

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

LAB MANUAL

OF

Programming C

Dr. Gopal Behera, Asst. Professor
Department Computer Science & Engineering



Government College of Kalahandi, Bhanuipatna

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

Step 3:- Introduce the variables F and C to store the temperature in Celsius and Fahrenheit.

Step 4:- Using printf() and scanf() function the temperature in C is accepted from the user and stored in C.

Step 5:- Using the formula $F=(1.8*C)+32$; or $F=((9/5)*C)+32$; the temperature is calculated in F and stored in F.

Step 6:- Using printf() statement the calculated temperature in F is displayed.

Step 7:- End of the program.

Program:

```
#include<stdio.h>

int main()
{
    float f,c;
    printf("enter the temperature in C");
    scanf("%f",&c);
    f=(1.8*c)+32; // or f=((9/5)*c)+32;
    printf("the temperature in F is %f",f);
    return 0;
}
```

Output:

```
D:\New folder\C to F.exe
enter the temperature in C
32
the temperature in F is 89.599998
-----
Process exited after 6.545 seconds with return value 0
Press any key to continue . . .
```

Experiment 2

Aim of the experiment: Write a program to calculate the SI and CI.

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Introduce the variables P,R,T,SI,CI for storing the principle amount, rate, time, simple interest and compound interest respectively.

Step 4:- Using the printf() and scanf() statement accept the values of the P,R and T.

Step 5: Using the formula $SI=(P*R*T)/100$; the simple interest is calculated.

Step 6:- Using the formula $CI=P*((1+(R/100))^T)$; the compound interest is calculated.

Step 7:- Using the printf() and scanf() the SI and CI is displayed.

Step 8: - End of the program.

Output:

```
D:\New folder\SI,CI.exe
enter the value of p
5000
enter the value of r
12
enter the value of t
3
The simple intrest is 1800
The compound intrest is 10000

-----
Process exited after 6.2 seconds with return value 0
Press any key to continue . . .
```

Experiment 3

Aim of the experiment: Write a program to display the grades of the students using switch case.

Procedure:

Step 1:- Start the program.

Step 2:- start the main function.

Step 3:- Initialise a variable m to store the marks of the student.

Step 4:- Using printf() and scanf() statement the marks of the student is accepted.

Step 5:- $m=m/10$; this is done in-order to decrease the no. of statement in the switch case.

Step 6:- The statements for the switch case is written to find out the grade according to the marks scored by the student.

Step 7:- End of the program.

Program:

```
#include<stdio.h>
int main()
{
    int m;
    printf("enter the marks of the student\n");
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
scanf("%d",&m);  
m=m/10;  
switch(m)  
{  
case 10:  
case 9:  
printf("grade A");  
break;  
case 8:  
printf("grade B");  
break;  
case 7:  
printf("grade C");  
break;  
case 6:  
printf("grade D");  
break;  
case 5:  
printf("grade E");  
break;  
default :  
printf("fail");  
}  
return 0;  
}
```

Output:

```
D:\New folder\marks.exe
enter the marks of the student
87
grade B
-----
Process exited after 3.642 seconds with return value 0
Press any key to continue . . .
```

Experiment 4(a)

Aim of the experiment: Write a program to display the prime numbers from 1 to 500.

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Initialise variables a, b and c. a and b for loop and c to check if the number is a prime number or not.

Step 4:- Using 2 for loop and if statement the numbers are checked if it is a prime number or not.

Step 5:- Using printf() statement the prime numbers are displayed.

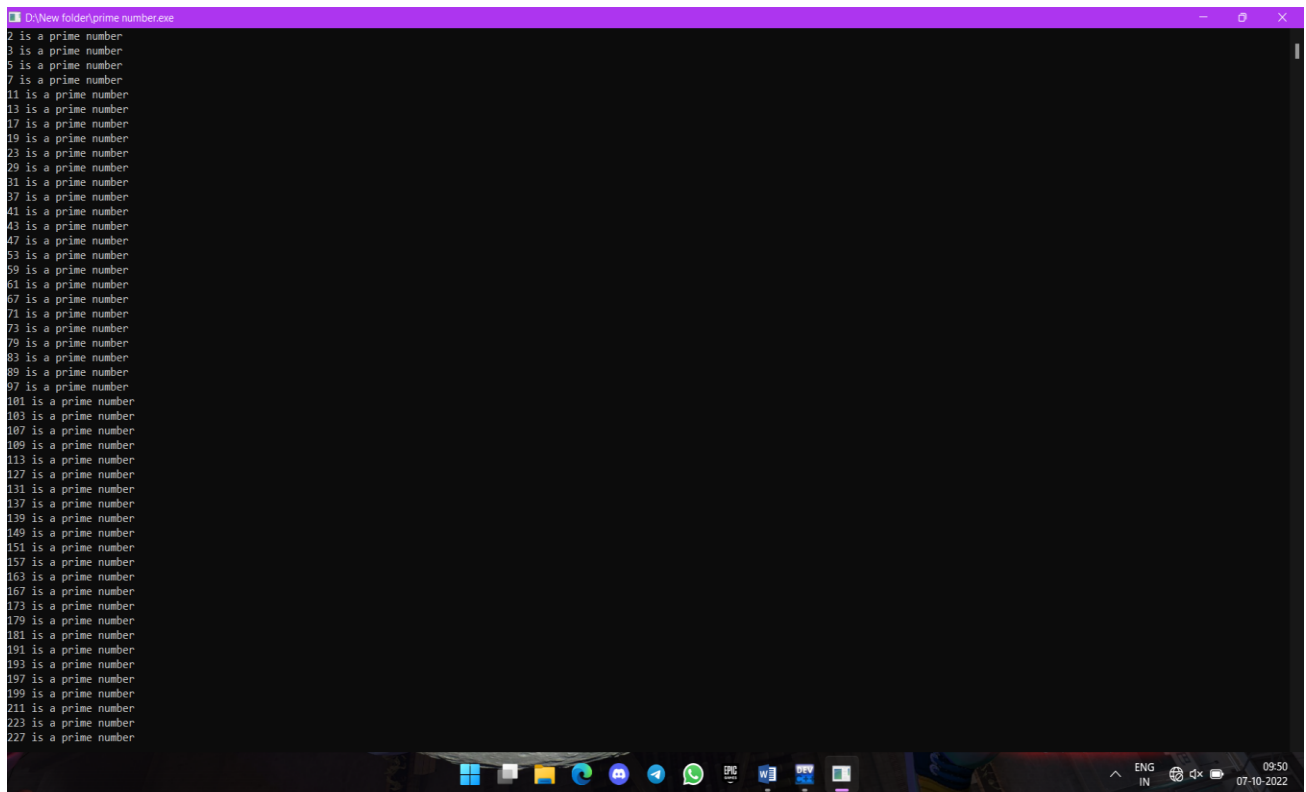
Program:

```
#include<stdio.h>
int main()
{
    int a,b,c=0;
    for(a=1;a<=500;a++)
    {
        for(b=1;b<=a;b++)
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
{  
    if(a%b==0)  
    {  
        c++;  
    }  
}  
if(c==2)  
{  
    printf("%d is a prime number \n",a);  
}  
c=0;  
}
```

Output:-



```
D:\New folder\prime number.exe  
2 is a prime number  
3 is a prime number  
5 is a prime number  
7 is a prime number  
11 is a prime number  
13 is a prime number  
17 is a prime number  
19 is a prime number  
23 is a prime number  
29 is a prime number  
31 is a prime number  
37 is a prime number  
41 is a prime number  
43 is a prime number  
47 is a prime number  
53 is a prime number  
59 is a prime number  
61 is a prime number  
67 is a prime number  
71 is a prime number  
73 is a prime number  
79 is a prime number  
83 is a prime number  
89 is a prime number  
97 is a prime number  
101 is a prime number  
103 is a prime number  
107 is a prime number  
109 is a prime number  
113 is a prime number  
127 is a prime number  
131 is a prime number  
137 is a prime number  
139 is a prime number  
149 is a prime number  
151 is a prime number  
157 is a prime number  
163 is a prime number  
167 is a prime number  
173 is a prime number  
179 is a prime number  
181 is a prime number  
191 is a prime number  
193 is a prime number  
197 is a prime number  
199 is a prime number  
211 is a prime number  
223 is a prime number  
227 is a prime number
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE
Experiment 4(b)

Aim of the experiment: Write a program to find the factorial of a number.

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Initialise 3 variables a, b and c. a for accepting the input number, b for loop and c for storing the factorial of the number.

Step 4:- Using printf() and scanf() statement the number whose factorial is to be found is accepted.

Step 5:- Using for loop the factorial is found and stored in c.

Step 6:- Using printf() statement the factorial is displayed.

Step 7:- End of the program.

Program:

```
#include<stdio.h>
int main()
{
    int a,b,c=1;
    printf("enter the number");
    scanf("%d",&a);
    for(b=1;b<=a;b++)
    {
        c=c*b;
    }
    printf("%d is the factorial of %d",c,a);
}
```

Output:-

```
D:\New folder\factorial.exe
enter the number
5
120 is the factorial of 5
-----
Process exited after 4.188 seconds with return value 0
Press any key to continue . . .
```

Experiment 4(c)

Aim of the experiment: Write a program to find the HCF and LCM of two numbers.

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Initialise variables a, b, i, hcf and lcm. a and b are used to store the two numbers given by the user. i is used in finding the hcf of the two numbers. Hcf and lcm are used to store the hcf and lcm respectively.

Step 4:- Using printf() and scanf() the two numbers are accepted and stored in a and b.

Step 5:- Using the for loop and if statements the hcf is calculated and stored in hcf.

Step 6:- Using the formula $lcm=(a*b)/hcf$; the lcm is calculated and stored in lcm.

Step 7:- Using printf() statement the hcf and lcm is displayed.

Step 8:- End of the program.

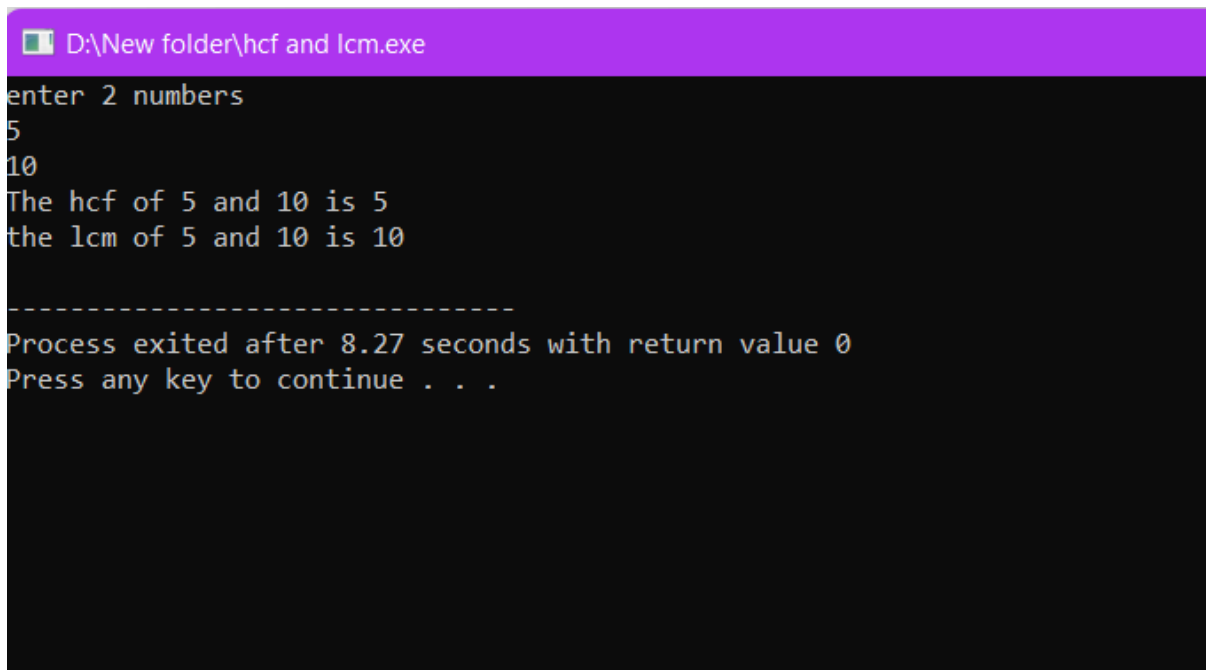
Program:-

```
#include<stdio.h>
int main()
{
    int a,b,i,hcf,lcm;
    printf("enter 2 numbers \n");
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
scanf("%d %d",&a,&b);
for(i=1;i<=a&& i<b;i++)
{
    if(a%i==0 && b%i==0)
    {
        hcf=i;
    }
}
lcm=(a*b)/hcf;
printf("The hcf of %d and %d is %d \n",a,b,hcf);
printf("the lcm of %d and %d is %d \n",a,b,lcm);
return 0;
}
```

Output:-



```
D:\New folder\hcf and lcm.exe
enter 2 numbers
5
10
The hcf of 5 and 10 is 5
the lcm of 5 and 10 is 10
-----
Process exited after 8.27 seconds with return value 0
Press any key to continue . . .
```

Experiment 5(a)

Aim of the experiment: Write a program to find check If a number is an Armstrong number or not.

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Initialise variables a, r, sum, num. a to store the number to be checked. R to find the Armstrong number. sum to store the number to check the number is an Armstrong number or not. Num to store the original number to check in the end if the sum is armstrong number or not.

Step 4:- Using printf() and scanf() statements accept the number in a.

Step 5:- Using for loop and the formula to find the Armstrong number, it is found and stored in the sum.

Step 6:- Using if and else statement and printf() statement, the sum found is compared to the num to check if the number is an Armstrong number or not and displayed.

Step 7:- End of the program.

Program :-

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int a,r,sum=0,num;
```

```
    printf("enter the number");
```

```
    scanf("%d",&a);
```

```
    num=a;
```

```
    for(;a>0;)
```

```
    {
```

```
        r=(a%10)*(a%10)*(a%10);
```

```
        a=a/10;
```

```
        sum=sum+r;
```

```
    }
```

```
    if(sum==num)
```

```
    {
```

```
        printf("%d is an armstrong number",num);
```

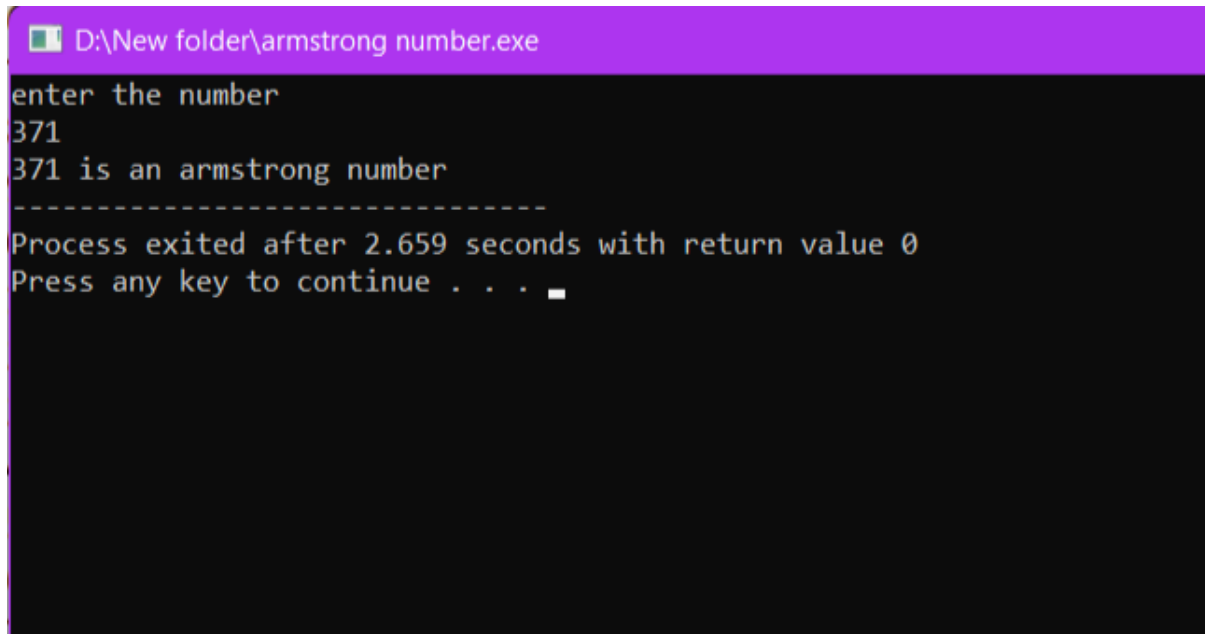
```
    }
```

```
    else
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
{  
    printf("%d is not an armstrong number",num);  
}  
return 0;  
}
```

Output:-



```
D:\New folder\armstrong number.exe  
enter the number  
371  
371 is an armstrong number  
-----  
Process exited after 2.659 seconds with return value 0  
Press any key to continue . . .
```

Experiment 5(b)

Aim of the experiment: Write a program to find the sum and average of an array.

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Initialise variables sum, avg, ar[], i and n. sum to store the sum of the elements of the array. Avg to store the average of the elements of the array. Ar[] to store the elements in the array ar. I for loop. N for number of elements in the array.

Step 4:- Using printf() and scanf() the number of elements to be accepted in the array is stored in n.

Step 5:- Using printf(), for loop and scanf() the array ar[] is filled with the n no. of elements.

Step 6:- using for loop the factorial is sum is added to sum one by one.

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

Step 7:- using the sum and number of elements the average is calculated.

Step 8:- Using printf() statement the sum and average of the array is displayed.

Step 9:- End of the program.

Program:

```
#include<stdio.h>

int main()
{
    int sum,avg,i,ar[40],n;
    printf("enter the number of elements\n");
    scanf("%d",&n);
    printf("enter the element\n");
    for(i=0;i<n;i++)
    {
        scanf("%d",&ar[i]);
    }
    for(i=0;i<n;i++)
    {
        sum=sum+ar[i];
    }
    avg=sum/n;
    printf("The sum of the array is %d\n",sum);
    printf("the average of the array is %d\n",avg);
    return 0;
}
```

Output:

```
D:\New folder\array sum avg.exe
enter the number of elements
5
enter the element
2
4
6
8
10
The sum of the array is 30
the average of the array is 6

-----
Process exited after 4.676 seconds with return value 0
Press any key to continue . . .
```

Experiment 5(c)

Aim of the experiment: Write a program to find an element in an array.

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Initialise variables i , $ar[]$, n , s , c . n to store number of elements. i for loop. S to accept the number to find in the array. $Ar[]$ to store the array elements. c to check if the element was found in the array or not.

Step 4:- Using `printf()` and `scanf()` statement the number of elements is accepted.

Step 5:- Using `printf()`, for loop and `scanf()` the elements in the array are filled.

Step 6:- Using `printf()` and `scanf()` the element to be searched is accepted from the user.

Step 7:- Using for loop and if statement the element to be searched is found.

Step 8:- Using `printf()` statement the statement and number is displayed which was to be found or if the number is not found.

Step 9:- End of the program.

Program:

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int i,ar[40],a,n,s,c=0;
```

```
    printf("enter the number of elements\n");
```

```
    scanf("%d",&n);
```

```
    printf("enter the elements\n");
```

```
    for(i=0;i<n;i++)
```

```
    {
```

```
        scanf("%d",&ar[i]);
```

```
    }
```

```
    printf("enter the element to be searched\n");
```

```
    scanf("%d",&s);
```

```
    for(i=0;i<n;i++)
```

```
    {
```

```
        if(ar[i]==s)
```

```
        {
```

```
            printf("%d is present in the array",s);
```

```
            c=1;
```

```
        }
```

```
    }
```

```
    if(c==0)
```

```
    {
```

```
        printf("%d is not present in the array",s);
```

```
    }
```

```
}
```

Output:

```
D:\New folder\search in array.exe
enter the number of elements
7
enter the elements
2
5
8
7
4
9
47
enter the element to be searched
7
7 is present in the array
-----
Process exited after 15.92 seconds with return value 0
Press any key to continue . . .
```

Experiment 5(d)

Aim of the experiment: Write a program to multiply two matrices.

Procedure:

Step 1:- Start the program.

Step 2:- Start the main function.

Step 3:- Initialise variables ar1[], ar2[], ar3[], c1, r1, c2, r2, l, j, k. ar1[], ar2[] and ar3[] to store the arrays. C1, r1, c2, r2 to store the number of rows and columns of the array. l, j, k for the loops.

Step 4:- using scanf() and printf() statement the number of rows and columns for both the matrix.

Step 5:- using for loops the elements is filled in both the matrix.

Step 6:- Using for loops both the matrix are displayed.

Step 7:- Using the formulas the matrix is multiplied.

Step 8:- Using printf() and for loop the multiplied matrix is displayed.

Step 9:- End of the program.

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

Program:

```
#include<stdio.h>

int main()
{
    int ar1[40][40],ar2[40][40],ar3[40][40],c1,r1,c2,r2,i,j,k;
    printf("enter the number of rows and columns of the first array \n");
    scanf("%d %d",&r1,&c1);
    printf("enter the number of rows and columns of the second array \n");
    scanf("%d %d",&r2,&c2);
    printf("enter the elements in the first matrix \n");
    for(i=0;i<r1;i++)
    {
        for(j=0;j<c1;j++)
        {
            scanf("%d",&ar1[i][j]);
        }
    }
    printf("enter the elements in the second matrix \n");
    for(i=0;i<r2;i++)
    {
        for(j=0;j<c2;j++)
        {
            scanf("%d",&ar2[i][j]);
        }
    }
    printf("the first matrix is \n");
    for(i=0;i<r1;i++)
    {
        for(j=0;j<c1;j++)
        {
            printf("%d ",ar1[i][j]);
        }
    }
}
```

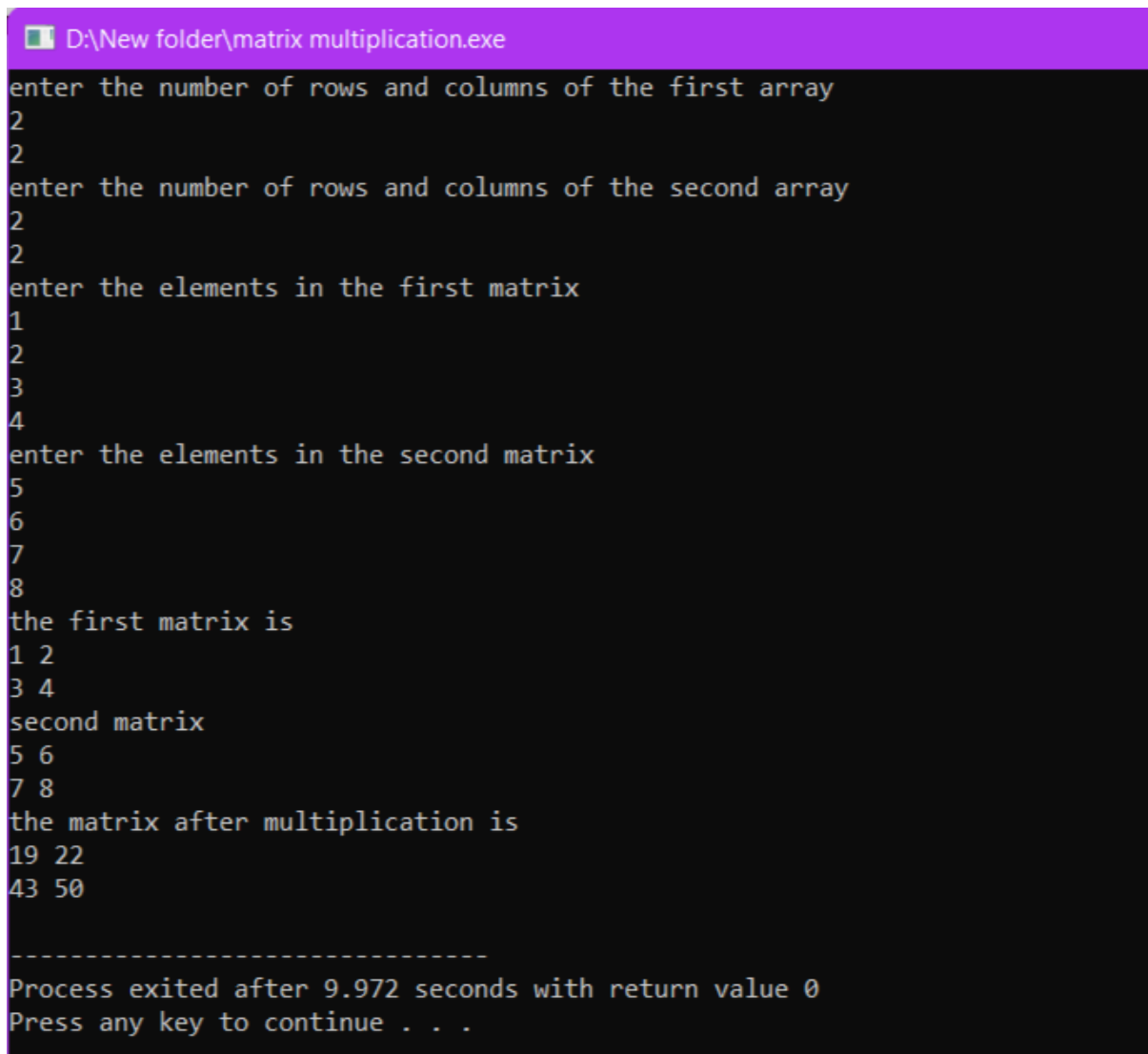
Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
    }
    printf("\n");
}
printf("second matrix \n");
for(i=0;i<r2;i++)
{
    for(j=0;j<c2;j++)
    {
        printf("%d ",ar2[i][j]);
    }
    printf("\n");
}
if(c1==r1)
{
    for(i=0;i<r1;i++)
    {
        for(j=0;j<c2;j++)
        {
            ar3[i][j]=0;
            for(k=0;k<c1;k++)
            {
                ar3[i][j]+=ar1[i][k]*ar2[k][j];
            }
        }
    }
    printf("the matrix after multiplication is \n");
    for(i=0;i<r2;i++)
    {
        for(j=0;j<c2;j++)
        {
            printf("%d ",ar3[i][j]);
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
    }  
    printf("\n");  
}  
}  
else  
{  
    printf("the matrix cannot be multiplied");  
}  
return 0;  
}
```

Output:-



```
D:\New folder\matrix multiplication.exe  
enter the number of rows and columns of the first array  
2  
2  
enter the number of rows and columns of the second array  
2  
2  
enter the elements in the first matrix  
1  
2  
3  
4  
enter the elements in the second matrix  
5  
6  
7  
8  
the first matrix is  
1 2  
3 4  
second matrix  
5 6  
7 8  
the matrix after multiplication is  
19 22  
43 50  
-----  
Process exited after 9.972 seconds with return value 0  
Press any key to continue . . .
```

Experiment-6

Program: WAP to swap or exchange two number using call by and reference.

```
#include <stdio.h>
// Function Prototype
void swapx(int x, int y);
// Main function
int main()
{
    int a = 10, b = 20;
    // Pass by Values
    swapx(a, b); // Actual Parameters

    printf("In the Caller:\na = %d b = %d\n", a, b);

    return 0;
}

// Swap functions that swaps
// two values
void swapx(int x, int y) // Formal Parameters
{
    int t;

    t = x;
    x = y;
    y = t;

    printf("Inside Function:\nx = %d y = %d\n", x, y);
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

}

Output:

Inside Function:

x = 20 y = 10

In the Caller:

a = 10 b = 20

```
#include <stdio.h>
```

```
// Function Prototype
```

```
void swapx(int*, int*);
```

```
// Main function
```

```
int main()
```

```
{
```

```
    int a = 10, b = 20;
```

```
    // Pass reference
```

```
    swapx(&a, &b); // Actual Parameters
```

```
    printf("Inside the Caller:\na = %d b = %d\n", a, b);
```

```
    return 0;
```

```
}
```

```
// Function to swap two variables
```

```
// by references
```

```
void swapx(int* x, int* y) // Formal Parameters
```

```
{
```

```
    int t;
```

```
t = *x;  
*x = *y;  
*y = t;  
  
printf("Inside the Function:\nx = %d y = %d\n", *x, *y);  
}
```

Output:

Inside the Function:

x = 20 y = 10

Inside the Caller:

a = 20 b = 10

Experiment-7

Program: Write a program to find the factorial a number using recursion function.

```
#include <stdio.h>
```

```
// function to find factorial of given number
```

```
int factorial(int n)
```

```
{
```

```
    int res = 1, i;
```

```
    for (i = 2; i <= n; i++)
```

```
        res *= i;
```

```
    return res;
```

```
}
```

```
int main()
```

```
{
```

```
    int num = 5;
```

```
    printf("Factorial of %d is %d", num, factorial(num));
```

```
    return 0;
```

```
}
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

Program: WAP to find HCF using recursion

```
#include <stdio.h>

int hcf (int n1, int n2);

int main () {
    int n1, n2;
    printf("Enter two positive integers: ");
    scanf("%d %d", &n1, &n2);
    printf("G.C.D of %d and %d is %d.", n1, n2, hcf(n1, n2));
    return 0;
}

int hcf(int n1, int n2) {
    if (n2 != 0)
        return hcf(n2, n1 % n2);
    else
        return n1;
}
```

Output:

Enter two positive integers: 366

60

G.C.D of 366 and 60 is 6.

Experiment- 8

Aim of Experiment: Write a program to check a string is a palindrome or not

Procedure:

1. Start
2. Declare a string variable and input a string from the user
3. Calculate the length of the string
4. Initialize a flag variable to 0 (This flag will be used to identify if the string is a palindrome or not)

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

5. Compare the first character and the last character of the string. If they are the same, compare the second character with the second last character, and so on. Continue this process until the middle of the string is reached.
6. If any pair of characters are not the same, set the flag variable to 1 and break the loop.
7. After the loop, check the flag variable. If the flag is 0, print "The string is a palindrome." If the flag is 1, print "The string is not a palindrome."
8. End

```
#include <stdio.h>
#include <string.h>

int main() {
    char string[100], rev_string[100];

    printf("Enter a string: ");
    gets(string);

    strcpy(rev_string, string);
    strrev(rev_string);

    if(strcmp(string, rev_string) == 0)
        printf("%s is a palindrome string.\n", string);
    else
        printf("%s is not a palindrome string.\n", string);

    return 0;
}
```

b)

```
#include <stdio.h>

int main() {
    char string[100];
    int i, length, flag = 0;

    printf("Enter a string: ");
    gets(string);

    for(length = 0; string[length] != '\0'; length++);

    for(i=0; i<length/2; i++) {
        if(string[i] != string[length-i-1]) {
            flag = 1;
            break;
        }
    }
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
}  
  
if(flag == 0)  
    printf("%s is a palindrome string.\n", string);  
else  
    printf("%s is not a palindrome string.\n", string);  
  
return 0;  
}
```

Input: *str = "madam"*

Output: *"madam" is palindrome.*

Experiment-9

Aim of Experiment: WAP to print employee details using Structure

```
#include<stdio.h>  
  
// Define the structure for employee details  
struct Employee {  
    int id;          // Employee ID  
    char name[50];  // Employee Name  
    float salary;   // Employee Salary  
};  
  
int main() {  
    // Declare a variable of type struct Employee  
    struct Employee emp;  
  
    // Prompt the user to enter the employee's ID and store it in emp.id  
    printf("Enter Employee ID: ");  
    scanf("%d", &emp.id);  
  
    // Prompt the user to enter the employee's name and store it in emp.name  
    // Note: %s reads a string until a space or newline is encountered  
    printf("Enter Employee Name: ");  
    scanf("%s", emp.name);  
  
    // Prompt the user to enter the employee's salary and store it in emp.salary  
    printf("Enter Employee Salary: ");  
    scanf("%f", &emp.salary);  
  
    // Display the employee's details  
    printf("\nEmployee Details:\n");  
  
    // Print the employee's ID  
    printf("ID: %d\n", emp.id);  
  
    // Print the employee's name
```

Prepared by Dr. Gopal Behera, Asst. Professor, Department of CSE

```
printf("Name: %s\n", emp.name);

// Print the employee's salary
printf("Salary: %.2f\n", emp.salary);

// Indicate that the program ended successfully
return 0;
}
```

Experiment-10

Aim of Experiment: convert experiment 9 into using union