# Dhaka University of Engineering & Technology (DUET), Gazipur

Department of Computer Science and Engineering (CSE)

**Course Title:** Microprocessor and Interfacing Sessional (CSE 3812)

**Lab # 04**

*Understanding* ***Branching Structure, Branching with Compound Condition, and Looping Structure of 8086*** *using Assembly Language.*

# Theory:

# IF-THEN: The If-then structure may be expressed as follows.

# If condition is true

# Then

# Execute true-branch statements

# End-if

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

# 

|  |  |
| --- | --- |
| If AX<0ThenReplace AX by –AXEnd-if | CMP AX, 0JNL END\_IFNEG AXEND\_IF: EXIT |

# IF-THEN-ELSE: The If-then-else structure may be expressed as follows.

# If condition is true

# Then

# Execute true-branch statements

# Else

# Execute false-branch statements

# End-if

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

|  |  |
| --- | --- |
| If AL< BLThenDisplay the character in ALElseDisplay the character in BLEnd-if | CMP AL, BLJNBE ElseMOV DL, ALJMP DisplayElse: MOV DL, BLDisplay: MOV AH, 2INT 21h |

# CASE: The Case structure may be expressed as follows.

# CASE expression

# Value-1: Statements-1

# Value-2: Statements-2

# Value-3: Statements-3

# .

# .

# .

# Value-n: Statements-n

# End-CASE

# 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<0: Put -1 in BX=0: Put 0 in BX>0: Put 1 in BXEnd-CASE | CMP AX, 0JL NegativeJE ZeroJG PositiveNegative: MOV Bx, -1JMP END\_CASEZero: MOV Bx, 0JMP END\_CASEPositive: MOV Bx, 1JMP END\_CASEEND\_CASE: |

# AND Condition: An AND condition is true if and only if condition-1 and condition-2 are both true.

# Example 6.6: Read a character, and if it’s an uppercase letter, Display it.

|  |  |
| --- | --- |
| Read a character (into AL)If (‘A’<= character) and (character <= ‘Z’)ThenDisplay the characterEnd-if | MOV AH, 1INT 21HCMP AL, ‘A’JNGE End-ifCMP AL, ‘Z’JNLE End-ifMOV DL, ALMOV AH, 2INT 21hEND\_if: EXIT |

# OR Condition: An AND condition is true if condition-1 OR condition-2 is true.

# Example 6.7: Read a character, and if it’s ‘y’ OR ‘Y’ letter, Display it; otherwise, terminate the program.

|  |  |
| --- | --- |
| Read a character (into AL)If (character = ‘y’) OR (character <= “Y”)ThenDisplay the characterEnd-if | MOV AH, 1INT 21HCMP AL, ‘y’JE ThenCMP AL, ‘Y’JE ThenJMP ExitThen: MOV DL, ALMOV AH, 2INT 21hEND\_if: EXIT |

# LOOP Structure:

# For Loop: This is a loop structure in which the loop statements are repeated a known number of times.

# The LOOP instruction can be used to implement a FOR loop, it has the form

# LOOP destination\_label

# The counter for the loop is the register CX which is initialized to loop\_count. Execution of the LOOP instruction causes CX to be decremented automatically. And if CX is not 0, control transfers to destination\_label. If CX=0, the next instruction after LOOP is done.

# Structure:

# Initialize CX to Loop-COUNT

# Top:

# ; Body of the loop

# LOOP Top

# 

# Example 6.8: Write a counter-controlled loop to display a row of 80 stars.

|  |  |
| --- | --- |
| For 80 times DODisplay ‘\*’End-for | MOV CX, 80MOV AH, 2MOV DL, ‘\*’;JCXZ SkipTOP: INT 21hLOOP TOPSkip: EXIT |

# While Loop: This is a loop depending on a condition.

# WHILE condition DO

# Statements

# END\_ WHILE

# The condition is checked at the top of the loop. If true, the statements executed, if false, the program goes on to whether follows. It is possible the condition will be false initially, in which case the loop body is not executed at all. The loop is executed as long as the condition is true.

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

|  |  |
| --- | --- |
| Initialize count to 0Read a characterWhile character <> carriage-return DOCount=count+1Read a characterEnd-WHILE | MOV DX, 0MOV AH, 1INT 21hWHILE: CMP AL, 0dhJE END\_ WHILEINC DXINT 21hJMP WHILEEND\_ WHILE : EXIT |

Note that because the WHILE loop checks the terminating condition at the top of the loop, you must make sure that any variables involved in the condition are initialized before the loop is entered. So you read a character before entering the loop, and read another one at the bottom.

**REPEAT LOOP:** Another conditional loop is the REPEAT LOOP***.***

REPEAT

statements

UNTIL condition

In a REPEAT .. , UNTIL loop, the statements are executed, and then the condition is checked. If true, the loop terminates; if false, control branches to the top of the loop.

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

|  |  |
| --- | --- |
| REPEATread a characterUNTIL character is a blank | MOV AH, 1REPEAT:INT 21hCMP AL, ‘ ’JNE REPEAT |

# Tasks to do:

1. Write an assembly language code to derive the final value of the number sequence 1+2+3+4+…..+N. (**Using Loop**). Take the input value of N (in between 2 to 9) as a single ASCII character and then adjust it to the actual decimal value in your program. Finally, store the output in a variable named RESULT. You do not need to display the output in the console.
2. Write an assembly language code to derive the final value of the number sequence 12+22+32+42+…..+N2. (**Using Loop**). Take the input value of N (in between 2 to 9) as a single ASCII character and then adjust it to the actual decimal value in your program. Finally, store the output in a variable named RESULT. You do not need to display the output in the console.
3. Write a program that will prompt the user to enter a hex digit character ("0"· ... "9" or "A" ... "F"), display it on the next line in decimal, and ask the user if he or she wants to do it again. If the user types "y" or "Y", the program repeats; If the user types anything else, the program terminates. If the user enters an illegal character, prompt the user to try again.

ENTER A HEX DIGIT: 9

IN DECIMAL IS IT 9

DO YOU WANT TO DO IT AGAIN? y

ENTER A HEX DIGIT: c

ILLEGAL CHARACTER - ENTER 0 .. 9 OR A .. F: C

IN DECIMAL IT IS 12

DO YOU WANT TO DO IT AGAIN? N

1. Write a program that reads a string of capital letters, ending with a carriage return, and displays the longest sequence of consecutive alphabetically increasing capital letters read.

ENTER A STKING OF CAPITAL LETTERS:

FGHADEFGHC

THE LONGEST CONSECUTIVELY INCREASING STRING IS:

DEFGH

Google Classroom Code: maqup5f