Lab Manual for Computer Organization and Assembly Language

Lab-12

Procedures

Table of Contents

1. Introduction

Large problems can be divided into smaller tasks to make them more manageable • A procedure is the ASM equivalent of a Java or C++ function. The CALL instruction calls a procedure. The RET instruction returns from a procedure.

2. Objective

- To know more about Assembly language, such as how to repeat a block of statements using Loop Instructions.

3. Concept Map

This section provides you the overview of the concepts that will be discussed and implemented in this lab.

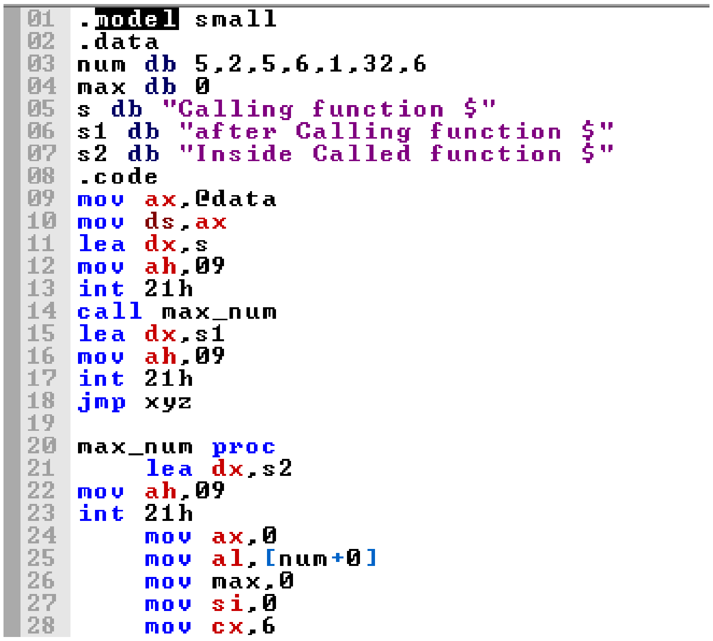
3.1 Procedure Instruction

The following instructions can be used for implementing procedures. For example, the following code snippet can be used for writing the procedure name **sample**.

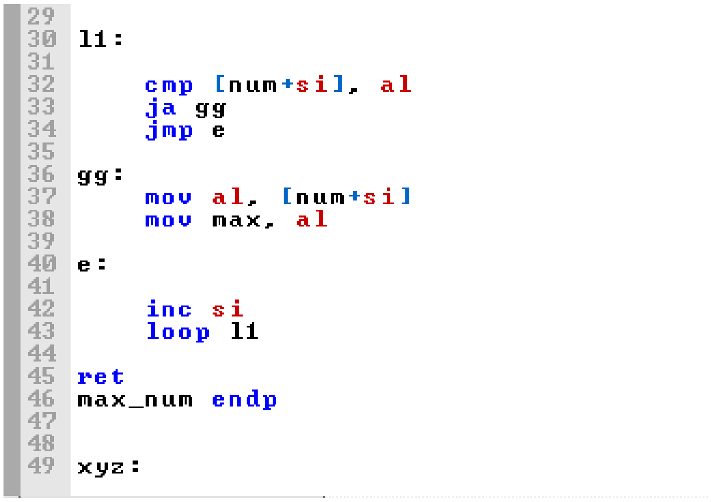
```
sample PROC

        function body here

ret

sample ENDP
```

4. Walkthrough Task

The following program find the maximum number.

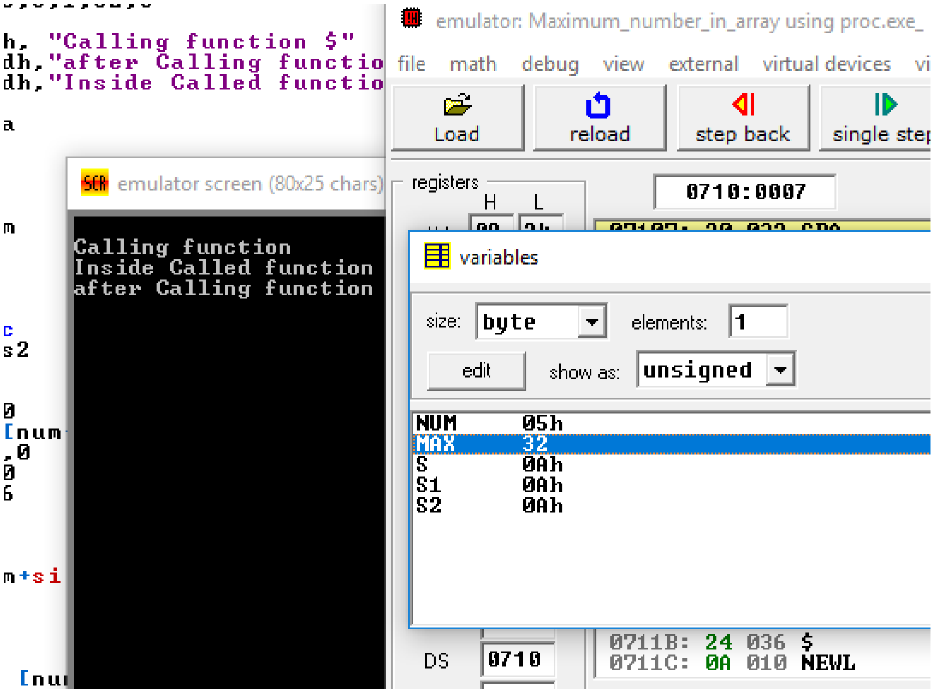


```
.model small
.data
num db 5,2,5,6,1,32,6
max db 0
s db "Calling function $"
s1 db "after Calling function $"
s2 db "Inside Called function $"
.code
mov ax,@data
mov ds,ax
lea dx,s
mov ah,09
int 21h
call max_num
lea dx,s1
mov ah,09
int 21h
jmp xyz

max_num proc
    lea dx,s2
mov ah,09
int 21h
    mov ax,0
    mov al,[num+0]
    mov max,0
    mov si,0
    mov cx,6
```



```
l1:

    cmp [num+si], al
    ja gg
    jmp e

gg:
    mov al, [num+si]
    mov max, al

e:

    inc si
    loop l1

ret
max_num endp


xyz:
```

When the above code is compiled and executed, it produces the following output.



```
h, "Calling function $"
dh,"after Calling functio
dh,"Inside Called functio
```

emulator: Maximum_number_in_array using proc.exe_

emulator screen (80x25 chars)

```
Calling function
Inside Called function
after Calling function
```

0710:0007

variables

size: byte   elements: 1

show as: unsigned

```
NUM      05h
MAX      32
S        0Ah
S1       0Ah
S2       0Ah
```

DS 0710

```
0711B: 24 036 $
0711C: 0A 010 NEWL
```

5. Procedure& Tools

In this section you will study how to setup and MASM Assembler.

5.1 Tools

- Download emu 8086 from (http://www.emu8086.com/files/emu8086v408r11.zip)
- Just extract the emu8086.15.zip on C
- Install emu8086

6. Practice Tasks

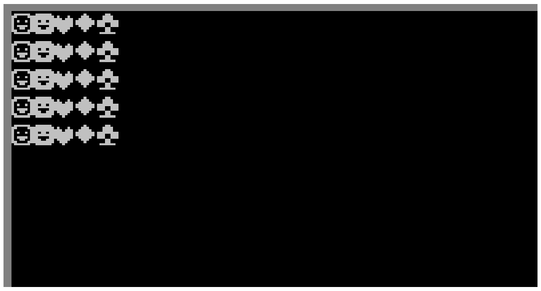
This section will provide more practice exercises which you need to finish during the lab. You need to finish the tasks in the required time. When you finish them, put these tasks in the following folder:
\\fs\assignments$\

6.1 Practice Task 1 [Expected time = 15mins]

Write an assembly code that arrange the numbers in array in ascending order using procedure.

6.2 Practice Task 2 [Expected time = 15mins]

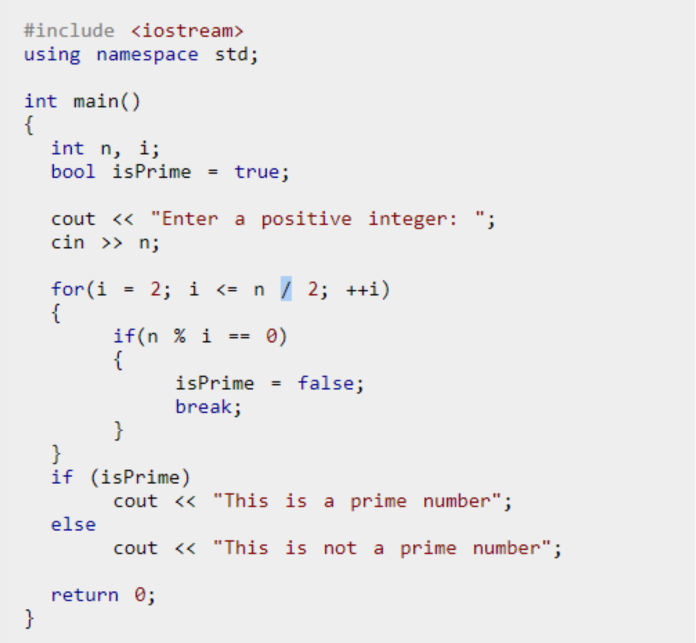
Write an assembly language program using the Loop instruction to print following pattern by using procedure.



```
☺☻♥♦♣
☺☻♥♦♣
☺☻♥♦♣
☺☻♥♦♣
☺☻♥♦♣
```

6.3 Practice Task 3 [Expected time = 15mins]

Convert the following C++ code into assembly using procedure



```
#include <iostream>
using namespace std;

int main()
{
   int n, i;
   bool isPrime = true;

   cout << "Enter a positive integer: ";
   cin >> n;

   for(i = 2; i <= n / 2; ++i)
   {
        if(n % i == 0)
        {
             isPrime = false;
             break;
        }
   }
   if (isPrime)
        cout << "This is a prime number";
   else
        cout << "This is not a prime number";

   return 0;
}
```

6.4 Out comes

After completing this lab, student will be able to setup emu 8086. He/ She will also be able to compile and run basic Assembly programs.

7. Evaluation Task (Unseen) [Expected time = 30mins for tasks]

The lab instructor will give you unseen task depending upon the progress of the class.

8. Evaluation criteria

The evaluation criteria for this lab will be based on the completion of the following tasks. Each task is assigned the marks percentage which will be evaluated by the instructor in the lab whether the student has finished the complete/partial task(s).

Table 3: Evaluation of the Lab

Sr. No.	Task No	Description	Marks
1	4	Problem Modeling	20
2	6	Procedures and Tools	10
3	7	Practice tasks and Testing	35
4	8	Evaluation Tasks (Unseen)	20
5		Comments	5
6		Good Programming Practices	10

9. Further Reading

This section provides the references to further polish your skills.

9.1 Slides

The slides and reading material can be accessed from the folder of the class instructor available at \\fs\lectures$