CPE101 Programming Languages I

Week 1
Introduction to Programming

Assoc. Prof. Dr. Caner ÖZCAN

Introduction

- Course Web Site: www.canerozcan.net

- Office Hours: **Monday 15:00-17:00**
  Tuesday 09:00-12:00
  or appointment by email:
  canerozcan@karabuk.edu.tr

- Textbooks:

  Doç. Dr. Fahri Vatansever, "Algoritma Geliştirme ve Programlamaya Giriş", Seçkin Yayıncılık, 12. Baskı, 2015

  Harvey Deitel, "C How to Program", Pearson Education

Introduction

- Work hard and practice!

- Grading
  - Midterm Exam: %40
  - Final Exam: %60

- Study to learn, not for grade. Grade is already won.

Term Topics

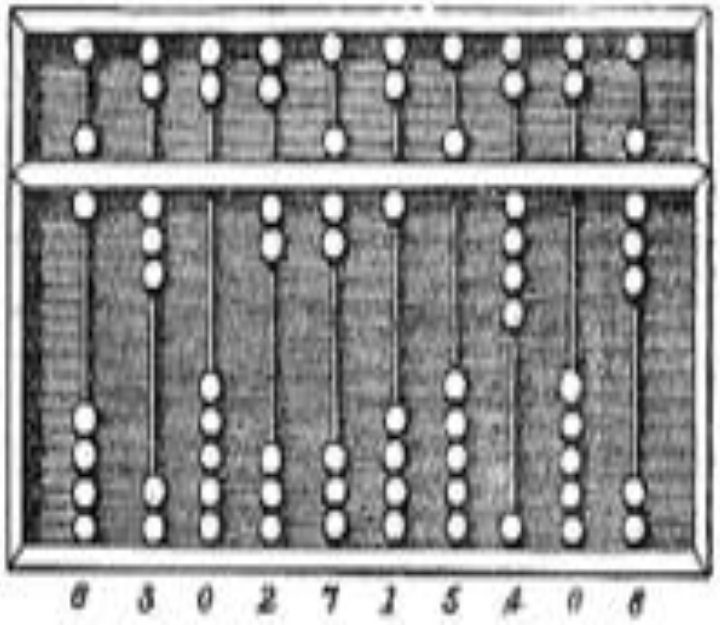
- Lecture 1- Basic Concepts and Algorithm
- Lecture 2- Variable and Operators
- Lecture 3- Control Structures
- Lecture 4- Repetition Structures
- Lecture 5- Algorithm Examples and Analysis-1
- Lecture 6- Algorithm Examples and Analysis-2
- Lecture 7- Introduction to C Programming and C Compiler
- Lecture 8- Basic Data Types of C and Input/Output Operations (printf, scanf)
- Lecture 9- Control and Repetition Structures of C
- Lecture 10- Arrays
- Lecture 11- Character Arrays
- Lecture 12- Functions-1
- Lecture 13- Functions-2

What is Computer?

- A computer is a device capable of performing computations and making logical decisions.
- Today's supercomputers can perform trillions of additions per second.
- A computer is comprised of various devices (such as the keyboard, screen, mouse, disks, memory, DVD, CD-ROM & processing units) that are referred to as **hardware**.
- Computer programs that run on a computer are referred to as **software**.

What is Computer?

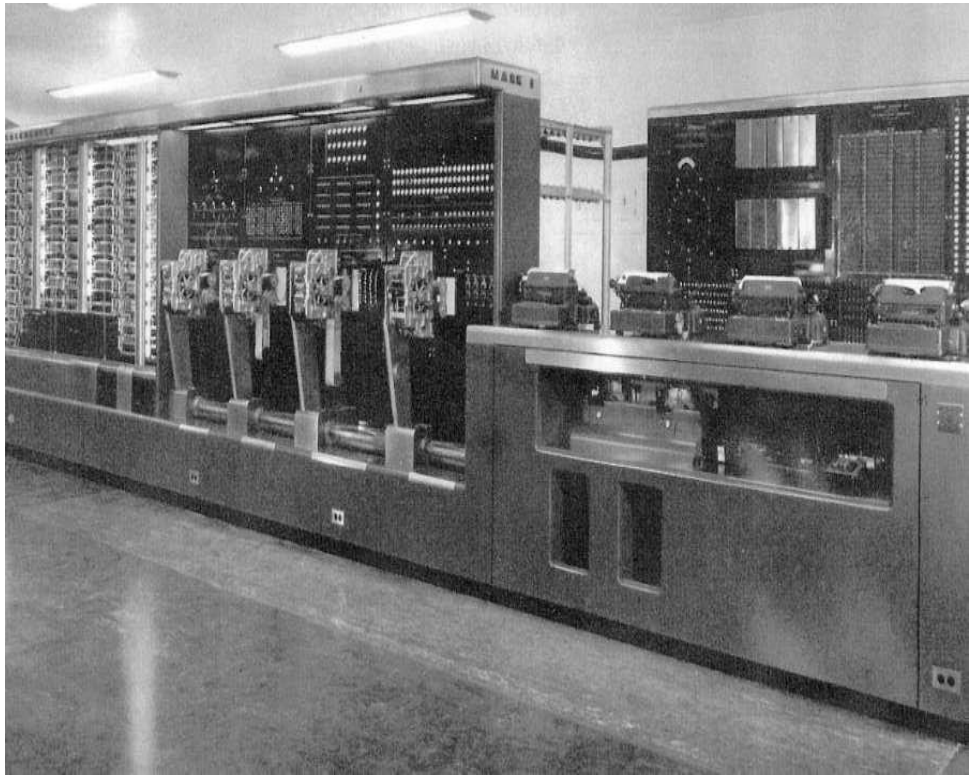
- One of the earlier computing devices was the abacus.

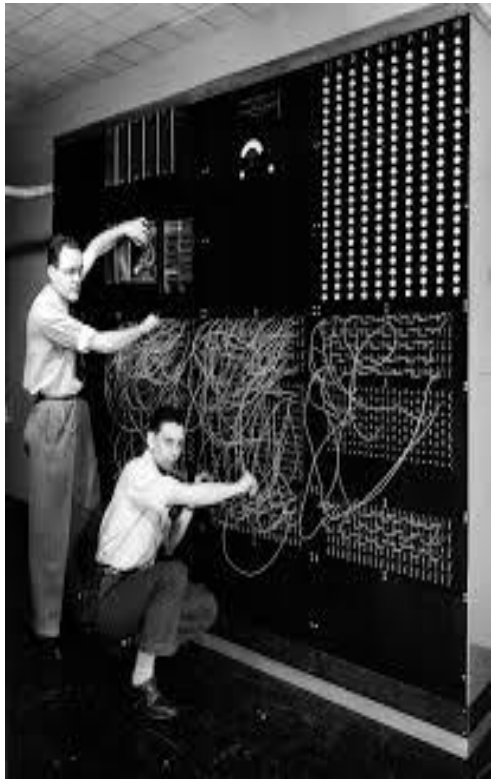


What is Computer?

- Advancements on computer technology chronologically:
  - B.C. 500 Abacus
  - 1642 Pascal's Mechanical Addition Machine
  - 1827 Babbage's subtraction machine
  - 1941 Binary Mechanical Calculator (Zuse)
  - 1944 Decimal Electromechanic Calculator (Aiken)
  - 1945-54 First Generation Vacuum Tubes and Beams
  - 1955-64 Second Generation Transistors and Magnetic Memories
  - 1965-71 Third Generation Integrated Circuits
  - 1971-90 Fourth Generation VLSI Circuits
  - 1982 IBM Pc & MS-DOS
  - 1984 MAC
  - 1990’s Parallel Processors

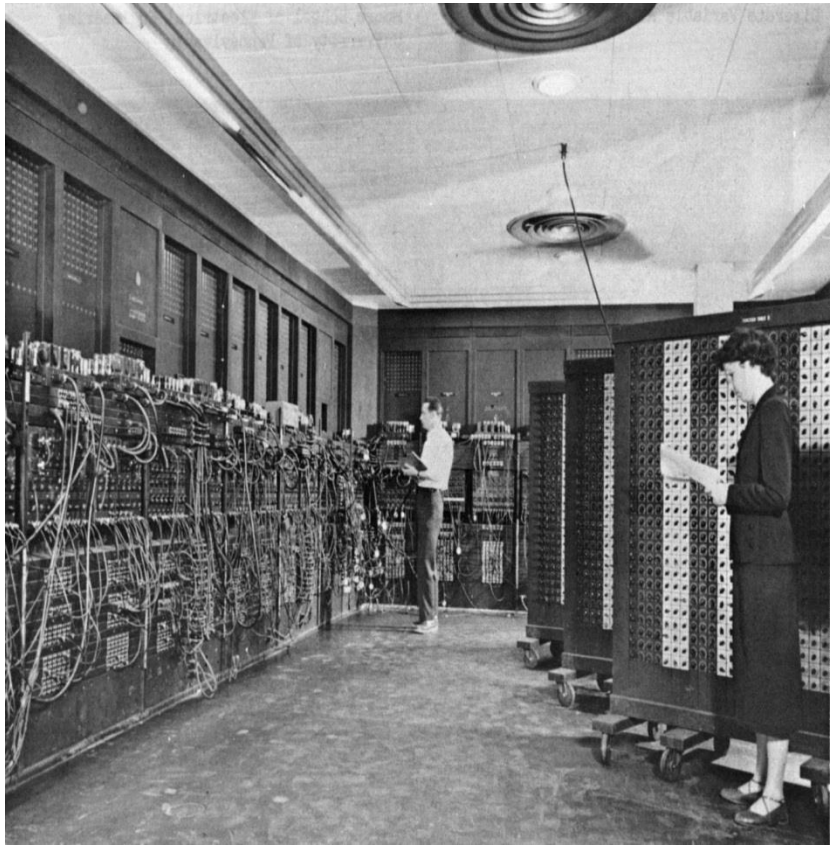
What is Computer?





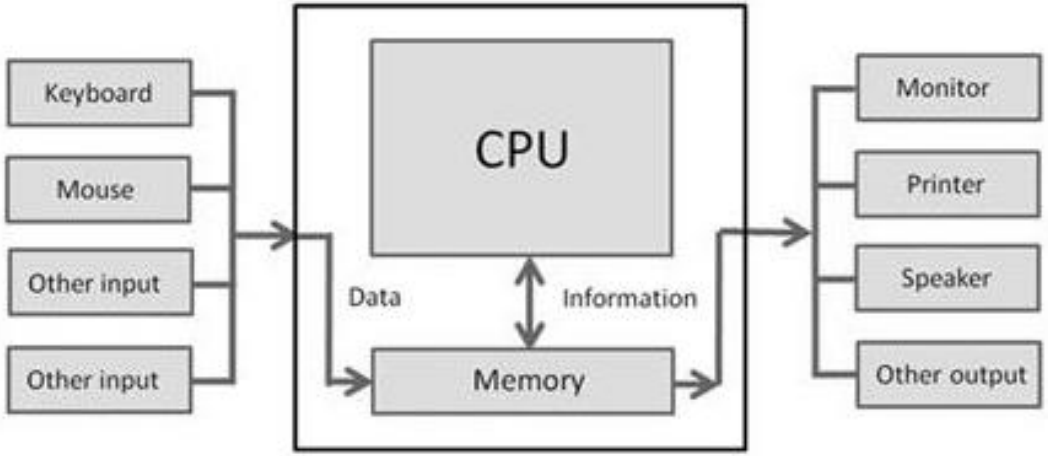
IBM MARK-I

What is Computer?



ENIAC- Electronic Numerical Integrator And Computer

Computer Organization



accepts data	Input
processes data	Processing
produces output	Output
stores results	Storage

Data Storage Units

- All data in computer systems are stored in electrical signals represented by binary **1 or 0**.
- Each digit in binary system is called **bit**.
- Bit is not enough to express a quantity. We use byte as basic memory unit.
  - 1 byte = 8 bits
- Each character in computer system is represented by 8 bits.
- Example: Character **A** is represented as 0100001 in binary. Each digit of this number is called bit.

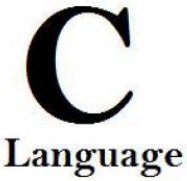
Data Storage Units

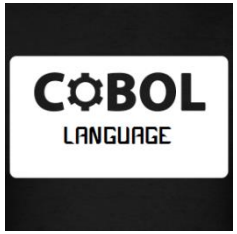
- Data storage units are as follows from lowest to highest: Bit, Byte ,KB(Kilo Byte), MB (Mega Byte), GB (Giga Byte), TB (Tera Byte).
- 1 Byte=8 Bits
- 1 KB (Kilo Byte)=1024 Byte
- 1 MB (Mega Byte)=1024 KB
- 1 GB (Giga Byte)=1024 MB
- 1 TB (Tera Byte)=1024 GB

Programming Language

- Unlike human being, computers do not have a functional brain. That is why we should give commands to do something with computers.
- **Programming Language**: Formal computer language designed to communicate instructions to a machine, particularly a computer.

Programming Language





[1]http://rigaux.org/language-study/diagram.html

What is program?

- There are variety of program types according to purpose of their usage :
  - **System programs**: Every program runs on an operating system. Operating system provides resources and environment for other softwares.
  - **Drivers**: Programs that allows communication between operating system and hardware.
  - **Applications**: Programs that run on an operating system for meeting users requirements.

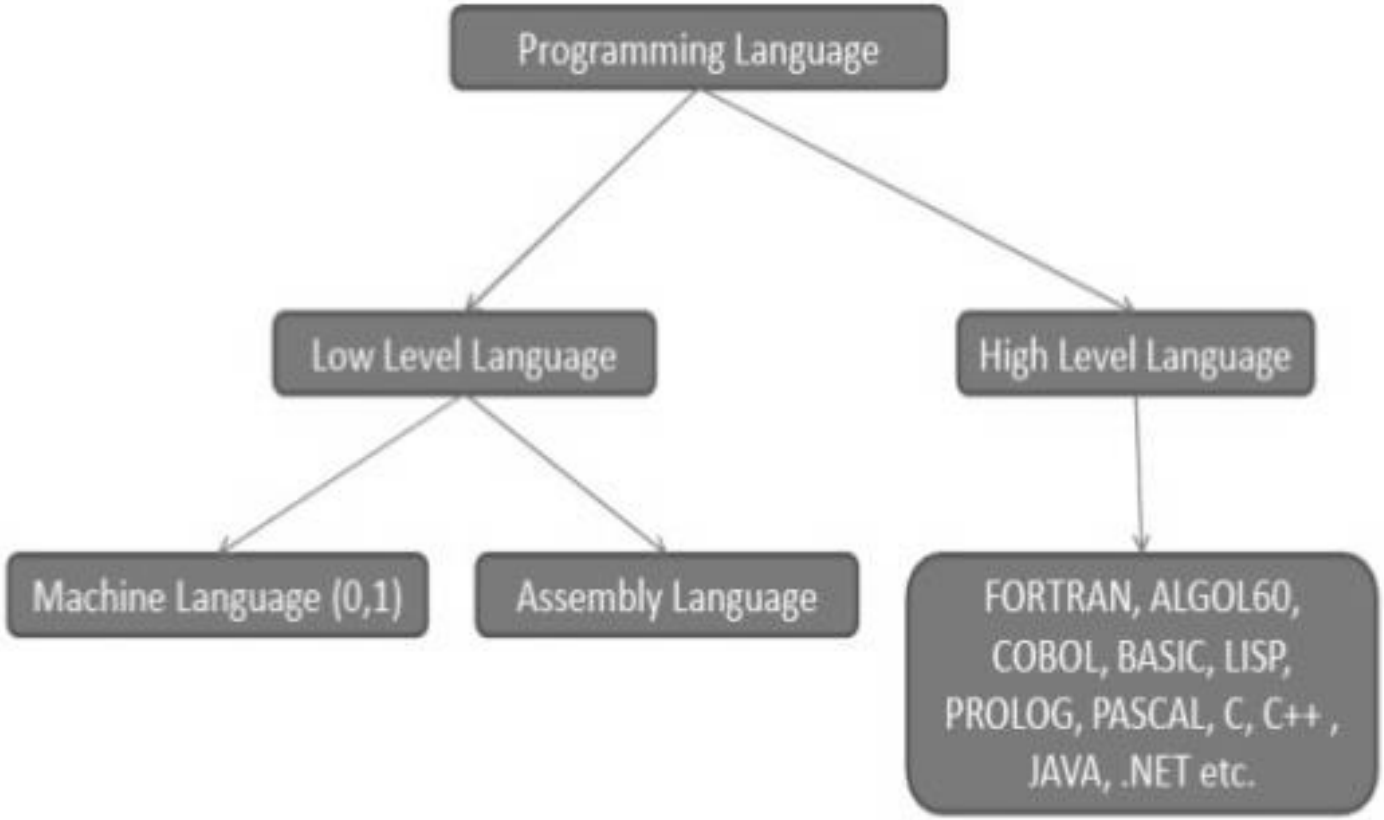
Who is programmer?

- Programmer, is a person who writes computer software.
- Programmer should know about platforms that is used for developing programs.
- Programmers:
  - **Builder:** Determines the necessary technologies for writing the program.
  - **Developer**: Person that writes programs.
  - **Analyst**: Person who tries to find out error and report them to coders during development process.

Classification of Programming Languages

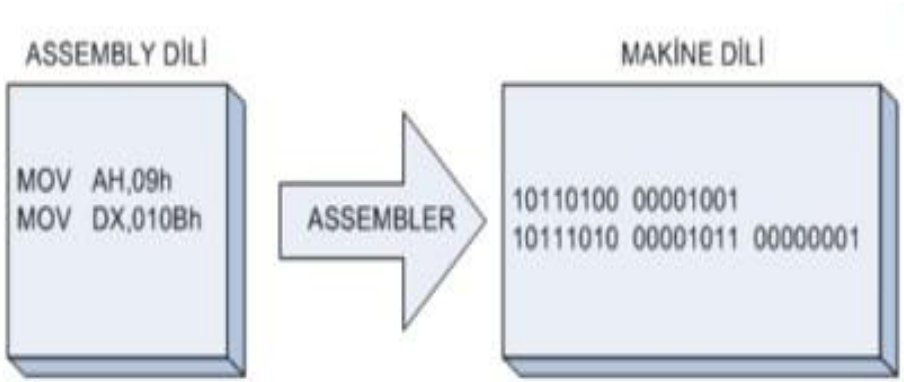
- Programming languages can be grouped according to their levels which is a measure of proximity for human perception.
- High level languages are close to human perception whereas low level languages are close to computer hardware.
- As level of programming language increase, things get easier for a programmer.
- The higher programming language makes the jobs easier but reduces the efficiency and flexibility in general.

Classification of Programming Languages



Classification of Programming Languages

- **Machine Language**
  - Natural language of a computer
  - Consist of strings of numbers (mostly reduced to 1 & 0)
  - Machine dependent
- **Low Level Language (Assembly Language)**
  - English-like abbreviations to represent elementary operations.
  - **Assemblers** were developed to convert assembly language programs to machine language.

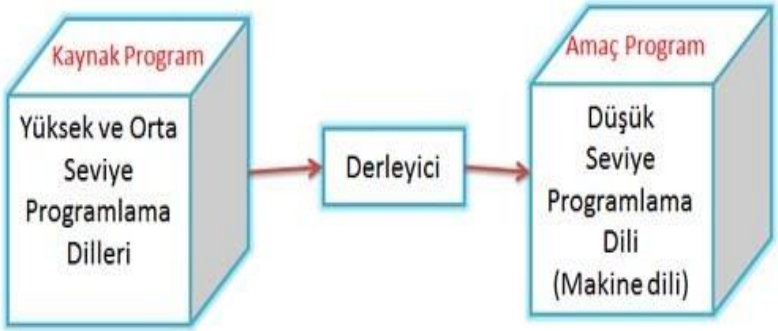


Classification of Programming Languages

- Machine language programming was simply too slow and tedious for most programmers.
- Instead using the strings of numbers that computers directly understand programmers began using **English-like abbreviations** to represent elementary operations.
- These abbreviations formed the basis of **assembly language.**
- **Assemblers** were developed to convert assembly language programs to machine language.

Classification of Programming Languages

- **Medium Level Languages**
  - Close to both human and computer
  - Easy like high level languages, flexible like low level languages.
  - Generally used for system programming.
  - C is a medium level language.
- **High Level Languages**
  - To speed the programming task
  - Compilers convert high level language programs into machine language.
  - Code similar to everyday English
  - Interpreter programs developed to execute high level languages without need of compiling



Classification of Programming Languages

- **Very High Level** languages allows to write programs almost visually using some tools.
- Languages such as C# and Visual Basic are used very often on Windows systems.
- Very high level languages are also called declarative languages.
- Languages that is used to manage database systems can be put in this group.

Compilers and Interpreters

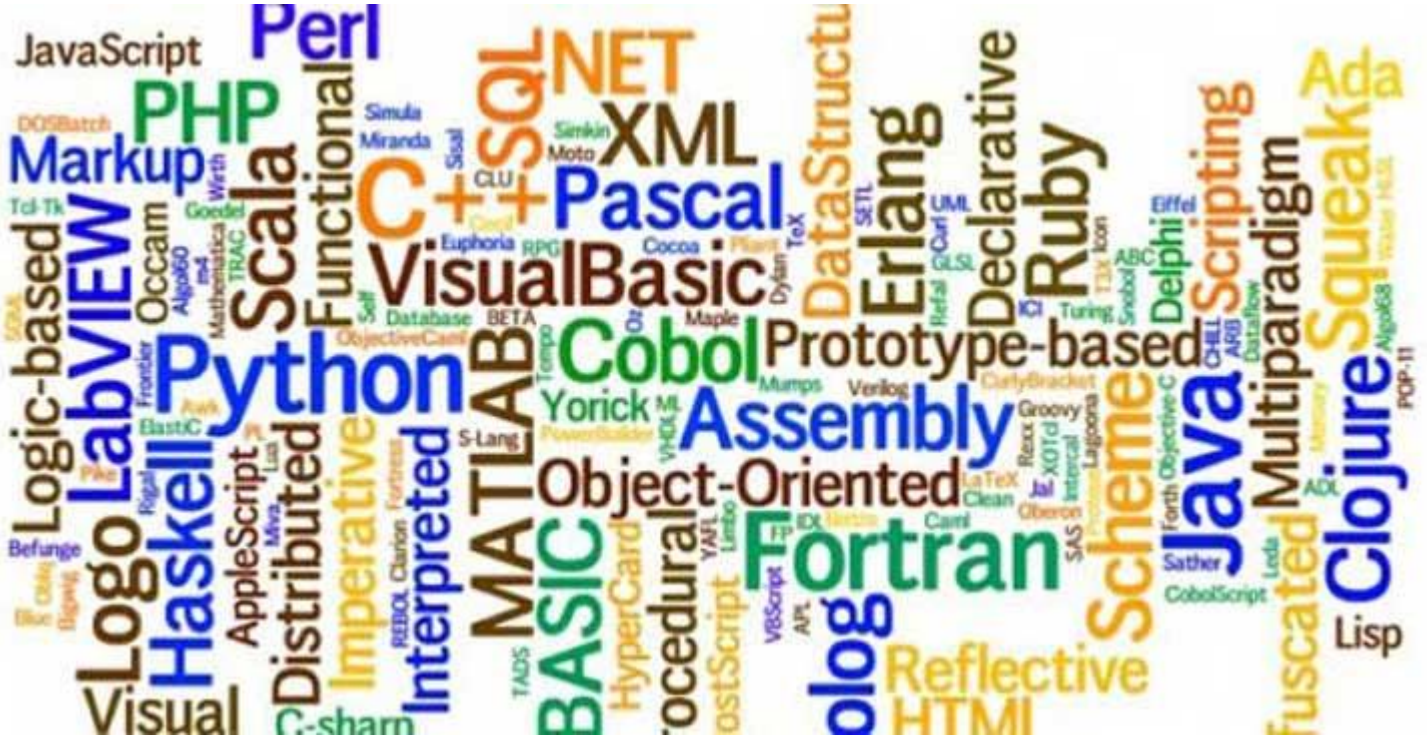
- Translator programs that convert high level language programs into machine language are called **compilers**.
- High level languages allow programmer to write instructions that look almost like everyday English and contain commonly used mathematical notations.
- **Interpreter programs** were developed to execute high-level language programs without the need for compiling them into machine language.
- Although compiled programs run faster than interpreted programs, interpreters are popular in program development environments.

Compilers and Interpreters

- Compilers firstly controls the written code.
- If code has errors, compiler stops process.
- When all errors are removed from code, compiler converts the code into machine code (exe file)
- Interpreters execute code row by row instead of compiling it totally.

Computers - Review

Why programming is important? Review



Problem and Problem Solving

- The idea of achieving an automation or solving a scientific computation using computers is called a problem.
- It is essential to have a thorough understanding of the problem and a carefully planned approach to solving the problem.
- A problem may have more than one solution. In this case we should find an optimal solution with computers.

Algorithm

- **Algorithm**: A procedure for solving problem in terms of
  - The action to be executed
  - The order in which these actions are to be executed..
  - All steps should be decisive. Nothing should be related with chance.
  - Algorithm should end in a finite step.
  - Algorithms should handle all possible cases.
- Algorithms are expressed with **pseudo codes** or **flow charts**.

Algorithm

Pseudo Code

- **Pseudo code** is an artificial and informal language that helps developing algorithms.
- Pseudo code is similar to everyday English;
  - Convenient
  - User friendly
  - Not an actual programming language.
- Not executed on computers.
- They help you think out a program before attempting to write it.
- Only consist of action statements.

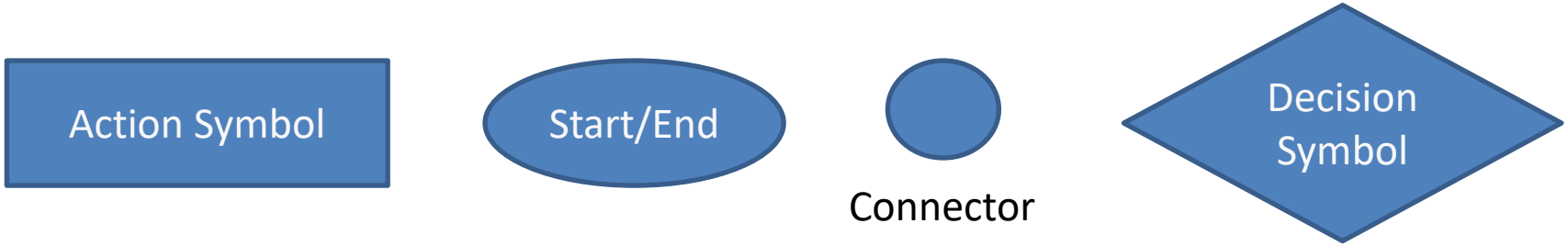
Pseudo Code

- Basic pseudo code commands

- **Start :** It indicates that the program has started.
- **End :** It indicates that the program has finished.
- **Read :** It is written for user input.
- **Print :** It is written to show users the information or results.
- **If … Else … :** It used in the change of flow according to the conditions.

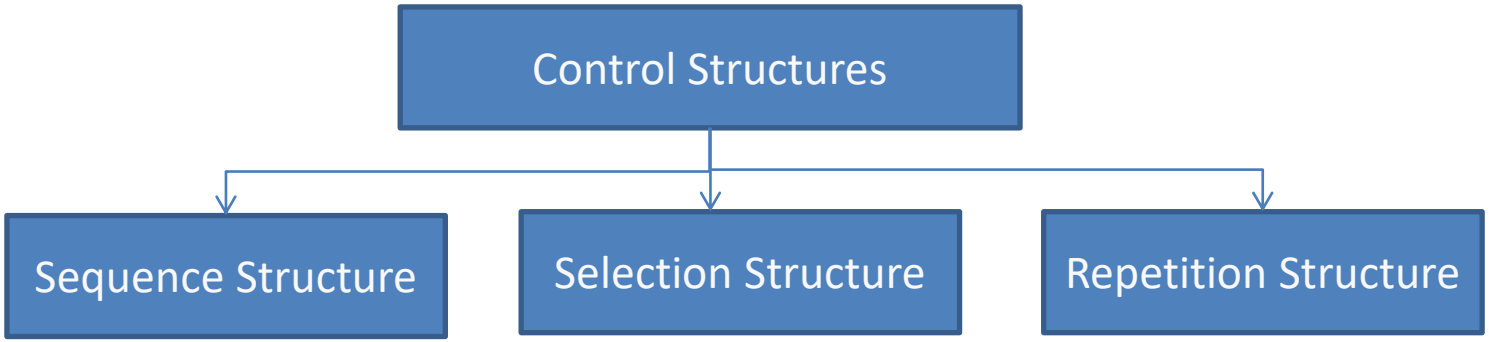
Flow charts

- **Flowchart** is a graphical representation of an algorithm or of a portion of an algorithm.
- Flowcharts are drawn using certain special purpose symbols such as Rectangles, Diamonds, Ovals, Small circles



Control Structures

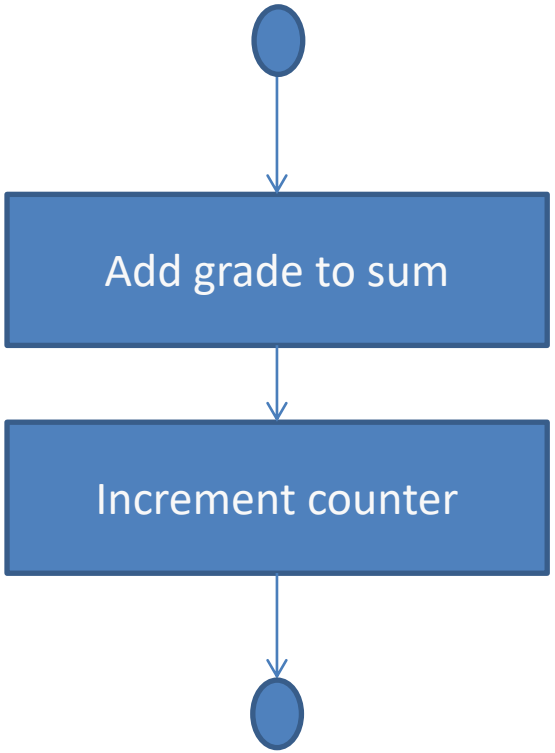
- Researches have shown that programs could be written without any go to statement in terms of **only three control structures**, namely **sequence structure**, the **selection structure** and **repetition structure**.



Control Structures

Sequence Structure

- Statements in a program executed one after the other in the order in which they are written
- Unless directed otherwise the computer automatically executes C statements one after the other in the order in which they are written.
- We use **rectangle symbol** also called **action symbol**, to indicate any type of action including a calculation or input/output operation.



toplam = toplam + not

sayac = sayac + 1

Control Structures (Sequence Structure)

- Example; if an aircraft wants to throw a missile to a target, steps should be as follows.
- If steps are not followed in order, process will fail.
  - 1. Get the coordinates of the target
  - 2. Get the current coordinate of aircraft
  - 3. Perform calculation to hit the target.
  - 4. Fire missile.

Control Structures

1. Refinement of the first step:

- a. Get coordinates from officer
- b. Be sure that target coordinates are valid.
- c. Store coordinates in memory.

- a. Refinement of sub step "a"
- i. Be sure of coordinates
- ii. Control whether target is the range of missile.
- iii. Control whether target is friend area or not.

Algorithm

- Basically, algorithms have three main parts :
- **Input**: Data that computer will work on is put by a user.
- **Process**: Computer processes data.
- **Output**: Computer shows results in a way that people can understand.

Algorithm

- **Input**:
- a. Get product name.
- b. Get amount of product.
- c. Read price of product from a file
- **Process**:
- a. Calculate the total price of the product
- b. Give discount

- **Output**:
- a. Print final price

Final algorithm:

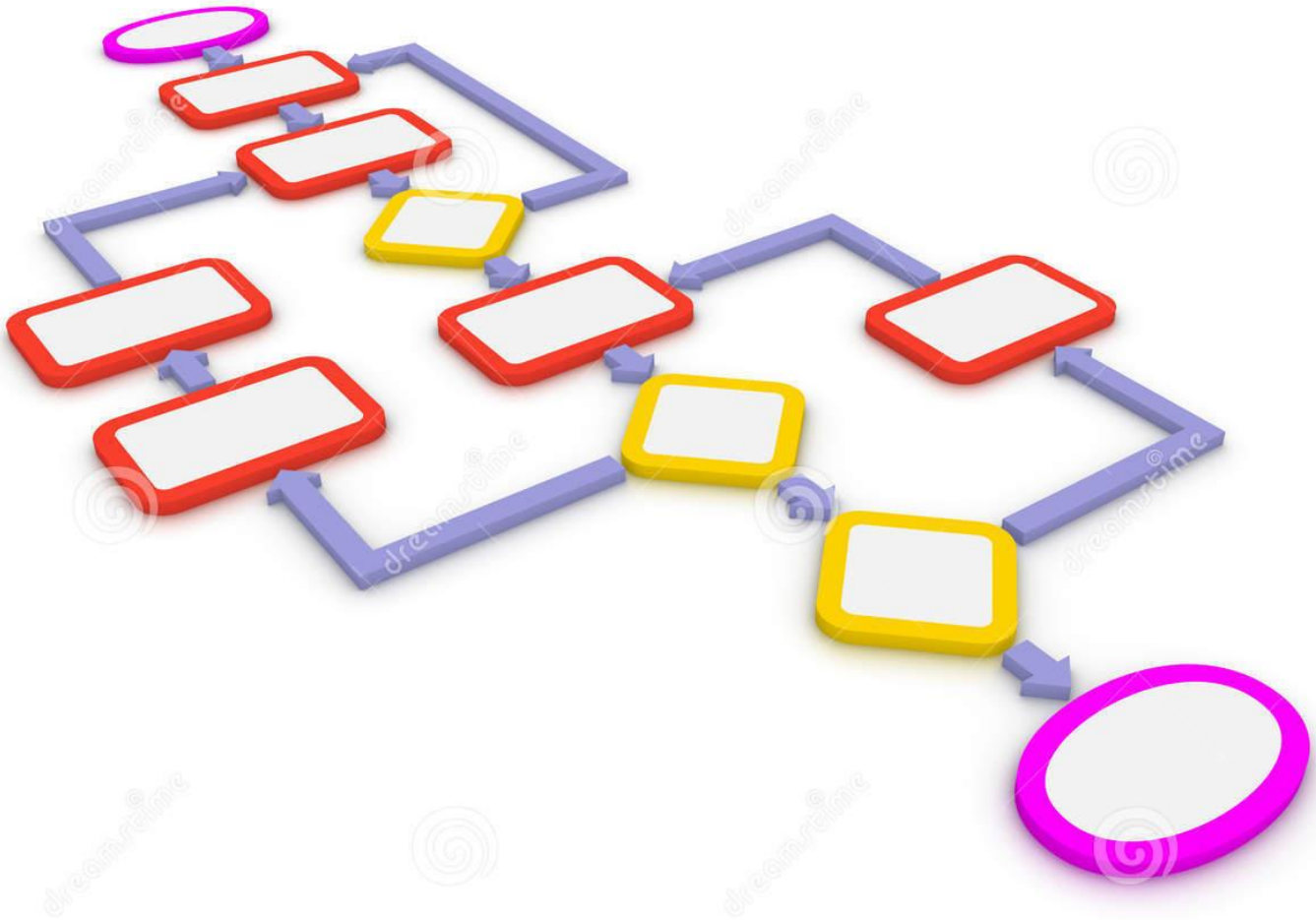
1. Start.
2. Get product name.
3. Get amount of product.
4. Read price of product from a file
5. Calculate the total price of the product
6. Give discount
7. Print final price
8. End

Flow chart

Pseudo Code

1. Get up from the bed. Switch off the alarm. Get out of your room.
2. Take the toothbrush. Take the toothpaste. ***Start brushing your teeth.***
3. Go to the bathroom. Take off your clothes. Choose hot/cold shower. ***Shower***.
4. Go to the dressing room. Get your dress for that day. ***Dress-up.***
5. Go to the parking lot. Take your car/bike out. ***Get into it***.
6. Start the car/bike. Check out your way. ***Drive to the work/College.***

Algorithm



References

- Doç. Dr. Fahri Vatansever, “Algoritma Geliştirme ve Programlamaya Giriş”, Seçkin Yayıncılık, 12. Baskı, 2015
- J. G. Brookshear, “Computer Science: An Overview 10th Ed.”, Addison Wisley, 2009
- Kaan Aslan, “A’dan Z’ye C Klavuzu 8. Basım”, Pusula Yayıncılık, 2002
- Paul J. Deitel, “C How to Program”, Harvey Deitel.