ASSEMBLY LANGUAGE LOOP / DECISION MAKING

Course Teacher

Shaila Rahman

LESSON PLAN

- Assembly program structure for Flow Control
  Make decision - The use of Jump Instruction
  Repeat a section of code – Using Loop Instructions

THE JUMP INSTRUCTIONS

Jump Instructions are two types

- Conditional Jump Instructions

  Ex: JZ, JG, JL

- Unconditional Jump Instructions

  Ex: JMP

CONDITIONAL JUMP INSTRUCTIONS

……

Statement # 1

Statement #2

……..

Label:

…..

Statement # n

……

JZ Label

…….

Statement # m

…..

UNCONDITIONAL JUMP INSTRUCTIONS

```
        ……
        statement # 1
        Statement #2
Label:
        Statement n
                ……
        JMP Label
          Statement m
        …..
```

CONDITIONAL JUMP INSTRUCTIONS

- JZ, JG, JL, JNZ, JNG, JNL, JNGE, JNLE,
- JG == JNLE
- JL == JNGE

Syntax:

- Jxxx destination_label

CONDITIONAL JUMP INSTRUCTIONS

Signed Jumps

Symbol	*Description*	*Condition for Jumps*
JG/JNLE	jump if greater than jump if not less than or equal to	ZF = 0 and SF = OF
JGE/JNL	jump if greater than or equal to jump if not less than or equal to	SF = OF
JL/JNGE	jump if less than jump if not greater than or equal	SF <> OF
JLE/JNG	jump if less than or equal jump if not greater than	ZF = 1 or SF <> OF

CONDITIONAL JUMP INSTRUCTIONS

Unsigned Conditional Jumps

Symbol	*Description*	*Condition for Jumps*
JA/JNBE	jump if above	CF = 0 and ZF = 0
	jump if not below or equal	
JAE/JNB	jump if above or equal	CF = 0
	jump if not below	
JB/JNAE	jump if below	CF = 1
	jump if not above or equal	
JBE/JNA	jump if equal	CF = 1 or ZF = 1
	jump if not above	

CONDITIONAL JUMP INSTRUCTIONS

Single-Flag Jumps

Symbol	Description	Condition for Jumps
JE/JZ	jump if equal	ZF = 1
	jump if equal to zero	
JNE/JNZ	jump if not equal	ZF = 0
	jump if not zero	
JC	jump if carry	CF = 1
JNC	jump if no carry	CF = 0
JO	jump if overflow	OF = 1
JNO	jump if no overflow	OF = 0
JS	jump if sign negative	SF = 1
JNS	jump if nonnegative sign	SF = 0
JP/JPE	jump if parity even	PF = 1
JNP/JPO	jump if parity odd	PF = 0

THE CMP INSTRUCTION

For making decision CMP instruction is used. This is subtraction not affecting destination, but the flags will reflect the result.

CMP destination, source

UNCONDITIONAL JUMP INSTRUCTIONS

Syntax:

JMP label

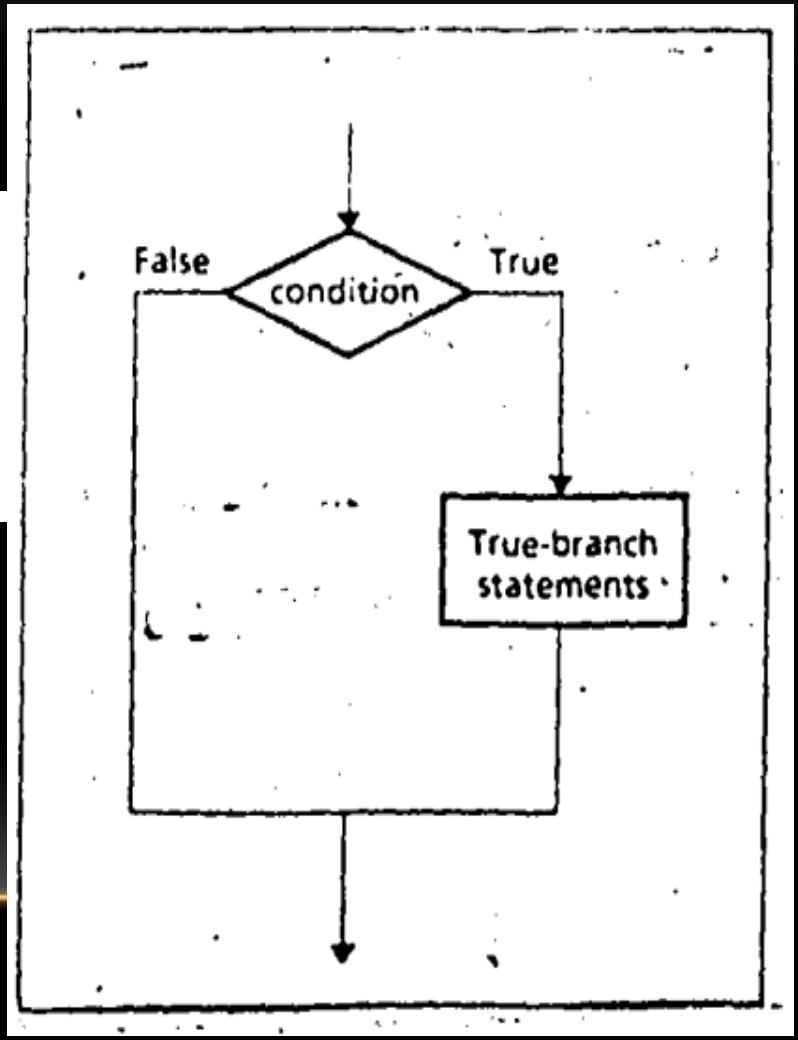
This instruction will not test any condition, just move the control to the label.

BRANCHING STRUCTURE

- IF – THEN(IF)
- IF – THEN – ELSE(IF, ELSE)
- CASE (IF, ELSE IF, ELSE OR SWITCH CASE)

IF-THEN (THE CONTROL FLOW)

```
IF condition is true
 THEN
  execute true-branch statements
END_IF
```



IF-THEN (EXAMPLE)

Example 6.2 Replace the number in AX by its absolute value.

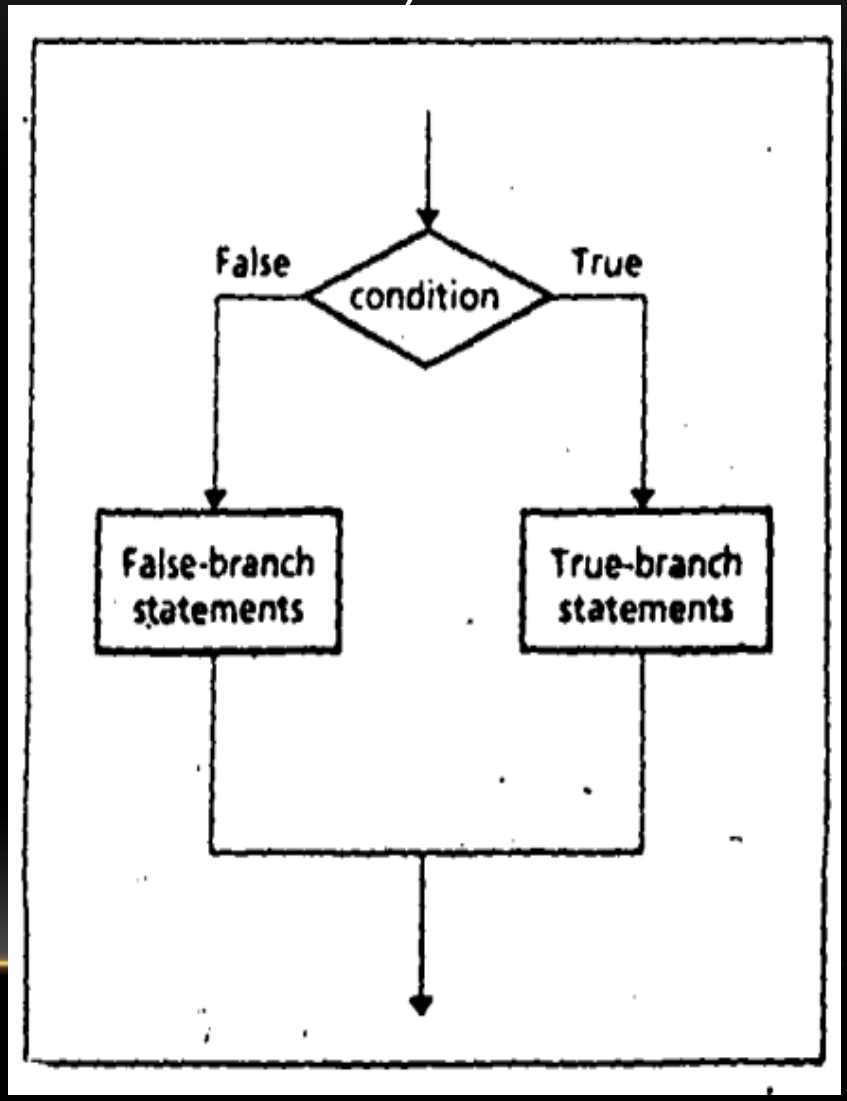
Solution: A pseudocode algorithm is

```
IF AX < 0
 THEN
  replace AX by -AX
END_IF
```

```
;if AX < 0
                CMP   AX,0        ;AX < 0 ?
                JNL   END_IF      ;no, exit
;then
                NEG   AX          ;yes, change sign
END_IF:
```

IF – THEN – ELSE (CONTROL FLOW)

```
IF condition is true
 THEN
  execute true-branch statements
 ELSE
  execute false-branch statements
END_IF
```



IF – THEN – ELSE (EXAMPLE)

Example 6.3 Suppose AL and BL contain extended ASCII characters. Display the one that comes first in the character sequence.

```
                MOV  AH,2       ;prepare to display
;if AL <= BL
                CMP  AL,BL      ;AL <= BL?
                JNBE  ELSE_     ;no, display char in BL
;then                           ;AL <= BL
                MOV DL,AL       ;move char to be displayed
                JMP DISPLAY     ;go to display
ELSE_:                          ;BL < AL
                MOV DL,BL
```

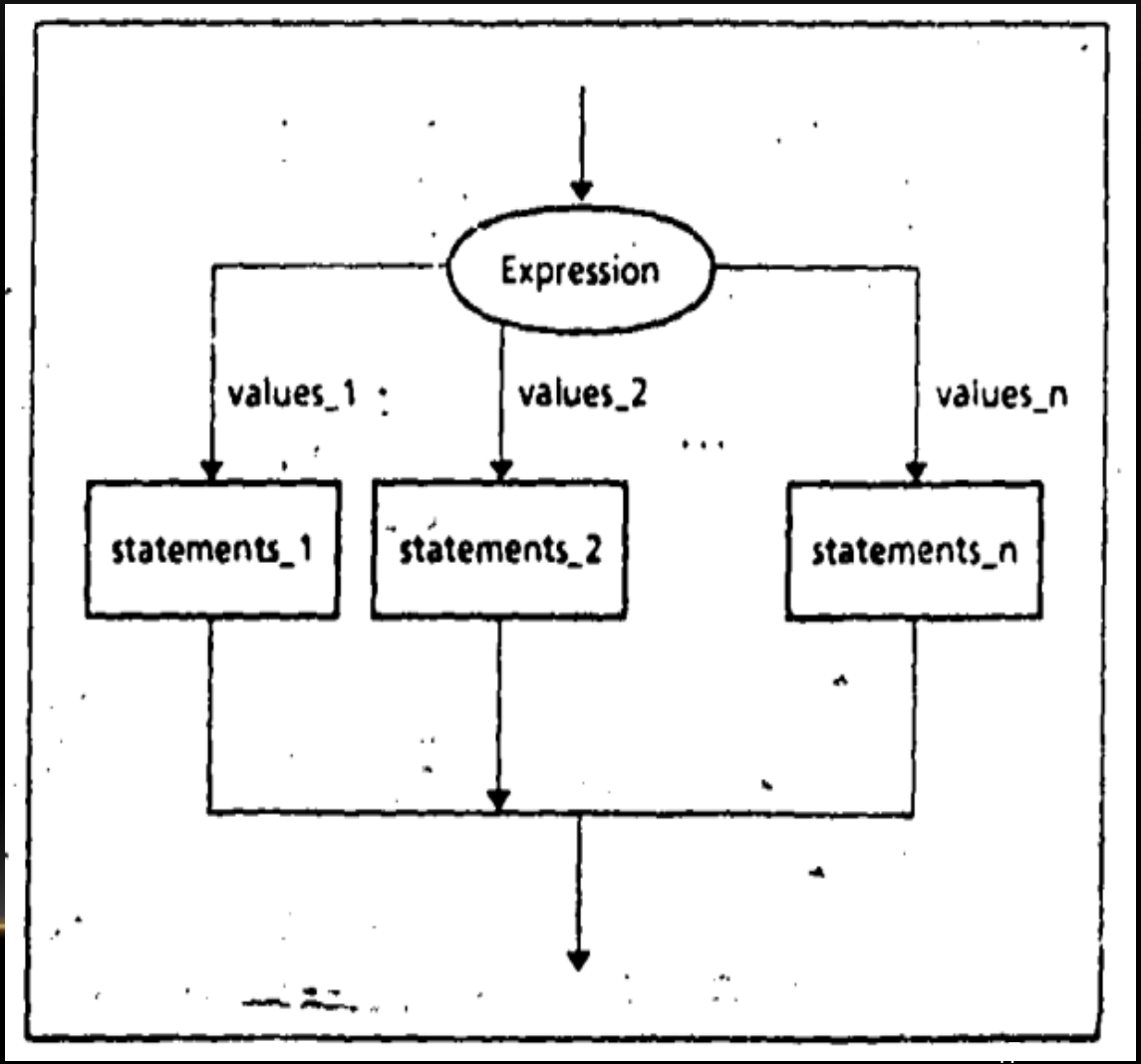
Solution:

```
IF AL <= BL
 THEN
  display the character in AL
 ELSE
  display the character in BL
END_IF
```

```
DISPLAY:
                INT 21h         ;display it
END_IF
```

CASE (CONTROL FLOW)

```
CASE expression
   values_1: statements_1
   values_2: statements_2
   .
   .
   .
   values_n: statements_n
END_CASE
```



CASE (EXAMPLE)

Example 6.4 If AX contains a negative number, put -1 in BX; if AX contains 0, put 0 in BX; if AX contains a positive number, put 1 in BX.

```
;case AX
                CMP AX,0        ;test ax
                JL  NEGATIVE    ;AX < 0
                JE  ZERO        ;AX = 0
                JG  POSITIVE    ;AX > 0
NEGATIVE:
                MOV  BX,-1      ;put -1 in BX
                JMP  END_CASE   ;and exit
ZERO:
                MOV  BX,0       ;put 0 in BX
                JMP  END_CASE   ;and exit
POSITIVE:
                MOV  BX,1       ;put 1 in BX
END_CASE:
```

Solution:

```
CASE AX
   <0: put -1 in BX
   =0: put 0 in BX
   >0: put 1 in BX
END_CASE
```

CASE (EXAMPLE)

Example 6.5 If AL contains 1 or 3, display "o"; if AL contains 2 or 4, display "e".

```
;case AL
; 1,3:
              CMP   AL,1        ;AL = 1?
              JE    ODD         ;yes, display 'o'
              CMP   AL,3        ;AL = '3'
              JE    ODD         ;yes, display 'o'
; 2,4:
              CMP   AL,2        ;AL = 2?
              JE    EVEN        ;yes, display 'e'
              CMP   AL,4        ;AL = 4?
              JE    EVEN        ;yes, display 'e'
              JMP   END_CASE    ;not 1.,4
ODD:                            ;display 'o'
              MOV   DL,'o'      ;get 'o'
              JMP   DISPLAY     ;go to display
EVEN:                           ;display 'e'
              MOV   DL,'e'      ;get 'e'
```

Solution:

```
CASE AL
   1,3: display 'o'
   2,4: display 'e'
END_CASE
```

```
DISPLAY:
              MOV   AH,2
              INT   21H         ;display char
END_CASE:
```

WARM UP PROBLEM (ADDER/SUBTRACTOR)

- Input three integer values x,y,z from user ,
  - where 0 <= x,y <= 4, x>y, and
  - z = 0 or 1.

If z=0 print x+y (ADD X,Y)

and if z = 1 print x-y (SUB X,Y) in the console.

PROBLEM (MAXIMUM OR MINIMUM VALUE)

- Input four integer values x,y,C from user where 0 <= x,y <= 9, and C = 0 or 1.
- If C=0 print the minimum of x,y and
- if C = 1 print the maximum of x,y in the console.

WARM UP PROBLEM (MULTIPLE CONDITIONS/CASES)

- Input two integer values x, y where 0<=x<=3.
- if y == 1 then print x+0,
- if y == 2 then print x+2,
- if y == 3 then print x+4,
- if y == 4 then print x+6,
- Else Print “invalid input”

BRANCHES WITH COMPOUND CONDITIONS

- Condition_1 AND Condition_2
- Condition_1 OR Condition_2

- If(CGPA>=2.0 && CGPA<=4.0)

{

PRINT(“VALID”)

}

- ELSE PRINT(“INVALID”)

CONDITION_1 AND CONDITION_2 (EXAMPLE)

Example 6.6 Read a character, and if it's an uppercase letter, display it.

Solution:

```
Read a character (into AL)
IF ('A' <= character) and (character <= 'Z')
 THEN
  display character
END_IF
```

CONDITION_1 AND CONDITION_2 (EXAMPLE)

Example 6.6 Read a character, and if it's an uppercase letter, display it.

```
;read a character
            MOV  AH,1       ;prepare to read
            INT  21H        ;char in AL
;if ('A' <= char) and (char <= 'Z')
            CMP  AL,'A'     ;char >= 'A'?
            JNGE END_IF     ;no, exit
            CMP  AL,'Z'     ;char <= 'Z'?
            JNLE END_IF     ;no, exit
;then display char
            MOV  DL,AL      ;get char
            MOV  AH,2       ;prepare to display
            INT  21H        ;display char
END_IF:
```

CONDITION_1 AND CONDITION_2 (EXAMPLE)

Example 6.7 Read a character. If it's "y" or "Y", display it; otherwise, terminate the program.

Solution:

```
Read a character (into AL)
IF (character = 'y') OR (character = 'Y')
 THEN
  display it
 ELSE
  terminate the program
END_IF
```

CONDITION_1 OR CONDITION_2 (EXAMPLE)

Example 6.7 Read a character. If it's "y" or "Y", display it; otherwise, terminate the program.

```
;read a character
                MOV  AH,1       ;prepare to read
                INT  21H        ;char in AL
;if (character = 'y') or (character = 'Y')
                CMP  AL,'y'     ;char = 'y'?
                JE   THEN       ;yes, go to display it
                CMP  AL,'Y'     ;char = 'Y'?
                JE   THEN       ;yes, go to display it
                JMP  ELSE_      ;no, terminate
THEN:
                MOV  AH,2       ;prepare to display
                MOV  DL,AL      ;get char
                INT  21H        ;display it
                JMP  END_IF     ;and exit
ELSE_:
                MOV  AH,4CH
                INT  21H        ;DOS exit
END_IF:
```

LOOP INSTRUCTION

- Loop destination_label

```
                ;initialize CX to loop_count
TOP:
                ;body of the loop
                LOOP TOP
```

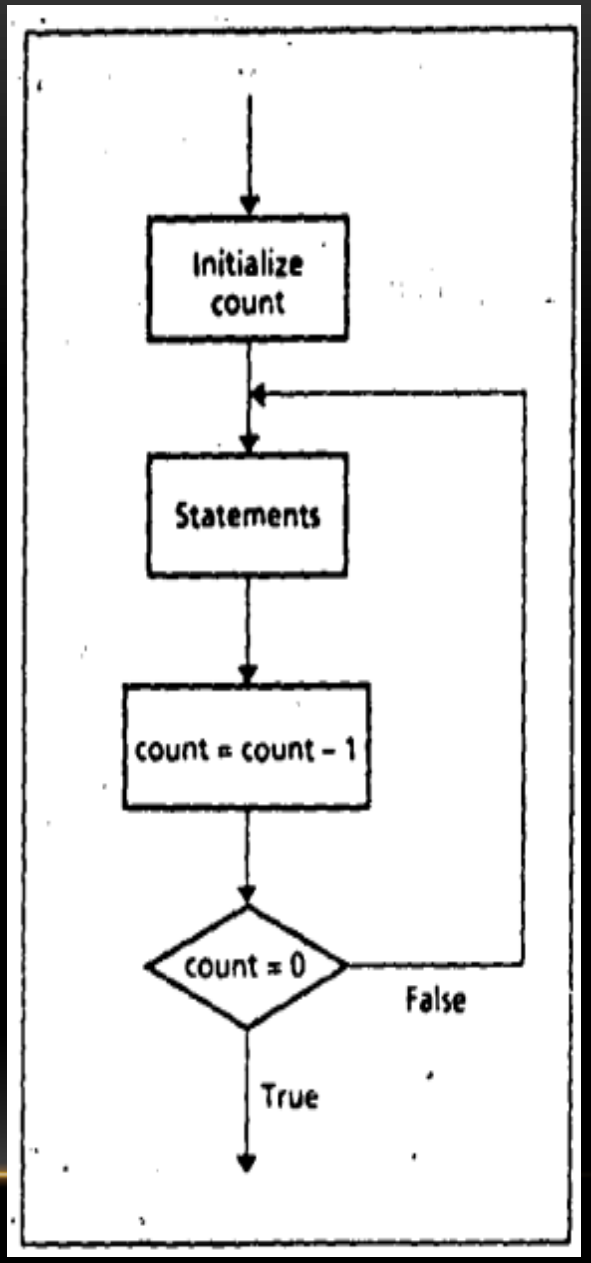
LOOP STRUCTURE

Three types of loops (Unconditional loop, entry controlled conditional Loop, exit controlled conditional loop)

- 1. For Loop -> Loop Instruction- Unconditional Loop
- 2. While Loop -> Jump Instruction – Entry Controlled Loop
- 3. Do While Loop(Repeat Loop) –> Jump Instruction – Exit Controlled Loop

FOR LOOP

```
FOR  loop_count  times  DO
   statements
END_FOR
```



FOR LOOP

Example 6.8 Write a count-controlled loop to display a row of 80 stars.

Solution:

```
FOR 80 times DO
            display '*'
END_FOR
```

```
            MOV  CX,80      ;number of stars to display
            MOV  AH,2       ;display character function
            MOV  DL,'*'     ;character to display
TOP:
            INT  21h        ;display a star
            LOOP TOP        ;repeat 80 times
```

PROBLEM WITH LOOP INSTRUCTION

- Loop instruction must execute to stop the looping
- If CX = 0h then, Loop instruction will decrease CX to FFFFh
- Which will eventually loop the code section for FFFFh more time

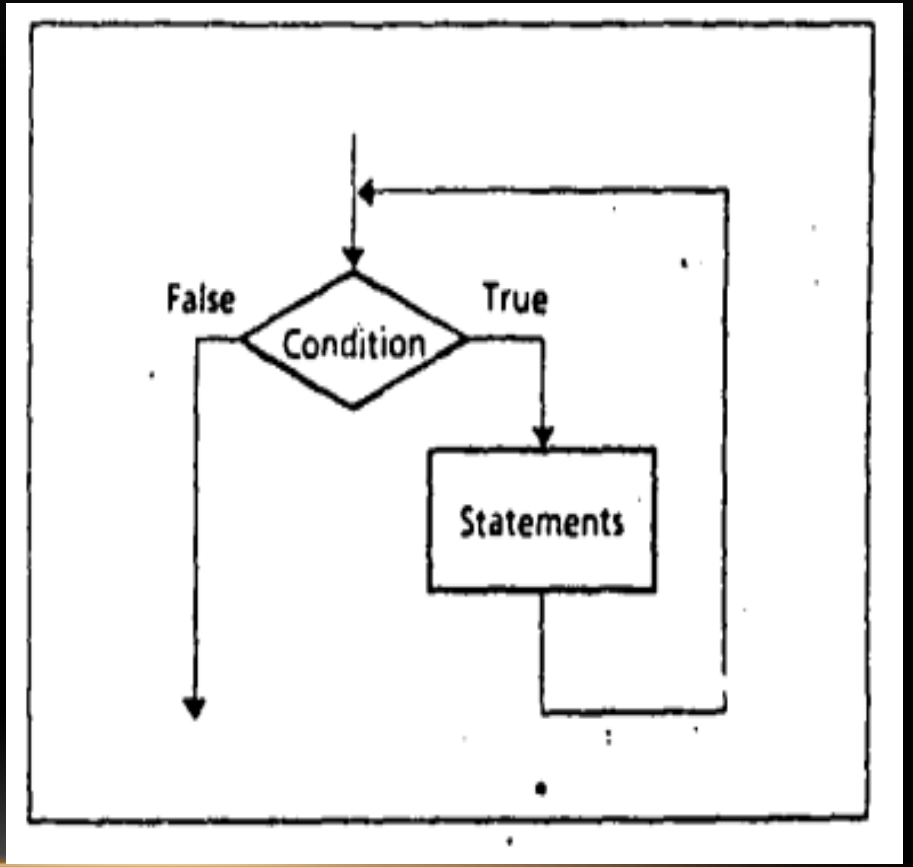
SOLUTION OF LOOP INSTRUCTION PROBLEM

- JCXZ (Jump if CX is Zero)
- JCXZ destination_label

```
        JCXZ  SKIP
TOP:
        ;body of the loop
        LOOP  TOP
SKIP:
```

WHILE LOOP (ENTRY CONTROLLED)

```
WHILE condition DO
  statements
END_WHILE
```



WHILE LOOP (EXAMPLE)

Example 6.9 Write some code to count the number of chara ters in : n input line.

Solution:

```
initialize count to 0
read a character
WHILE character <> carriage_return DO
 count = count + 1
 read a character
END_WHILE
```

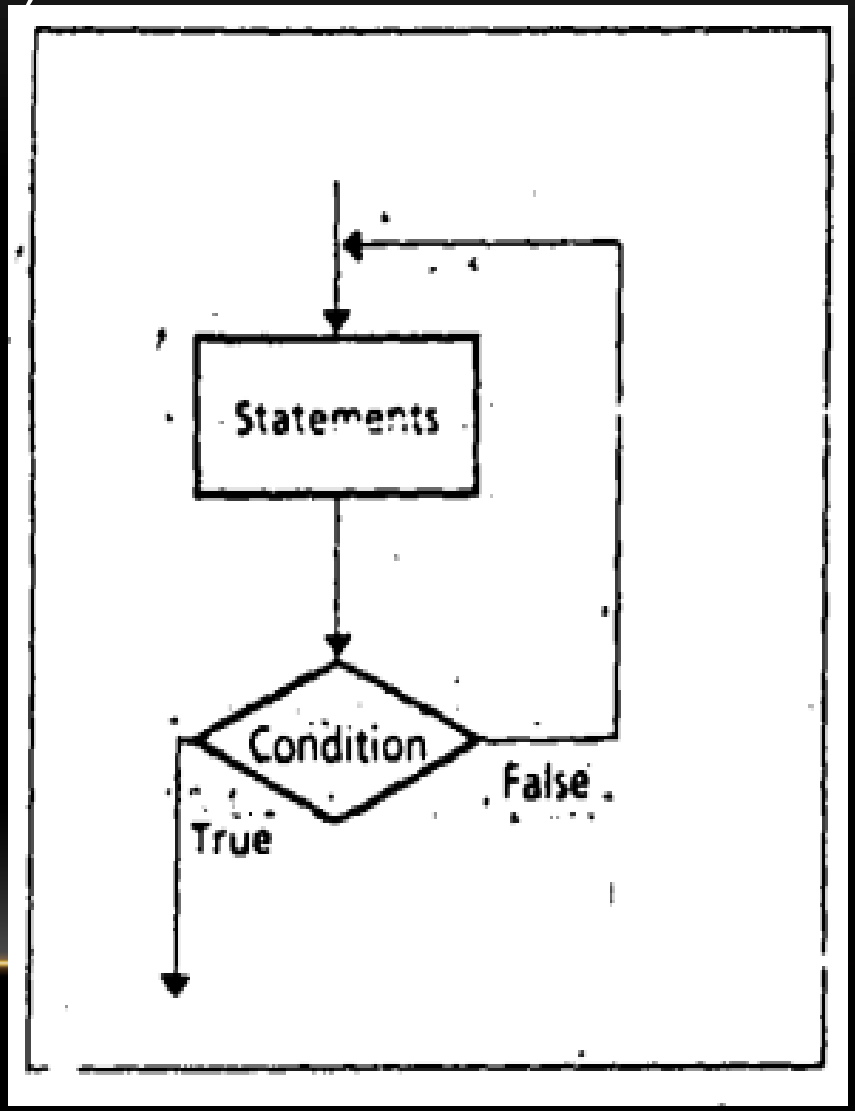
WHILE LOOP (EXAMPLE)

Example 6.9 Write some code to count the number of characters in an input line.

```
                MOV   DX,0        ;DX counts characters
                MOV   AH,1        ;prepare to read
                INT   21H         ;character in AL
WHILE_:
                CMP   AL,0DH      ;CR?
                JE    END_WHILE   ;yes, exit
                INC   DX          ;not CR, increment count
                INT   21H         ;read a character
                JMP   WHILE_      ;loop back
END_WHILE:
```

REPEAT LOOP(DO WHILE)

```
REPEAT
statements
UNTIL condition
```



REPEAT LOOP(DO WHILE - EXAMPLE)

Example 6.10 Write some code to read characters until a blank is read.

Solution:

```
REPEAT
 read a character
UNTIL character is a blank
```

```
                MOV   AH,1        ;prepare to read
REPEAT:
                INT   21H         ;char in AL
;until
                CMP   AL,' '      ;a blank?
                JNE   REPEAT      ;no, keep reading
```