KALINGA INSTITUTE OF
INDUSTRIAL TECHNOLOGY (KIIT)

Deemed to be University U/S 3 of UGC Act, 1956

Name: Prajukta Dey

Section: CSE 13

Subject: Data Structures and Algorithms Lab

Date: 22.07.2022

1. Take n numbers as input in to an 1-d dynamic array A from user; write a C program to compute and display the minimum, maximum, sum, and average of the elements in A.

quest1.c

dsa-lab > lab-01 > C quest1.c > main()

```c
/*Take n numbers as input in to an 1-d dynamic array A from user; write a C program
to compute and display the minimum, maximum, sum, and average of the elements in A.*/

#include<stdio.h>

int main()
{

    int a[50],i,n,max,min,sum=0;
    float average;

    printf("How many elements: ");
    scanf("%d",&n);
    printf("\n");

    printf("Enter the array elements: ");

    for(i=0;i<n;++i)
        scanf("%d",&a[i]);

    max=min=a[0];
    for(i=1;i<n;++i)
    {
        if(a[i]>max)
            max=a[i];
        if(a[i]<min)
            min=a[i];
    }

    for(i=0;i<n;++i)
    {
        sum= sum+a[i];
    }

    average= sum/n;

    printf("The largest element is %d",max);
    printf("\nThe smallest element is %d",min);
    printf("\nThe sum of the elements: %d", sum);
    printf("\nThe average: %f", average);

    return 0;
}
```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; if ($?) { gcc quest1.c -o quest1 } ; if ( $?) { .\quest1 }
How many elements: 5

Enter the array elements: 1 2 3 4 5
The largest element is 5
The smallest element is 1
The sum of the elements: 15
The average: 3.000000
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```

2. Write a program to reverse the contents of an array.

dsa-lab > lab-01 > C quest2.c > main()

```c
//wap to reverse the contents of an array
#include<stdio.h>

void main()
{
    int a[100],reverse[100],i,n;

    printf("Enter number of elements: \n");
    scanf("%d",&n);

    printf("Enter the elements: \n");
    for (i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }

    printf("\n");
    printf("The original array: \n");
    for (i=0;i<n;i++)
    {
       printf("%d ", a[i]);
    }

    for(i=0;i<n;i++)
    {
        reverse[i]=a[n-i-1];
    }

    printf("\nOn reversing the array elements we get: \n");
    for(i=0;i<n;i++)
    {
       printf("%d ",reverse[i]);
    }
}
```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; if ($?) { gcc quest2.c -o quest2 } ; if (
$?) { .\quest2 }
Enter number of elements:
5
Enter the elements:
1 2 3 4 5

The original array:
1 2 3 4 5
On reversing the array elements we get:
5 4 3 2 1
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```

3. Write a program to search an element in an array of n numbers.

```c
//WAP to search an element in an array of n numbers.

#include<stdio.h>
int main()
{
  int arr[100],n1,n,i,count=0;
  int pos=0;

  printf("Enter the size of array: \n");
  scanf("%d",&n1);

  printf("Enter the element of array: \n");
  for(i=0;i<n1;i++)
  {
    scanf("%d",&arr[i]);
  }

  printf("Enter the number to be searched: \n");
  scanf("%d",&n);
  for(i=0;i<n1;i++)
  {
   if(arr[i]==n)
    {
      int pos=i+1;
      count++;
      printf("The position of the element searched: %d\n", pos);
    }
  }

  printf("\n");

  if(count!=0)
  {
   printf("The number is in array and it appeared %d time(s).",count);
  }

  else
  {
   printf("The number is not in array.");
  }

return 0;
}
```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; if ($?) { gcc quest3.c -o quest3 } ; if (
$?) { .\quest3 }
Enter the size of array:
5
Enter the element of array:
1 45 23 78 45
Enter the number to be searched:
45
The position of the element searched: 2
The position of the element searched: 5

The number is in array and it appeared 2 time(s).
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```

4. Given an unsorted array of size n, WAP to find and display the number of elements between two elements a and b (both inclusive). range[2>= and <=5]

dsa-lab > lab-01 > C quest4.c > main()

```c
/*Given an unsorted array of size n, WAP to find and display the number of
elements between two elements a and b (both inclusive). range [2&gt;= and &lt;=5]*/

#include <stdio.h>

int main()
{
    int n;
    printf("Enter the size of array: ");
    scanf("%d",&n);
    printf("\n");

    int a[n];

    printf("Enter the array elements: \n");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }

    int l,u;
    printf("Enter the lower: ");
    scanf("%d" ,&l);
    printf("\n");
    printf("Enter the upper limit: ");
    scanf("%d" ,&u);


    int count=0;

    printf("The elements inside the range are as follows: \n");
    for(int i=0;i<n;i++)
    {
        if(a[i]>=l && a[i]<=u)
        {
        count+=1;
        printf("%d ", a[i]);
        }
    }

    printf("\n");
    printf("Number of elements between the limits: %d\n",count);

    return 0;
}
```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; if ($?) { gcc quest4.c -o quest4 } ; if (
$?) { .\quest4 }
Enter the size of array: 8

Enter the array elements:
1 34 23 67 89 56 45 47
Enter the lower: 1

Enter the upper limit: 50
The elements inside the range are as follows:
1 34 23 45 47
Number of elements between the limits: 5
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```

5. Given an array, WAP to print the next greater element (NGE) for every element. The next greater element for an element x is the first greater element on the right side of x in array. Elements for which no greater element exist, consider next greater element as -1.

```c

#include<stdio.h>

//prints the next greater element
void printNGE(int arr[], int n) //parameters: array[] with size n
{
    int next, i, j;
    for (i=0; i<n; i++)
    {
        next = -1; //default of next is -1
        for (j = i+1; j<n; j++)
        {
            if (arr[i] < arr[j]) //if the next element in the array is greater, then store the number
            {
                next = arr[j];
                break;
            }
        }
        printf("%d  |  %d\n", arr[i], next);
    }
}

```

```c

int main()
{
    int num;
    printf("Enter the size of array: ");
    scanf("%d",&num);
    printf("\n");

    int arr[num];

    printf("Enter the array elements: \n");
    for(int i=0;i<num;i++)
    {
        scanf("%d",&arr[i]);
    }

    int n = sizeof(arr)/sizeof(arr[0]);
    printNGE(arr, n);
    return 0;
}
```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; i
f ($?) { gcc quest5.c -o quest5 } ; if ($?) { .\quest5 }
Enter the size of array: 5

Enter the array elements:
2 5 3 9 7
2  |  5
5  |  9
3  |  9
9  |  -1
7  |  -1
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```

6. Let A be square matrix. WAP by using appropriate user defined functions for the following:

a) Find the number of nonzero elements in A

b) Find the sum of the elements above the leading diagonal.

c) Display the elements below the minor diagonal.

d) Find the product of the diagonal elements.

dsa-lab > lab-01 > C quest6.c > main()

```c
/*Let A be square matrix. WAP by using appropriate user defined functions
for the following:
a) Find the number of nonzero elements in A
b) Find the sum of the elements above the leading diagonal.
c) Display the elements below the minor diagonal.
d) Find the product of the diagonal elements.*/

#include <stdio.h>
int non_zero(int n,int a[n][n]);
int leading_diagonal(int n,int a[n][n]);
void minor_diagonal(int n,int a[n][n]);
void diagonal_product(int n,int a[n][n]);
int main()
{
    int n,i,j;
    printf("Enter no. of elements\n");
    scanf("%d",&n);
    int a[n][n];
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            printf("Enter element of index %d,%d:",i,j);
            scanf("%d",&a[i][j]);
        }
    }
```

```c
    printf("Matrix: \n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            printf("%d  \t",a[i][j]);
        }
        printf("\n");
    }
    int k=non_zero(n,a);
    printf("Number of non-zero elements: %d\n",k);
    int l=leading_diagonal(n,a);
    printf("Sum of elements above leading diagonal are: %d\n",l);
    minor_diagonal(n,a);
    diagonal_product(n,a);
    return 0;
}
int non_zero(int n,int a[n][n])
{
    int i,j,z=0,c=0;

    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            if(a[i][j]==0)
            c++;
```

```c
            else z++;
        }
    }
    return z;
}
int leading_diagonal(int n,int a[n][n])
{
    int i,j,s=0;
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            if(i<=j)
            {
                s+=a[i][j];
            }
        }
    }
    return s;
}
void minor_diagonal(int n,int a[n][n])
{
    printf("Elements below minor diagonal are: \n");
    int i,j;
    for(i=0;i<n;i++)
    {
```

```
    {
        for(j=0;j<n;j++)
        {
            if((i+j)>=(n-1))
            printf("%d  ",a[i][j]);
            else printf("   ");
        }
        printf("\n");
    }
}
void diagonal_product(int n,int a[n][n])
{
    int i,j,m=1;
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            if((i==j)||((i+j)==(n-1)))
            m*=a[i][j];
        }
    }
    printf("\nProduct of leading and minor diagonal elements are: %d\n",m);
}


```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; if ($?) { gcc quest6.c -o quest6 } ; if (
$?) { .\quest6 }
Enter no. of elements
3
Enter element of index 0,0:1
Enter element of index 0,1:2
Enter element of index 0,2:3
Enter element of index 1,0:4
Enter element of index 1,1:5
Enter element of index 1,2:6
Enter element of index 2,0:7
Enter element of index 2,1:8
Enter element of index 2,2:9
Matrix:
1       2       3
4       5       6
7       8       9
Number of non-zero elements: 9
Sum of elements above leading diagonal are: 26
Elements below minor diagonal are:
      3
   5  6
7  8  9

Product of leading and minor diagonal elements are: 945
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```

7. Define a structure for representing a point in two-dimensional Cartesian co-ordinate system. Now, write a C program to perform the following tasks:
a. Compute the distance between two given points.
b. Compute the area of a triangle, given the co-ordinates of its three vertices.

```c
/*Define a structure for representing a point in two-dimensional Cartesian co-ordinate
system. Now, write a C program to perform the following tasks:
a. Compute the distance between two given points.
b. Compute the area of a triangle, given the co-ordinates of its three vertices.*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

struct Point
{
float x,y;
};


double getDistance(struct Point a, struct Point b)
{
    double distance;
    distance = sqrt((a.x - b.x) * (a.x - b.x) + (a.y-b.y) *(a.y-b.y));
    return distance;
}

double AreaTriangle(struct Point a, struct Point b, struct Point c)
{
    double area;
    area= (1.0/2.0)*(((a.x)*(b.y-c.y))+(b.x*(c.y-a.y))+(c.x*(a.y-b.y)));
}

    int main()
{
    struct Point a, b;
    printf("Enter coordinate of point a(x1,y1): ");
    scanf("%f %f", &a.x, &a.y);
    printf("Enter coordinate of point b(x2, y2): ");
    scanf("%f %f", &b.x, &b.y);
    printf("Distance between a and b: %f\n", getDistance(a, b));

    struct Point p,q,r;
    printf("Enter coordinate of point p(x1,y1): ");
    scanf("%f %f", &p.x, &p.y);
    printf("Enter coordinate of point q(x2, y2): ");
    scanf("%f %f", &q.x, &q.y);
    printf("Enter coordinate of point r(x3, y3): ");
    scanf("%f %f", &r.x, &r.y);
    printf("Area of a triangle: %f\n", AreaTriangle(p, q, r));


    return 0;
}
```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; if ($?) { gcc quest7.c -o quest7 } ; if (
$?) { .\quest7 }
Enter coordinate of point a(x1,y1): 1 2
Enter coordinate of point b(x2, y2): 3 4
Distance between a and b: 2.828427
Enter coordinate of point p(x1,y1): 1 2
Enter coordinate of point q(x2, y2): 3 4
Enter coordinate of point r(x3, y3): 5 6
Area of a triangle: 2.000000
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```

8. Write a C program that:

a. Uses Structure to store name, roll no., marks, and address of 5 students in C programming subject.

b. Displays the stored information.

dsa-lab > lab-01 > C 1.c > main()

```c
#include <stdio.h>

int main()
{
  // creating a student structure template
  struct student
  {
    char name[64];
    char address[64];
    int roll;
    int marks;
  };

  // creating a student structure array variable
  struct student studentarray[5];

 //loop variables
  int i;

  // taking user input
  for (i = 0; i < 5; i++) {
    printf("Enter detail of student %d\n", (i+1));

    printf("Enter name: ");
    scanf("%s", studentarray[i].name);

    printf("Enter address: ");
    scanf("%s", studentarray[i].address);

    printf("Enter roll number: ");
    scanf("%d", &studentarray[i].roll);

    printf("Enter marks: ");
    scanf("%d", &studentarray[i].marks);
  }

  // display
  for (i = 0; i < 5; i++)
  {
    printf("\nStudent #%d Detail:\n", (i+1));
    printf("Name: %s\n", studentarray[i].name);
    printf("Address: %s\n", studentarray[i].address);
    printf("Roll: %d\n", studentarray[i].roll);
    printf("Marks: %d\n", studentarray[i].marks);
  }

  return 0;
}
```

Output:

```
PS C:\PRAJUKTA\learning-languages\cpp-programming> cd "c:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01\" ; i
f ($?) { gcc 1.c -o 1 } ; if ($?) { .\1 }
Enter detail of student 1
Enter name: Prajukta
Enter address: Kolkata
Enter roll number: 1
Enter marks: 98
Enter detail of student 2
Enter name: Alan
Enter address: NYC
Enter roll number: 2
Enter marks: 87
Enter detail of student 3
Enter name: Tom
Enter address: NYC
Enter roll number: 3
Enter marks: 87
Enter detail of student 4
Enter name: Kristen
Enter address: Texas
Enter roll number: 4
Enter marks: 90
Enter detail of student 5
Enter name: Karen
Enter address: NYC
Enter roll number: 5
Enter marks: 98
```

```
Student #1 Detail:
Name: Prajukta
Address: Kolkata
Roll: 1
Marks: 98

Student #2 Detail:
Name: Alan
Address: NYC
Roll: 2
Marks: 87

Student #3 Detail:
Name: Tom
Address: NYC
Roll: 3
Marks: 87

Student #4 Detail:
Name: Kristen
Address: Texas
Roll: 4
Marks: 90

Student #5 Detail:
Name: Karen
Address: NYC
Roll: 5
Marks: 98
PS C:\PRAJUKTA\learning-languages\cpp-programming\dsa-lab\lab-01>
```