технических особенностей «железа»)

Sintaksis

Общепринятого стандарта для синтаксиса языков ассемблера не существует. Однако, существуют стандарты де-факто — традиционные подходы, которых придерживаются большинство разработчиков языков ассемблера. Основными такими стандартами являются Intel-синтаксис и AT&T-синтаксис.

Общий формат записи инструкций одинаков для обоих стандартов:

`[метка:] опкод [операнды] [;комментарий]`

Опкод — непосредственно мнемоника инструкции процессору. К ней могут быть добавлены префиксы (повторения, изменения типа адресации и пр.). В качестве операндов могут выступать константы, названия регистров, адреса в оперативной памяти и пр.. Различия между стандартами Intel и AT&T касаются, в основном, порядка перечисления операндов и их синтаксиса при различных методах адресации.

Используемые мнемоники обычно одинаковы для всех процессоров одной архитектуры или семейства архитектур (среди широко известных — мнемоники процессоров и контроллеров Motorola, ARM, x86). Они описываются в спецификации процессоров.

Например, процессор Zilog Z80 наследовал систему команд Intel i8080, расширил ее и поменял мнемоники (и обозначения регистров) на свой лад. Например, сменил интеловские mov на ld. Процессоры Motorola Fireball наследовали систему команд Z80, несколько её урезав. Вместе с тем, Motorola официально вернулась к мнемоникам Intel. и в данный момент половина ассемблеров для Fireball работает с интеловскими мнемониками, а половина с мнемониками Zilog.

Директивы

Кроме инструкций, программа может содержать директивы: команды, не переводящиеся непосредственно в машинные инструкции, а управляющие работой компилятора. Набор и синтаксис их значительно разнятся и зависят не от аппаратной платформы, а от используемого компилятора (порождая диалекты языков в пределах одного семейства архитектур). В качестве набора директив можно выделить:

- определение данных (констант и переменных)
- управление организацией программы в памяти и параметрами выходного файла
- задание режима работы компилятора
- всевозможные абстракции (т.е. элементы языков высокого уровня) — от оформления процедур и функций (для упрощения реализации парадигмы процедурного программирования) до условных конструкций и циклов (для парадигмы структурного программирования)
- макросы

Происхождение и критика термина «язык ассемблера»

Данный тип языков получил свое название от названия транслятора (компилятора) с этих языков — ассемблера (англ. assembler — сборщик). Название последнего обусловлено тем, что на первых компьютерах не существовало языков более высокого уровня, и единственной альтернативой созданию программ с помощью ассемблера было программирование непосредственно в кодах.

Язык ассемблера в русском языке часто называют «ассемблером» (а что-то связанное с ним — «ассемблерный»), что, согласно английскому переводу слова, неправильно, но вписывается в правила русского языка. Однако, сам ассемблер (программу) тоже называют просто «ассемблером», а не «компилятором языка ассемблера» и т. п.

Использование термина «язык ассемблера» также может вызвать ошибочное мнение о существовании единого языка низкого уровня, или хотя бы стандарта на такие языки. При именовании языка, на котором написана конкретная программа, желательно уточнять, для какой архитектуры она предназначена и на каком диалекте языка написана.

Assemblerdə

İKİ ƏDƏDİN TOPLANMASI PROQRAMI

Программа складывает два числа, и проверяет результат. Если сумма равна 0 – выводится одно сообщение, если нет – другое.
В данной задаче самое интересное – это работа с метками, командами jz, jmp и test.
test – это операция логического сравнения двух операндов, размерностью байт, слово или двойное слово. В процессе выполняется операцию логического умножения: бит результата равен 1, если соответствующие биты операндов равны 1, в остальных случаях бит результата равен 0. Затем устанавливаются флаги, в том числе флаг ZF (zero flag), который равен 1, если результат логического умножения равен нулю.
Флаг ZF в дальнейшем используется для анализа результата.
jnz – выполняет переход по указанной метке, если не установлен флаг ZF. Данная команда обычно используется с операциями сравнения, которые влияют на состояние флага ZF. Например, test и cmp.
jz – выполняет переход по указанной метке, если установлен флаг ZF. Данная команда обычно используется с операциями сравнения, которые влияют на состояние флага ZF. Например, test и cmp.
jmp – выполняет безусловный переход по указанной метке.

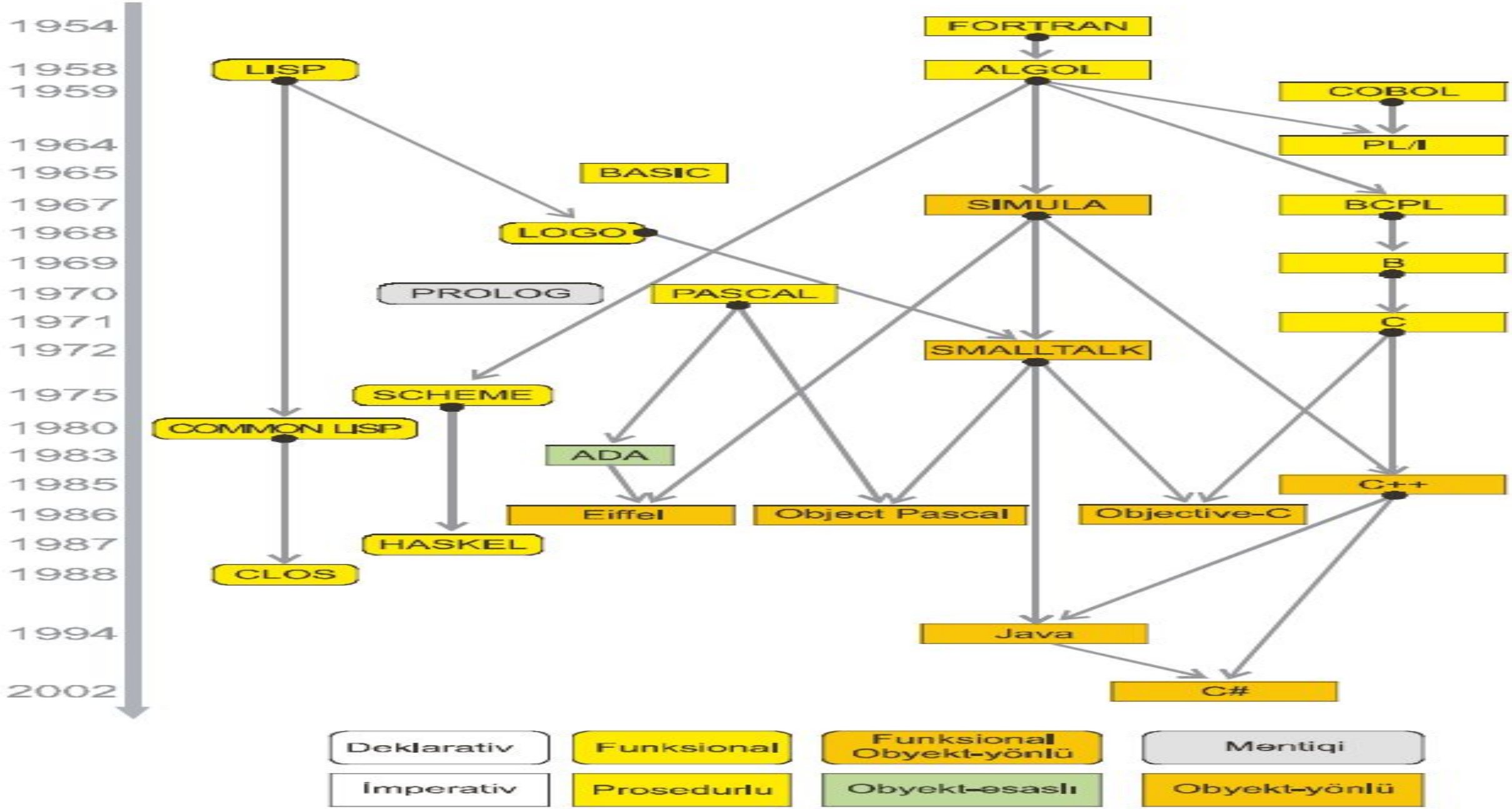
В данной задаче самое интересное – это работа с метками, командами jz, jmp и test.
test – это операция логического сравнения двух операндов, размерностью байт, слово или двойное слово. В процессе выполняется операцию логического умножения: бит результата равен 1, если соответствующие биты операндов равны 1, в остальных случаях бит результата равен 0. Затем устанавливаются флаги, в том числе флаг ZF (zero flag), который равен 1, если результат логического умножения равен нулю.
Флаг ZF в дальнейшем используется для анализа результата.
jnz – выполняет переход по указанной метке, если не установлен флаг ZF. Данная команда обычно используется с операциями сравнения, которые влияют на состояние флага ZF. Например, test и cmp.
jz – выполняет переход по указанной метке, если установлен флаг ZF. Данная команда обычно используется с операциями сравнения, которые влияют на состояние флага ZF. Например, test и cmp.
jmp – выполняет безусловный переход по указанной метке.

```
.486
.model flat, stdcall
option casemap: none
include /masm32/include/windows.inc
include /masm32/include/user32.inc
include /masm32/include/kernel32.inc
includelib /masm32/lib/user32.lib
includelib /masm32/lib/kernel32.lib

include /masm32/macros/macros.asm
uselib masm32, comctl32, ws2_32

.data

.code
start:

mov eax, 123
mov ebx, -90
add eax, ebx

test eax, eax

jz zero
invoke MessageBox, 0, chr$("В eax не 0!"), chr$("Info"), 0
jmp lexit

zero:
invoke MessageBox, 0, chr$("В eax 0!"), chr$("Info"), 0

lexit:
invoke ExitProcess, 0

end start


```

təkrarlamalı” və s.
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 gerçəkləş dir - di yindən qısa müddət ərzində yüzlərlə yeni dil meydana çıxdı (1993-cü ilin hesablamalarına görə, 1950-ci ildən həmin vaxta kimi 1000-dən artıq proqramlaşdırma 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. Digər tərəfdən, assemblerdə yazılmış proqram, onun yüksək səviyyəli dildə yazılmış bənzərindən praktik olaraq həmişə səmərəlidir. Yəni yüksək səviyyəli dilin kompilyatorunun yaratdığı icra faylı funksiona cəhətdən eyni olan assembler proqramından daha çox yer tutur və yavaş işləyir. Doğrudur, son illər mikroprosessorların inkişafı nəticəsində kompilyatorlar daha optimal kodlar generasiya edir

İndi dünyada ***bir neçə min proqramlaşdırma dili*** mövcuddur. 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 dilind yazılmış proq - ramların 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 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;



Yüksək səviyyəli dillərin təkamül sxemi

PASCAL

Pas (Paskal) – 1971-ci ildə professor Niklaus Virt (Niclaus Wirth) tərəfindən yaradılmışdır. Dil fransız riyaziyyatçısı və filosofu Blez Paskalın (Blaise Paskal) şərəfinə adlandırılmışdır və qısa zaman içində universitetlərdə istifadə edilən proqramlaşdırma dili halına gəlmişdir.

Paskal proqramının strukturu

program <Proqramın adı>;

Təqdimat Bölümü

uses – gərəklì kìtabxana və proqram modullarının göstərìlməsì;

label – nìşanları (metkaları) tanıtmaq;

const – sabìtlərì tanıtmaq;

type – tìplərì tanıtmaq;

var – dəyìşənlərì tanıtmaq;

procedure – proseduraları tanıtmaq;

function – funksìyaları tanıtmaq;

İcra Bolümü

Begin

<əmrlər bölümü>

End.

XİDMƏTİ SÖZLƏR

and – və	mod – qalıq
array – massìv	nil – sıfır
begin – başlanõıc	not – yox (xeyr)
case – seçìm	of – dan (başlayaraq)
const – konstantlar	or – və ya
div – qalıqsız bölmə	packed – sıxlaşmış
do – ìcra et	procedure – prosedura
downto – aşağı qədər	program – proqram
else –əks halda	record – yazı
end – son	repeat – təkrar et
file – fayl	set – çoxluq
for – üçün (dövr üçün)	then – onda
function – funksìya	to – qədər
goto – keç	type – tìp
if – əgər	until – hələlìk
in – daxìlìndə	var – dəyìşənlər
label – nìşan	while – hələ ki (nə qədər ki)

İfadə	Qiymət	Nümunə	Qiymət
not true	false	-1 < 0	true
true and true	true	0 > -1	true
true and false	false	16 div 5 = 3	false
false and true	false	16 mod 5 = 1	true
false and false	false	not(b > 4)	true
not false	true	(-3 > -1) or (3 > 5)	false
true or true	true	(10 div 5=1) and (3=3)	false
true or false	true	(17 * 3 = -51) or (1>0)	true
false or true	true		
false or false	false		

Əməl	Əməlin təyinatı
+, - , not	işarələmə, məntiqi inkar
***, / div, mod and**	vurma, bölmə- tam ədəd əməlləri məntiqi hasil
+, - or, xor	toplama, çıxma məntiqi cəm
=, <>, <, >, <=, >=	müqayisə

Tipin adı	Qiymət diapazonu
byte	0..255
shortint	-128..+127
Word	0..65535
integer	-32768..+32767
longint	-2147483648..+214748

inc(x, y)	**integer**	**integer**	**x-ì y qədər böy-r**
inc(x)	integer, char	integer , char	x-ì 1 vahìd böy-r
dec(x, y)	integer	integer	x-ì y qədər az-r
dec(x)	integer, char	integer , char	x-ì 1 vahìd az-r

Tìpìn adı	**QìymƏt dìapazonu**
single	1.5*10-45 .. 3.4*1038
real	2.9*10-39 .. 1.7*1038
double	5.0*10-324 .. 1.7*10308
extended	3.4*10-4951 .. 1.1*104932

Funksiya	Arqumentin tipi	Nəticənin Tipi	Riyazi yazılış
t	integer, real	integer, real	|x|
arctan(x)	integer, real	Real	arctg(x)
cos(x)	integer, real	Real	cos(x)
sin(x)	integer, real	Real	sin(x)
exp(x)	integer, real	Real	
ln(x)	integer, real	Real	ln(x), x>0
sqrt(x)	integer, real	Real	
sqr(x)	integer, real	integer, real	
ord(x)	ordered	integer	ASCII –simvol kodları
pred(x)	ordered	nizamlanmış	x –in əvvəlki qiymətini verir
round(x)	real	nizamlanmış	x -i tam ədədə yuvarlaqlaşdırır
trunc(x)	real	Real	x –in tam hissəsi
frac(x)	real	Real	x–in kəsr hissəsi
odd(x)	integer	Boolean	true (x –tək) false(x –cüt) çevirir

Sətri hərflərin baş hərflərə çevrilməsi

```
Uses Crt;
Var S : String;
    I : Integer;
Begin
 Write('Sətri Hərflərlə mətn daxil et! : ');
 ReadLn(S);
 For I:=1 To lenght[S] do S[I]:=upcase(S[I]);
 WriteLn('Baş hərflər sətri: ', S);
End.
```

PASKAL-da psixoloji test proqramı

- **uses** crt;
- **var** a,b,c,d,m:longint;
- g,r,y,k:text;
- **begin** clrscr;
- a:=0; b:=0; c:=0; d:=0;
- //textmode(2);
- textcolor(3);
- textbackground(1);
- write('Her teqdim olunan nomredeki xarakterik xususiyyetden size en cox uygun olanini');
- Readln;
- write('secib qeyd edin.');
- Readln;
- write('I.');

Birinci və axırdan iki hərfi baş hərf edən proqram

- **Var** S,n:String;
- İ:İnteger;
- **Begin**
- n:=' ';
- Write ('Hərfləri Sətri daxil et');
- Readln(S);
- S[1]:=UpCase(S[1]);
- **For** I:=1 **to** Length(S) **Do**
- **If** S[I]=n **then**
- S[I+1]:=UpCase(S[I+1]);
- Writeln('Duzgun yazılış: ',S);
- **End**.

Yazılan mətnin bir hərfini baş digərini sətri edən proqram.

- **Var** S,n:String;
- İ:İnteger;
- **Begin**
- Readln(S);
- Begin
- **For** I:=1 **to** Length(S) **Do**
- S[İ]:=UpCase(S[İ]);
- **End;**
- **Begin**
- **For** I:=1 **to** Length(S) **Do**
- **If I mod 2=0 then**
- **S[I]:=LowerCase(S[I]);**
- **End;**
- Writeln(S);
- **End**.

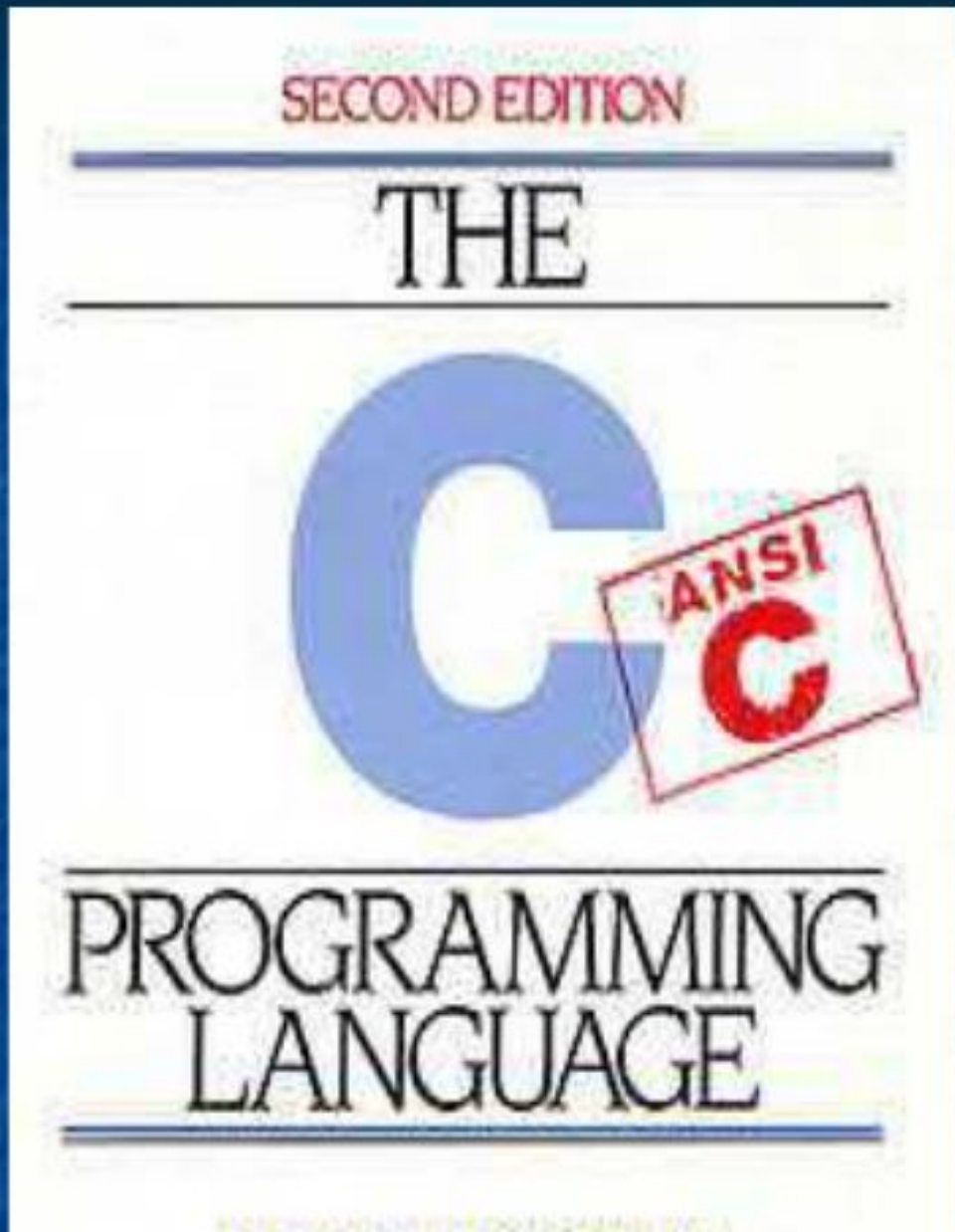
EVKLİD məsələsi

- **program** evklid;
- **var** x,y1,y2,n:longint; //dəyişənlərin təsviri,longint -2147483648... +2147483647,integer -32768...+32767
- **begin** {programın govdəsinin başlanğıcı}
- writeln('n-i daxil edin!'); {məlumatın ekrana cıxarılması}
- readln(n); {n-in oxuması }
- **for** x:=1 **to** n **do** //for operatoru
- y1:=2*X-3; //y1 ve y2 daxil edilmesi
- Y2:=X+2;
- **if** y1=y2 **then** writeln('y1=',y1,' y2=',y2,' y1=y2'); //if şərt operatoru
- **if** y1>y2 **then** writeln('kicik eded daxil edin!');
- **if** y1<y2 **then** writeln('Daha boyuk eded daxil edin!');
- **end**.

Fibonaçi ədədləri

```
program  fibonaci;                          {program adı}
var n,i,x1,x2,summa:integer;                {dəyişənləirin təsviri}
label 1;                                    {nişanın təsviri }
begin                                       {programın govdəsinin başlanğıcı}
1:                                          {goto operatorunun kecidi}
x1:=0;     x2:=1;                           {sabitlər}
write('Nece element lazimdir? ');           {məlumatın ekrana cıxarılması}
readln(n);                                  {n-in oxuması }
if  n>0  then                               {şərt operatoru}
      begin                                 {2}
      writeln ('1-ci element= ',x2);
      for  i:=2 to n do                     {for operatoru }
            begin                           {3}
            summa:= x1+x2 ;
            writeln( i,'-ci element= ',summa);
            x1:=x2;     x2:= summa ;
            end;                            {3}
      end                                   {2}
else
Writeln ('Sehv! Novbeti defe musbet eded goturun!');
goto 1;                                     {kecid operatoru}
end.                                        {govdənin sonu}
```

C proqramlaşdırma dili 1970-ci illərin əvvəllərində Denis Ritçi və Ken Tompson tərəfindənUNIX əməliyyat sistemi üçün yaradılmış proqramlaşdırma dil. C proqramlaşdırma dili "A" və "B" dilinin inkişafı məqsədilə yaradılmışdır. Müasir dövrdə bu dil sistem proqramramlaşdırması üçün ən güclü proqramlaşdırma dillərindən biri hesab olunur. Windows,Linux, Unix, FreeBSD və



C++

Bu dildə proqramı yazarkən adətən, kompilyator olaraq **Ms Visual Studio** proqramlaşdırma sistemindən istifadə edilir. Ms Visual Studio kompilyatorunun quraşdırma faylını http://www.microsoft.com/express/Downloads keçidindən kompüterinizə endirə bilərsiniz.

Ilk test:

Ms Visual Studio proqramını yükləyin. Daha sonra ***File -> New -> Project*** Seçimini edin. ***New Project*** pəncərəsi açılacaq. Project types: panelindən ***Win32 , Templates***: panelindən isə ***Win32 Console Application*** seçimini edin.

Daha sonra Name: pəncərəsindən yeni yaradacağımız proqramın adını daxil edirik. Bu pəncərəyə ***prog1*** yazıb Ok düyməsini basırıq. Açılan yeni pəncərədə Finish düyməsini basırıq. Bu əməliyyatlar ilk proqramımızı tərtib etmək üçün bütün lazımı faylları yaradacaq və proqramın mətn faylı redaktoru pəncərəsi - ***prog1.cpp*** aktivləşdirəcək. Bu fayl aşağıdakına bənzər formada olur:

```
// prog1.cpp : Defines the entry point for the console // application. #include "stdafx.h" int _tmain(int argc, _TCHAR* argv[]) { return 0; }
```

Bu mətn kompiyator tərəfindən avtomatik yaradılıb. Proqramın mətn faylında aşağıdakı kimi dəyişikliklər edirik. ***#include "stdafx.h"*** sətrindən sonra ***#include*** , ***{*** mötərəzəsindən sonra ***isə std::cout<<"Salam dunya \n"; , return 0;*** sətrindən əvvəl isə ***int x; std::cin>>x***; sətrlərini daxil edirik. Aşağıdakı kimi:

```
// prog1.cpp : Defines the entry point for the console // application. #include "stdafx.h" #include int _tmain(int argc, _TCHAR* argv[]) { std::cout<<"Salam dunya \n"; int; std::cin>>x; return 0; }
```

Etdiyimiz dəyişiklikləri saxlamaq üçün File -> Save all düyməsini basırıq. Artıq proqramımızın mətn faylı hazırdır və biz onu kompilyasiya edə bilərik. Kompilyasiya nəticəsində kompilyator bizim mətn faylından prosessor tərəfindən icra oluna bilən ikilik proqram alacaq. Proqramı kompilyasiya etmək üçün Build -> Build Solution əmrini daxil edirik. Proqramımızın kompilyasiyası başlayacaq və Output pəncərəsinə ötürüləcək. Əgər sonda Build: 1 Succeeded ... sətri çap olunursa deməli proqramımız uğurla kompilyasiya olunmuşdur. İndi isə proqramımızı yerinə yetirək. Bunun üçün Debug -> Start Debugging düyməsini basırıq. Nəticədə kansole pəncərəsi açılacaq və *Salam dunya* mətni çap olunacaq. Proqramı söndürmək üçün klaviaturadan hər-hansı simvol daxil edib enter düyməsinə basmağımız kifayətdir.

Daxil edilən məlumatın növünə və yaddaşda tutduğu yerin həcminə görə dəyişənlər tiplərə ayrılır. Misal üçün
tam ədədlər tipi – int (4 bayt),
kəsr ədədlər tipi – double (8 bayt) ,
simvol tipi – char (1 bayt),
sətir tipi – char [], char * v.s.

Bu tiplərə standart tiplər deyilir. Bundan əlavə C++ dilində *ünvan dəyişənləri tip, struktur tiplər, siyahılar* və *siniflər (klasslar)* da çox geniş istifadə olunur. Dəyişənlərə istədiyimiz kimi ad verə bilərik yalnız və yalnız hərflərdən(ingilis əlifbasının) , '_' simvolundan və rəqəmlərdən istifadə etməklə. Dəyişənin adı mütləq hərflə başlamalıdır və operator, tip v.s. adlarından da dəyişən adı kimi istifadə etmək olmaz. Operatorlarla gələn mövzularda tanış olacayıq. Beləliklə, cəm proqrmında hər iki toplananı və onların cəmini yerləşdirmək üçün biz tam tipli 3 dəyişən təyin etməliyik. Gəlin bu dəyişənləri uyğun olaraq top1, top2 və cem kimi adlandıraq. Bu dəyişənləri təyin etmək üçün proqram kodu aşağıdakı kimi olacaq. int top1; int top2; int cem; Qeyd edək ki, eyni tipdən olan dəyişənləri vergüllə ayırmaqla bir sətirdə də elan edə bilərik. Aşağıdakı kimi :

```
 int top1,top2,cem;
// prog1.cpp : Defines the entry point for the console application.
 #include "stdafx.h" #include int _tmain(int argc, _TCHAR* argv[])
 { int top1, top2, cem; top1 = 4; top2 = 6;
cem = top1 + top2; std::cout<<" 4 ile 6 -nin cemi ="<>x;
return 0; }
```

Tək ədəd daxil edirik. 10-dan > və ya < şərtinə görə C++ da budaqlanma:

İF OPERATORU

```
#include <iostream> using namespace std;
 int main() { setlocale(0, ""); double num;
cout << «İxtiyari ədəd daxil et: "; cin >> num;
 if (num < 10) {                    // 10-dan kiçik olarsa. cout << «Bu ədəd 10-dan kiçikdir." << endl; } else
 { // əks halda cout << «Ədəd 10-dan böyük ya da =-dir." << endl;                    }
 return 0;
 }
```

10-dan > və ya < şərtinə görə C++ proqramı müvafiq mətn verəcək **if** operatorunda *C++* - da şərt həmişə mötərizədə verilir Fiqur mötərizədə şərtin ***bodisi*** olur. Şərt ödəndikdə fiqur mötərizə arasındakı əmrlər icra olunur.

Budaqlanmaya nümunə:

```
        if (num < 10) { // ədəd 10-dan kiçik olsa. cout << "Bu ədəd 10-dan kiçikdir." << endl; } else { // əks halda cout << «Bu ədəd
10-dan böyük və ya =-dir." << endl; }
```

num dəyişəninin 10-a = halı üçün əlavələr edək:

```
if (num < 10) { // Əgər daxil edilmiş ədəd 10-dan kiçikdirsə. cout << "Bu ədəd 10-dan kiçikdir." << endl; } else
 if (num == 10) { cout << "Bu ədəd 10-a bərabərdir." << endl; } else
 { //  əks halda cout << "Bu ədəd 10-dan böyükdür." << endl; }
```

3 şərt yoxlanır:

- **İ —10-dan kiçik**
- **İİ — ədəd 10-a = olduqda**
- **İİİ—10-dan böyük**

II şərtdə bərabərlik şərt operatoru == istifadə olunur, bu mənimsətmə deyil!
num == 10 yazılışı dəyişəni 10 ədədiylə müqaisə edir.
•Bir = - lik, yəni mənimsətmə yazəlsaydı dəyişənin qiymətini yeniləşdirərdi (10 edərdi).
Hər **if** operatorunun 1 *else operatoru olur*. Yuxarıdakı proqramı belə yazmaq olar:

```
#include <iostream> using namespace std; int main() { setlocale(0, ""); double num; cout << "Введите произвольное число: "; cin >> num; if (num < 10) // Если введенное число меньше 10. cout << "Это число меньше 10." << endl; else if (num == 10) cout << "Это число равно 10." << endl; else // иначе cout << "Это число больше 10." << endl; return 0; }
```

Bu üsul daha yığcamdır. Şərt ödəndikdə 1-dən çox əmr yerinə yetirilirsə fiqur mötərizələrdərn istifadə etmək zəruridir.
Nümunə:

```
#include <iostream> using namespace std; int main() { setlocale(0, ""); double num; int k; cout << "Введите произвольное число: "; cin >> num; if (num < 10) { // Если введенное число меньше 10. cout << "Это число меньше 10." << endl; k = 1; } else if (num == 10) { cout << "Это число равно 10." << endl; k = 2; } else { // иначе cout << "Это число больше 10." << endl; k = 3; } cout << "k = " << k << endl; return 0;
}
```

Bu proq. num. dəyişəninin qiymətini yoxlayır. 10-dan kiçik olarsa, k dəyişəninə 1 verir. Əgər num 10-a =-dirsə, k dəyişəninə iki qiyməti verilir. Əks halda, 3 qiyməti verilir. Budaqlanmadan sonra k -nın qiyməti ekrana çıxarılır.

1 dən 1000-ə qədər ədədlərin cəmi proqramı:

```
#include <iostream> using namespace std;
int main()
{ int i;                                    // dövr sayğacı
int sum = 0;                                // cəm 1 –dəm  1000 - ə qədər.
setlocale(0, "");
 for (i = 1; i <= 1000; i++)                // ilkin qiymət 1, son 1000 dövrün addımı  1.
{ sum = sum + i; }
cout << "1 dən  1000-ə qədər ədədlərin cəmi = " << sum << endl;
 return 0;
 }
```

Proqramın cavabı : **500500-dür.**

Yash, boy, chəki nisbətinin analizi proqramı

```
#include<iostream>
using namespace std;
main (){
	int boy,cheki,ferq , yash;// int tipli "boy,cheki,ferq" deyisenlerini  daxil edirik
	int d,d1,netice; // int tipli "d,d1" deyisenlerini  daxil edirik
	string ad; // string tipli "ad" deyiseni daxil edirik
	string needek; // string tipli "needek" deyiseni daxil edirik
	int orta=0;// int tipli "orta" deyiseni daxil edib 0 reqemine menimsedirik
	cout << "Adamlarin sayi :";// 'Adamlarin sayi :' ekrana yazilir
	cin >> d1 ;// klavisden d1 deyisenine istenilen edede menimsede bilerik
	for (d=0;d<d1;d++)// sonra ise d-e 0 menimsedirik , adamlarin sayi yeni d1 boyuk olmalidi d-den ve d adamlarin sayina qeder catanda dovr diyansin
	{
	cout << "Adinizi daxil edin : ";// 'Adinizi daxil edin :' ekrana yazilir
	cin >> ad;// klavişden ad deyisenine istenilen adi menimsede bilerik
	cout << "Boy ededi daxil edin : "; // 'Boy ededi daxil edin :' ekrana yazilir
	cin >> boy;// klavisden boy deyisenine istenilen edede menimsede bilerik
	cout << "Cheki. Ededi daxil edin : ";// `Cheki. Ededi daxil edin : ' ekrana yazilir
	cin >> cheki;// klavisden cheki deyisenine istenilen edede menimsede bilerik
	cout << "Yaşinizi daxil edin  : ";// Yaşinizi daxil edin  : ' ekrana yazilir
	cin >> yash;// klavisden cheki deyisenine istenilen ededi menimsədə bilərik
	// bu hissede ise ferq deyiseninə boy - 100 - cheki - ni menimsedirik
		ferq= boy - 100 - cheki;
	// bu hissede  int tipinde verilmis orta deyisenine 0 menimsetdiyimiz halda funksiyamiz orta=orta+ferq (yeni 0+(yuxaridaki)ferqin cavabi)
	orta = orta +ferq ;
	//bu hissede int tipinde verilmis netice deyiseni menimsedirik yash deyiseni- 18 / 3
	netice = (yash - 18)/3 ;
	//burada ise  ferq deyiseni beraberdir (ferq= boy - 100 - cheki;) burda alinan ferq + (netice = yash - 18/3 ;) burada alinan netice
	ferq = ferq + netice ;
```

```
	if(ferq==0 ) needek="   Ideal ceki !      Yasiniz ucun ela neticedi !" ;// bu shertimizde ise ferq deyiseninin cavabi 0 olarsa ekrana needek deyisenine daxil etdiyimiz Ideal ceki fikri cixir
	else // shertimiz odemesse eks halda asagidaki sherte baxiriq
	{
		if (ferq<0 ) needek =  "Siz Ariqlamalisiniz  ";// bu sertimizde ise ferq deyiseninin cavabi 0 dan kicik olarsa ekrana needek deyisenine daxil etdiyimiz Ariqlamali  fikri cixir
		else // shertimiz odemesse eks halda ekrana needek deyisenine daxil etdiyimiz Kokelmeli fikri cixir
		needek = "Siz Kokelmelisiniz" ;
	}
	// sonra ise ekrana ad deyiseninin neticesi 'ucun : ' ferq deyiseninin neticesi ve nehayet needek deyiseninin neticesi cixir
	cout << ad  << "  ucun :   " << needek    << "   " << ferq << "   kq" ;
	cout << "\n";
	cout << "-----------------------------------------------" << "\n";
	}
	// burada ise orta deyiseninin (orta = orta +ferq ;)cavabi bolunsuz yeni daxil etdiyimiz adamlarin sayi
	orta =  orta / d1;
	// burada ise ekrana 'Orta beraberdir :' ve orta deyisenin son neticesi (orta =  orta / d1) cixir
	cout << "Cekiye orta  teleb  : " << orta   << "\n" ;
	return 0;    // bu kod ise neticemizi gorene qeder ekrani sonmeye imkan vermir. Klavisden ne ise daxil etdikde kompilyasiya sona catir
}
```

C# proqramlaşdırma dili

C # proqramlaşdırma dili (si şarp şəklində tələffüz edilir), Microsoftun inkişaf etdirmiş olduğu yeni nəsil proqramlaşdırma dilidir. Microsoft tərəfindən yaradılmış .NET Texnologiyası üçün yazılmışdır.

Microsoft məhsulu olsa da ecmain və ISO standartları altına alınmışdır.

C proqramlaşdırma dilində N tam ədəd dəyişənini 1 vahid atırmaq üçün N++ istifadə edilir. C ++ dili adını, C diliylə obyektyönümlü Proqramlaşdırma üçün əlavələr (C With Classes) almışdır. Bənzər şəkildə C ++ dilinə yeni əlavələr edilərək ((C ++) ++) bir addım daha irəliyə götürülmüş və tamamilə obyektə istiqamətli hazırlanmış C # dilinin isimlendirilmesinde, + xarakterlərinin bir-birlərinə yakınlaşmış halı və bir melodiya açarı olan C # Major istifadə edilmişdir.

Bu dilin hazırlanması Pascal, Delphi derleyiciler və J ++ proqramlaşdırma dilinin dizaynlarıyla bilinən Anders Hejlsberg liderlik etmişdir.

Bir çox sahədə Java'yı özünə nümunə götürər və C # da java kimi C və C ++ kod sözdizimine bənzər bir kod quruluşundadır. .NET Kitabxanalarını istifadə məqsədiylə yazılan proqramların çalışdığı kompüterlərdə uyğun bir kitabxananın və yorumlayıcının olması lazımlıdır. Bu, Microsoftun .Net Framework'u ola biləcəyi kimi ecmain standartlarına uyğun hər hansı bir kitabxana və şərh da ola bilər. Məşhur digər kitabxanalara nümunə olaraq Portable.Net və Mono verilə bilər.

Xüsusilə obyekt yönümlü proqramlaşdırma anlayışının inkişafına kömək olan ən aktiv proqramlaşdırma dillərindən biridir .NET platformasının ana dili olduğu bəzi seqmentlər tərəfindən qəbul edilsə də bəzi mütəxəssislər bunun doğru olmadığını göstərirlər.

C #, .NET orta səviyyəli proqramlaşdırma dillerindendir. Yəni həm maşın dilinə həm də insan qəbuluna bərabər səviyyədədir. Buradakı orta ifadəsi dilin gücünü deyil maşın dili ilə gündəlik danışıq dilinə olan məsafəsini göstərir. Məsələn; Visual Basic .NET (VB.NET) yüksək səviyyəli bir dildir desək bu, dilin insanların gündəlik həyatlarında danışma formasına yaxın şəkildə yazıldığını ifadə etməkdədir. Bu səbəbdən VB.NET, C # .NET'ten daha güclü bir dildir deyə bilmərik. Proqramın çalışması istənən kompüterlərdə framework heyəti olması lazımdır. (Windows 7 və Windows Vistada .NET Framework quruludur)

C# da Yazılmış proqram

```
{
    int x, y, z;
    Console.Write("Birinci ededi daxil edin : ");
    x = Convert.ToInt32(Console.ReadLine());
    Console.Write("Ikinci ededi daxil edin : ");
    y = Convert.ToInt32(Console.ReadLine());
    Console.Write("Hesabinizi secin - \n 1.Toplama \n 2.Cixma \n 3.Vurma \n 4.Qismet \n ");
    z = Convert.ToInt32(Console.ReadLine());
    switch (z)
    {
        case 1:
            {
                Console.Write("Toplama Emeliyyatinizin Neticesi : ");
            Console.Write(x + y);
                break;}
        case 2:
            {
                Console.Write("Cixma Emeliyyattinizin Neticesi : ");
            Console.Write(x - y);
                break;}
        case 3:
            {
                Console.Write("Vurma Emeliyyatinizin Neticesi : ");
            Console.Write(x * y);
                break;}
        case 4:
            {
                Console.Write("Qismet Emeliyyatinizin Neticesi : ");
            Console.Write(x / y);
                break;}
        default :{Console.Write("Bu cur emeliyyat tetbiq edilmeyib");
                    break;}
    }
    Console.ReadLine();
```

C++ da Yazılmış proqram

```
#include <iostream>
using namespace std;
main (){

int x, y, z;
   cout << "Birinci ededi daxil edin : ";
   cin >> x;
   cout <<"Ikinci ededi daxil edin : ";
   cin >> y;
   cout << "Hesabinizi secin - \n 1.Toplama \n 2.Cixma \n 3.Vurma \n 4.Qismet \n ";
   cin >> z;
            switch (z)
            {
                case 1:
                    {
                    cout << "Toplama Emeliyyatinizin Neticesi : ";
                    cout << x + y;
                        break;}
                case 2:
                    {
                    cout << "Cixma Emeliyyattinizin Neticesi : ";
                    cout << x - y;
                        break;}
                case 3:
                    {
                    cout << "Vurma Emeliyyatinizin Neticesi : ";
                    cout << x * y;
                        break;}
                case 4:
                    {
                    cout << "Qismet Emeliyyatinizin Neticesi : ";
                    cout << x / y;
                        break;}
                default :{cout << "Bu cur emeliyyat tetbiq edilmeyib";
                            break;}
            }
                return 0;
        }
```

C#

Yash, boy, chəki nisbətinin analizi proqramı

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace C_sharp_insanin_cekisi_ve_yasi__gosder
{
   class Program
   {
      static void Main(string[] args)
      {
         int boy, cheki, ferq, yash; // int tipli "boy,cheki,ferq" deyisenlerini  daxil edirik
         int d, d1,netice; // int tipli "d,d1" deyisenlerini  daxil edirik
         string ad; // string tipli "ad" deyiseni daxil edirik
         string needek; // string tipli "needek" deyiseni daxil edirik
         int orta = 0;  // int tipli "orta" deyiseni daxil edib 0 reqemine menimsedirik
         Console.WriteLine("Adamlarin sayini daxil edin : ");  // `Adamlarin sayini daxil edin : ' ekrana yazilir
         d1 = Convert.ToInt32(Console.ReadLine());   // klaviaturadan d1 deyisenine istenilen ededi menimsədirik
         for (d = 0; d < d1; d++)  // d-e 0 menimsedirik , adamlarin sayi yeni d1 boyuk olmalidi d-den ve d adamlarin sayina qeder
catanda dövr diyansin
         {
            Console.WriteLine("Adinizi daxil edin : ");   // 'Adinizi daxil edin :' ekrana çıxarılır
```

```
 ad = Convert.ToString(Console.ReadLine());  // klavisden ad deyisenine istenilen adi menimsede bilerik
            Console.WriteLine("Boy ededi daxil edin : ");// 'Boy ededi daxil edin :' ekrana yazilir
            boy = Convert.ToInt32(Console.ReadLine());// klavisden boy deyisenine istenilen edede menimsede bilerik

            Console.WriteLine("Ceki eded daxil edin : ");// 'Cheki ededi daxil edin :' ekrana yazilir
            cheki = Convert.ToInt32(Console.ReadLine());// klavisden cheki deyisenine istenilen edede menimsede bilerik

            Console.WriteLine("Yasinizi daxil edin : "); // 'Yasinizi daxil edin :' ekrana yazilir
            yash = Convert.ToInt32(Console.ReadLine()); // klavisden yash deyiseni istenilen edede menimsede bilerik
            // bu hissede ise ferq deyisenini boy - 100 - cheki - e menimsedirik
            // yeni klavisden boy-a menimsetdiyimiz eded - 100 - cheki-e menimsetdiyimiz eded
            ferq = boy - 100 - cheki;
            // bu hissede ise int tipinde verilmis orta deyisenine 0 menimsetdiyimiz halda funksiyamiz orta=orta+ferq (yeni 0+(yuxaridaki)ferqin cavabi)
            orta = orta + ferq;
            //bu hissede int tipinde verilmis netice deyiseni menimsedirik yash deyiseni- 18 / 3
            netice = yash - 18 / 3;
            //burada ferq deyiseni beraberdir (ferq= boy - 100 - cheki;). alinan ferq + netice  yəni (yash - 18/3 ;) hesablamalıyıq:
            ferq = ferq + netice;
if (ferq == 0) needek = "  Ideal Ceki !     Yasiniz ucun ela neticedi  !";  // bu shertimizde ise ferq deyiseninin cavabi 0 olarsa ekrana needek deyisenine daxil etdiyimiz Ideal ceki fikri cixir
            else              // shertimiz odenməzse asagidaki sherte baxiriq
            {
               if (ferq < 0) needek = "Siz Ariqlamalisiniz "; // ferq deyiseninin cavabi 0 dan kicik olarsa ekrana needek deyisenine daxil etdiyimiz Ariqlamali  fikri cixmalıdır
               else // eks halda
                   needek = "Siz Kokelmelisiniz";  // needek deyisenine Siz Kokelmelisiniz mətni daxil edilir}
            // sonra ise ekrana ad deyiseninin neticesi, 'ucun : ‘, ferq deyiseninin neticesi ve needek deyiseninin neticesi cixir:
            Console.WriteLine(ad + " ucun : " + ferq + "  " + needek+"\n");
            Console.WriteLine("-------------------------------------");
        }
        // orta deyiseninin (orta = orta +ferq ;)cavabi bolunsuz yeni daxil etdiyimiz adamlarin sayi
        orta = orta / d1;
        // ekrana 'Orta beraberdir :' ve orta deyisenin son neticesi (orta =  orta / d1; esasen) cixir
        Console.WriteLine("Orta Beraberdir : " + orta+"\n");
        Console.ReadLine(); // klaviaturadan nə ise daxil etdiyiniz halda cari proqramın icra pəncərəsi bağlanır
    }
  }
}
```

JAVA

Java – Sun Microsystems şirkəti tərəfindən təqdim olunan obyekt
yonlü proqramlaşdırma dilinə və texnologiyalara verilən ümumi ad
Java proqram dili struktur və sintaksisinə görə C proqram dilinə ço
yaxındır. Rəsmi olaraq 23 may 1995-ci ildə təqdim olunmuşdur.

Başlanğıcda Oak ("palıd") adlandırılan bu dil Ceyms Qoslinq (ing.
James Gosling) tərəfindən məişət cihazlarının proqramlaşdırılması
üçün yaradılmışdı. Daha sonra bu ad Java ilə əvəz olundu və serve
proqram təminatının yazılması üçün istifadə olunmağa başladı. Bu
proqramlaşdırma dili Java kofe markasının şərəfinə adlandırılmışdı
və buna görə də onun rəsmi emblеminin üzərində fincanda
buxarlanan kofe təsvir olunmuşdur.

JavaScript və Java arasında heç bir qohumluq yoxdur.

Java dilində dəyişənlər

Dəyişən istənilən vaxt dəyişdirilə bilən qiymətdir. Java dilində əsas 7 baza tipi var:
int, long, float, double, char, String, boolean. Əlavə tiplər : byte, short, Bigİnteger və object.
Javada dəyişənin təsviri: dəyişənin_tipi dəyişənin_adı başlanğıc_qiymət; .
Javada dəyişənin adı: rəgəm, +, *, #, %, ^, &, !, @, (,), [,], ~, \, /, | simvolları ilə başlaya bilməz!

Əsas tiplərin təsvirinə aid proqram:

```
public class main
{
 public static void main(String[] args)
{
int deyisen1; int deyisen2 =18; //deyisene baslangic qiymet 18 menimsedilib
long deyisen3; long deyisen4 = 13000; //deyisene baslangic qiymet 13000 menimsedilib
float deyisen5; float deyisen6 = -3.14f; //deyisene baslangic qiymet -3.14 menimsedilib
double deyisen7; double deyisen8 = 2.7; //deyisene baslangic qiymet 2.7 menimsedilib
char deyisen9; char deyisen10 = 'a'; //deyisene baslangic qiymet a simvolu menimsedilib
String deyisen11; String deyisen12 = «Koroğlu Rəhimov küçəsi"; //deyisene baslangic qiymet - mətn menimsedilib
boolean deyisen13; boolean deyisen14 = true; //deyisene baslangic qiymet kimi true menimsedilib
}
}
```

JAVADA TİPLƏR

byte tipi - 8 bitli tam tipdir. Minimum qiymət -128(-2^7), maksimum qiymət 127 (2^7 -1). Məsələn: byte a = -50; byte b = 100;
short tipi - 16 bitli tam tipdir. Minimum qiymət -32,768 (-2^15), maksimum qiymət 32,767 (2^15 -1). Məsələn: short a = -3350; short b = 10001;
int tipi - 32 bitli tam tipdir. Minimum qiymət - 2,147,483,648.(-2^31), maksimum qiymət 2,147,483,647.(2^31 -1). Məsələn: int a = -9837350; int b = 10804901;
long tipi - 64 bitli tam tipdir. Minimum qiymət -9,223,372,036,854,775,808.(-2^63), maksimum qiymət 9,223,372,036,854,775,807. (2^63 -1). Məsələn: long a = 100000L; long b = -200000L;
float tipi - 32 bitli, sürüşən vergüllü həqiqi tipdir. Məsələn: float deyisen = 345.3453f;
double tipi - 64 bitli, sürüşən vergüllü, ikiqat dəqiqli həqiqi tipdir. Məsələn: double deyisen = 345.3453;
boolean tipi - məntiqi tipdir. true/false qiymətlərdən birini alır. Susmaya görə false qiymət verir. Məsələn: boolean bool = true;
char tipi - simvol tipidir. Min. '\u0000' (yaxud 0), Max. '\uffff' (yaxud 65,535). Məsələn: char letter = 'A';

NÜMUNƏ

```
package javadersleri.eu.pn;
public class DataTypes {

    public static void main(String[] args) {
        System.out.println("Byte type");
        byte a = -99;
        byte b = 120;
        System.out.println("a = "+a+" b = "+b);
        System.out.println(Byte.MIN_VALUE);
        System.out.println(Byte.MAX_VALUE);
```

```
System.out.println("Short type");
        short c = -919;
        short d = 27120;
        System.out.println("c = "+c+" d = "+d);
        System.out.println(Short.MIN_VALUE);
        System.out.println(Short.MAX_VALUE);
 System.out.println("Integer type");
        int e = -999999999;
        int f = 1287438790;
        System.out.println("e = "+e+" f = "+f);
        System.out.println(Integer.MIN_VALUE);
        System.out.println(Integer.MAX_VALUE);

        System.out.println("Long type");
        long g = -92372036854775808L;
        long h = 12577584375895998 50L;
        System.out.println("g = "+g+" h = "+h);
        System.out.println(Long.MIN_VALUE);
        System.out.println(Long.MAX_VALUE);
```

```
        System.out.println("Float type");
        float k = -34.443535f;
        float l = 44.444545f;
        System.out.println("k = "+k+" l = "+l);
        System.out.println(Float.MIN_VALUE);
        System.out.println(Float.MAX_VALUE);

        System.out.println("Double type");
        double m = -374.4485;
        double n = 7587483.487;
        System.out.println("m = "+m+" n = "+n);
        System.out.println(Double.MIN_VALUE);
        System.out.println(Double.MAX_VALUE);

        System.out.println("Char type");
        char letter = 'A';
        System.out.println("letter = "+letter);

        System.out.println("Boolean type");
        boolean bul1 = false;
        boolean bul2 = true;
        System.out.println("bul1 = "+bul1+" bul2 = "+bul2);
    }

}
```

Java programında dövrlərin qurulması

Dövrlər program içərisində dəfələrlə icra olunan komandalardan ibarət fraqmentlərdir. Java dilində 2 növ fraqmentlərdən istifadə olunur:

1. “n dəfə” dövrlər

2. “nə qədər ki” dövrlər

Adından göründüyü kimi 1 tip verilmiş əməliyyatın tələb olunan sayda icrası üçündür. Məsələn “n” faktorialda olduğu kimi yəni eyni ədəd “n” dəfə 1-dən n-ə qədər ardıcıl ədədlərə vurulur.

“n” dəfə dövr tipi (for operatoru)

For operatorunun 3 parametri var. 1 İnisializasiya 2 təkrarlanma şərti 3 İterasiya For (inisializasiya; şərt ;İterasiya) {

//dövrün bədəni yəni dövrü təkrar olunan hərəkət

}

Birinci parametrdə adətən bir dəyişən seçilir ki, onun köməyilə dövrün təkrarlanmaları sayılır. Bu dəyişənə sayğac deyilir. Sayğaca tələb olunan ilkin qiymət verilir.

İkinci parametrdə sayğacın məhdudlasdırılması şərti göstərilir. Yəni sayğacın hansı qiymətə dəyişəcəyi təyin edilir.

Üçüncü parametrdə dövrün hər addımından sonra sayğacı dəyişdirən ifadə göstərilir . Bir qayda olaraq iterasiya imkrement və ya dekrement olur . yəni bir bir artır və ya bir -bir azalır. Amma sayğaca zəruri qiymət mənimsədən ixtiyari ifadədən istifadə etmək olar.

Dövrün birinci addımından qabaq sayğaca ilkin qiymət mənimsədilir. Buna inisializasiya deyilir. Bu əməliyyat yalnız bir dəfə icra olunur.

İnisializasiyadan sonra dövrün hər bir addımından qabaq təkrarlanma şərti yoxlanılır. Əgər şərt doğrudursa dövrün bədəni növbəti dəfə icra olunur. Bu zaman dörün bədəni bir dəfədə icra olunmaya bilər. Bunun üçün birinci yoclamada şərt yalan olmalıdır. Dövrün hər bir addımı qutardıqdan sonra təkrarlanma şərti yoxlanmaqdan qabaq iterasiya icra olunur.

Aşağıdakı program 1-dən 100-ə qədər ədədləri ekrana çıxarır:

```
For(int i=1;i<=100;i++) {
System .out.print(i + " ");
```

7-dən mənfi 7-ə qədər ədədləri ekrana çıxaran program

```
For( int s =7; s> -8 ; s--) {
System.out.print(s + " ");
```

Aşağıdakı program 2-dən 100-ə qədər cüt ədədlərin cəmini və hasilini hesablayır:

```
Double prod =1;        // sum cəm,  prod hasil üçün
İnt sum=0
For (int j=2 j<=100; j=j+2)
 { sum=sum+j}
System.out.println(sum);
System.out.println(prod);
Prod=prod * j
```

1 dövr daxilində bir neçə sayğac istifadə etmək olar. Bu zaman iterasiya və inisializasuya daxilində olan ifadələr vergüllə ayrılır. Təkrarlanmaq şərti yalnız bir dənə olur. Amma o da özündə bir neçə sayğac saxlıyan ifadədən ibarət ola bilər.

Maaşı hesablayan sadə proqram:

```
import java.util.Scanner;
public class maas {

public static void main(String[] args) {

double yekunmaas;
int maas;
double cem = 0;
System.out.println("n-adamlarin sayi");
Scanner m = new Scanner(System.in);
int n = m .nextInt();
System.out.println("adamlarin sayi n="+n);
double reis = 0;
System.out.println("reisin payi-");
Scanner r = new Scanner(System.in);
 reis = r .nextInt();//reise ayrilmis faiz
int s;

 //emekdaslarin sayi,iw gunleri
 //tutulan faiz
```

// DAVAMI

```
int saatsay;

int i=0 ;
while (i < n)
{ //fiqur moterize icerisinde tekrarlanan emeliyatlar verilir
    System.out.println("isci");
    Scanner a = new Scanner(System.in);
      String isci = a.nextLine();

System.out.println("gunluk maas:");
 Scanner gunluk = new Scanner(System.in);
int gunm =  gunluk.nextInt();

System.out.println("is gunlerinin sayi:");
Scanner gun  = new Scanner(System.in);
int gunsayi = gun.nextInt();//is gunlerin sayin daxil edir
System.out.println("Vergi faizi:");
Scanner ver  = new Scanner(System.in);
int vergi = ver.nextInt();//her adamin verdiyi verginin faizi
```

```
System.out.println( "iscinin Adi:" +isci + " vergi faizi = " +vergi+ " reisin faizi =  "  + reis +  " gunsayi= " +gunsayi+ " gunluk maas= " +gunm );
maas =gunsayi*gunm;//novbeti adamin maasi hesablanir
System.out.println(" ayliq maas= " + maas);
yekunmaas=maas-maas/100*vergi;
System.out.println("yekun maas= "+yekunmaas);
 double rhaqqi=yekunmaas/100*reis;
yekunmaas=yekunmaas-maas/100*reis;
System.out.println(" cari adamin "+ isci+ " yekunmaas=" + yekunmaas);
    cem=cem+=yekunmaas;
 i++;
double reiscem = reiscem+rhaqqi;
// TODO Auto-generated method stub
}
double orta=cem/n;
System.out.println("reisin adi nedir?");
 Scanner a = new Scanner(System.in);
      String reisad = a.nextLine();
     System.out.println(" Son! *** adamlarin sayi - i= " +i+ " cem = " +cem +  " orta = " +orta + " "+ reisad  +"  reisin haqqi= " +reis );
 System.out.println("birinci maashinizi tebrik edirem!" );
}
}   } }
```

Aşağıdakı ifadə verilmiş ardıcıllığı ekranda yaradacaq: “0 -1 -4 -9 -16 -25”

For(int a=3 i=1 i<=10 a=2*a-2 i++) { system.out.print(a+ “ “); }

Nəqədəki tipli dövr (while və do...while)

While operatoru onun parametric doğru qiymət aldığı sürəcdə verilmiş əməliyyatı icra edir. məsələn aşağıdakı dövr 5 dəfə icra olunaraq “ 1 2 3 4 5 “ ədədlərini çıxarır.

```
İnt i=1;
While (i<6)
 {system.out.print(i+””); i++}
```

Aşağıdakı dövr dayanmadan təkrarlanaraq ekrana” 1*2*3*4*5*6*7.....”

```
İnt i=1
While (true)
{ system.out.print(i + “ *“ );
İ++;
}
```

Göründüyü kimi şərt birincidən qabaq bütün addımlarda yoxlanılır (öncə yoxlamaq) şərtin əməliyyatın icrasından sonra yoxlanılması üçün do…while operatorundan istifadə olunur. (sonra yoxlamaq)

Asağıdakı dövr 4 dəfə icra olunaraq ekrana”2/3/4/5/” çıxarır:

```
İnt i=1
Do{
      I++;
System.out.print(i + “/”);
} while (i<5);
```

Aşağıdakı program 1-10 intervalında hər hansı tam ədəd götürüb istifadəçiyə tapmağı təklif edir. İstifadəçi klaviaturada program axtarılan ədədin istifadəçinin daxil etdiyindən böyük və ya kiçik olduğunu göstərir:

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {        // prog — число созданное программой      user — число введённое пользователем
        int prog, user;
        // Генерируем случайное целое число от 1 до 10
        prog = (int)(Math.random() * 10) + 1;
        System.out.println("Я загадала число от 1 до 10, отгадайте его.");
        System.out.print("Вводите ваше число: ");
        Scanner input = new Scanner(System.in);
        // Проверяем, есть ли в потоке ввода целое число
        if( input.hasNextInt() ) {
            do {                                                // tam ədədi daxil edirik
                user = input.nextInt();
                if(user == prog) {
                    System.out.println("Вы угадали!");
                } else {
                    // Проверяем, входит ли число в отрезок [1;10]
                    if (user > 0 && user <= 10) {
                        System.out.print("Вы не угадали! ");
                        // Если число загаданное программой меньше...
                        if( prog < user ) {
                            System.out.println("Моё число меньше.");
                        } else {
                            System.out.println("Моё число больше.");
                        }
                    } else {
                        System.out.println("Ваше число вообще не из нужного отрезка!");
                    }
                }
            } while( user != prog );
        } else {
            System.out.println("Ошибка. Вы не ввели целое число!");
        }
       System.out.println("До свиданья!");
    }

}
```

Əməliyyat	İstifadə	True-olur. Əgər:
&&	op1 && op2	op1 и op2 оба истины (конъюнкция)
||	op1 || op2	один из op1 или op2 истинен (дизъюкция)
!	!op	op — ложь (отрицание)
^	op1 ^ op2	op1 и op2 различны (исключающее или)

Əməliyyat	İstifadə	True-olur. Əgər:
>	op1 > op2	op1 больше чем op2
>=	op1 >= op2	op1 больше или равен op2
<	op1 < op2	op1 меньше op2
<=	op1 <= op2	op1 меньше или равно op2
==	op1 == op2	op1 и op2 равны
!=	op1 != op2	op1 и op2 не равны

Əməliyyat	İstifadə	Funksiyası
+	op1 + op2	Складывает op1 и op2
-	op1 - op2	Вычитает op1 из op2
*	op1 * op2	Умножает op1 на op2
/	op1 / op2	Делит op1 на op2
%	op1 % op2	Вычисляет остаток от деления op1 на op2

Hər bir ikilik əməliyyat üçün məxsusi mənimsətmə operatoru var.

```
x += 8; // то же самое, что x = x + 8 (x увеличивается на 8)
```

*=, -=, /=, %=, &= ^= və s..

```
x *= 3; // то же самое, что x = x * 3 (x увеличивается в 3 раза)
b1 ^= b2; // то же самое, что b1 = b1 ^ b2
```

Tapşırıq

2 tam ədəd elan edib onlara qiymət mənimsədin. Cəm və hasili və ədədi ortanə tapın! Ekrana çıxarın!

İnkrement və Dekrement

Операции `&&` и `||` отличаются тем, что не обязательно вычисляют значение второго операнда. Например, `&&` вычисляет значение первого операнда и, если оно ложно, сразу возвращает `false`, а `||` возвращает `true` сразу, если видит, что первый операнд – истина. В Java есть аналогичные по действию операции `&` и `|`, они вычисляют значения обоих операндов, прежде чем произвести над ними операцию.

Mənimsətmə

После того, как переменная описана, с ней можно работать в программе. В частности, ей можно присвоить значение соответствующего типа. Тогда в дальнейшем при использовании этой переменной в каком-либо выражении вместо нее будет автоматически подставляться это текущее значение.

Значение связывается с переменной с помощью оператора присваивания. В языке Java он записывается простым знаком равенства:

```
переменная = выражение;
```

Слева от оператора присваивания всегда указывается переменная. Выражение справа должно соответствовать переменной по типу. Оно может представлять собой просто литерал (например, число или символ):

```
x = 7; // переменной x присваивается значение 7
letter = 'Q'; // переменной letter присваивается значение 'Q'
```

В общем случае выражение — это то, что может быть вычислено (например, результат математической операции или результат, возвращаемый некоторым методом):

```
a = 7.5 + 2.4; // переменной a присваивается 9.9 как результат вычислений
```

В выражении наряду с литералами могут участвовать другие переменные. Вместо них подставляется их текущее значение. В результате выполнения команды:

```
b = a + 1;
```

переменная b примет значение 10.9.

Итак, оператор присваивания действует следующим образом. Сначала вычисляется значение выражения в правой части, а затем полученный результат присваивается переменной, указанной в левой части.

Возможна даже следующая ситуация:

```
x = x + 4;
```

Эта команда увеличивает текущее значение целочисленной переменной x на 4.

А следующие команды записаны неправильно и работать не будут:

```
5 = x + 7; // слева должна стоять переменная
x + 3 = 14; // слева должна стоять просто одна переменная
x = 4.5; // переменная x может принимать только целочисленные значения
```

Eclipse попытается указать на ошибку в этих строчках еще до выполнения программы, расставив предупреждающие знаки на полях редактора кода.

JAVAda riyazi funksiyalar

Math.abs(n) — **modul**. возвращает модуль числа n.

Math.round(n) — **Yuvarlaqlaşdırma**. возвращает целое число, ближайшее к вещественному числу n (округляет n).

Math.ceil(n) — **Yuvarlaqlaşdırma**. возвращает ближайшее к числу n справа число с нулевой дробной частью (например, Math.ceil(3.4) в результате вернёт 4.0).

Math.cos(n), Math.sin(n), Math.tan(n) — **Triqonometrik**. тригонометрические функции sin, cos и tg от аргумента n, указанного в радианах.

Math.acos(n), Math.asin(n), Math.atan(n) — **Əks Triqonometrik**обратные тригонометрические функции, возвращают угол в радианах.

Math.toDegrees(n) — **Bucaq**. возвращает градусную меру угла в n радианов.

Math.toRadians(n) — **Radian**. возвращает радианную меру угла в n градусов.

Math.sqrt(n) — **Kvadrat Kök.** возвращает квадратный корень из n.

Math.pow(n, b) —**Kvadrat Kök.** возвращает значение степенной функции n в степени b, основание и показатель степени могут быть вещественными.

Math.log(n) — **Loqaritm**. возвращает значение натурального логарифма числа n.

Math.log10(n) — **Onluq Loqarifm.** возвращает значение десятичного логарифма числа n.

Math.random() — **Təsadüfi ədədlər.** возвращает псевдослучайное вещественное число из промежутка [0;1).

Scratch.mit.edu

Sign in Login : tahir5282 Parol: 3088533t

My stuff Untitled-3

Условные операторы: if | Download | Downloads | Untitled-3 on Scratch

https://scratch.mit.edu/projects/85328116/#editor

SCRATCH File Edit Tips About Tahir5282

Untitled-3
by Tahir5282 (unshared)

x: 240 y: -85

Sprites New sprite:

Stage
4 backdrops

New backdrop:

Giga2

Scripts Costumes Sounds

Share See project page

Motion Looks Sound Pen Data Events Control Sensing Operators More Blocks

move 10 steps
turn 15 degrees
turn 15 degrees
point in direction 90
point towards mouse-pointer
go to x: -69 y: -11
go to mouse-pointer
glide 1 secs to x: -69 y: -11
change x by 10
set x to 0
change y by 10
set y to 0
if on edge, bounce
set rotation style left-right
x position
y position
direction

```
when clicked
hide
switch backdrop to welcome
wait 1 secs
show
switch costume to catherine-c
wait 1 secs
say Salam
wait 1 secs
say Hello!
go to x: -13 y: 9
wait 1 secs
switch costume to Giga walk4
wait 1 secs
glide 1 secs to x: -202 y: -30
wait 1 secs
ask What's your name? and wait
say Good!!!
wait 1 secs
switch backdrop to motherboard-0
wait 1 secs
switch costume to giga-b
wait 1 secs
ask What is it? and wait
repeat until answer = motherboard
    switch costume to giga-d
    say mistake
    wait 1 secs
    switch costume to giga-a2
    ask Give the right answer and wait
switch costume to giga-c
say you are quite right
```

x: -202
y: -30

Backpack

ENG 5:17 PM 11/4/2016

Scratch proqramının mətni