Practical 1

Name: Aaron Emmanuel Rocque
Batch: A2 Roll No: 31
Subject: DAA Lab

Problem Statements:

Stock prices of an “ABC” company on a per-day basis are stored in a file “Stock.dat” for a particular month. Write a program to read the file and find the Minimum and Maximum stock price using the Brute force approach. Also, propose and implement an improved algorithm based on the Divide and Conquer Strategy.

Stock.txt:

45
5
63
2
99
234
6969

Code (C):

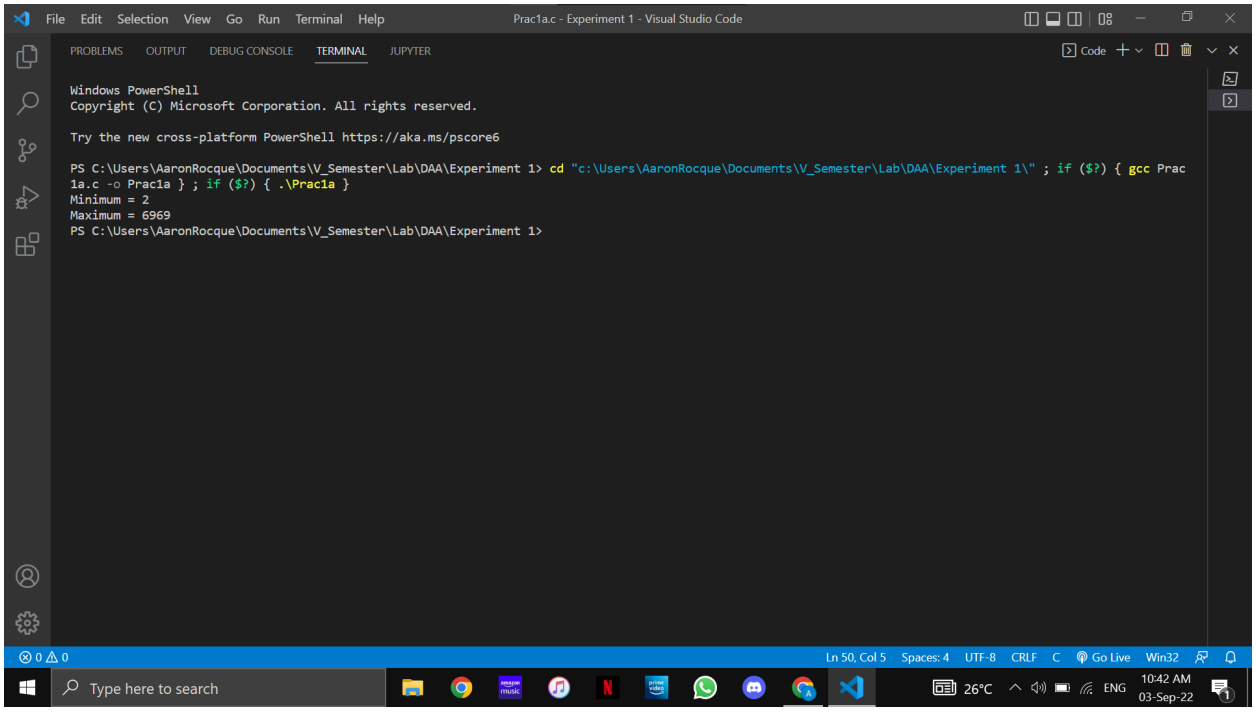
Prac1a.c

```c
#include <stdio.h>

int minimum(int arr[]);
int maximum(int arr[]);

int main(void)
{
  int numbers[50];
  int i = 0;
```

```c
  FILE *file;

  if (file = fopen("Stock.txt", "r"))
  {
    while (fscanf(file, "%d", &numbers[i]) != EOF)
    {
      i++;
    }
    fclose(file);

    numbers[i] = '\0';

    int min = minimum(numbers);
    int max = maximum(numbers);

    printf("Minimum = %d\n", min);
    printf("Maximum = %d\n", max);
  }

  return 0;
}

int minimum(int arr[])
{
    int min = arr[0];

    for(int i = 0; arr[i] != '\0'; i++)
    {
        if(arr[i] <= min)
        {
            min = arr[i];
        }
    }

    return min;
}

int maximum(int arr[])
{
```

```
    int max = arr[0];

    for(int i = 0; arr[i] != '\0'; i++)
    {
        if(arr[i] >= max)
        {
            max = arr[i];
        }
    }

    return max;
}
```

Output:



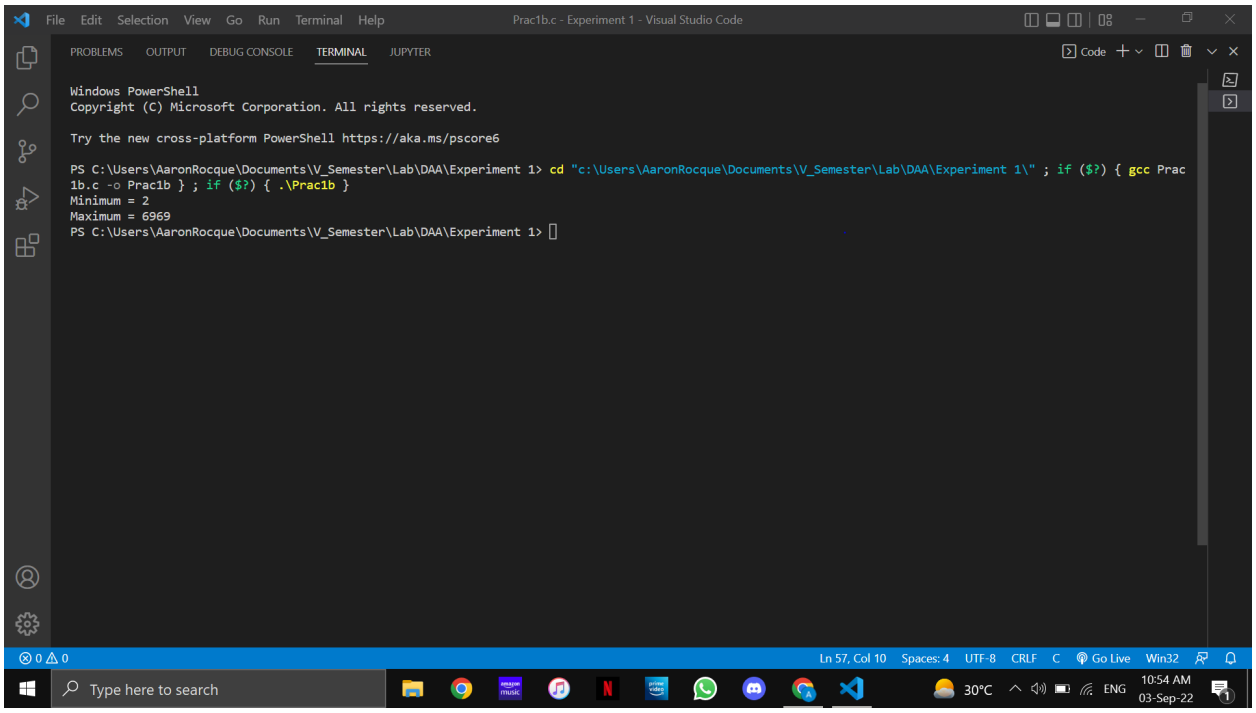
Prac1b.c

```c
#include <stdio.h>

int minimum(int i, int j, int min, int a[]);
int maximum(int i, int j, int max, int a[]);

int main(void)
{
  int numbers[50];
  int i = 0;
  FILE *file;

  if (file = fopen("Stock.txt", "r"))
  {
    while (fscanf(file, "%d", &numbers[i]) != EOF)
    {
      i++;
    }
    fclose(file);

    numbers[i] = '\0';


  }

  int min = minimum(0, i-1, numbers[0], numbers);
  printf("Minimum = %d\n", min);

  int max = maximum(0, i-1, numbers[0], numbers);
  printf("Maximum = %d\n", max);

  return 0;
}


int minimum(int i, int j, int min, int a[])
```

```
{
    int min1 = a[0];

    if(i == j) //for one element
    {
        min = a[j];
    }
    else
    {
        if(i == j-1) //for two elements
        {
            if(a[i] < a[j])
            {
                min = a[i];
            }
            else
            {
                min = a[j];
            }
        }
        else //for more than two elements
        {
            int mid = (i+j)/2;

            min = min123(i, mid, min, a);
            min1 = min123(mid+1, j, min1, a);

            if(min1 < min)
            {
                min = min1;
            }
        }
    }

    return min;

}

int maximum(int i, int j, int max, int a[])
```

```
{
    int max1 = a[0];

    if(i == j) //for one element
    {
        max = a[i];
    }
    else
    {
        if(i == j-1) //for two elements
        {
            if(a[i] < a[j])
            {
                max = a[j];
            }
            else
            {
                max = a[i];
            }
        }
        else //for more than two elements
        {
            int mid = (i+j)/2;

            max = max123(i, mid, max, a);
            max1 = max123(mid+1, j, max1, a);

            if(max < max1)
            {
                max = max1;
            }

    }

    return max;

    }

}
```

Output:



Code (Python):

```
f = open('Stock.txt','r')
contents = f.read().split('\n')
for i in range(0, len(contents)):
    contents[i] = int(contents[i])
print("The contents of the file are \n")
```

```
print(contents)

#-----------------------------------------------------------------

def find_min_and_max(a,i,j,min,max):
    min1 = a[0]
    max1 = a[0]

    if i == j: #for one element
        min = a[j]
        max = a[i]
    else:
        if i == j-1: #for two elements

            if a[i] < a[j]:
                max = a[j]
                min = a[i]
            else:
                max = a[i]
                min = a[j]
        else: #for more than two elements
            mid = (i + j) //2

            (min, max) = find_min_and_max(a, i, mid, min , max)
            (min1, max1) = find_min_and_max(a, mid+1, j, min1, max1)

            if max < max1:
                max = max1

            if min1 < min:
                min = min1

    return min, max


#-----------------------------------------------------------------

(min, max) = find_min_and_max(contents, 0, len(contents)-1, contents[0],
contents[0])
```

```
print("The minimum element in the list is", min)
print("The maximum element in the list is", max)
```