Name: JAHANVI AGRAWAL
Roll No: IMT2019506

ASSIGNMENT-2

To sort the given input numbers (stored at memory locations starting from X), I have implemented the following **Selection Sort Algorithm:**

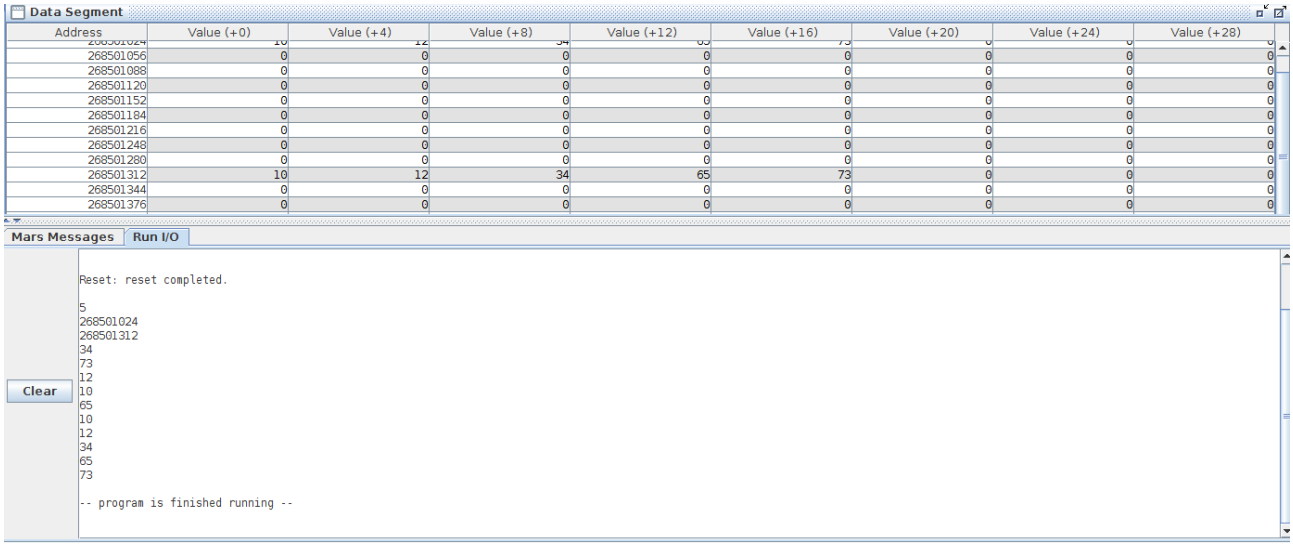
```
//In this algorithm, input numbers are stored in the array arr
int i, j, min_idx;
for ( i = 0 ; i < N – 1 ; i++)
{
        min_idx = i;
        for ( j = i+1; j < N; j++)
        {
                if ( arr[j] < arr[min_idx] )
                {
                        min_idx = j;
                }
        }
        //Swapping arr[i] and arr[min_idx]
        int temp = arr[i];
        arr[i] = arr[min_idx];
        arr[min_idx] = temp;
}
```

Some important comments regarding my code:

- To actually access the elements stored in the memory we need to some manipulation like:
  For accessing arr[i], we need to go to the memory location where this element is stored and because the source array begins at location X, so we can say that arr[i] is stored at **X+4*i memory location**. (So arr[0] is at location X, then arr[1] is at location X+4, arr[2] is at location X+8 and so on).
  I have implemented this using the by shifting i left by 2 (sll instruction) and then adding X to it and storing this X+4*i in a temporary register like $t4 or $t5.
  Then the element is loaded from this X+4*i memory location.
  This is done everytime we need to access the array elements.
- $s3 register keeps track of the min_idx (i.e. the index of the minimum element found at the end of each outer loop iteration)
- $s4 register gives the destination address. It is initialized to Y and gets incremented by 4 as we keep on storing the numbers in the destination array (which starts from Y).
- After the inner loop ends, we swap the elements arr[i] and arr[min_idx] and also store arr[min_idx] in the destination array (address given by $s4) as it is the next number in the sorted sequence.
- At the end, we load the highest element (i.e. last element left in the input array as the input array has been changed due to sorting) at the end of the destination array i.e. at location Y+4*(N-1)

Now, the following are snapshots of the output:

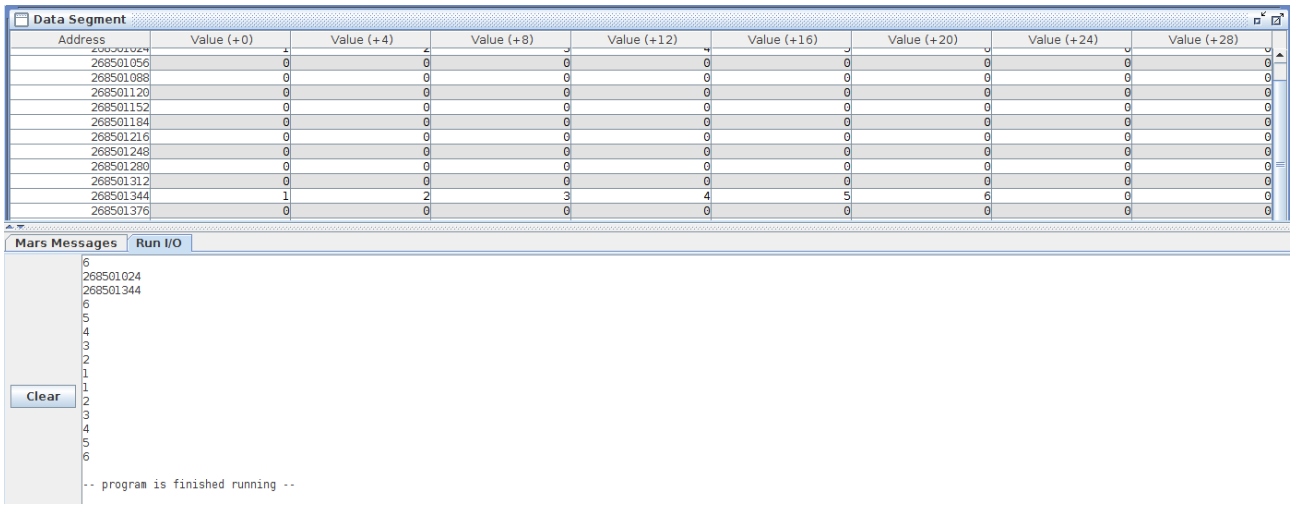
1)



We can see for the Input:
N=5, X=268501024, Y=268501312, and the input numbers as **[34, 73, 12, 10, 65]**
We have got the Output:
[10, 12, 34, 65, 73] which are sorted and are stored in the memory location starting from Y=268501312 (which can be seen in the Data Segment)

Initially the input numbers were stored at locations starting from 265801024 but during the sorting algorithm they get swapped and thus the order changes. Thus, at the end we don't have the input numbers in the same input order at X.

2)



We can see for the Input:
N=6, X=268501024, Y=268501344, and the input numbers as **[6, 5, 4, 3, 2, 1]**
We have got the Output:
[1, 2, 3, 4, 5, 6] which are sorted and are stored in the memory location starting from Y=268501344 (which can be seen in the Data Segment)