**Name: SOUMYADIP GHOSH**

**Roll: 2005133**

**CSE\_SLOT1**

**LAB 1**

**Q1**

#include<stdio.h>

#include<stdlib.h>

void main()

{

    int \*p,n,sum=0;

    float avg;

    printf("Enter the value of n: ");

    scanf("%d",&n);

    p = (int\*)malloc(n\*sizeof(int)); //(int\*)calloc(n,sizeof(int));

    for(int i=0;i<n;i++){

        printf("Enter the no. : ");

        scanf("%d",&p[i]);

        sum += p[i];

    }

    for(int i=0;i<n;i++){

        printf("Number: %d\n ",p[i]);

    }

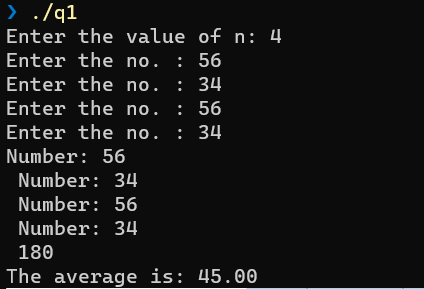
    avg = sum/n;

    printf("%d\n",sum);

    printf("The average is: %.2f",avg);

    free(p);

}



#include <stdio.h>

**Q2**

#include <stdlib.h>

int main()

{

    int n, max = -1;

    printf("enter number: ");

    scanf("%d", &n);

    int \*a = (int \*)malloc(n \* sizeof(int));

    printf("enter elements: \n");

    for (int i = 0; i < n; i++)

    {

        scanf("%d", a + i);

        if (max < a[i])

        {

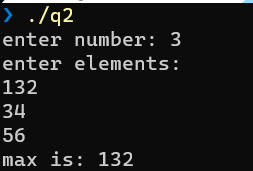
            max = a[i];

        }

    }

    printf("max is: %d\n", max);

}



#include <stdio.h>

**Q3**

#include <stdlib.h>

struct student

{

    int roll;

    int age;

    float cgpa;

    char sub[25];

};

void main()

{

    int n, large;

    struct student \*p;

    printf("Enter the number of students: ");

    scanf("%d", &n);

    p = (struct student \*)malloc(n \* sizeof(struct student)); //(struct student\*)calloc(n,sizeof(struct student))

    for (int i = 0; i < n; i++)

    {

        printf("Details of student %d\n", i + 1);

        printf("\nEnter the roll of student: ");

        scanf("%d", &(p + i)->roll);

        printf("Enter the age of student: ");

        scanf("%d", &(p + i)->age);

        printf("Enter the cgpa of student: ");

        scanf("%f", &(p + i)->cgpa);

        printf("Enter the subject of student: ");

        scanf(" %[^\n]%\*c", (p + i)->sub);

    }

    for (int i = 0; i < n; i++)

    {

        printf("Roll of student: %d\n", p[i].roll);

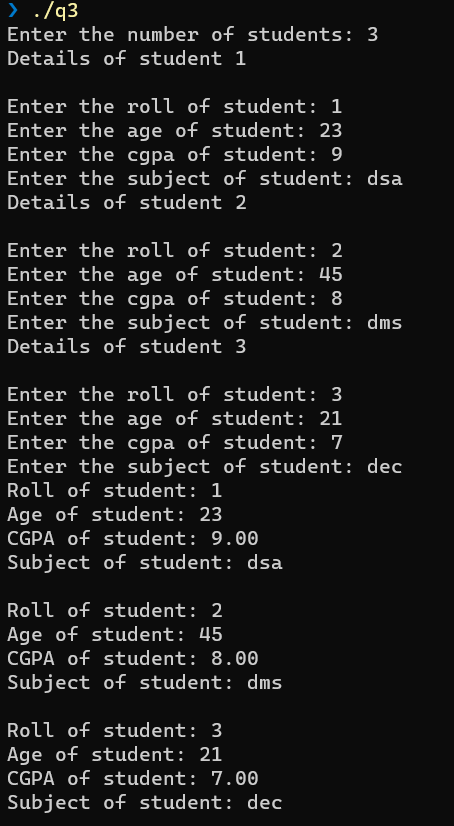
        printf("Age of student: %d\n", p[i].age);

        printf("CGPA of student: %.2f\n", p[i].cgpa);

        printf("Subject of student: %s\n\n", p[i].sub);

    }

    free(p);

}

#include <stdio.h>

**Q4**

#include <stdlib.h>

#include <math.h>

float mean(int \*x, int n)

{

    float mean = 0;

    for (int i = 0; i < n; i++)

    {

        mean += x[i];

    }

    mean = (mean / n);

    return mean;

}

float std\_dev(int \*x, int n)

{

    float std = 0;

    float avg = mean(x, n);

    for (int i = 0; i < n; i++)

    {

        std += ((x[i] - avg) \* (x[i] - avg));

    }

    float deviation = sqrt(std / n);

    return deviation;

}

void main()

{

    int n = 20, \*p;

    p = (int \*)malloc(n \* sizeof(int)); //(int\*)calloc(n,sizeof(int))

    for (int i = 0; i < n; i++)

    {

        printf("Enter the no. : ");

        scanf("%d", &p[i]);

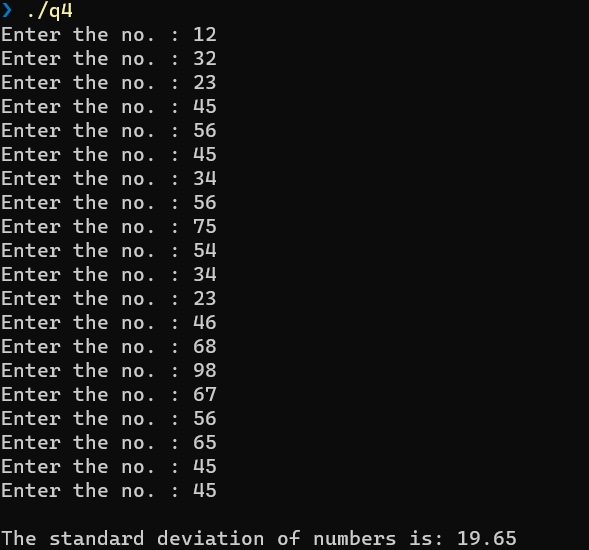
    }

    float standard\_deviation = std\_dev(p, n);

    printf("\nThe standard deviation of numbers is: %.2f", standard\_deviation);

    free(p);

}



#include <stdio.h>

**Q5**

#include <stdlib.h>

void main()

{

    int x, y, \*arr, n;

    printf("Enter the number of elements: ");

    scanf("%d", &n);

    arr = (int \*)malloc(sizeof(int) \* n);

    for (int i = 0; i < n; i++)

    {

        printf("Enter element no. %d: ", i + 1);

        scanf("%d", &arr[i]);

    }

    int even\_num = 0, odd\_num = 0;

    int \*odd, \*even;

    odd = (int \*)malloc((n / 2) \* sizeof(int));

    even = (int \*)malloc((n / 2) \* sizeof(int));

    for (int i = 0; i < n; i++)

    {

        if (arr[i] % 2 == 0)

        {

            even[even\_num] = arr[i];

            even\_num++;

        }

        else

        {

            odd[odd\_num] = arr[i];

            odd\_num++;

        }

    }

    for (int i = 0; i < odd\_num; i++)

    {

        arr[i] = odd[i];

    }

    int s = odd\_num;

    for (int i = 0; i < even\_num; i++)

    {

        arr[s] = even[i];

        s++;

    }

    for (int i = 0; i < n; i++)

    {

        printf("%d ", arr[i]);

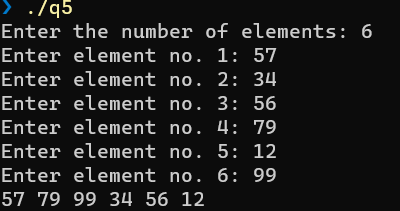
    }

    free(arr);

    free(odd);

    free(even);

}



#include <stdio.h>

**Q6**

#include <stdlib.h>

void swap(int \*x, int n)

{

    int first = 0;

    int last = n - 1;

    int temporary;

    while (first < last)

    {

        temporary = x[first];

        x[first] = x[last];

        x[last] = temporary;

        first++;

        last--;

    }

}

void main()

{

    int n, \*p;

    printf("Enter the number of elements: ");

    scanf("%d", &n);

    p = (int \*)malloc(n \* sizeof(int)); //(int\*)malloc(n,sizeof(int))

    for (int i = 0; i < n; i++)

    {

        printf("Enter the no. : ");

        scanf("%d", &p[i]);

    }

    swap(p, n);

    printf("Reversed Array: \n");

    for (int i = 0; i < n; i++)

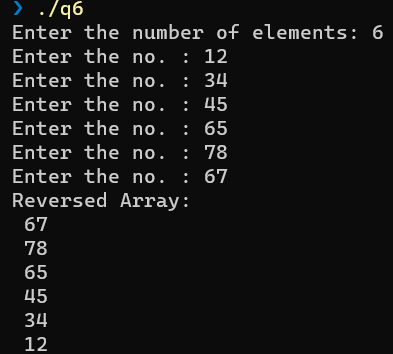
    {

        printf(" %d \n", p[i]);

    }

    free(p);

}



**Q1**

**Lab 2**

#include <stdio.h>

#include <stdlib.h>

void row\_fixed()

{

    int coloumn[4];

    int \*ptr[4];

    for (int i = 0; i < 4; i++)

    {

        printf("Enter the no. of coloumn for row %d : ", i + 1);

        scanf("%d", &coloumn[i]);

        ptr[i] = (int \*)malloc(sizeof(int) \* coloumn[i]);

    }

    for (int i = 0; i < 4; i++)

    {

        printf("\n");

        for (int j = 0; j < coloumn[j]; j++)

        {

            printf("Enter the element for row %d column %d: ", i + 1, j + 1);

            scanf("%d", &ptr[i][j]);

        }

    }

    printf("\nThe array: \n");

    for (int i = 0; i < 4; i++)

    {

        for (int j = 0; j < coloumn[j]; j++)

        {

            printf("%d ", ptr[i][j]);

        }

        printf("\n");

    }

    free(ptr);

}

void coloumn\_fixed()

{

    int row;

    int(\*ptr)[5];

    printf("Enter the no of rows: ");

    scanf("%d", &row);

    ptr = (int(\*)[5])malloc(row \* 5 \* sizeof(int));

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < 5; j++)

        {

            printf("Enter the element on row %d coloumn %d: ", i + 1, j + 1);

            scanf("%d", &ptr[i][j]);

        }

    }

    printf("\nThe array: \n");

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < 5; j++)

        {

            printf("%d ", ptr[i][j]);

        }

        printf("\n");

    }

    free(ptr);

}

void nothing\_fixed()

{

    int row, \*ptr\_coloumn;

    int \*\*ptr;

    printf("Enter the no of rows: ");

    scanf("%d", &row);

    ptr = (int \*\*)malloc(row \* sizeof(int));

    ptr\_coloumn = (int \*)malloc(row \* sizeof(int));

    for (int i = 0; i < row; i++)

    {

        printf("Enter coloumn size for row %d: ", i + 1);

        scanf("%d", &ptr\_coloumn[i]);

        ptr[i] = (int \*)malloc(sizeof(int) \* ptr\_coloumn[i]);

    }

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < ptr\_coloumn[i]; j++)

        {

            printf("Enter the element for row %d column %d: ", i + 1, j + 1);

            scanf("%d", &ptr[i][j]);

        }

    }

    printf("\nThe array: \n");

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < ptr\_coloumn[i]; j++)

        {

            printf("%d ", ptr[i][j]);

        }

        printf("\n");

    }

    free(ptr\_coloumn);

    free(ptr);

}

void main()

{

    int n;

    printf("enter case: ");

    scanf("%d", &n);

    switch (n)

    {

    case 1:

        row\_fixed();

        break;

    case 2:

        coloumn\_fixed();

        break;

    case 3:

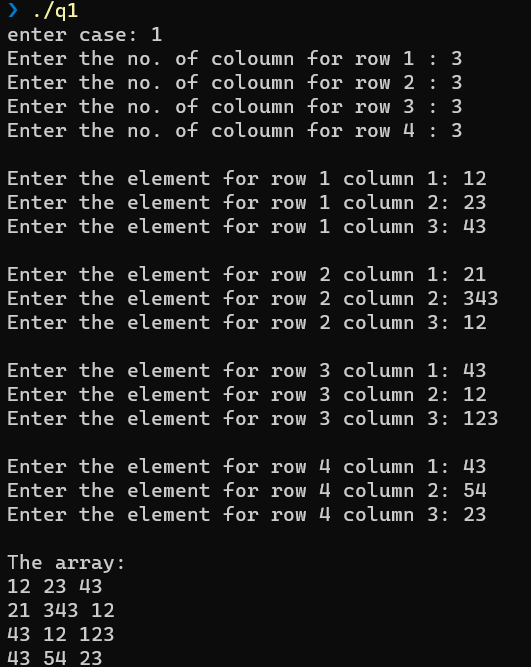
        nothing\_fixed();

        break;

    default:

        break;

    }

}

#include <stdio.h>

**Q2**

#include <stdlib.h>

int non\_zero(int \*\*\*ptr, int n)

{

    int nonzero = 0;

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            if (\*(\*((\*ptr) + i) + j) != 0)

            {

                nonzero++;

            }

            //printf("%d ",\*(\*((\*ptr)+i)+j));

        }

        //printf("\n");

    }

    return nonzero;

}

int diag\_sum(int \*\*\*ptr, int n)

{

    int sum = 0;

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            if (j > i)

            {

                sum += \*(\*((\*ptr) + i) + j);

            }

        }

    }

    return sum;

}

void below\_diag(int \*\*\*ptr, int n)

{

    printf("\nArray below diag: \n");

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            if (j < i)

            {

                printf("%d ", \*(\*((\*ptr) + i) + j));

            }

        }

        printf("\n");

    }

}

int diag\_product(int \*\*\*ptr, int n)

{

    int prod = 1;

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            if (j == i)

            {

                prod \*= \*(\*((\*ptr) + i) + j);

            }

            if (i + j == n - 1)

            {

                prod \*= \*(\*((\*ptr) + i) + j);

            }

        }

    }

    return prod;

}

void main()

{

    int \*\*arr;

    int n;

    printf("Enter the size of square matrix: ");

    scanf("%d", &n);

    arr = (int \*\*)malloc(n \* sizeof(int));

    for (int i = 0; i < n; i++)

    {

        arr[i] = (int \*)malloc(n \* sizeof(int));

    }

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            printf("Enter element for row %d column %d: ", i + 1, j + 1);

            scanf("%d", &arr[i][j]);

        }

    }

    int nonzero = non\_zero(&arr, n);

    printf("\nThe number of non-zero element in the array: %d", nonzero);

    int sum\_lead\_diag = diag\_sum(&arr, n);

    printf("\nThe sum of elements above diagonal %d ", sum\_lead\_diag);

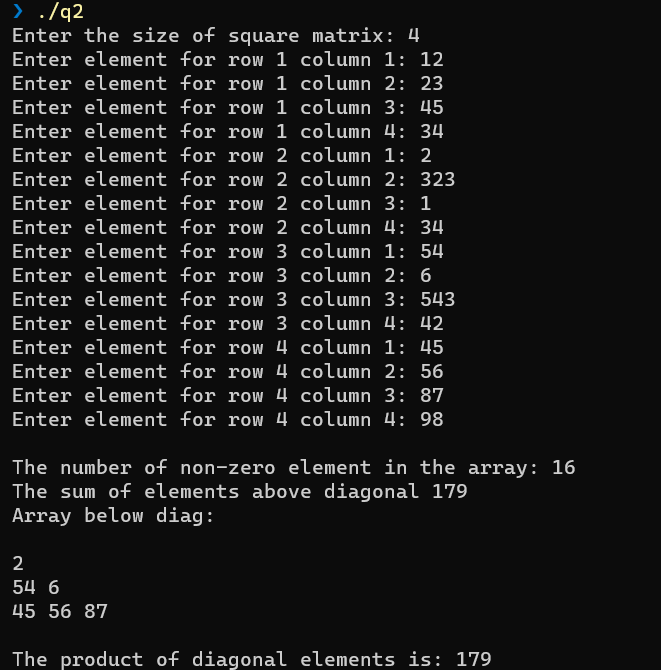
    below\_diag(&arr, n);

    int diag = diag\_sum(&arr, n);

    printf("\nThe product of diagonal elements is: %d ", diag);

    free(arr);

}



#include <stdio.h>

**Q3**

#include <stdlib.h>

void swap(int \*\*\*ptr, int n)

{

    for (int x = 0; x < n; x++)

    {

        int end = n - 1;

        int start = 0;

        while (end >= start)

        {

            int temp = (\*ptr)[x][start];

            (\*ptr)[x][start] = (\*ptr)[x][end];

            (\*ptr)[x][end] = temp;

            start++;

            end--;

        }

    }

}

void main()

{

    int \*\*arr;

    int n;

    printf("Enter the size of square matrix: ");

    scanf("%d", &n);

    arr = (int \*\*)malloc(n \* sizeof(int));

    for (int i = 0; i < n; i++)

    {

        arr[i] = (int \*)malloc(n \* sizeof(int));

    }

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            printf("Enter element for row %d column %d: ", i + 1, j + 1);

            scanf("%d", &arr[i][j]);

        }

    }

    swap(&arr, n);

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

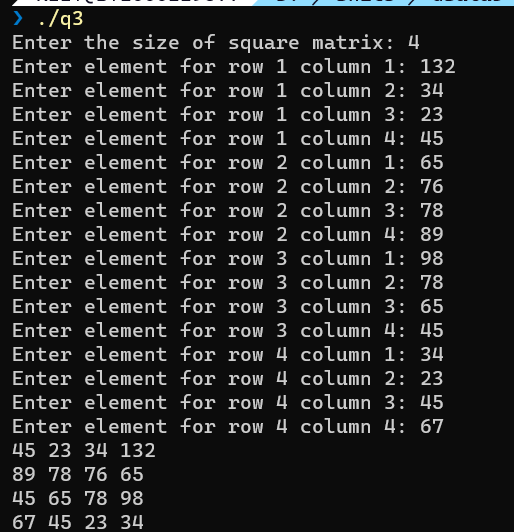
            printf("%d ", arr[i][j]);

        }

        printf("\n");

    }

    free(arr);

}

**Lab 3**

**Q1**

//km-m distance

#include <stdio.h>

#include <stdlib.h>

struct distance

{

    float km;

    float m;

};

struct distance add(struct distance \*\*ptr)

{

    struct distance temp;

    temp.km = (\*ptr)[0].km + (\*ptr)[1].km;

    temp.m = (\*ptr)[0].m + (\*ptr)[1].m;

    return temp;

}

void main()

{

    struct distance \*d;

    int n = 2;

    d = (struct distance \*)malloc(n \* sizeof(struct distance));

    for (int i = 0; i < n; i++)

    {

        printf("\nEnter distance in km-m for distance %d: ", i + 1);

        scanf("%f%f", &d[i].km, &d[i].m);

    }

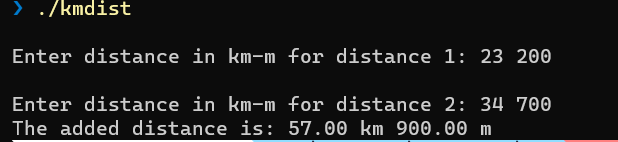
    add(&d);

    struct distance added = add(&d);

    printf("The added distance is: %.2f km %.2f m", added.km, added.m);

    free(d);

}



//You are given an array of 0s and 1s in random order.

**Q2**

//Segregate 0s on left side and 1s on right side of the array. Traverse array only once.

#include <stdio.h>

void segregate(int arr[], int n)

{

    int count = 0;

    for (int i = 0; i < n; i++)

    {

        if (arr[i] == 0)

            count++;

    }

    for (int i = 0; i < count; i++)

        arr[i] = 0;

    for (int i = count; i < n; i++)

        arr[i] = 1;

}

int main()

{

    int n;

    printf("Enter number. of elements: ");

    scanf("%d", &n);

    int arr[n];

    printf("Enter elements:\n");

    for (int i = 0; i < n; i++)

    {

        scanf("%d", &arr[i]);

    }

    segregate(arr, n);

    printf("Segregated array: \n");

    for (int i = 0; i < n; i++)

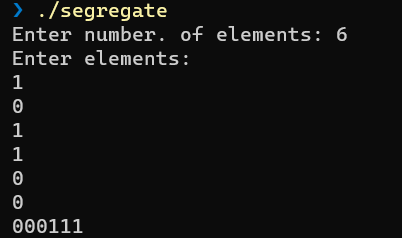
    {

        printf("%d", arr[i]);

    }

    return 0;

}



//min distanv=ce between two elements

**Q3**

#include <stdio.h>

#include <limits.h>

#include <stdlib.h>

int minDist(int \*a, int n, int x, int y)

{

    int i, j;

    int min\_dist = INT\_MAX;

    for (i = 0; i < n; i++)

    {

        for (j = i + 1; j < n; j++)

        {

            if ((x == a[i] && y == a[j] || y == a[i] && x == a[j]) && min\_dist > abs(i - j))

            {

                min\_dist = abs(i - j);

            }

        }

    }

    if (min\_dist > n)

    {

        return -1;

    }

    return min\_dist;

}

int main()

{

    int n;

    printf("Enter number of elements: ");

    scanf("%d", &n);

    int \*a = (int \*)malloc(n \* sizeof(int));

    printf("Enter the elements\n");

    for (int i = 0; i < n; i++)

    {

        scanf("%d", a + i);

    }

    int x, y;

    printf("Enter x and y: ");

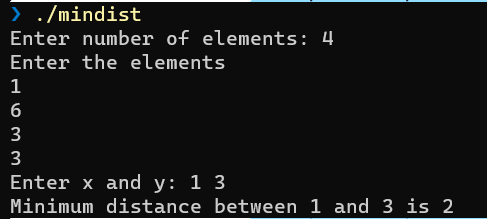
    scanf("%d %d", &x, &y);

    printf("Minimum distance between %d and %d is %d\n", x,

           y, minDist(a, n, x, y));

    return 0;

}



//lcf hcf gcd

**Q4**

#include <stdio.h>

int lcm(int a, int b)

{

    static int temp = 1;

    if (temp % a == 0 && temp % b == 0)

    {

        return temp;

    }

    else

    {

        temp++;

        lcm(a, b);

        return temp;

    }

}

int hcf(int a, int b)

{

    while (a != b)

    {

        if (a > b)

        {

            return hcf(a - b, b);

        }

        else

        {

            return hcf(a, b - a);

        }

    }

    return a;

}

int gcd(int a, int b)

{

    if (b != 0)

        return gcd(b, a % b);

    else

        return a;

}

int main()

{

    int c, x, y;

    printf("Enter case:(1 for lcm, 2 for hcf, 3 for gcd) ");

    scanf("%d", &c);

    printf("Enter two positive integer: ");

    scanf("%d %d", &x, &y);

    switch (c)

    {

    case 1:

        printf("LCM = %d", lcm(x, y));

        break;

    case 2:

        printf("HCF = %d", hcf(x, y));

        break;

    case 3:

        printf("GCD = %d", gcd(x, y));

    default:

        break;

    }

}

