PROBLEM SOLVING WITH LOOP AND CASE LOGIC STRUCTURE

PROBLEM SOLVING WITH LOOP AND CASE LOGIC STRUCTURE

Objective:

- Develop problems using the loop logic structure
- Using nested loop constructs
- Distinguish the different uses of three loop constructs.

Pre-requisites

Short hand Assignment Operators

Assignment operators in an expression c = c + 3 can be abbreviated as

c +=3

(using the addition short hand assignment operator)

Examples of other short hand assignment operators:

d -= 4	(d = d -4)
e *= 5	(e = e * 5)
f /= 3	(f = f / 3)
g %= 9	(g = g % 9)

Pre-requisites contd…

- **Increment operator (++)**
  - **Can be used instead of c+=1**
- **Decrement operator (--)**
  - **Can be used instead of c --= 1**
- **Pre increment / Pre decrement**
  - **Operator is used before the variable (++c or --c)**
  - **Variable is changed before the expression is evaluated**
- **Post increment / Post decrement**
  - **Operator is used after the variable (c++ or c --)**
  - **Expression executes before the variable is changed**

ITERATION

- Have you found yourself doing certain things over and over again?
- Think of three things you do over and over again
- Maybe you go shopping a few times a week
- *Monday*
  Wake up
  Get into car
  Do shopping
  Come home

  Tuesday
  Wake up
  Get into car
  Do shopping
  Come home

  Wednesday
  Wake up
  Get into car
  Do shopping
  Come home

Go to statement

- Transfers the system control to another instruction in the solution instead of processing the next instruction in sequence
- Disadv:Reduces readability of the program.
- Thus replaced by loop constructs.
- Two standard tasks accomplished through the use of loop constructs:
  - Counting (Incrementing and decrementing)
  - Accumulating (calculating sum or total)

Types of loops

1. While/ WhileEnd statement
2. Repeat/Until
3. Automatic-Counter loop

While Statement

- The while_statement is used when the program needs to perform repetitive tasks.
- While the condition is true, repeat all instructions between While and the WhileEnd.
- The while statement has the form:

```
while <condition(s)>
        Instruction
        Instruction
        Instruction
        .
        .
WhileEnd
```

While/While End

Algorithm:

While<condition(s)>

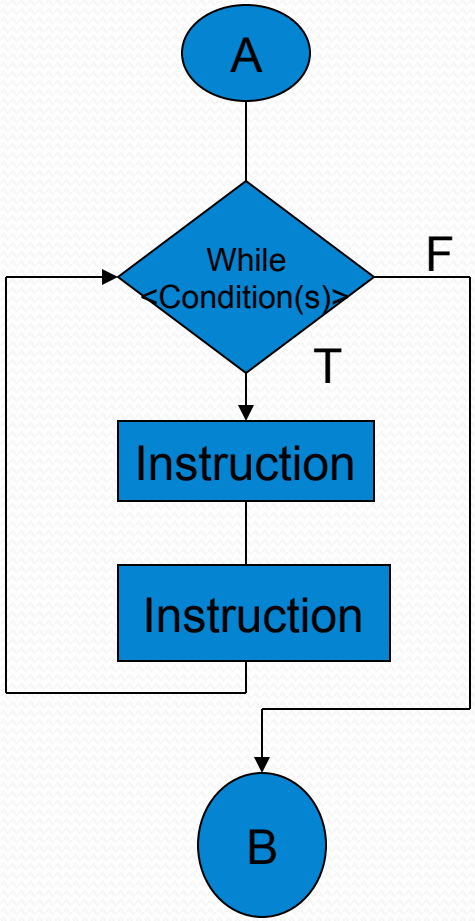
Instruction

Instruction

.

.

WhileEnd



Decision equivalent to While/whlile end

Algorithm:

If<conditions)>

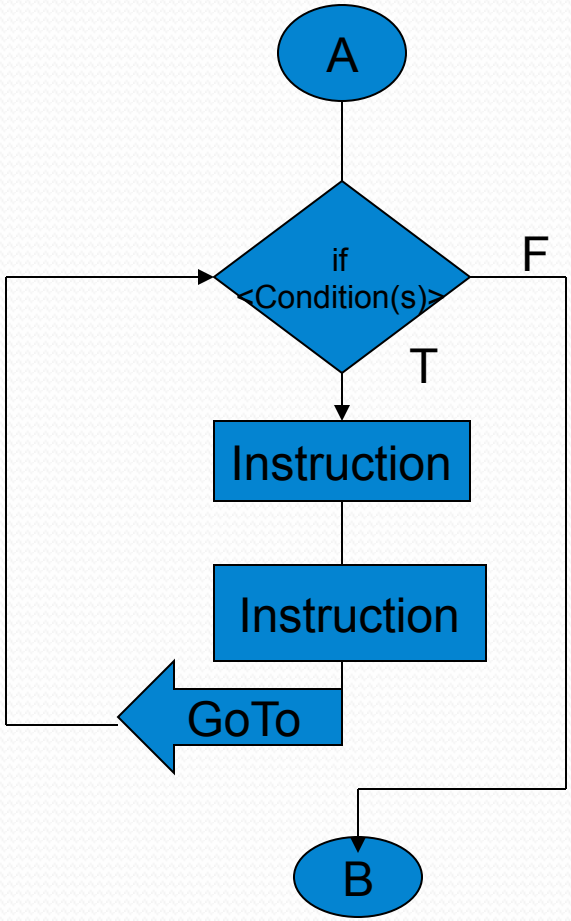
Then

T

Instruction

Instruction

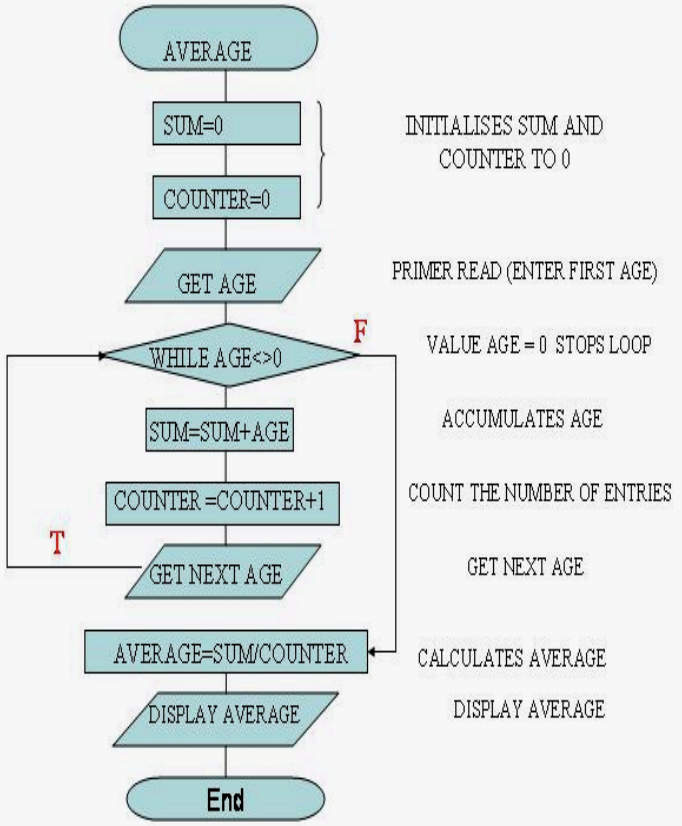
GoTo100



To calculate average of input ages

ALGORITHM

- Set sum to zero
- Set counter to zero
- Get age (priming Read)
- WHILE age <> 0
  - Sum = sum + age
  - Counter = counter + 1
  - Get next age
- WHILE END
- Average =sum/counter
- Display average
- End

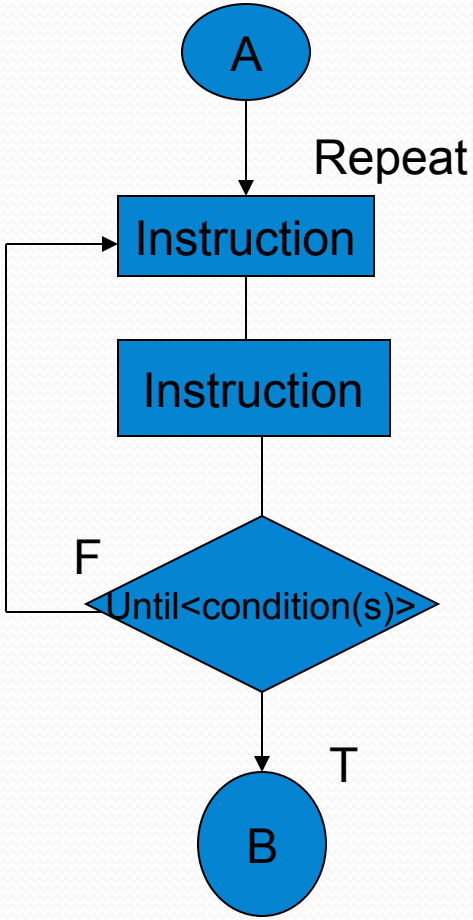


Repeat/Until

It tells the computer to repeat the set of instructions between the Repeat and until , until a condition is true

Algorithm:

Repeat
  Instruction
  Instruction
  .
  .
Until<condition(s)>



Decision equivalent to Repeat-until loop

10 Instruction

11 Instruction

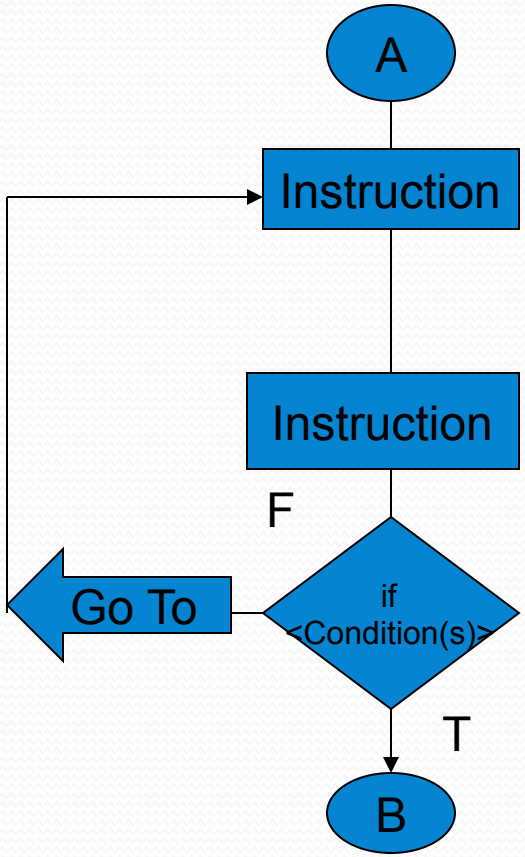
12 If<condition(s)>

T — Then

Continue

F — Else

Go To 10

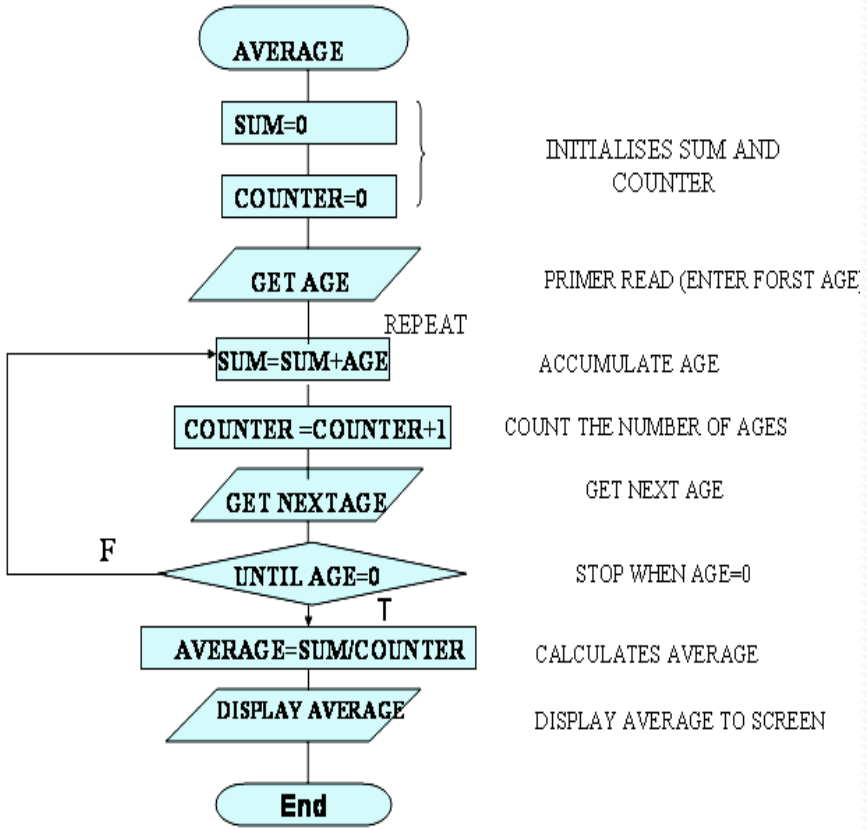


To calculate average of input ages

ALGORITHM

- Set sum to zero
- Set counter to zero
- Get age (priming lead)
- REPEAT
  - Sum = sum + age
  - Counter = counter + 1
  - Get next age

  UNTIL AGE = 0
- Average =sum/counter
- Display average
- End



Difference between While/WhileEnd and Repeat/Until loop structures

While/WhileEnd

- Program continues to loop as long as the condition is true.
- Condition is evaluated at the beginning.
- If the condition fails at the beginning itself, then instructions are not executed even once.

Repeat/Until

- Program stops the loop process if the condition is true.
- Condition is evaluated at the end.
- Ensures execution of the instruction inside loop at least once in the program.

AUTOMATIC COUNTER LOOP

- It increments or decrements a variable each time the loop is repeated.
- A variable acts as a counter that is initialized and incremented/decremented each time the loop is processed.
- The loop terminates when the counter exceeds the upperlimit.
- The test for whether or not ot continue is present at the beginning or end of the loop depending on the language.

AUTOMATIC COUNTER LOOP

FLOWCHART

ALGORITHM

Loop: Counter =Begin To End Step S

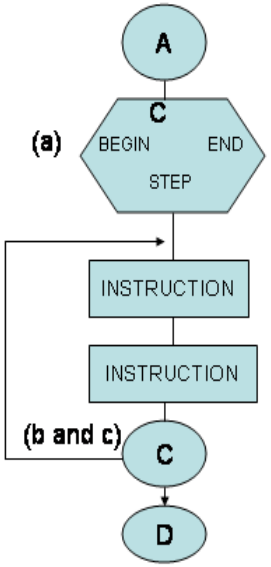
Instruction

Instruction

.

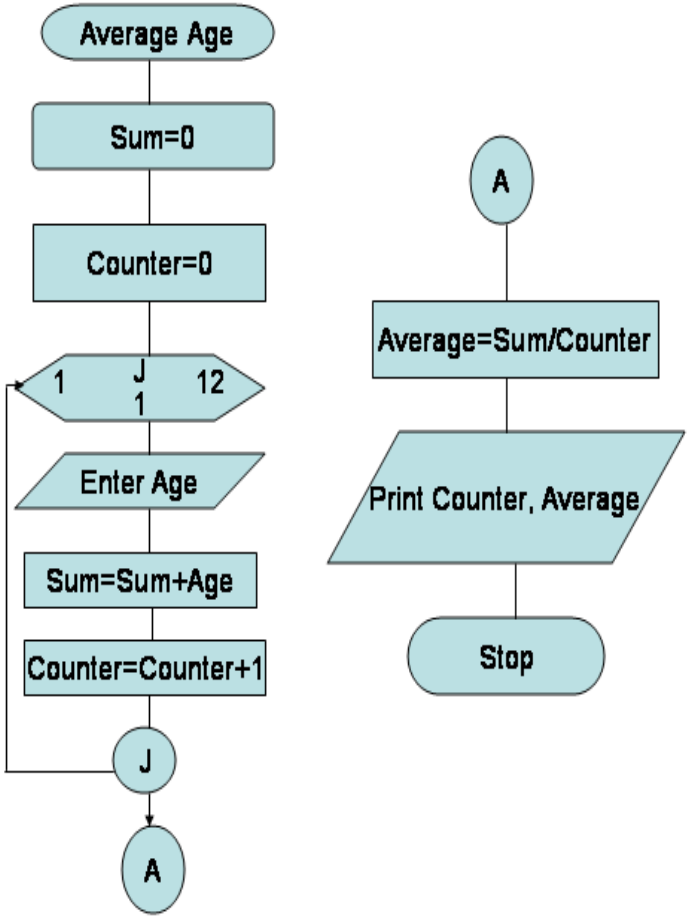
.

Loop-End :Counter



Algorithm and Flowchart Using Automatic Counter Loop

1.AverageAge
2.Sum=0
Counter=0
3.Loop:J=1 to 12
Enter Age
Sum= Sum +Age
Counter=Countet+1
Loop-End: J
4.Average=Sum/Counter
5.Print Counter, Average
6.End



RECURSION

- When a module or a function calls itself.
- The condition that ends the loop must be within the module
- Usually faster.
- Ex. Recursive function Factorial(N) continues to call itself until N=1

Control

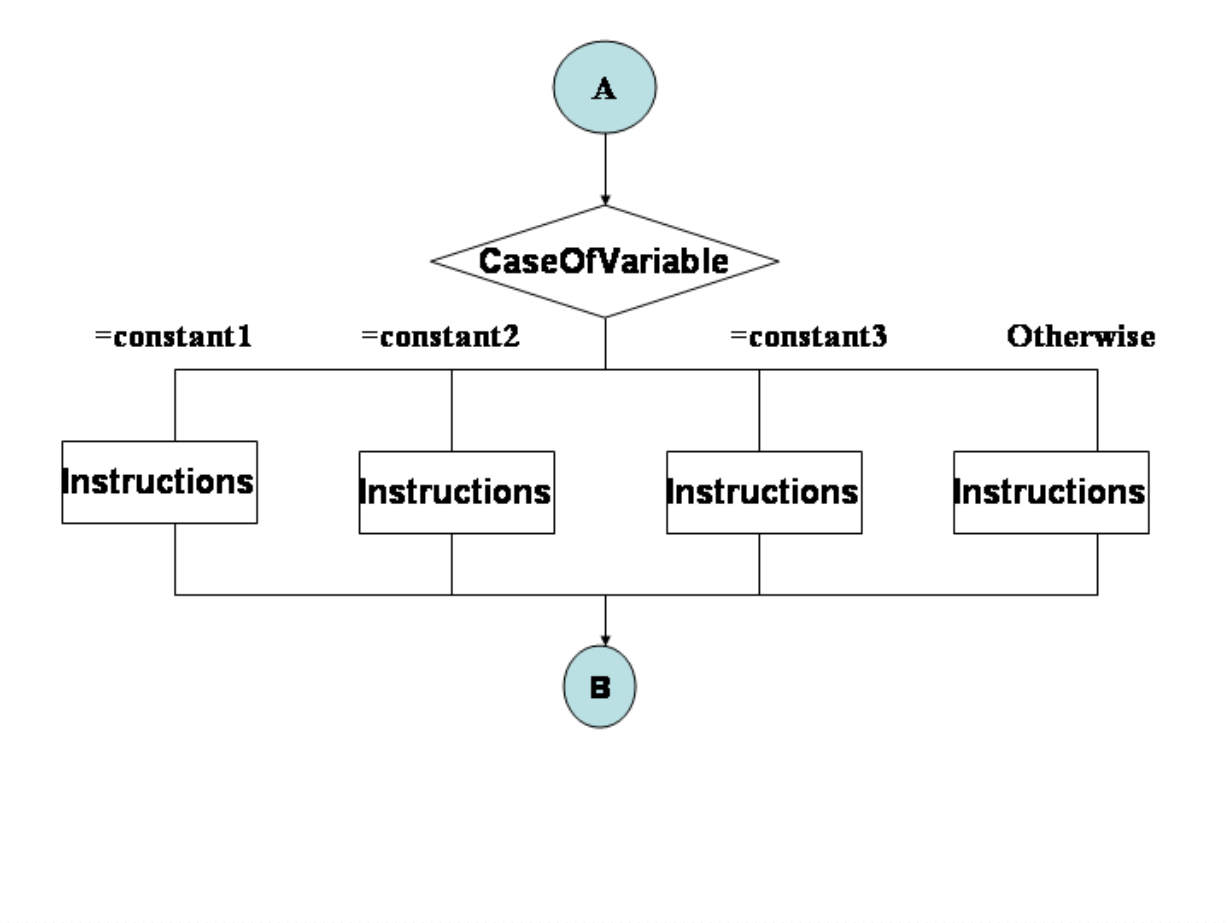
1. Enter N
2. Nfact=Fact(N)
3. Print Nfact
4. End

Fact(N)

1. If(N>1) then
   Factorial=N*fact(N-1)
   Else
   Factorial=1
2. Exit

CASE Structure Flowchart

•Made up of several sets of instructions, of which only one will be selected and executed by the user input.

```
CaseOfVariable
        =CONSTANT1:
                actions for variable = CONSTANT1
        =CONSTANT2:
                actions for variable = CONSTANT2
        =CONSTANT3:
                actions for variable = CONSTANT3
                .
                .
        Otherwise:
                actions for variable = Anything else
EndOfCase
```

CASE Structure Flowchart