PROGRAMMING LOGIC AND TECHNIQUES

Algorithms and analysis

What is a program?

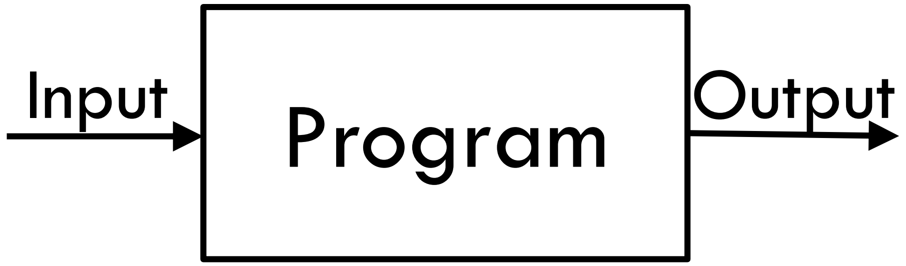
- A program is a set of directives, called instructions or commands, that specify which operations are to be performed and determine how those operations are linked. The purpose of a program is to serve as an interface between the device (a computer) and the user (a human being).

Input vs. output

Input: Information necessary for the execution of the program (sometimes called data).

ex. Get data from the keyboard, a file, a sensor, etc.

Input can be manual (keyboard) or automatic (file, database, or other programs).

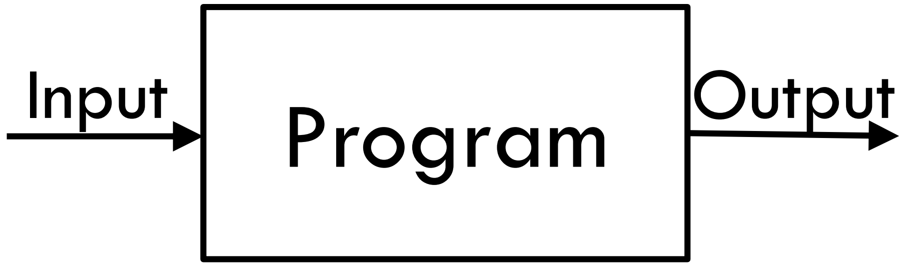


Input vs. output

Output: Information that the program produces (sometimes called the result).

ex. Display data on the screen, send data to a file, etc.

Output can be displayed on the screen, printed, stored in a file, stored in a database, or returned to another program.



Human-machine communication

As was mentioned before, a program aims to serve as an **interface** between a human and a machine.

Such an interface is necessary because computers have a very specific language that is composed of symbols, numbers and commands. This is called **machine language.**

Humans also have their own dialect, but it is very different from machine language.

Therefore, programs are used to simplify and facilitate **communication** between these two entities.

Knowing a language vs. knowing how to program

There is a big difference between knowing a programming language and knowing how to use the commands of a language in order to create a program that does what we want it to do.

Creating a program can be a complicated task for both a novice programmer and an experienced programmer. This is why certain development tools and methods have been invented. In fact, we can break down software development into six main steps.

Software development life cycle (SDLC)

The main phases of the software development life cycle are:

1. Requirements
2. Analysis/design
3. Implementation
4. Testing
5. Deployment
6. Maintenance

In this course, the requirements will be given to you. We will focus on step 2, and, time permitting, steps 3 and 4.

Knowing a language vs. knowing how to program

The **analysis step** is to determine which is the best method by which to achieve the desired outputs. This design and formulation process of the solution is called the **algorithm.**

An **algorithm** is a sequence of steps that specifies how to solve a problem.

Programming is the process of breaking down a large, complex task into smaller and smaller subtasks. The process continues until the subtasks are simple enough to be performed with the basic instructions provided by the computer.

Knowing a language vs. knowing how to program

The **next step** is successfully expressing, in a given language, the result of the first step. It is important to note that if the algorithm is correctly executed, this step is only a transcription (translation into computer-speak).

Each **programming language** has its own syntax, i.e. its own format that the computer will accept.

Algorithms

- An **algorithm** is the logical structure of a program. It is expressed through the writing of its set of procedures, regardless of the programming language or the technical constraints of the environment it will be used in.

- A **program** is a sequence of instructions that follows the rules of a specific programming language , written to perform a specified task with a computer.

Algorithms

- So once you've created an algorithm that will solve a problem, <u>this solution will always be valid</u> no matter what computer you're using, what language you're programming in, etc.
- Each program you code will be in a specific language, using a combination of symbols that are considered to be correctly structured for that language, OS, computer, etc.

Analysis

- The first step in designing an algorithm is to analyze the problem that the program must solve in order to achieve its desired results. The analysis consists of establishing four blocks of information:
    - Input(s)
    - Output(s)
    - Internal data
    - The procedure

Inputs (unknown data)

- **Inputs** are the information that is read by the program and required to do its job. Input data can come from a keyboard entry by a user or from a data file.

Outputs (unknown data)

- **Outputs** are the **results** that the program produces. These results can be displayed on the screen, saved to a database, written in a file, etc.

Internal data (known data)

- **Internal data** is the information that the program needs to perform its task, but which is not read as input. This data is entered directly into the program code.

The procedure

- The **procedure** consists of a short, simple, textual description of what the program needs to do to produce the desired results.
  - How does it treat the information it receives?
  - What must it calculate?

- These four blocks of information do not necessarily exist in all programs. For example, programs often have no internal data. It is rare, however, that a program has no output.

Example

- Suppose we want a program that reads an amount and adds a 15% tax. The program must then display the new amount, after tax, on the screen. The analysis of this program is as follows:

  Input:

  The amount

  Output:

  The amount after tax

  Internal data:

  15 % tax

  Procedure:

  Read the amount, multiply it by 1.15 to get
  the amount after tax, and then write it on the screen

Exercises

- Write an algorithm that takes an integer value and returns the cube of that value.

Answer

Input:

An integer value

Output:

The cube of the value

Internal data:

--

Procedure:

Raise the value to the power of 3

Exercises

- Write an algorithm that receives a numeric value and returns 0 if the value is less than 0, and otherwise returns 1.

Answer

Input:

A numeric value

Output:

A value of 0 or 1 depending on the procedure

Internal data:

--

Procedure:

If the value is less than 0, return 0, else, return1

Exercises

- Write an algorithm that takes two numbers as input and displays the sum, the difference, the product, and the quotient of the two numbers (without validation).

Answer

Input:

The first number

The second number

Output:

The sum, the difference, the product, and the quotient of the two numbers.

Internal data:

--

Procedure:

Display the sum of the two numbers
Display the difference of the two numbers
Display the product of the two numbers
Display the quotient of the two numbers

Exercises

- Write an algorithm that asks for a starting number, and then calculates the sum of integers from 1 up to this number. For example, if 4 is entered, the program should calculate the following series:1 + 2 + 3 + 4 = 10

Answer

Input:

The number

Output:

The sum of integers up to the entered number

Internal data:

--

Procedure:

Add the integers from 1 (inclusive) until the number being added is equal to the value entered by the user

Exercises

- Make a program that converts US dollars and Euros into Canadian dollars according to the following exchange rates:

US dollars	Canadian dollars	Rate: $1.48
Euros	Canadian dollars	Rate: $1.55

This program will need to ask the user to select which type of currency they would like to convert. The user will have to type 1 or 2 to indicate their choice. The program will then require the user to enter the amount of foreign currency they would like to convert. The program will then display the equivalent amount in Canadian dollars. (No validation required)

Answer

Input:

The type of currency to convert

The amount of currency to convert

Output:

The equivalent amount in Canadian dollars

Internal data:

US dollar exchange rate = $1.48
Euro exchange rate = $1.55

Procedure:

Ask for type of foreign currency (1 for US or 2 for Euro)
Ask for the amount of money to convert
Display the converted amount of money