CPS 188 Lab 3 : Arrays

***Instructor**: Dr. Ufkes*

***TA**: Mohammed Emrul Hasan*

***Section**: 18*

Sayeed Ahamad

***Student Number**: 501209136*

1 Problem Sets

1.1 Problem 1

1.1.1 Computer Program

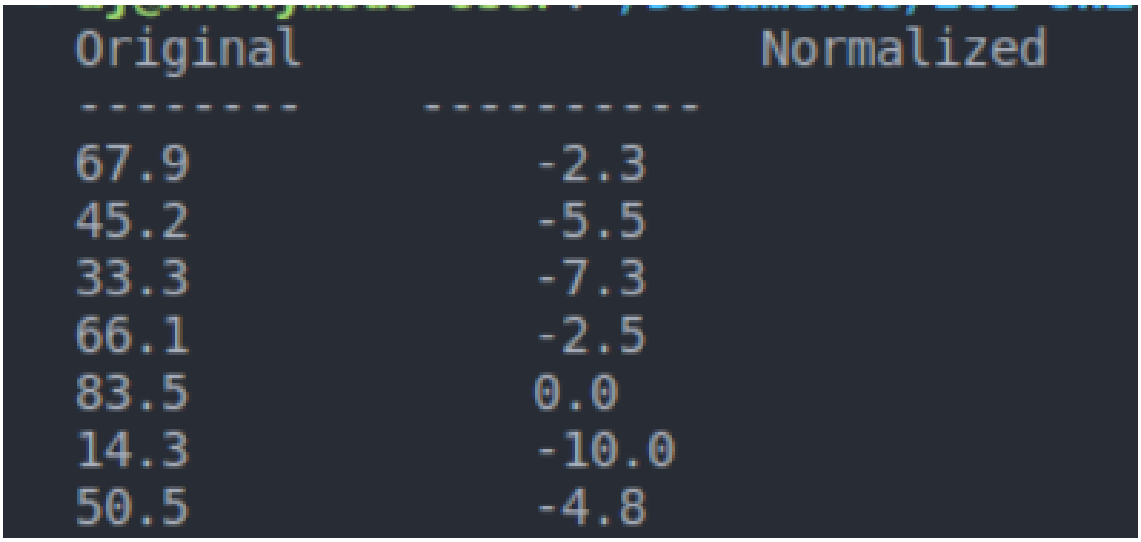
```c
#include <stdio.h>

void main(void)
{
    FILE* file;
    int amt, i;
    double min, max, x1, xh;
    file = fopen("data.txt", "r");
    while (!feof(file))
    {
        fscanf(file, "%d %lf %lf", &amt, &min, &max);
        double nums[amt], norm[amt];
        for (int i = 0; i < amt; i++)
        {
            fscanf(file, "%lf", &nums[i]);
        }
        x1 = nums[0];
        xh = nums[1];
        for (int i = 0; i < amt; i++)
        {
            if (nums[i] < x1)
            {
                x1 = nums[i];
            }
            else if (nums[i] > xh)
            {
                xh = nums[i];
            }
        }
        for (int i = 0; i < amt; i++)
        {
            norm[i] = min + ((nums[i] - xh) * (max - min) / (xh - x1));
        }
        printf("Original\t\tNormalized\n");
        printf("--------    ----------\n");
        for (int i = 0; i < amt; i++)
        {
            printf("%.1lf\t\t%.1lf\n", nums[i], norm[i]);
```

```
        }
        fclose(file);
    }
}
```

Listing 1.1:

1.1.2 Program Output Screenshot



1.2 Problem 2

1.2.1 Computer Program

```

#include <stdio.h>
#define ARRAY_SIZE 8
//  finds the position of the smallest element in the subarray
//  list[first] through list[last].
//  Pre: first < last and elements 0 through last of array list are defined.
//  Post: Returns the subscript k of the smallest element in the subarray;
//  i.e., list[k] <= list[i] for all i in the subarray
int get_min_range (int list[], int first, int last)
{
    int j;
    int min = sizeof(double);
    for (int i = 0; i < last; i++)
    {
```

```
16          if (min > list[i])
17          {
18              min = list[i];
19              j = i;
20          }
21      }
22      return j;
23  }
24  //  sorts the data in array list
25  void select_sort(int list[], int n)
26  {
27      int fill,   /*  index of first element in unsorted
        subarray */
28      temp,   /*  temporary storage   */
29      index_of_min;   /*  subscript of next smallest element
        */
30      for (fill = 0; fill < n-1; ++fill)
31      {
32          /*  Find position of smallest element in unsorted
        subarray */
33          index_of_min = get_min_range (list, fill, n-1);
34          /* Exchange elements at fill and index_of_min */
35          if (fill != index_of_min)
36          {
37              temp = list[index_of_min];
38              list[index_of_min] = list[fill];
39              list[fill] = temp;
40          }
41      }
42  }
43  int main (void)
44  {
45      int array[] = {67, 98, 23, 11, 47, 13, 94, 58};
46      int i;
47      select_sort (array, ARRAY_SIZE);
48      for (i=0; i < 8; ++i)
49      {
50          printf ("%d ", array[i]);
51      } printf("\n");
52      return (0);
53  }
54
55
```

Listing 1.2:

1.2.2 Program Output Screenshot

1.3 Problem 3

1.3.1 Computer Program

```
#include <stdio.h>
#define STACK_EMPTY '0'
#define STACK_SIZE 20

void push(char stack[], /* input/output - the stack */
char item, /* input - data being pushed onto the stack */
int *top, /* input/output - pointer to top of stack */
int max_size) /* input - maximum size of stack */
{
    if (*top < max_size-1)
    {
        ++(*top);
        stack[*top] = item;
    }
}
char pop (char stack[], /* input/output - the stack */
int *top) /* input/output - pointer to top of stack */
{
    char item; /* value popped off the stack */
    if (*top >= 0)
    {
        item = stack[*top];
        --(*top);
    }
    else
    {
        item = STACK_EMPTY;
    }
    return (item);
}
void main(void)
{
    char s [STACK_SIZE];
    int s_top = -1; // stack is empty
    int *ptr = &s_top;
    push(s, 'A', ptr, STACK_SIZE);
    push(s, 'B', ptr, STACK_SIZE);
    push(s, 'C', ptr, STACK_SIZE);

```

```
40      for (int i = 0; i < *ptr; i++)
41      {
42          printf("%c", s[i]);
43      }
44      printf("\n");
45      pop(s, ptr);
46      for (int i = 0; i < *ptr; i++)
47      {
48          printf("%c", s[i]);
49      }
50  }
51
52
```

Listing 1.3:

1.3.2 Program Output Screenshot