**Computer Organization & Assembly Language**

**Lab 11**

**Topics:**

* Interrupts

**Interrupt**

* An interrupt is an event that causes the processor to suspend its present task and transfer control to a new program called the interrupt service routine (ISR).
* Three sources of interrupts
* Software interrupts
* Hardware interrupts
* Processor interrupts

**Common Software Interrupts:**

The Common Interrupts are (according to Irvine32 library):

* ReadInt
* ReadStr
* WriteInt
* WriteStr
* Check Irvine [documentation](https://csc.csudh.edu/mmccullough/asm/help/) for more information

1. **Procedures (uses keyword):**

Procedures can also be used in a nested fashion. Any procedure can be called inside any other procedure. In this way the stack will save all the values of instruction pointers and are retrieved in the way of last in first out.

Keyword uses can be used if a register is to be used inside a procedure.

Example:

Array\_Sum proc uses si, cx

<instructions>

ret

Array\_Sum endp

1. **Macros:**

* Macros are just like procedures, but they exist only until your code is compiled, after compilation all macros are replaced with real instructions.
* Macro Definition
  + name MACRO [parameters,...]
  + <instructions>
  + ENDM

**Using Macros:**

* When you want to use a macro, you can just type its name. For example:
  + MyMacro
* Macro is expanded directly in program's code. So if you use the same macro 100 times, the compiler expands the macro 100 times, making the output executable file larger and larger, each time all instructions of a macro are inserted.

**Passing Arguments to Macro:**

* To pass parameters to macro, you can just type them after the macro name. For example:
  + MyMacro 1, 2, 3
* To mark the end of the macro ENDM directive is enough

Example:

* Unlike procedures, macros should be defined above the code that uses it.
* For Example
  + .code
  + mymacro macro p1,p2,p3
  + mov ax,p1
  + mov bx,p2
  + mov cx,p3
  + endm
  + main proc
  + mymacro 1,2,3
  + mov ah,4ch
  + int 21h
  + main endp
  + end

**Defining Macros in Separate file:**

* To define Macros in Separate file;
  + Open your assembler Source Directory
    - C:\masm615\include\
  + Create a File named “mymacros.inc”
  + Write your Macro in this file and save. Make sure your file have extension .inc
  + Include this file in your source program (\*.asm), by writing below line on top of your code
    - include mymacros.asm

Compile your Code.

**TASK**

1. Write an assembly program that checks if a number is prime or not
2. Write an assembly program which checks if a string is palindrome or not
3. Write an assembly program that checks if a number is multiple of 5
4. Write an assembly program that counts the number of vowels and consonants in a string,
5. Write an assembly program that takes 10 numbers as an input in array from user and then does the following:
   1. Calculate array sum
   2. Calculate mean
   3. Calculate mode (most occurring number of array)
6. Write an assembly program to print the following pattern for any generic input. (The pattern given below is for n=5)

****

**Submission Instructions:**

* Write your name, roll no and section on top of your code file
* Do all of your code work in procedures and just call that procedure in main to execute the code. You will make 7 different procedures for each question.
  + Eg:

Q1 proc

;code

ret

Q1 endP

* **Self-Evaluation: (bonus 2 marks)**

You are to self-evaluate all the questions you do. Write a one-liner comment for each question on top of code file below your name.

Eg. Q1, all done, works completely fine for all use cases

Q2, partially done, exception thrown error

You must be honest in self-evaluating yourself.

**If during evaluation you are found being dishonest, your lab will be marked zero.**

* Submit only one .asm file (Format: i22-1234\_LAB11.asm