Introduction to informatics

Piroska Biró

Course 08

Algorithms

+

Data Structures

=

Programs

Niklaus Wirth , 1976

Definition of the algorithm

A finite, well-defined, unambiguously determined sequence of basic steps used for the solution of a problem/task and within an infinite period of time it comes to a halt and reaches its aim.

Shortly:

Algorithm is the solution of a problem with an infinite number of partial steps, which are unambiguous and complete.

Requirements of algorithm

- should be made up of basic steps
- the order of the steps should be unambiguous
- they solve the problem within a finite period of time and in finite number

Correctness of the algorithm

- Algorithm can be called **correct** if it takes every concrete input and produces correct output then stops (the correct algorithm solves the problem).

- Algorithm can be called **incorrect** which produces unexpected solution, can stop at intermediary steps or it becomes infinite (does not stop).

Segments of the algorithm

Input: those initial values from which the algorithm generates output

Output: the results coming from the input values generated by the algorithm

Running time (complexity): the number of the executed basic operations or the completed steps (state-transitions) during the running of the algorithm; the length of the running time depends on the input

- **The worst:** any kind of input gives the upper bound of the running time. Knowing this the algorithm will not last longer than expected.
- **The best:** any kind of input gives the lower bound of the running time. The algorithm runs during the shortest possible time.
- **Average:** It is frequently bad like the worst case.

Segments of the algorithm

- **sequences** - statements coming one after the other,
- **iterations or loops** - repetition of statements, until the fulfilment of an examined condition,
- **selections** - branching of algorithm depending on the condition.

Languages for describing algorithm

- **Textual description**: natural verbal language, fast and understandable by everyone but unpunctual.
- **Pseudocode**: basic statements, the operating or frame statements which give the skeleton of the algorithm. Not a specific program language, informal, it has a program language like symbol system. It is often referred to as **PDL** (Problem Definition Language). Its weakness is that no information is given (size, type, aim, etc.) during encoding, so it always contains a data definition (identifiers, size, type, meaning)
- **Flow chart or block diagram**: with the help of a chart we can follow the steps.

Languages for describing algorithm

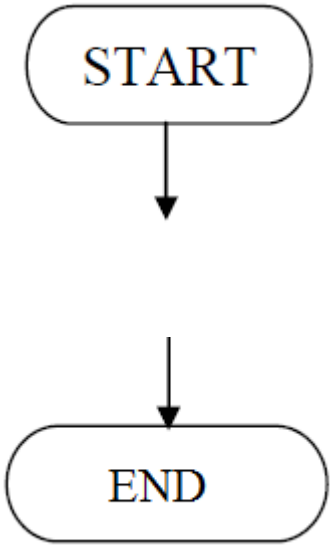
- **Structogram,** structure diagram, Chinese box or Chapin-chart: a very important characteristic feature is that the language is independent device of description that is the description of algorithm does not depend on the programming language.
- **Meta language ("the language of language"):** a language with unique vocabulary, symbols, and grammar, with the help of which the formal and contextual analysis of a language can be done. For example: BNF (Backus-Naur Form).
- **High level programming language:** Pascal, FORTRAN, ADA, BASIC, C, C++, Java, LISP, Algol etc.

The steps of problem solution

1. The definition of the problem, precision and generalization.
2. The definition of the data structure, the specification of input-output.
3. The selection of the mathematical or other model, its definition if needed or possible.
4. The planning and the implementation of the algorithm for the solution
5. Program writing, encoding (on the basis of the data structure and the algorithm)
6. testing, debugging
7. Documentation (for users and developers)

Flow chart

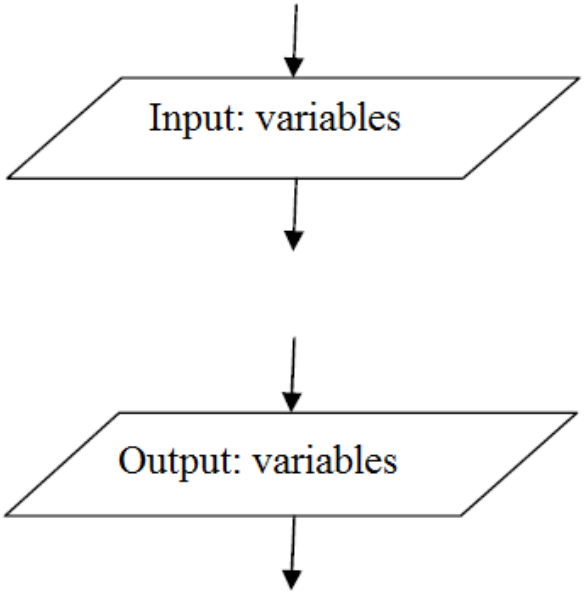
Start and **end** symbols of the alghorithm represented as circles, ovals or rounded rectangles.



Flow chart

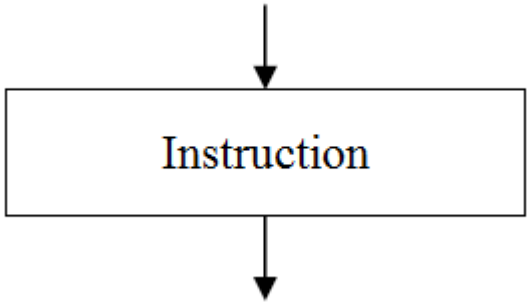
Input/Output

Represented as a parallelogram.



Flow chart

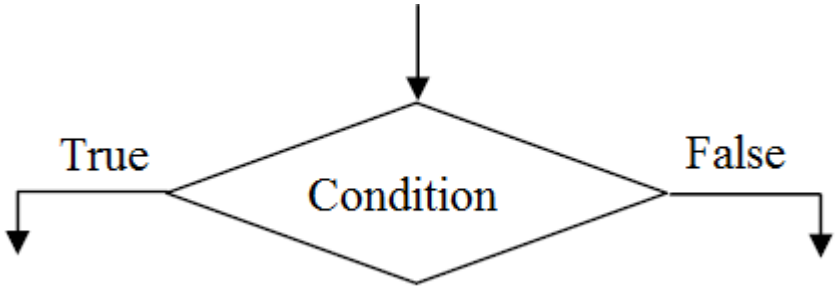
Instructions



Flow chart

Conditional or decision

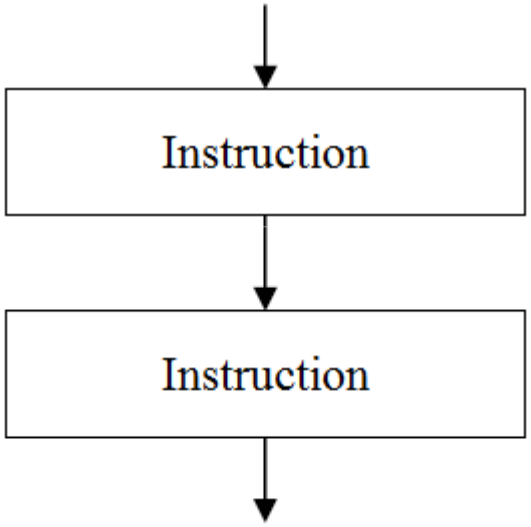
Represented as a diamond (rhombus).



Flow chart

Program structures

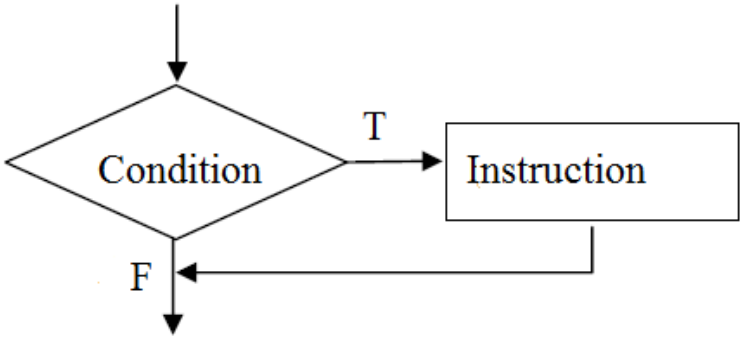
Sequences



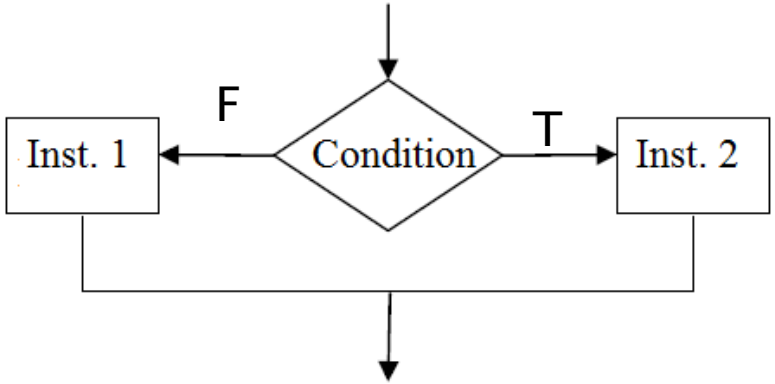
Flow chart

Program structures

Conditional branching



single branch

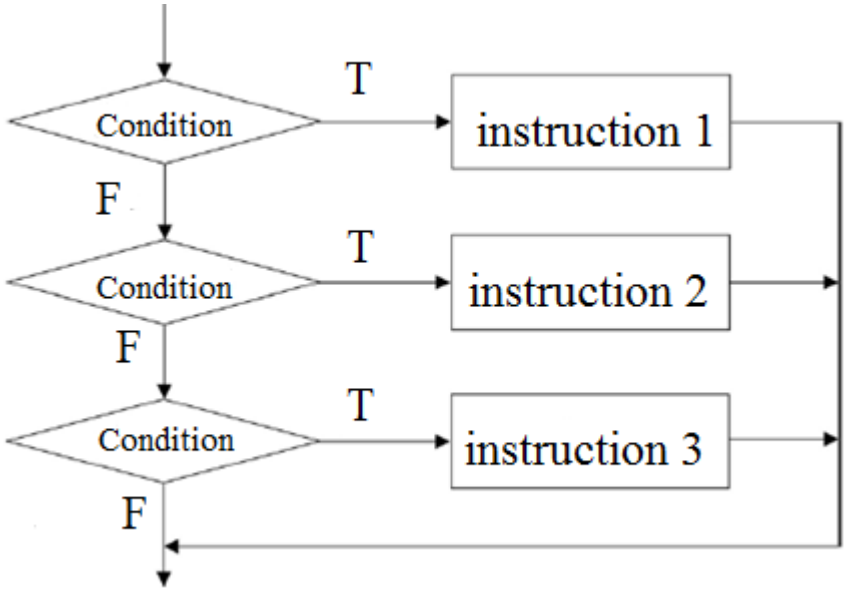


double branch

Flow chart

Program structures

Embedded condition

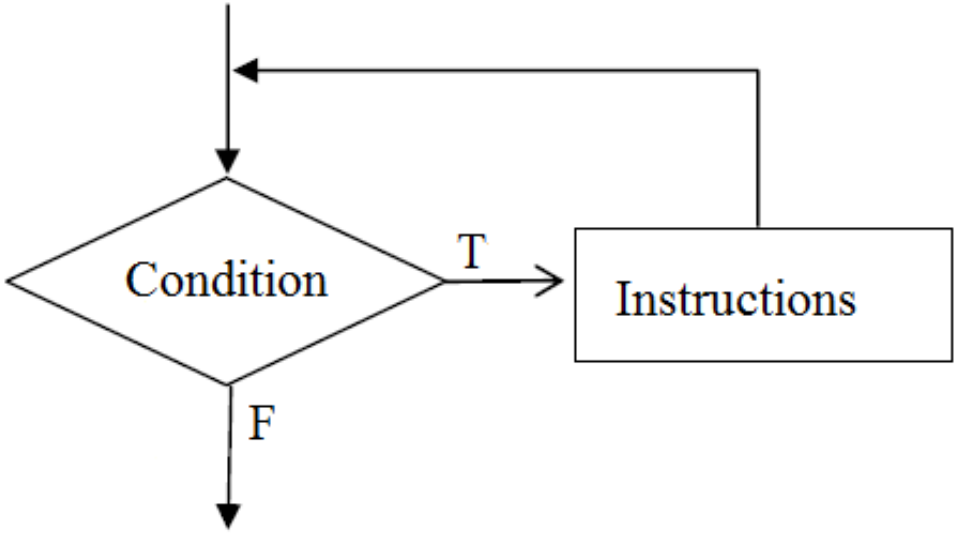


multiple branching

Flow chart

Program structures

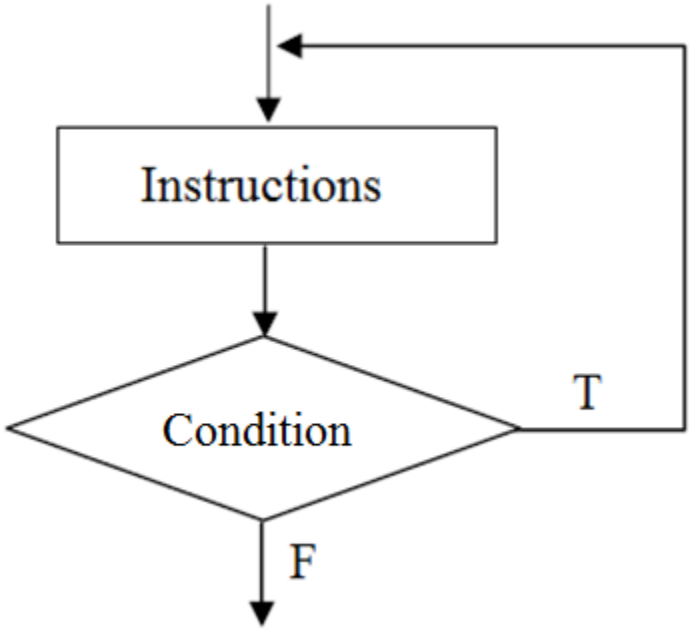
Pre-test loop



Flow chart

Program structures

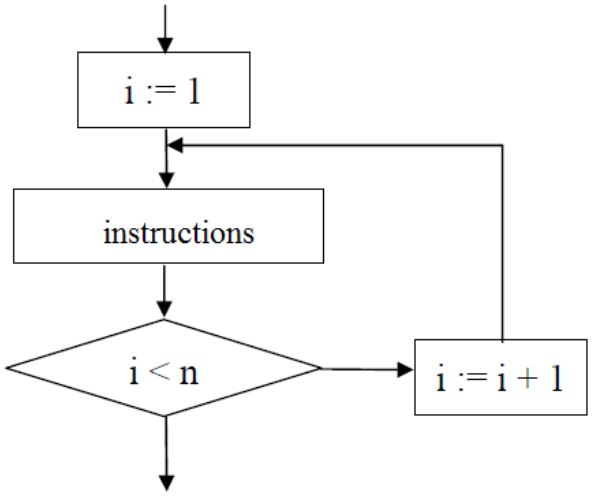
Post-test loop

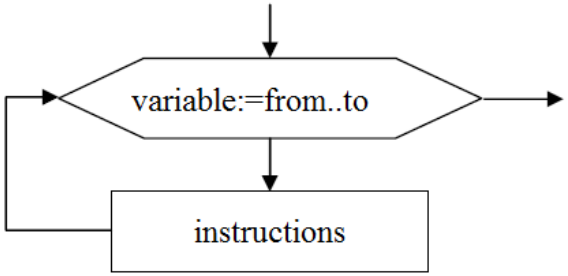


Flow chart

Program structures

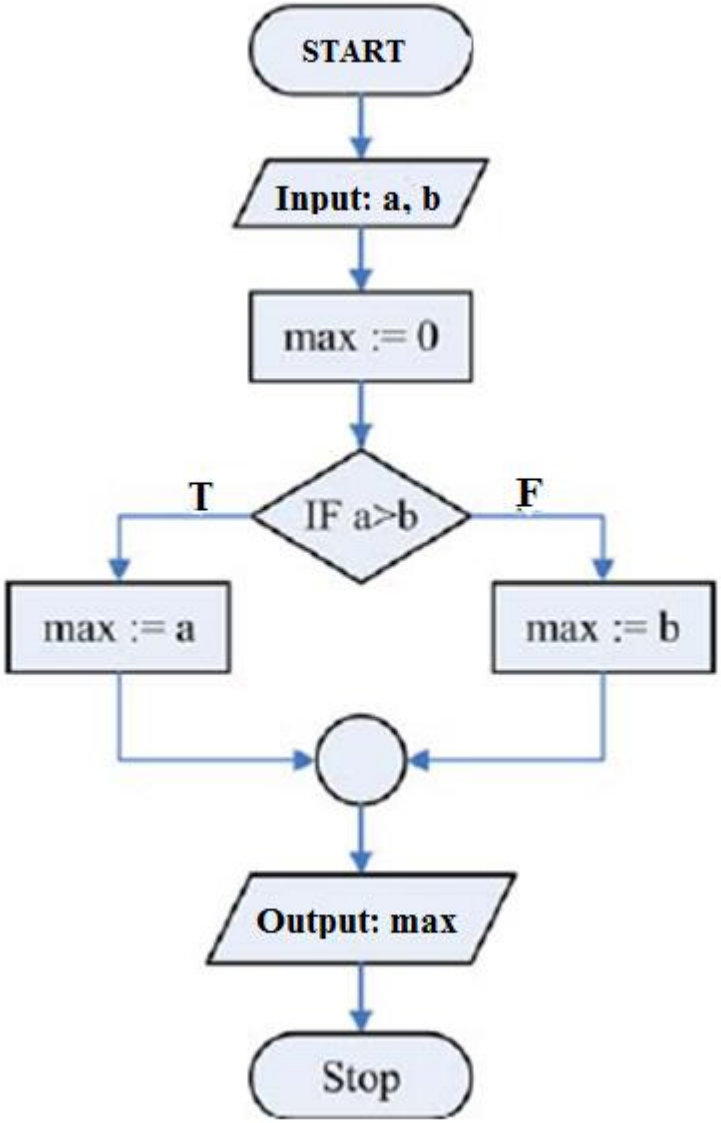
For loop





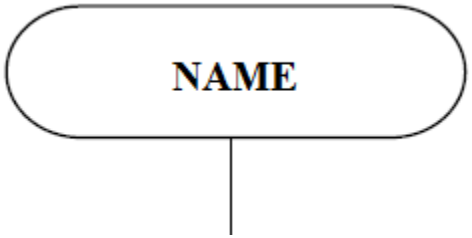
Example

The maximum of two numbers.



Structogram

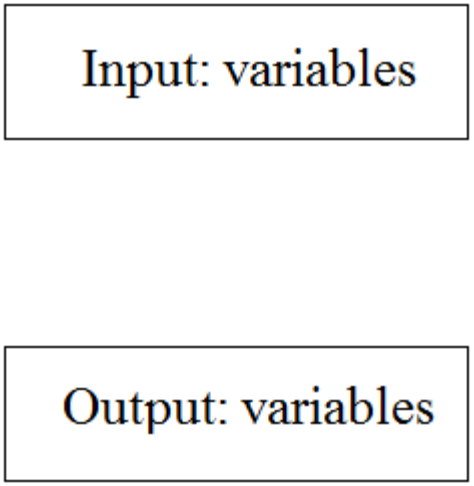
Start symbols of the alghorithm represented as circles, ovals or rounded rectangles.



Structogram

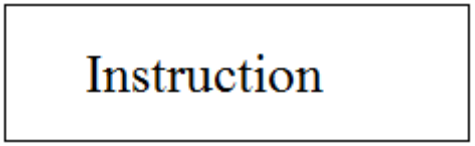
Input/Output

Represented as a rectangular.



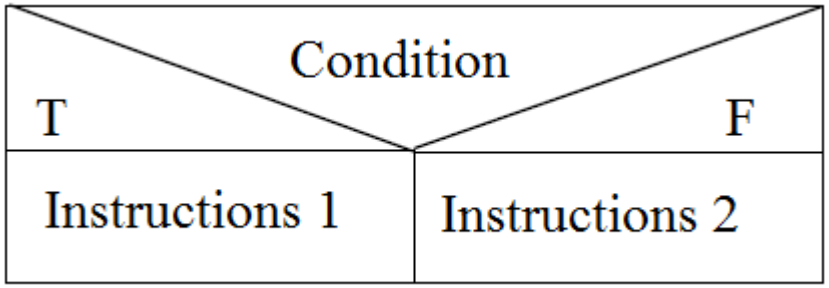
Structogram

Instructions



Structogram

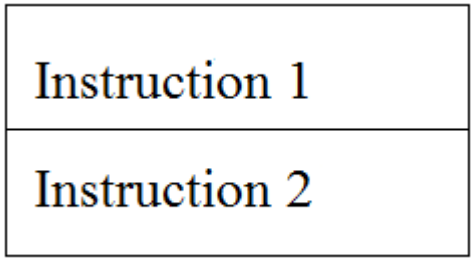
Conditional or decision



Structogram

Program structures

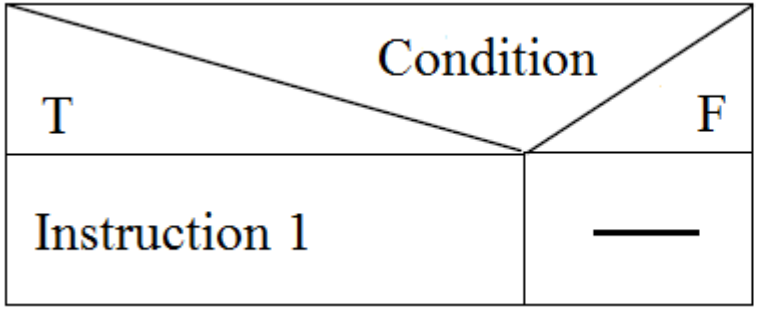
Sequences



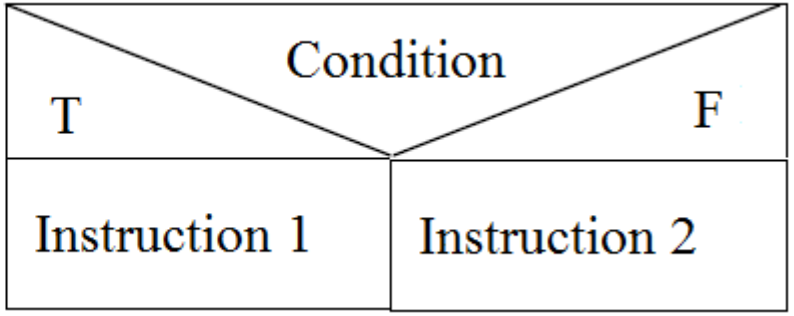
Structogram

Program structures

Conditional branching



single branch



double branch

Structogram

Program structures

Embedded conditions

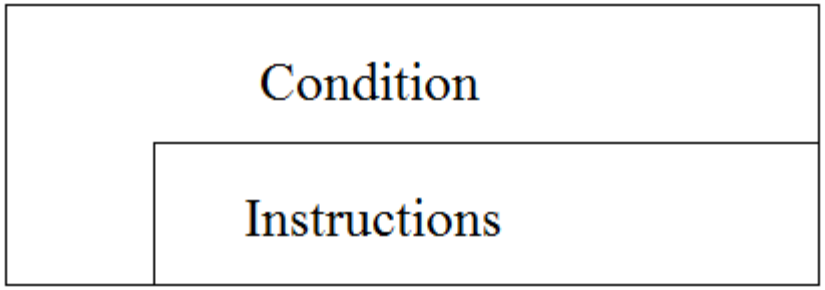
Condition 1	Condition 2	...	Condition n
Instruction 1	Instruction 2	...	Instruction n

multiple branching

Structogram

Program structures

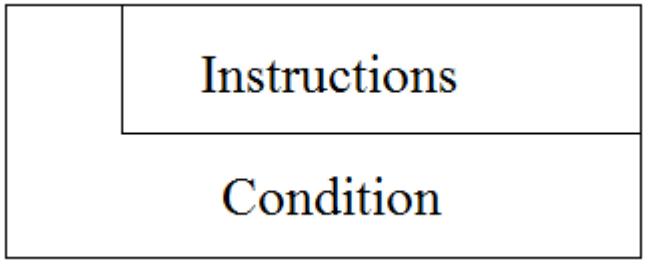
Pre-test loop



Structogram

Program structures

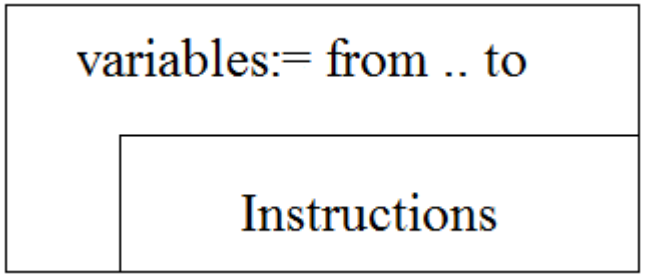
Post-test loop



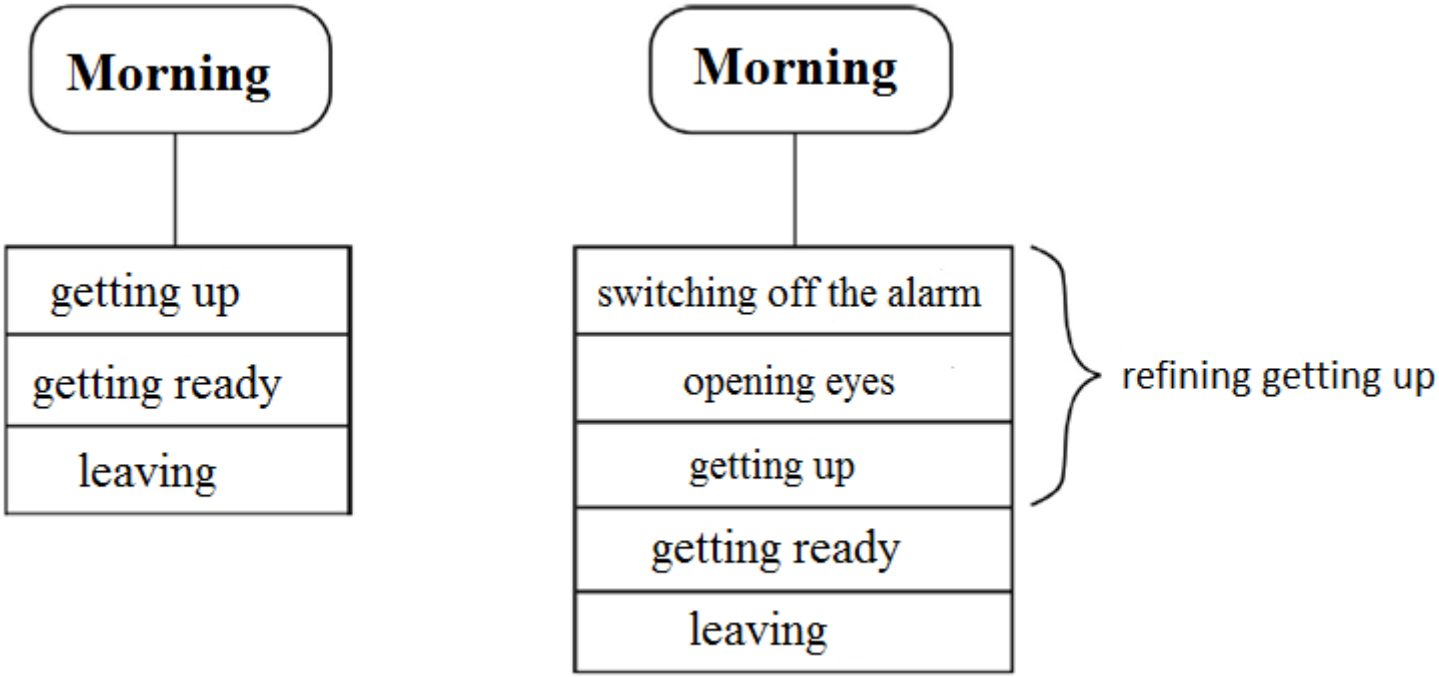
Structogram

Program structures

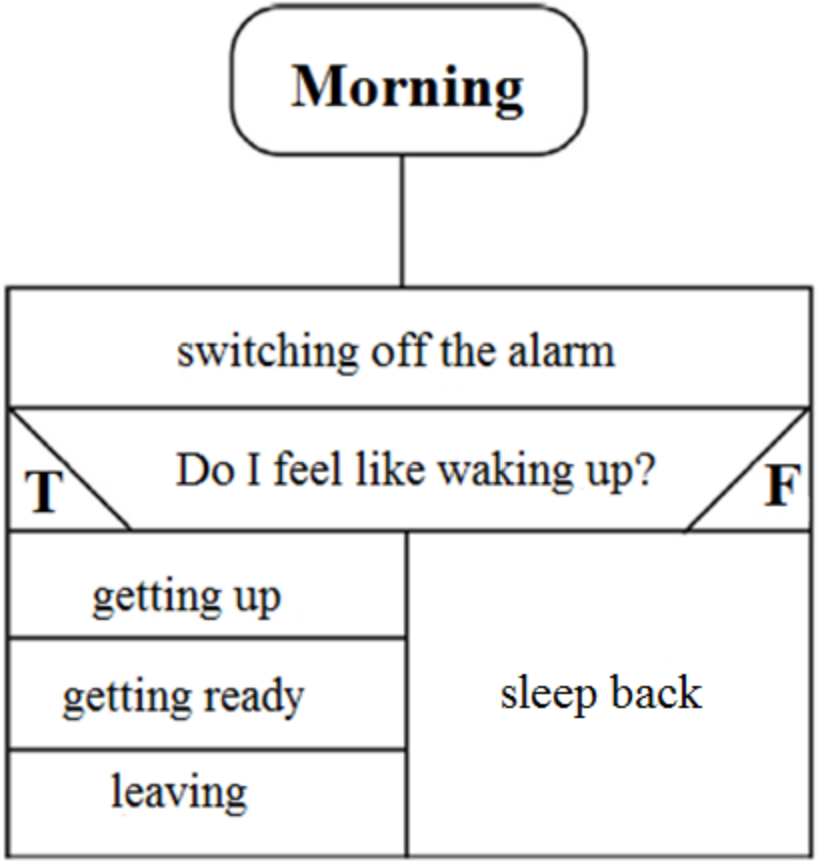
For loop



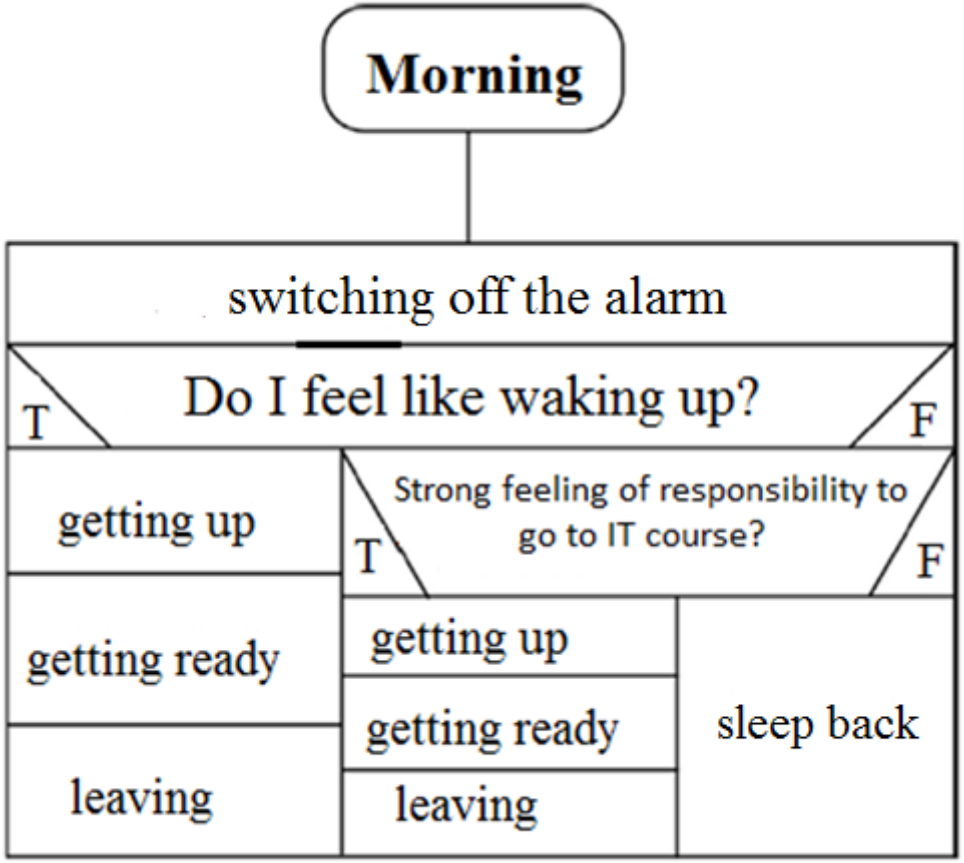
Example – Sequence



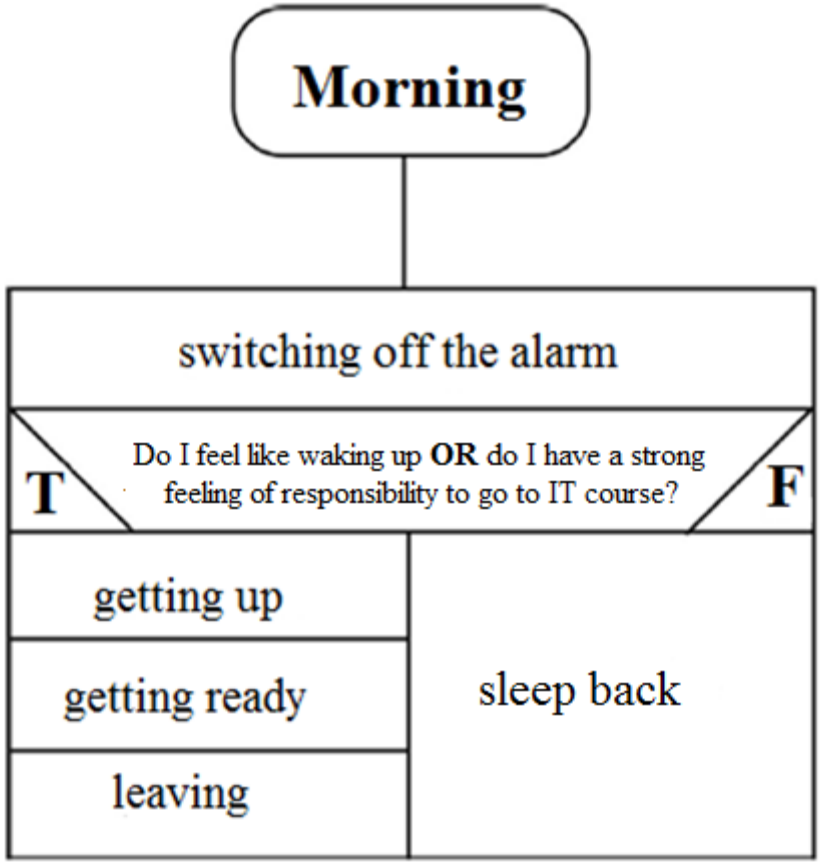
Example 1 – Selection



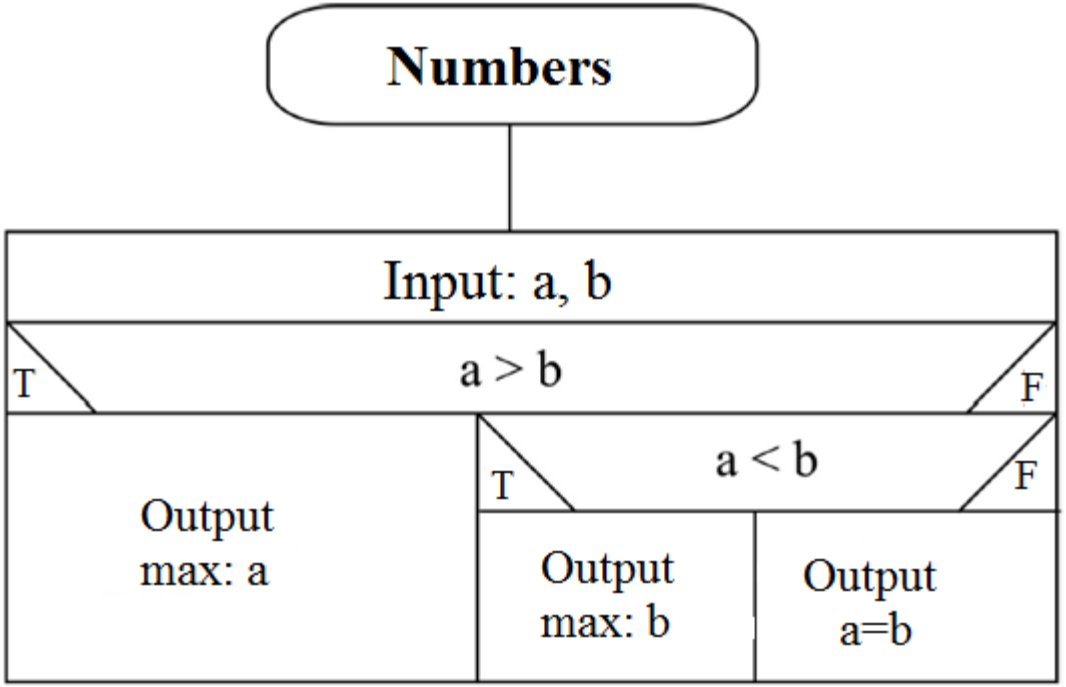
Example 2 – Selection



Example 3 – Selection



Example 4 – Selection

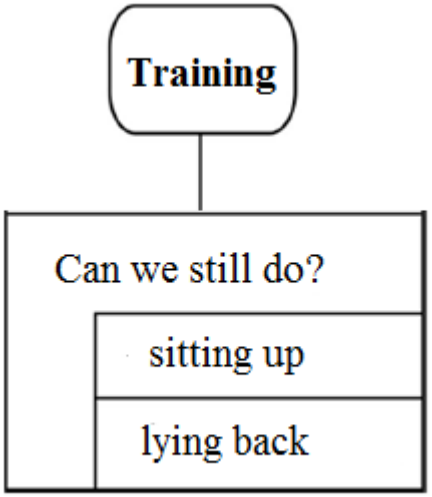


Example 5 – Multiple selection

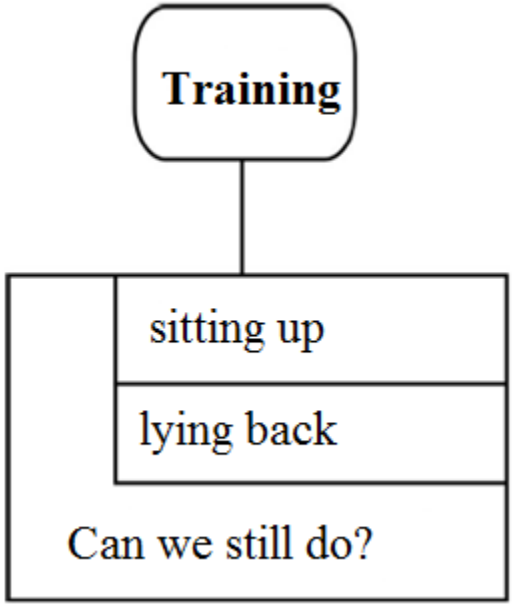
Dresses

Input: t (temperature)					
t <=-2	-2 < t <= 5	5 < t < 15	15 < t <= 20	20 < t <= 30	30 < t
Output: fur coat	Output: sweater and jacket	Output: jacket	Output: sweater	Output: light dress	Output: swimming dress

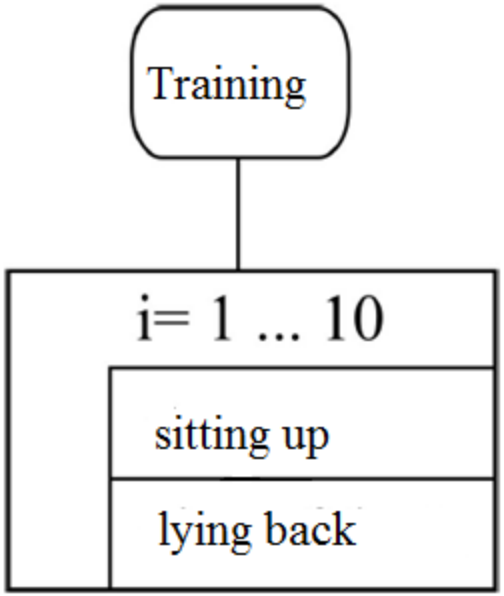
Example 6 Pre-test Loop



Example 7 Post-test Loop



Example 8 For Loop



Pseudocode – keywords

- START, BEGIN, END, STOP
- DO, WHILE, DO WHILE, FOR, UNTIL, DO UNTIL, REPEAT, END WHILE, END UNTIL, END REPEAT
- IF THEN, IF, ELSE, IF ELSE, END IF, THEN, ELSE THEN, ELSE IF, SO, CASE
- EQUAL, LT, LE, GT, GE, NOT, TRUE, FALSE, AND, OR, XOR
- GET, WRITE, PUT, UPDATE, CLOSE, OPEN, CREATE, DELETE, EXIT, FILE, READ, EOF, EOT, WITH, RETURN

Pseudocode

Selection

if *condition* **then**
  instructions
[**else if** *condition* **then**
  instructions] . . .
[**else**
  instructions]
end if

Iterations – Loops

For loop

for *loop variable* ← *initial value* **to** *end value* **do**

    instructions

end for

Pre-test loop

while *condition* **do**

    instructions

end while

Iterations – Loops

Post-test loop

repeat

instructions

until *condition*

do

instructions

while *condition*

Function and procedure

function Function_name[(*parameters*)]
  function body/instructions
end function

procedure Procedure_name [(*parameters*)]
  procedure body/instructions
end procedure

Pseudocode

- The maximum of two numbers

```
BEGIN
        INPUT: a,b
        IF (a>b)
                max := a
        ELSE
                max := b
        END
        OUTPUT: max
END
```

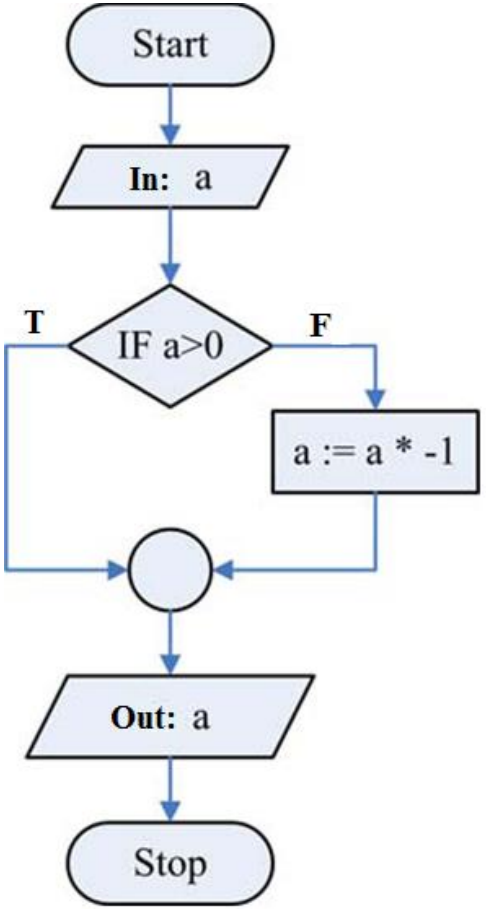
Exercises

Write the algorithm of the following exercices using the following language of description:
(Pseudocode, Flow chart, Structogram)

- Absolute value of the given number
- The maximum of three numbers
- The factorial of the N numbers
- The maximum of N numbers

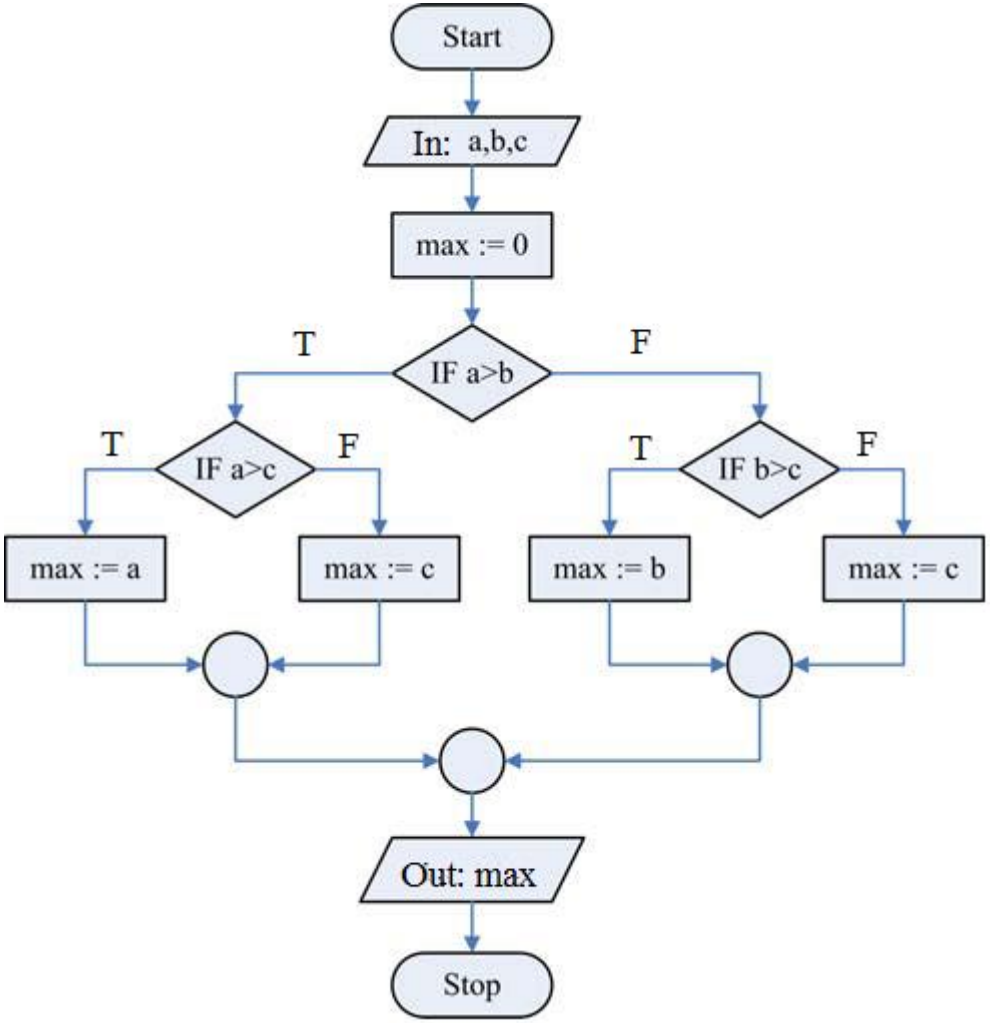
Absolute value of the given number

```
BEGIN
        INPUT: a
        IF a<0
                a := a*(-1)
        END
        OUTPUT: a
END
```



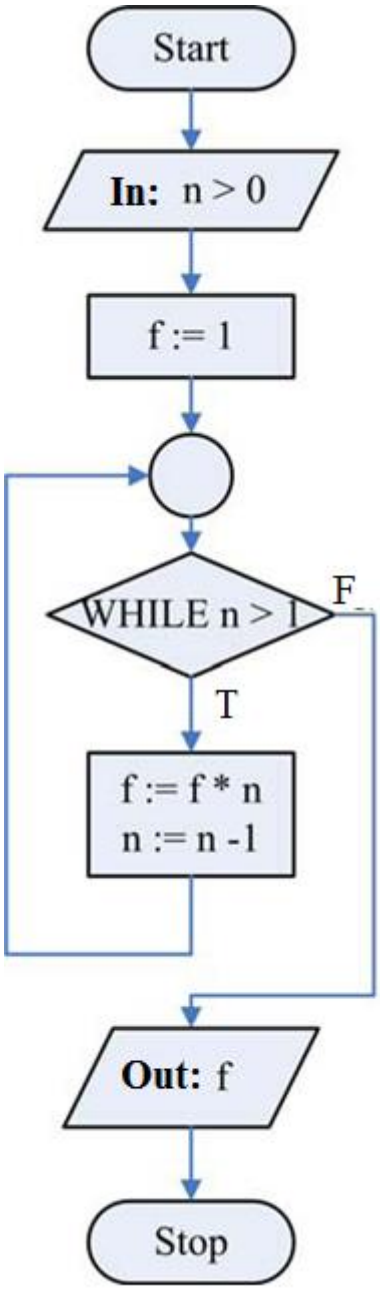
Maximum of three numbers

```
BEGIN
        INPUT: a,b,c
        IF (a>b)
                IF (a>c)
                        max := a
                ELSE
                        max := c
                END
        ELSE
                IF (b>c)
                        max := b
                ELSE
                        max := c
                END
        END
        OUTPUT: max
END
```



The factorial of the N numbers

```
BEGIN
        INPUT: n
        f := 1
        WHILE (n>1)
                f := f*n
                n := n-1
        END
        OUTPUT: f
END
```



Maximum of N numbers

```
BEGIN
        INPUT: a[N]
        max := a[1],  i := 2
        WHILE (i<=N)
                IF (a[i]>max)
                        max := a[i]
                END
                i := i+1
        END
        OUTPUT: max
END
```