Algorithm

- A logical description of a computer program or process that
- Includes step-by-step instructions
- Designed to solve a particular problem or
- Perform a spesific task

→An algorithm is a sequence of operations that follows a finite number of specific steps from the starting point to the goal.

→It presents a specific method to solve a problem.

→This method takes some inputs, and then performs relevant operations on these inputs and finally produces outputs.

Basic Features and Components of an Algorithm

- Input
- Output
- Clarity: The algorithm should be clear and unambiguous, understandable by everyone and should not contain fuzzy expressions.
- Finiteness: Every algorithm has a finite number of steps, with a definite start and end point.
- Efficiency: Each command should contain operations that are simple and clear enough for the user to express with pen and paper.

Algorithms

- Linear algorithms
  - Algorithms for basic operations such as summation and multiplication
  - Example: Algorithm to calculate the area of a rectangle given two sides
- Logical algorithms
  - Algorithms that return a value as a result of a logical test
  - Example: Algorithm that finds the maximum value in a sequence of numbers
- Recursive algortihms
  - Algorithms in which an operation is repeated more than once
  - Logical operations are used
  - Example: Algorithm to calculate the factorial of a number, algorithm to calculate Greatest Common Divisor (GCD) and Least Common Multiple (LCM) values of two values

Program

- A sequence of commands written in a computer language that
- Uses inputs to solve a given problem and
- Obtains the appropriate outputs

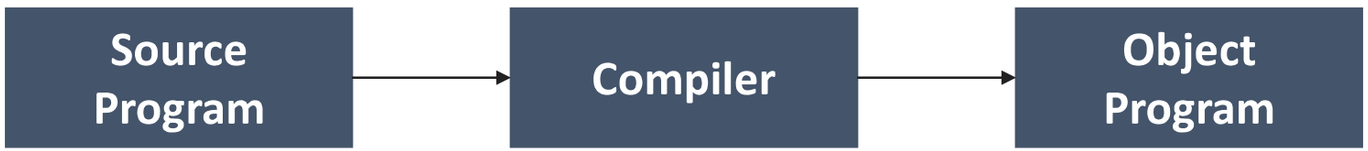
→Transfer of the algorithm to the computer in the appropriate language

Programming Language

- A special structure used to create, run and control computer programs
- A notation for communicating with machine language
- Expressing commands that the machine can understand in a language close to everyday speech
- There are different levels of machine-understandable languages, from low-level languages to high-level languages

Programming Language

- Low-level languages: Contains expressions that are closer to the computer's processor and usually closer to machine language. i.e.: Assembly
- High-level languages: Allows programming at a level that users can understand and write more easily. i.e.: Python, Java, C++.
- Compiler: Translates a program written in a high-level language (source program) into machine language (object program).



Programming Languages

- **C:** It is a general purpose programming language. Due to its modular structure and high performance, it is used in many areas such as system programming, game development and embedded systems.
- **C++:** A language that extends the features of C and supports object-oriented programming. C++ is often used in large software projects and game development.
- **Java:** It is a platform-independent programming language. It is known for its "Write once, run anywhere" principle. It is mainly used for developing large-scale web applications and mobile applications.
- **JavaScript:** A programming language that runs in web browsers. It is especially used to add dynamic content to web pages. It can also be used on the server side.
- **PHP:** PHP is a language specifically designed for web development. It is often used for server-side web development.
- **Python:** It is a high-level programming language that is easy to read and easy to learn. It is general-purpose and used in many fields, especially popular in areas such as artificial intelligence, data science, web development and automation.

Computer Programs

- Mathematical (arithmetic) operations
- Comparison (decision) operations
- Logical operations

Mathematical (Arithmetic) Operations

- Bost commonly used basic operations
- Order of operations is important
  - PEMDAS: parentheses, exponents, multiplication, division, addition, and subtraction

Operation	Math	Computer
Addition	a + b	a + b
Subtraction	a - b	a - b
Multiplication	a x b	a * b
Division	a ÷ b	a / b
Exponent	a^b	a ^ b

Comparison (Decision) Operations

- Used for decisions to be made as a result of comparing two situations
- At the end of the comparison decision, the program proceeds in different ways

Operation	Description
=	is equal to
<> or !=	is not equal to
<	is less than
>	is greater than
<=	is less than or equal to
>=	is greater than or equal to

Logical Operations

- Used in basic logic operations
- Used when comparison operations are required to satisfy more than one condition
- Priority of operations: Parentheses, not, and, or

Operation	Mathematical Symbol	Description
AND	&	The result is true if all conditions are true
OR	|	The result is true if only one of the conditions is true
NOT	~	The result is the opposite of the condition.

Developing Algorithms

- Setting the instructions step by step
  - Natural language NOT programming language
  - Conceptual algorithm design
- Drawing the flowchart
  - The steps of the algorithm are drawn using some special geometric shapes
- Writing the pseudo code
  - The steps of the algorithm are written in clear text or abbreviations similar to computer commands
  - Computer terms and daily language are used together

Setting the instructions step by step

- Algorithm that calculates the sum of two numbers entered by the user

(Linear Algorithm)

  - STEP 1: Start
  - STEP 2: Enter two numbers: x, y
  - STEP 3: Calculate the sum of the numbers: sum = x+y
  - STEP 4: Print/return the sum of numbers: sum
  - STEP 5: End

Setting the instructions step by step

- Algorithm that finds the maximum out of three given numbers

(Logical Algorithm)

- STEP 1: Start
- STEP 2: Enter three numbers: a,b,c
- STEP 3: Set a as the maximum value: max = a
- STEP 4: If b is greater than the maximum value (b > max), b will be the maximum: max = b
- STEP 5: If c is greater than the maximum value (c > max), c will be the maximum: max = c
- STEP 6: Return the maximum value: max
- STEP 7: End

Setting the instructions step by step

- Algorithm to calculate the factorial of an input number

(Recursive Algorithm)

- STEP 1: Start
- STEP 2: Get the number to calculate the factorial: n
- STEP 3: Initialize the sytem;

  Set the factorial value equal to 1: f = 1

  Set the counter value to 1: i = 1
- STEP 4: Increase the counter value and multiply the factorial value by the counter and equate this calculated value to the factorial: i = i + 1, f = f * i
- STEP 5: If the counter value is less than n (i < n) go to Step 4, otherwise go to Step 6.
- STEP 6: End

MUDANYA
UNIVERSITY

Practical Examples

Logical Operations

Find the result (X OR Y) AND NOT X

X	Y
1	1
1	0
0	1
0	0

X	Y	X OR Y	NOT X	(X OR Y) AND NOT X
1	1	1	0	0
1	0	1	0	0
0	1	1	1	1
0	0	0	1	0

Logical Operations

You are asked for the names of workers at a workplace who are over 25 years of age and earn the minimum wage. (Minimum wage is 11.500)

IF Age > 25 AND Salary = Minimum wage THEN write the name

Name	Age (X)	Salary (Y)	X	Y	X AND Y	Result
Adams	30	15000	1	0	0	-
Barney	23	11500	0	1	0	-
Clara	24	9600	0	0	0	-
Danielle	29	11500	1	1	1	Danielle

Linear Algorithm

- Algorithm that calculates the sum of the squares of two entered numbers by the user
  - STEP 1: Start
  - STEP 2: Enter two numbers: num1, num2
  - STEP 3: Calculate the square of the first number: sqr1 = num1 * num1
  - STEP 4: Calculate the square of the second number: sqr2 = num2 * num2
  - STEP 5: Calculate the sum of squares: sum = sqr1 + sqr2
  - STEP 6: Return the sum : sum
  - STEP 7: End

Logical Algorithm

- Algorithm that determines whether the entered number is odd or even
  - STEP 1: Start
  - STEP 2: Enter the number: num
  - STEP 3: If (num mod 2 = 0) then return even, otherwise return odd
  - STEP 4: End

Logical Algorithm

- Algorithm that indicates the state of the water according to a given temperature value
  - STEP 1: Start
  - STEP 2: Enter the temperature value: temp
  - STEP 3: If (temp <= 0) then water is in the form of ice: status: ice
  - STEP 4: If (temp > 0 and temp < 100) then water is in liquid form: status: water
  - STEP 5: If (temp >= 100) then water is in the gaseous state: status: steam
  - STEP 6: Return state of the water: status
  - STEP 7: End

Logical Algorithm

- Algorithm that sorts three numbers entered by the user from smallest to largest
  - STEP 1: Start
  - STEP 2: Enter three numbers: a, b, c
  - STEP 3: Set a dummy number equal to zero: dummy = 0
  - STEP 4: If (a > b) then dummy = a, a = b, b = dummy
  - STEP 5: If (a > c) then dummy = a, a = c, c = dummy
  - STEP 6: If (b > c) then dummy = b, b = c, c = dummy
  - STEP 7: Return a, b, c
  - STEP 8: End

Recursive Algorithm

- Algorithm that calculates the sum of numbers from 1 to the entered number
  - STEP 1: Start
  - STEP 2: Enter a number: x
  - STEP 3: Define the sum and set it equal to zero: sum = 0
  - STEP 4: Define a counter and set it equal to one: counter = 1
  - STEP 5: Calculate the sum: sum = sum + counter
  - STEP 6: Increase counter by 1. If (counter <= x) then go to Step 5, otherwise go to Step 7
  - STEP 7: Return the result: sum
  - STEP 8: End

Recursive Algorithm

- Algorithm that calculates the average of the numbers entered so far until the user enters a negative value
  - STEP 1: Start
  - STEP 2: Initialize the system: counter = 0, total = 0, average = 0
  - STEP 3: Enter a number: x
  - STEP 4: If (x > 0) then go to Step 5, otherwise go to Step 7
  - STEP 5: Increase counter by 1 and calculate total value and the average value:
    counter = counter + 1, total = total + x, average = total / counter
  - STEP 6: Go to Step 3
  - STEP 7: Return the result: average
  - STEP 8: End

Recursive Algorithm

- Algorithm that calculates the GCD (Greatest Common Divisor) of two given numbers
  - STEP 1: Start
  - STEP 2: Enter two numbers: a, b
  - STEP 3: If (a = b) then (gcm = a) and go to Step 9
  - STEP 4: Define a counter and set it equal to 1: counter = 1
  - STEP 5: If (a < b) then the minimum value is equal to a: min = a
  - STEP 6: If (b < a) then the minimum value is equal to b: min = b
  - STEP 7: If (a MOD counter = 0) AND (b MOD counter = 0) then gcm = counter
  - STEP 8: If (counter < min) then counter = counter + 1 and go to Step 7, otherwise go to Step 9
  - STEP 9: Return the result: gcm
  - STEP 10: End

HOMEWORK

1. Write the algorithm that calculates the LCM (Least Common Multiple) of two given numbers
2. Write the algorithm that determines whether a given number is a prime number or not
3. Write the algorithm that sorts four given numbers from smallest to largest
4. Create some random variables and apply the algorithm by hand for Assignment #1 , #2 , #3