HOMEWORK 3: ASSEMBLY LANGUAGE

Module: Informational technologies FMISB18100

Vilnius Gediminas Technical University

Department of Information Systems

Lecturer: Dr. D.Zabulionis

Liene Malkalne, group 2

liene.malkalne@gmail.com

2022

Contents

Assignment

Homework 3. Student: *LIENE MALKALNE* **group:** *2 group* 1

Module: Informational technologies FMISB18100; Date: Wednesday 23rd February, 2022

Vilnius Gediminas Technical University
Department of Information Systems

Assignment of homework 3, Variant 2: Programming with the Assembly language

The main problem

Write the working C++ program with the embedded inline Assembly language code. In the Assembly code there must be implemented code to evaluate the functions f_1 and f_2 given in Eqs. 1 and 2 respectively. In the program, the initial values of the independent variable x can be input by keyboard or can be hard-coded while the result of the calculation has to be printed by using the method `printf()`. The output of the results must also include the initial values of thee independent variable x. For example:

```
The value of the function f1 at x = xxx is: f(xxx) = XXX.
```

Where `xxx` and `XXX` stand for the numbers dependent on the input and the evaluated values of f_1 and f_2

The programs for the functions f_1 and f_2 can be written in separate files (projects).

$$f_1(x) = \begin{cases} 50 - 4x, & \text{when } x < 9 \\ \frac{(x-5)^3}{16}, & \text{when } x = 9 \\ 3x - (2x). & \text{when } x > 9 \end{cases} \quad (1)$$

where $x \in \mathbb{Z}$, $\mathbb{Z} = (\infty, \ldots, -2, -1, 0, 1, 2, \ldots \infty)$ is set of integers.

$$f_2(x) = \sum_{i=-10}^{x} ((2i-3)^2 - i), \text{ when } x \in \{-5, ..., 10\} \subset \mathbb{Z} \quad (2)$$

Where $x \in \mathbb{Z}$, $\mathbb{Z}$ is the set of integers.

The homework must contain

(I) Code of the program that calculates Eq. (1)

(II) Code of the program that calculates Eq. (2)

(III) Report on the programs with explanations and examples of usage of these programs

Remarks.

The report of the homework must contain: the title page, assignment, abstract page, table of contents, the main text of the homework whose chapters and numbering of the chapters correspond to the above listed items, the list of the references (if literature were used)

All calculations, derivations of formulas assumptions etc. must be commented in the text of the report; variables of the formulas must be explained when they are used for the first time in the text; figures and tables must be labeled and numbered in the text of the report.

Exhibit1: The assignment of the homework

Abstract

The purpose of this homework is to familiarize with assembly language and write assembly language program to solve two functions. This work includes the assembly language code, comments, and explanation of the code step by step. This homework includes images of code snippets and assignments.

This work contains 11 pages.

References:

- Lecture notes and recordings
- To test the result: https://www.mathsisfun.com/numbers/sigma-calculator.html

PART I: Code of the program that calculates $f_1(x)$

$$f_1(x) = \begin{cases} 50 - 4x, & \text{when } x < 9 \\ \frac{(x-5)^3}{16}, & \text{when } x = 9 \\ 3x - (2x). & \text{when } x > 9 \end{cases} \quad (1)$$

where $x \in \mathbb{Z}$, $\mathbb{Z} = (\infty, \ldots, -2, -1, 0, 1, 2, \ldots \infty)$ is set of integers.

Exhibit2: Calculation task Nr1

The Code for task 1

```
/*
This is an Assembly language program to calculate the value of the function:
f(x) = 50 - 4x      ,when x < 9
f(x) = ((x-5)^3)/16 ,when x = 9
f(x) = 3x - (2x)    ,when x > 9
*/

#include <iostream>
using namespace std;

int x, rez;

int main()
{
    printf("Enter the value of variable x: ");
    cin >> x;

    printf("\n--------------------Results of the calculation--------------------\r\n");
    printf("\nWhen x < 9, the arithmetical operation is f(x) = 50 - 4x");
    printf("\nWhen x = 9, the arithmetical operation is f(x) = ((x-5)^3)/16");
    printf("\nWhen x > 9, the arithmetical operation is f(x) = 3x - (2x)\r\n");


    __asm
    {
        mov eax, x      // move the value of x in eax

        cmp eax, 9      // 9 is the breaking point for this calculation
        jl less9
        je equal9
        jg more9

    less9:              // 50-4x
        imul eax, -4    // x * -4
        add eax, 50     // add 50
        jmp endIfThenStatement

    equal9:             // ((x-5)^3)/16
        sub eax, 5      // x - 5
        mov ebx, eax    // move the result of eax in ebx for power operation
```

```
        imul eax, ebx   // (x - 5) * (x - 5)
        imul eax, ebx   // ((x - 5) * (x - 5)) * (x - 5)
        mov ecx, 16
        cdq;            // clear the register
        idiv ecx        // divide eax with 16
        jmp endIfThenStatement

    more9:              // 3x-2x
        mov ebx, eax    // eax = x
        imul eax, 3     // 3x
        imul ebx, 2     // 2x
        sub eax, ebx    // 3x-2x
        jmp endIfThenStatement

    endIfThenStatement:
        mov rez, eax    // put the result in rez

    }


    printf("\nThe value entered for variable x: %d\n", x);
    printf("\nResult of the arithmetical operation: f(%d) = %d\r\n", x, rez);
}
```

PART II: Code of the program that calculates $f_2(x)$

$$f_2(x) = \sum_{i=-10}^{x} ((2i-3)^2 - i), \text{ when } x \in \{-5, ..., 10\} \subset \mathbb{Z} \quad (2)$$

Where $x \in \mathbb{Z}$, $\mathbb{Z}$ is the set of integers.

Exhibit3: Calculation task Nr2

The Code for task 2

```
/*
This is an Assembly language program to calculate the value of the function:
sigma(x) (i = -10) : (2i - 3)^2 - i, where -5 <= x <= 10
*/

#include <iostream>
using namespace std;

int x, rez, i = -10;

int main()
{
    printf("Enter the value of variable x: ");
    cin >> x;
    printf("\n-------Result of the calculation-------\r\n");
    if ((x >= -5) && (x <= 10)) {
        cout << "\r\nThe number you entered is: " << x << endl;
    }
    else {
        cout << "\r\nWrong value. Enter value from -5 to 10" << endl;
    }

    __asm
    {
        mov eax, x
        cmp eax, -5          // compare x to -5
        jl invalid_value     // if x < -5 it is ivalid
        cmp eax, 10          // compare x to 10
        jg invalid_value     // if x > 10 it is invalid
        invalid_value:
            mov rez, 0       // display 0 as result

        mov eax, 0           // clear the eax registry
        mov ecx, i           // place the i value in the ecx registry
        mov ebx, 0           // assign initial value for ebx, this will be result

        while_loop:
            mov eax, ecx     // move i to eax for calculations
            imul eax, 2      // 2*i
            sub eax, 3       // 2i - 3
            mul eax          // (2i - 3)^2
            sub eax, ecx     // (2i - 3)^2 - i

            add ebx, eax     // place the result in ebx
            inc ecx          // increment i by 1 = i++
```

```
            cmp ecx , x     // compare i to x
            jg exit_loop    // go to result if i > x
            jmp while_loop  // continue the loop

        exit_loop:
            mov rez, ebx
    }

    printf("\r\nResult: %d\n", rez);
}
```

PART III: Report on the program with explanations and examples of the programs

For both tasks user input of the value of the variable x is allowed. It is possible to hardcode the value as well.

Calculation of $f_1(x)$

The calculations are performed by determining if the value is less, equal, or greater than 9, considering the value that is assigned to the variable x. At first the user is asked to enter the value of x in the console, then the arithmetical operations, which will be executed based on said value, are shown. Then the assembler language program is initialized and comments inside the program explain what is achieved in each step. At the end the is ready to be printed. Finally, the result of the calculation is printed in the console.

```
Enter the value of variable x: 3


------------------Results of the calculation------------------


When x < 9, the arithmetical operation is f(x) = 50 - 4x
When x = 9, the arithmetical operation is f(x) = ((x-5)^3)/16
When x > 9, the arithmetical operation is f(x) = 3x - (2x)


The value entered for variable x: 3


Result of the arithmetical operation: f(3) = 38
```

Exhibit4: Code Snippet of the console output when value x<9

```
Enter the value of variable x: 9

------------------Results of the calculation------------------

When x < 9, the arithmetical operation is f(x) = 50 - 4x
When x = 9, the arithmetical operation is f(x) = ((x-5)^3)/16
When x > 9, the arithmetical operation is f(x) = 3x - (2x)

The value entered for variable x: 9

Result of the arithmetical operation: f(9) = 4
```

Exhibit5: Code Snippet of the console output when value x= 9

```
Enter the value of variable x: 38

------------------Results of the calculation------------------

When x < 9, the arithmetical operation is f(x) = 50 - 4x
When x = 9, the arithmetical operation is f(x) = ((x-5)^3)/16
When x > 9, the arithmetical operation is f(x) = 3x - (2x)

The value entered for variable x: 38

Result of the arithmetical operation: f(38) = 38
```

Exhibit6: Code Snippet of the console output when value x>9

Calculation of $f_2(x)$

For the second function the calculation is performed using a while loop, the calculation is done step by step and then i is increased and the loop continues until i=x.

NB! The loop works, and the correct result is achieved, however the lines 25-31 do not achieve the intended result and I was unable to find how to do it. My idea was to evaluate if -5<=x<=10 and if not assign it as invalid value and not enter the loop. My intended steps are described in the program.

```
23  __asm
24  {
25      mov eax, x
26      cmp eax, -5          // compare x to -5
27      jl invalid_value     // if x < -5 it is ivalid
28      cmp eax, 10          // compare x to 10
29      jg invalid_value     // if x > 10 it is invalid
30      invalid_value :
31          mov rez, -1      // display 0 as result
32
33      mov eax, 0           // clear the eax registry
34      mov ecx, i           // place the i value in the ecx registry
35      mov ebx, 0           // assign initial value for ebx, this will be result
36
37      while_loop :
38          mov eax, ecx     // move i to eax for calculations
39          imul eax, 2      // 2*i
40          sub eax, 3       // 2i - 3
41          mul eax          // (2i - 3)^2
42          sub eax, ecx     // (2i - 3)^2 - i
43
44          add ebx, eax     // place the result in ebx
45          inc ecx          // increment i by 1 = i++
46          cmp ecx, x       // compare i to x
47          jg exit_loop     // go to result if i > x
48          jmp while_loop   // continue the loop
49
50      exit_loop :
51          mov rez, ebx
52
53  }
```

Exhibit7: Code Snippet of assembler for $f_2(x)$

Here is an example of a result in the console:

```
Enter the value of variable x: 5


-------Result of the calculation-------


The number you entered is: 5


Result: 2424
```

Exhibit8: Code Snippet of $f_2(x)$ result