In this project, you are required to implement some procedures in MIPS assembly language. You will use SPIM simulator [1] to develop and test your code. There will be three questions in the project which are unrelated.

QUESTION 1. *(25 points)* In this question, you are required to implement a set of min-heap operations in MIPS assembly language.
Fundamental property of a min-heap is that *for every node i other than the root, A[parent(i)] ≤ A[i], that is, the value of a node is at most the value of its parent*. (For more information check a textbook on data-structures [2]).
The first element in the heap array A[0] is the size of the heap, i.e. the number of elements in the heap (i.e the integer "size") is in A[0]. The rest of the elements in A (A[1] to A[size]) are the values in the nodes of the heap. If a node is at the *k*th location in the array, its children will be at the *(2k)* and the*(2k+1)* locations of the array. The following figures gives three different min-heaps which have 6, 4 and 3 nodes.

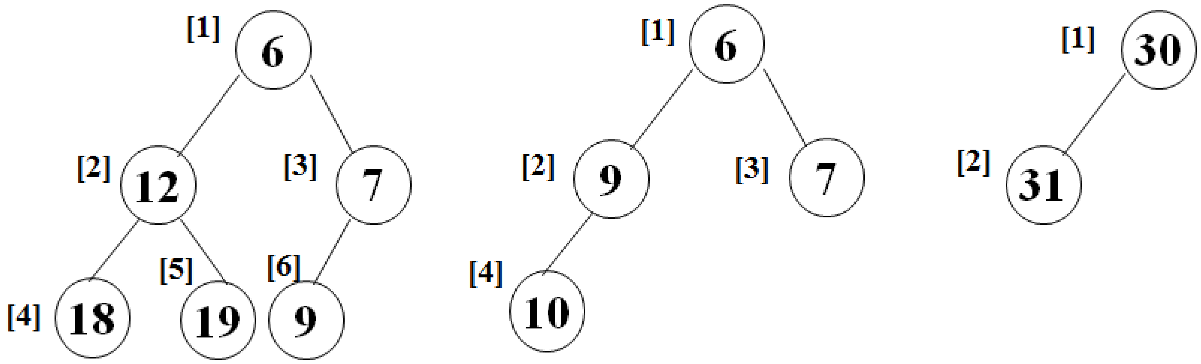

Figure 1.a

Figure 1.b

Figure 1.c

Figure 1 – Three sample min-heaps with a) 6 nodes, b) 4 nodes and c) 2 nodes.

Procedures of the Question 1
There will be three procedures for this question: **insert**, **delete**, and **print**.

a) insert (*value*, *heap*)
This procedure creates a new element (where the **value** is given in $a0 register) and insert it at the correct location in the min-heap by preserving heap-order property. The address of the place where the procedure should create the data structure is in **heap** (given in $a1). You should create a sufficient amount of memory for the heap. The register $v0 will contain the position number in the heap array A; i.e., it will be n, if A[n] is the location where the new value was inserted.

b) delete *(heap)*

This procedure deletes the minimum value (i.e., the root node in logical structure of the tree) from the min-heap, where the address of the first item of the data structure is in ***heap*** argument. You should do appropriate modifications so that resulting min-heap should preserve heap-order property for all existing nodes.

c) print (*heap*)

This procedure prints both the size (i.e., the number of integers) and the content of the min-heap structure on the screen. Each line of the output is for one level of the tree nodes stored in the array A. If there is no children (left or right), there will be “x” character to represent no node.

Example output of the print procedure for the min-heap structure represented in Figure 1.a is :

```
   The size of the heap is 6.
   [1] 6
   [2] 12 –  [3] 7
   [4] 18 –  [5] 19      [6] 9 – x
```

Example output of the same procedure for the min-heap structure represented in Figure 1.b is :

```
   The size of the heap is 4.
   [1] 6
   [2] 9 –  [3] 7
   [4] 10 –  x  x – x
```

QUESTION 2. *(10 points)* Write a MIPS program that takes an expression as a string, then it should calculate the actual result of the expression. The numbers might be multi-digit numbers and you should only consider the non-negative numbers (It should be considered that intermediate or final results might be negative!). You should consider mathematical operations for addition (+), subtraction (-), multiplication (*), and division (/). All the numbers can fit in a word!

An example run:

```
Enter the input string: ((7-(13+14))*(6-9)
The output is 60

Enter the input string: ((5*((105-2)+9))/(70-79))
The output is -62
```

QUESTION 3. *(12 points)* Write a MIPS program that takes an input string and an integer for the number of rows, then it should construct a 2D integer array in the memory. Then, several information about the 2D array will be given to the user based on the following arguments: a row index, and a column index. Firstly, it should give the beginning address of the 2D array and the memory address of the corresponding cell (specified with row and column indexes). You should not use **lw** instruction. You should return the address (in hexadecimal format) of the given array slot. Then, it should give the minimum value (MIN) and the maximum value (MAX) in the subarray Array[0][0] through Array[row][column].

For example:

```
Enter the input string: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
Enter the number of rows: 3
The 2D array is:
0   1  2  3  4
5   6  7  8  9
10 11 12 13 14

Enter the row index: 2
Enter the column index: 3
The beginning address of the 2D array is 0x100000c8
The memory address of the cell Array[2][3] is 0x100000fc
The MIN value between Array[0][0] and Array[2][3] is: 0
The MAX value between Array[0][0] and Array[2][3] is: 13
```

MENU *(8 points):* Your program should support a *Menu* including all questions above. A sample execution scenario given below:

```
Welcome to our MIPS project!
Main Menu:
1. Build a min-heap
2. Evaluate an expression
3. Construct a 2D array
4. Exit
Please select an option: 1
```

These options must be printed inside a loop until “Exit” option is selected.
When the user select option 1, you should print the followings:

```
The address of the heap: 0x10000000
Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option: 1
Please enter the value of item: 6
The value of 6 is placed at location 1 of the heap array!
```

```
Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option:1
Please enter the value of item: 9
The value of 9 is placed at location 2 of the heap array!

Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option:1
Please enter the value of item: 7
The value of 7 is placed at location 3 of the heap array!

Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option:1
Please enter the value of item: 10
The value of 10 is placed at location 4 of the heap array!


Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option:3
The size of the heap is 4.
[1] 6
[2] 9 -  [3] 7
[4] 10 -  x    x - x

Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option:2
The value of 6 is deleted from the heap!
```

```
Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option:3
The size of the heap is 3.
[1] 7
[2] 9 - [3] 10

Menu of Q1:
1. Insert an item to the heap
2. Delete an item from the heap
3. Print the heap
4. Return back
Please select an option:4

Main Menu:
1. Build a min-heap
2. Evaluate an expression
3. Construct a 2D array
4. Exit
Please select an option: 2
Enter the input string: ((7-(13+14))*(6-9)
The output is 60

Main Menu:
1. Build a min-heap
2. Evaluate an expression
3. Construct a 2D array
4. Exit
Please select an option: 3
Enter the input string: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
Enter the number of rows: 3
The 2D array is:
0   1  2  3  4
5   6  7  8  9
10 11 12 13 14

Enter the row index: 2
Enter the column index: 3
The beginning address of the 2D array is 0x100000c8
The memory address of the cell Array[2][3] is 0x100000fc
The MIN value between Array[0][0] and Array[2][3] is: 0
The MAX value between Array[0][0] and Array[2][3] is: 13
Main Menu:
1. Build a min-heap
2. Evaluate an expression
3. Construct a 2D array
4. Exit
Please select an option: 4
Program ends. Bye :)
```

References

[1] http://spimsimulator.sourceforge.net/

[2] Data Structures and Algorithm Analysis in C, Mark A. Weiss, Addison Wesley