DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering) Shahbad Daulatpur, Bawana Road, Delhi 110042

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

 s

**CO102: Programming Fundamentals**

**Lab File**

Submitted To: Submitted By:

**Dr. Ashish Girdhar Sanskar Verma**

**Assistant Professor B.Tech (ME) 1st Year**

**Department of Computer Roll NO:2K20/A10/62**

Science and Engineering

**INDEX**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **S.NO** | **TOPIC** | **DATE** | **SIGNATURE** | | 1. | Program to find the sum and average of two numbers. |  |  | | 2. | Program to find the greatest of two numbers |  |  | | 3. | Program to find the simple interest |  |  | | 4. | Program to print the following pattern. (triangle of stars) |  |  | | 5. | Program to find whether the entered number is prime |  |  | | 6. | Program to find the sum of a five-digit number. |  |  | | 7. | Program to reverse a five-digit number. |  |  | | 8. | Program to convert decimal to binary and vice versa |  |  | | 9. | Program to implement switch case statement |  |  | | 10 | Program to generate the Fibonacci sequence |  |  | | 11. | Program to find the exponential function |  |  | | 12. | Program to search a number from an array using linear search |  |  | | 13. | Program to search a number from array using binary search |  |  | | 14. | Program to sort an array using bubble sort |  |  | | 15. | Program to sort an array using selection search |  |  | | 16. | Program to sort an array using insertion sort |  |  | | 17. | Program to find factorial of a number using recursion |  |  | | 18. | Program to find the length of the string without using strlen and then pass the string to characters |  |  | | 19. | Program to count the number of vowels in a given string |  |  | | 20. | Program to check if a given string is palindrome or not |  |  | | 21. | Program to string concatenation. |  |  | | 22. | Program to string comparison. |  |  | | 23. | Program to string reverse. |  |  | | 24. | Program to convert a string from lower case to upper case and vice versa. |  |  | | 25. | Program for addition of two 3 x 3 matrices |  |  | | 26. | Program to multiply two 3 x 3 matrices |  |  | | 27. | Program to swap two numbers using pointers. |  |  | | 28. | Program to generate employee details using structure |  |  | | 29. | Program to find the area and perimeter of a circle, square, rectangle and triangle using functions |  |  | | 30. | Program to pass and return pointer to a function hence calculate average of an array |  |  | | 31. | Program to pass an array as pointer to a function that calculates the sum of all elements of array |  |  | | 32. | Program to demonstrate the example of array of pointers |  |  | | 33. | Program which copies one file contents to other |  |  | | 34. | Program to find size of a given file |  |  | | 35. | Program to read a file and after converting all lower case to upper case letters write it to another file. |  |  | | 36. | Program to find the size of a given file. |  |  | |

**EXPERIMENT-1**

**OBJECTIVE-** Program to find the sum and average of two numbers.

**INTRODUCTION-** To find the sum of two numbers we use the ‘+’ operator, and for average we divide the sum of the numbers using ‘/’operator.

**PROGRAM CODE-**

#include<stdio.h>

int main()

{

int a,b,c,d;

printf("Enter the value of a :");

scanf("%d",&a);

printf("Enter the value of b : ");

scanf("%d",&b);

c=a+b; // variable c stores the sum of a and b

printf("Sum of a and b is = %d ",c);

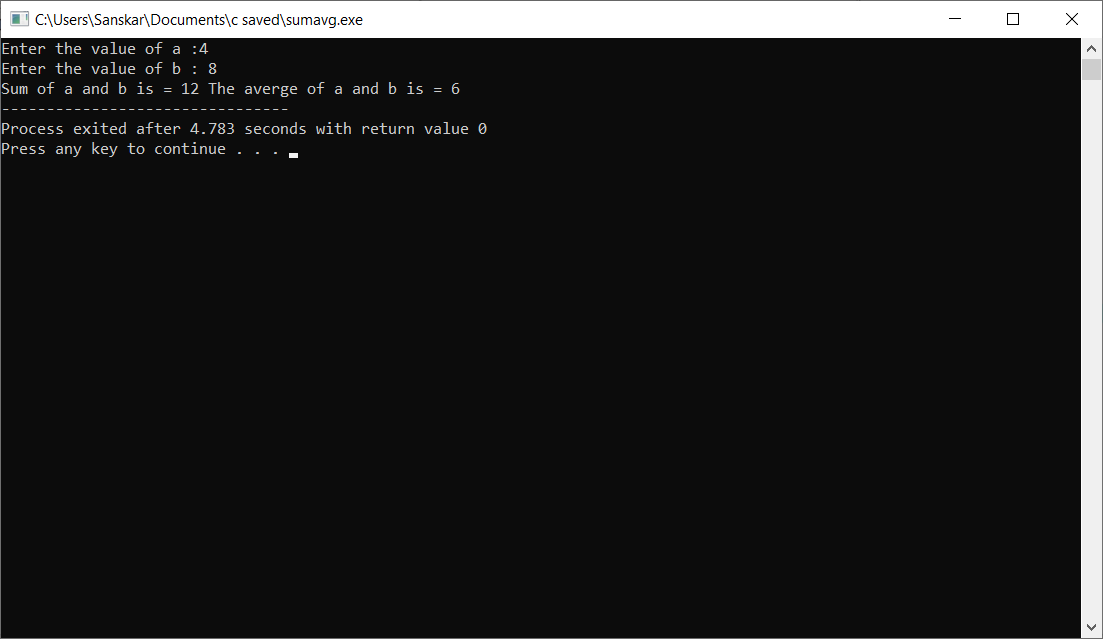
d= c/2; // variable d stores the value of average of a and b

printf("The averge of a and b is = %d",d);

return 0;

}

**OUTPUT-**



**EXPERIMENT-2**

**OBJECTIVE-**Program to find the greatest of 10 numbers.

**INTRODUCTION-** To take input of 10 numbers, we make an 1D integer array (contains all elements of same data type) with 10 elements. Then we use a loop to input the numbers. After taking input we assume the first element (a [0]) to be the greatest, then we use a loop to compare the greatest value with the rest of the array elements, if any element is greater than the assumed greatest, we update the value of the greatest variable, we run this loop till the end of the array.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

int a[10]; // taking array of 10 elements

int i;

int greatest;

printf("Enter ten values:");

//Store 10 numbers in an array

for (i = 0; i < 10; i++)

{

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

}

//Assuming that a[0] is greatest

greatest = a[0];

for (i = 0; i < 10; i++) {

if (a[i] > greatest)

{

greatest = a[i];

}

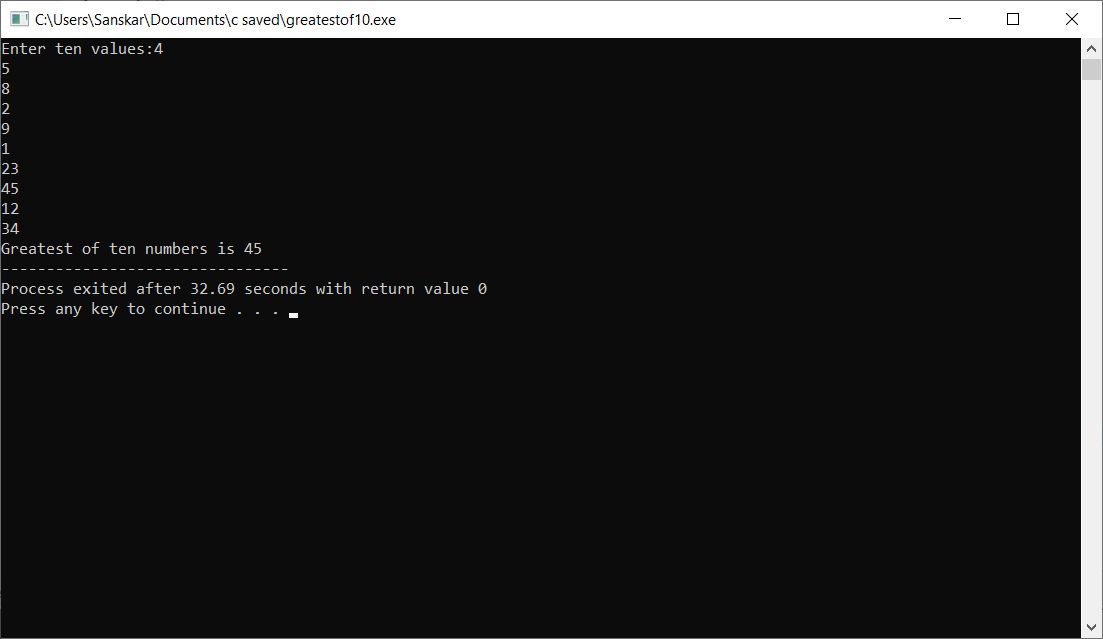
}

printf("Greatest of ten numbers is %d", greatest);

return 0;

}

**OUTPUT-**



**EXPERIMENT-3**

**OBJECTIVE-**Program to find the Simple Interest

**INTRODUCTION-** The formula for simple interest is (p\*r\*t)/100 [p is the principal amount, r is the rate, t is the time period, we use the ‘\*’and ‘/’operators.

**PROGRAM CODE-**

#include<stdio.h>

int main()

{

int p,r,t,si;

printf("Enter the principal amount : ");

scanf("%d",&p);

printf("Enter the rate per annum : ");

scanf("%d",&r);

printf("Enter the time : ");

scanf("%d",&t);

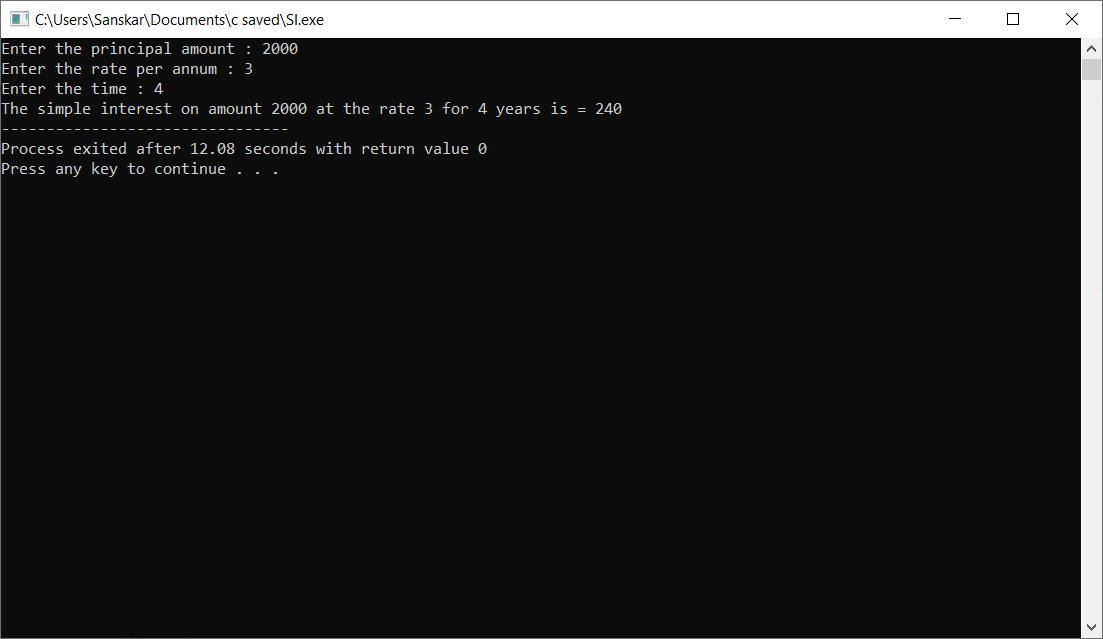
si = (p\*r\*t)/100;

printf("The simple interest on amount %d at the rate %d for %d years is = %d",p,r,t,si);

return 0;

}

**OUTPUT-**



**EXPERIMENT-4**

**OBJECTIVE-** Program to print the triangle of stars

**INTRODUCTION-** In this program we use the concept of nested loops. The main(outer) loop runs from 1 till the value of rows that user has input, and the inner loop runs from 1 to ‘i’ for every value of i.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

int i, j, rows;

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

scanf("%d", &rows); //taking input from user

for (i = 1; i <= rows; ++i)

{

for (j = 1; j <= i; ++j) //using for loop to print the stars

{

printf("\*");

}

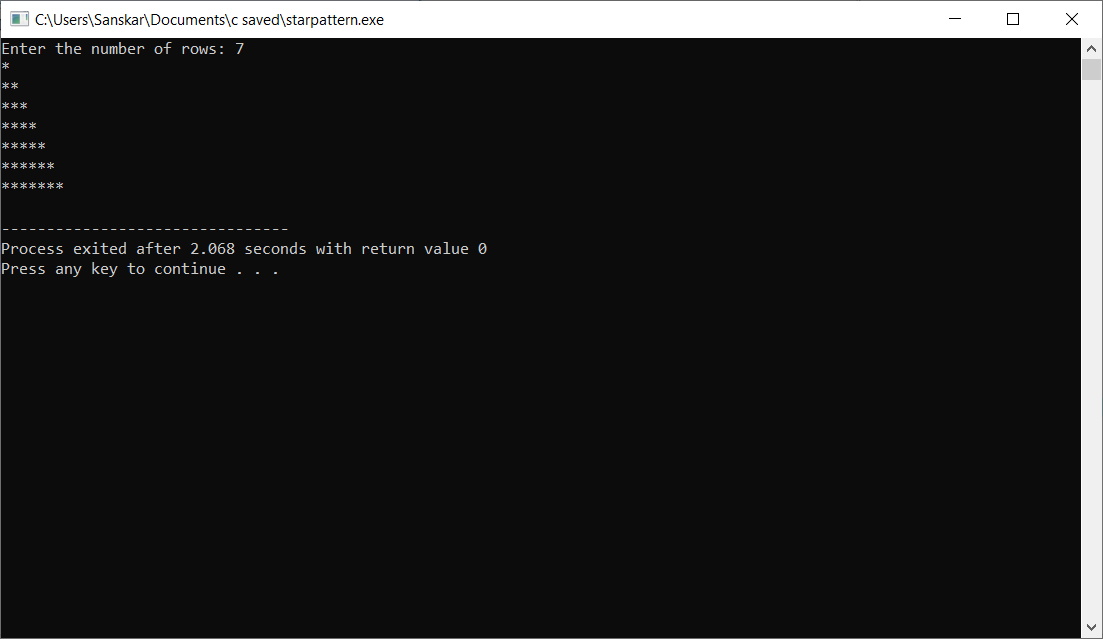
printf("\n");

}

return 0;

}

**OUTPUT-**



**EXPERIMENT-5**

**OBJECTIVE-** Program to find whether the given number is prime

**INTRODUCTION-** To check whether the given number (n) is prime or not, we divide the number by all integral values from 1 to n/2, if the remainder for any one of these cases comes out to be 0, then the number is not prime, otherwise it is prime. Special case – when n=1, the number the neither prime nor composite.

**PROGRAM CODE-**

#include <stdio.h>

int main() {

int n, i, num = 0;

printf("Enter a positive integer: ");

scanf("%d", &n);

for (i = 2; i <= n / 2; ++i) {

// condition for non-prime

if (n % i == 0) // loop to find if their is any factor

{

num = 1;

break;

}

}

if (n == 1) {

printf("1 is neither prime nor composite.");

}

else {

if (num == 0)

printf("%d is a prime number.", n);

else

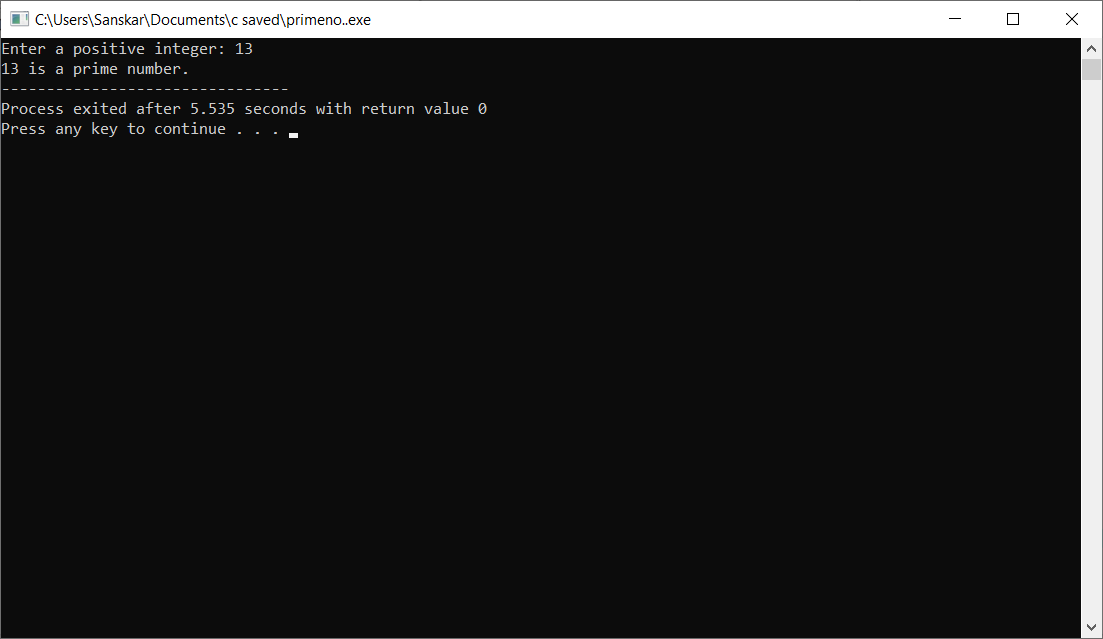
printf("%d is not a prime number.", n);

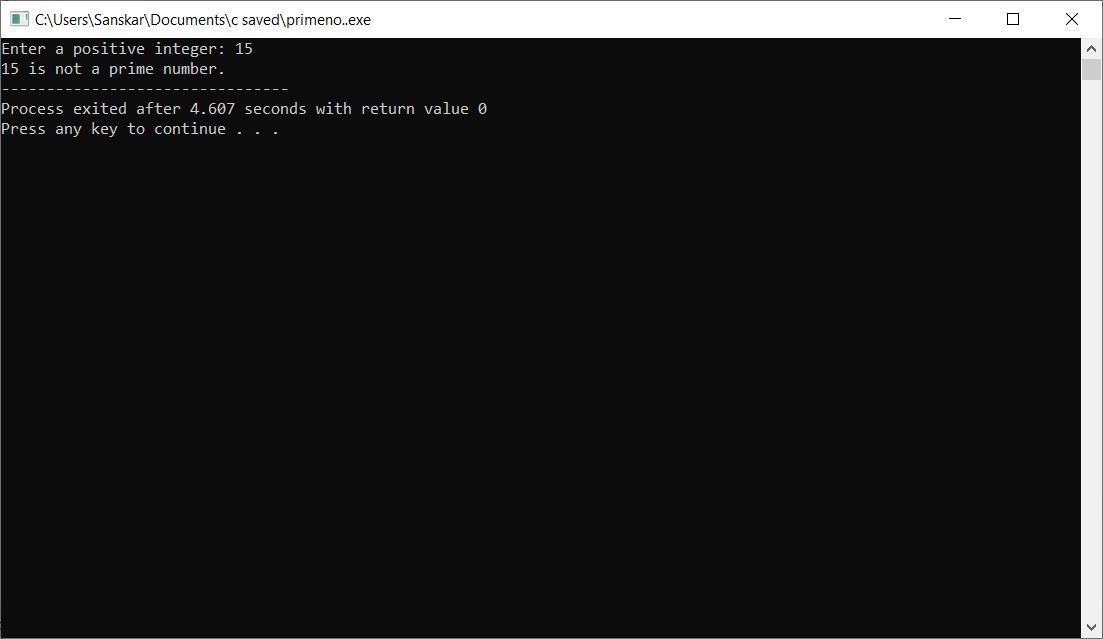
}

return 0;

}

**OUTPUT-**





**EXPERIMENT-6**

**OBJECTIVE-**Program to find the sum of a five-digit number

**INTRODUCTION-** To find the sum of digits of a five-digit number, we will divide the number by 10 then store the value of the remainder in ‘sum’ variable, then we will keep on repeating this loop and keep updating the value of sum as sum = sum + remainder, till we get the remainder = 0.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

int n, t, sum = 0, remainder;

printf("Enter an integer\n");

scanf("%d", &n);

t = n;

while (t != 0)

{

remainder = t % 10;

sum = sum + remainder;

t = t / 10;

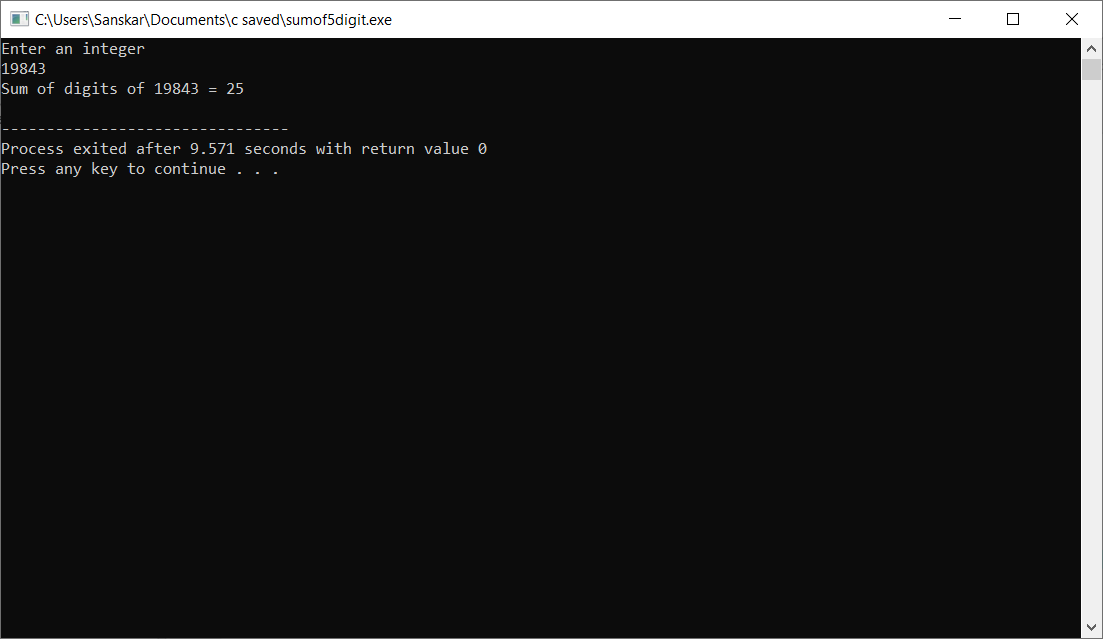
­­­}

printf("Sum of digits of %d = %d\n", n, sum);

return 0;

}

**OUTPUT-**



**EXPERIMENT-7**

**OBJECTIVE-**Program to reverse a five-digit number

**INTRODUCTION-** To reverse a five-digit number, we divide it by 10 (n =n/10) then we put variable reverse = remainder, then we divide the number by 10 again and then put reverse = reverse\*10 + remainder, and we keep on repeating this loop till n/10 = 0

**PROGRAM CODE-**

#include<stdio.h>

int main()

{

int n, reverse=0, rem;

printf("Enter a number: ");

scanf("%d", &n);

while(n!=0)

{

rem=n%10;

reverse=reverse\*10+rem;

n/=10;

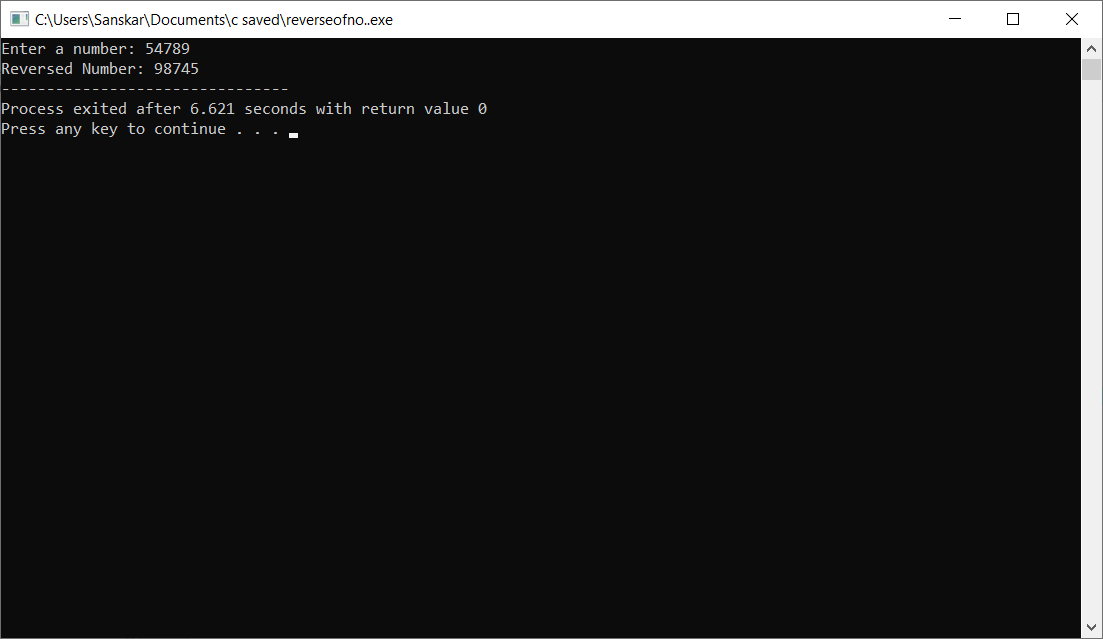
}

printf("Reversed Number: %d",reverse);

return 0;

}

**OUTPUT-**



**EXPERIMENT-8**

**OBJECTIVE –** Program to convert decimal to binary and vice versa

**INTRODUCTION –** Decimal to Binary conversion involves a number getting divided by 2 and the remainder noted, now the quotient of previous calculation is treated as a decimal and the process is repeated. As long as quotient is either 0 or 1, this process is repeated. The remainders are noted together and the reverse of this is the binary form of given decimal number. Binary to Decimal conversion: The total number of terms in binary is noted, say n. To get decimal, the first term is multiplied by 2n-1 , second by 2n-2 , and so on till n=1. The sum of all these is taken and the resulting number is the decimal

**PROGRAM CODE –**

**Binary to decimal-**

#include <math.h>

#include <stdio.h>

int convert(long long n);

int main() {

long long n;

printf("Enter a binary number: "); // enter binary number

scanf("%lld", &n);

printf("%lld in binary = %d in decimal", n, convert(n));

return 0;

}

int convert(long long n) {

int dec = 0, i = 0, rem;

while (n != 0) {

rem = n % 10;

n /= 10;

dec += rem \* pow(2, i);

++i;

}

return dec; // return the decimal value

}

**Decimal to binary –**

#include <math.h>

#include <stdio.h>

long long convert(int a);

int main() {

int a;

printf("Enter a decimal number: "); // enter decimal value

scanf("%d", &a);

printf("\n %d in decimal = %lld in binary", a, convert(a));

return 0;

}

long long convert(int a) {

long long binary = 0;

int rem, i = 1, step = 1;

while (a != 0) {

rem = a % 2;

printf(" \n Step %d: %d/2, Remainder = %d, Quotient = %d\a", step++, a, rem, a / 2);

a /= 2;

binary += rem \* i;

i \*= 10;

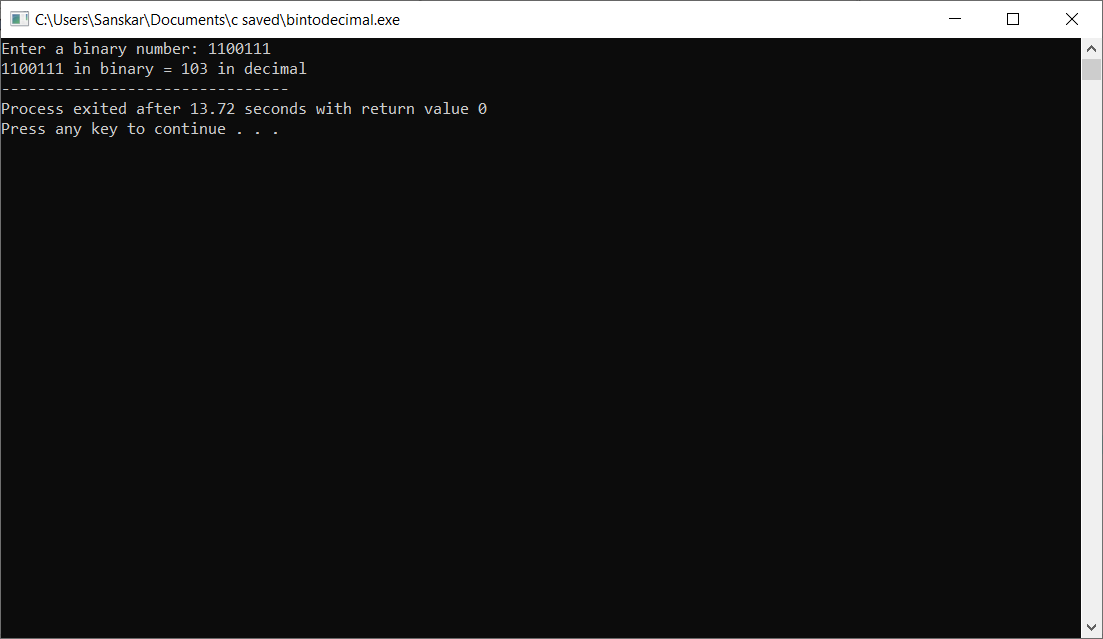
}

return binary; // returning binary value

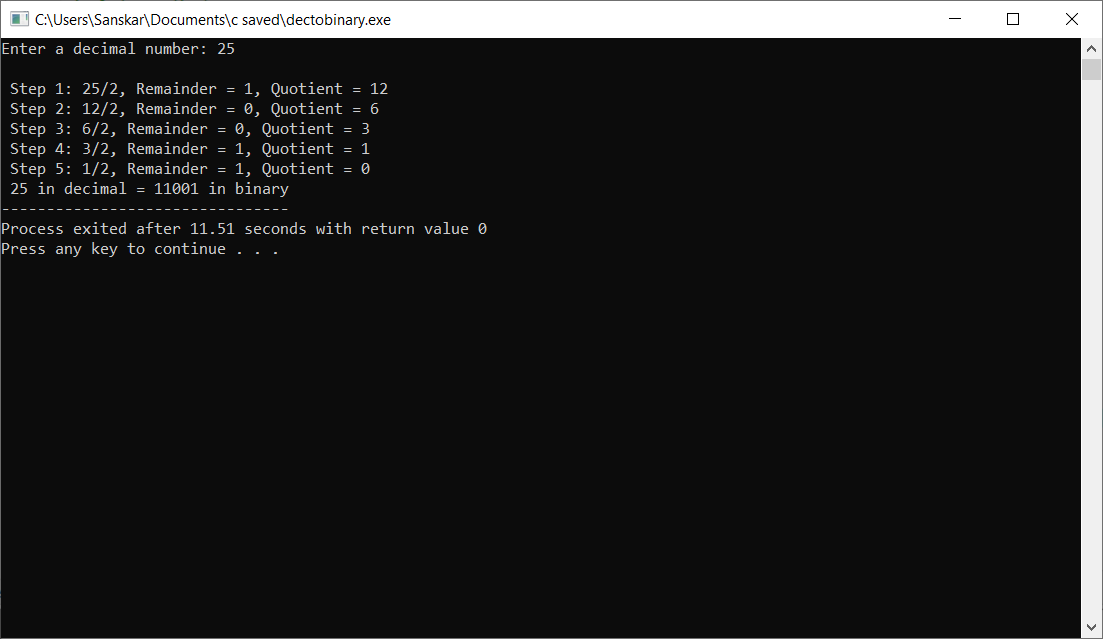
}

**OUTPUT –**

**Binary to decimal-**



**Decimal to binary-**



**EXPERIMENT-9**

**OBJECTIVE –** Program to implement switch case statement

**INTRODUCTION –** A switch allows a variable to be tested for equality against a list of values. Each value is called case, and the variable being switched on is checked for each switch case. The break; statement causes an immediate exit from the switch.

**PROGRAM CODE –**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int a,b,op;

printf(" 1.Addition\n 2.Subtraction\n 3.Multiplication\n 4.Division\n");

printf("Enter the values of a & b: ");

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

printf("Enter your Choice : ");

scanf("%d",&op);

switch(op)

{

case 1 :

printf("Sum of %d and %d is : %d",a,b,a+b);

break;

case 2 :

printf("Difference of %d and %d is : %d",a,b,a-b);

break;

case 3 :

printf("Multiplication of %d and %d is : %d",a,b,a\*b);

break;

case 4 :

printf("Division of Two Numbers is %d : ",a/b);

break;

default :

printf(" Enter Correct Choice.");

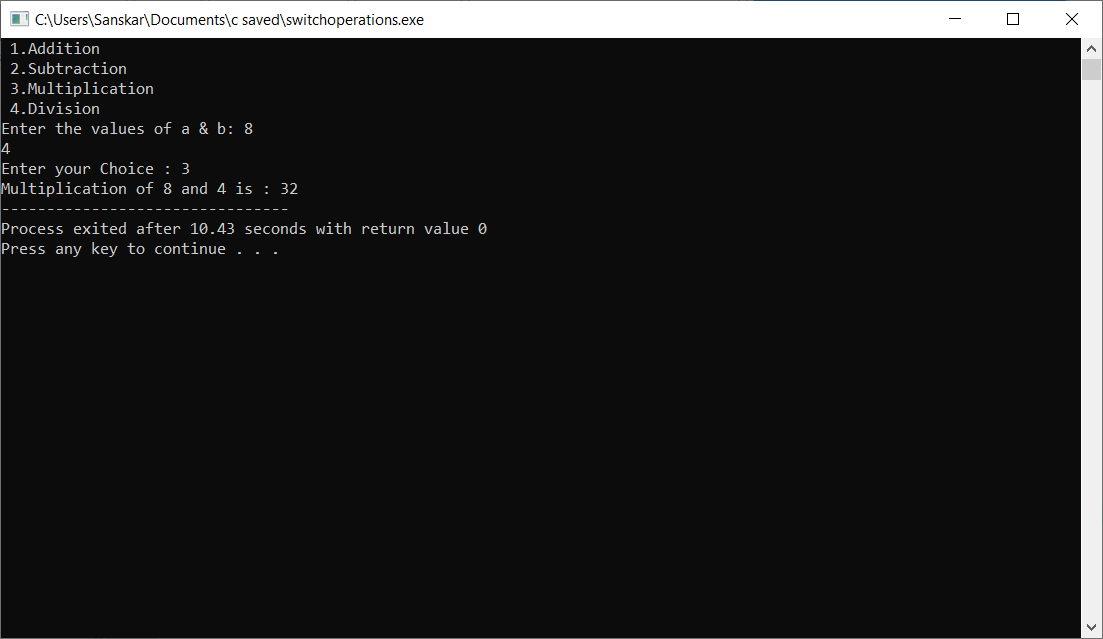
break;

}

return 0;

}

**OUTPUT –**



**EXPERIMENT-10**

**OBJECTIVE –** Program to generate the Fibonacci sequence

**INTRODUCTION –** Fibonacci sequence is a sequence in which each n umber is the sum of the preceding two numbers. This series starts from 0 and 1

**PROGRAM CODE –**

#include <stdio.h>

int main() {

int i, n, t1 = 0, t2 = 1, nextTerm;

printf("Enter the number of terms: "); // number of terms

scanf("%d", &n);

printf("Fibonacci Series: ");

for (i = 1; i <= n; ++i) // loop to generate series

{

printf("%d, ", t1);

nextTerm = t1 + t2;

t1 = t2;

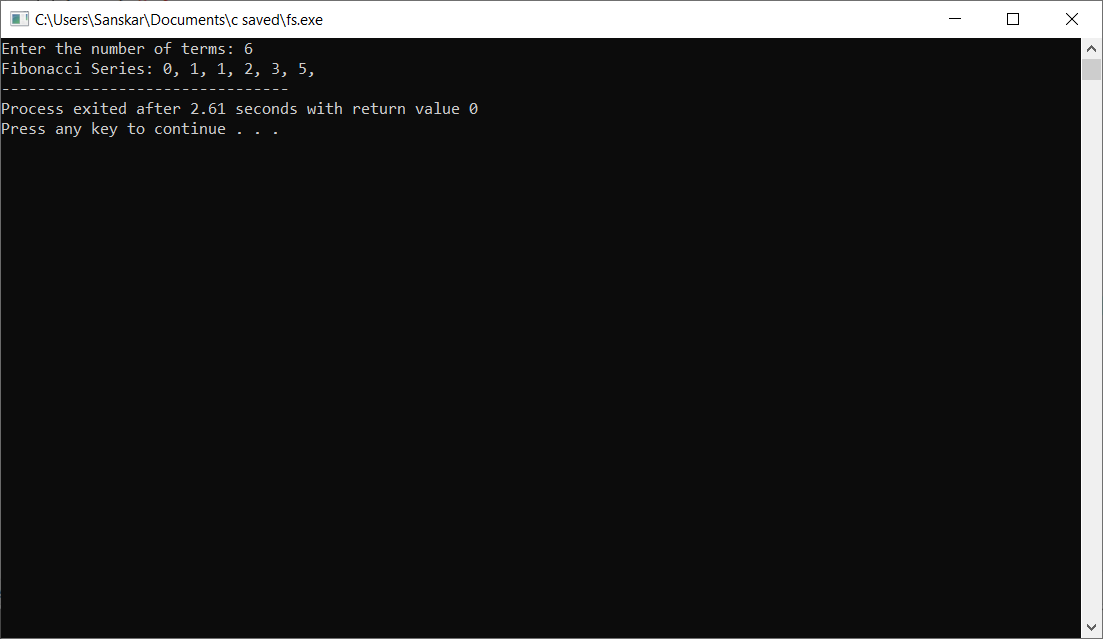
t2 = nextTerm;

}

return 0;

}

**OUTPUT –**



**EXPERIMENT-11**

**OBJECTIVE –** To find exponential function

**INTRODUCTION –** We use the header file <math.h> which contains the declaration for mathematical operations.

**PROGRAM CODE –**

#include<stdio.h>

#include<math.h>

int main()

{

double num,k;

printf("Enter a number- \n ");

scanf("%lf",&num);

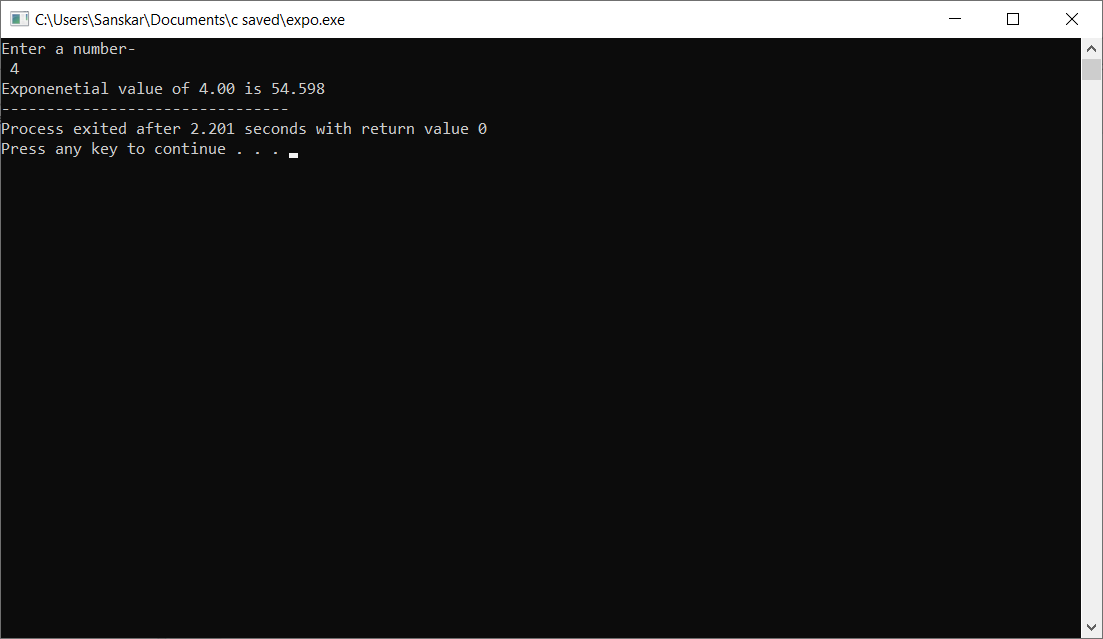
k=exp(num);

printf("Exponenetial value of %.2lf is %.3lf" ,num ,k);

return 0;

}

**OUTPUT –**



**EXPERIMENT-12**

**OBJECTIVE –** Program to search a number from an array using linear search

**INTRODUCTION –** It is an important array application. With help of it we can find numbers in big data files. In Linear search it starts searching value from starting, comparing the with the given number and stops when the desired value is found.

**PROGRAM CODE –**

#include <stdio.h>

void main()

{ int num;

int i, tosearch, found = 0;

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

scanf("%d", &num);

int array[num];

printf("Enter the elements one by one : \n");

for (i = 0; i < num; i++)

{

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

}

printf("Enter the element to be searched- ");

scanf("%d", &tosearch);

/\* Linear search begins \*/

for (i = 0; i < num ; i++)

{

if (tosearch == array[i] )

{

found = 1;

break;

}

}

if (found == 1)

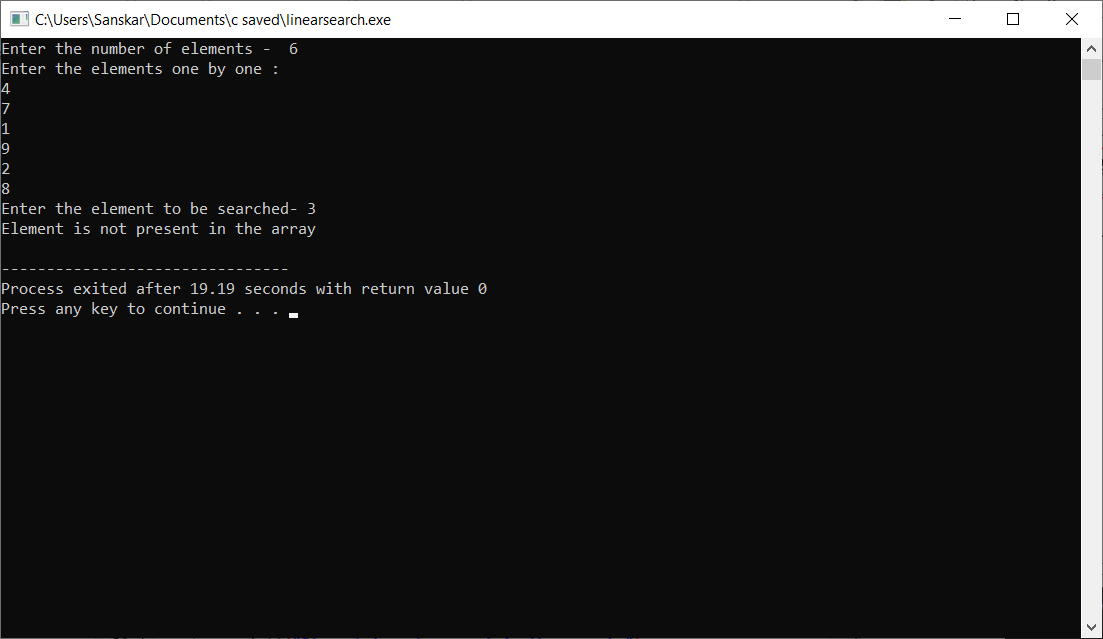
printf("Element is present in the array at position %d",i+1);

else

printf("Element is not present in the array\n");

}

**OUTPUT –**



**EXPERIMENT-13**

**OBJECTIVE -**Program to search a number from an array using binary search

**INTRODUCTION –** In Binary Search, we search a sorted array by repeatedly dividing the search interval in half. Begin with an interval covering the whole array. If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise narrow it to the upper half. Repeatedly check until the value is found or the interval is empty.

**PROGRAM CODE –**

#include <stdio.h>

int main()

{

int c, first, last, middle, n, tosearch, array[100];

printf("Enter number of elements\n");

scanf("%d", &n);

printf("Enter %d integers\n", n);

for (c = 0; c < n; c++)

scanf("%d", &array[c]);

printf("Enter value to find\n");

scanf("%d", &tosearch);

first = 0;

last = n - 1;

middle = (first+last)/2;

while (first <= last) {

if (array[middle] < tosearch)

first = middle + 1;

else if (array[middle] == tosearch) {

printf("%d found at location %d.\n", tosearch, middle+1);

break;

}

else

last = middle - 1;

middle = (first + last)/2;

}

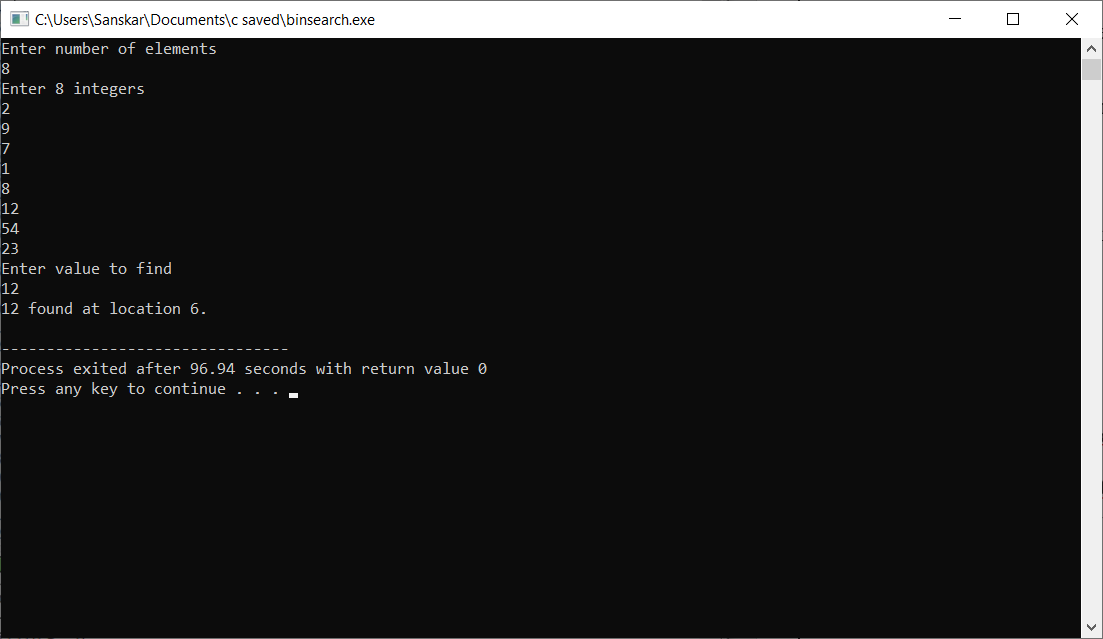
if (first > last)

printf("Not found! %d isn't present in the list.\n", tosearch);

return 0;

}

**OUTPUT –**



**EXPERIMENT-14**

**OBJECTIVE –** Program to sort an array using bubble sort

**INTRODUCTION -**Bubble Sort is the simplest sorting algorithm that works by repeatedly

swapping the adjacent elements if they are in the wrong order.

**PROGRAM CODE –**

#include <stdio.h>

int main()

{

int array[100], n, c, d, swap;

printf("Enter number of elements\n");

scanf("%d", &n);

printf("Enter %d integers\n", n);

for (c = 0; c < n; c++)

scanf("%d", &array[c]);

for (c = 0 ; c < n - 1; c++)

{

for (d = 0 ; d < n - c - 1; d++)

{

if (array[d] > array[d+1]) /\* For decreasing order use '<' instead of '>' \*/

{

swap = array[d];

array[d] = array[d+1];

array[d+1] = swap;

}

}

}

printf("Sorted list in ascending order:\n");

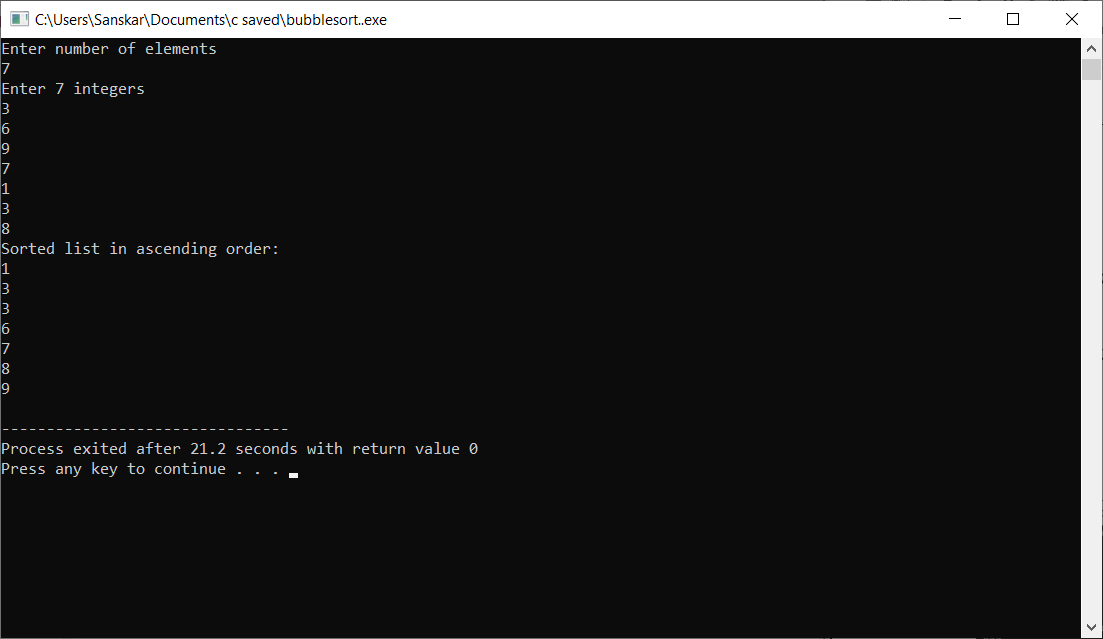
for (c = 0; c < n; c++)

printf("%d\n", array[c]);

return 0;

}

**OUTPUT –**



**EXPERIMENT-15**

**OBJECTIVE-** Program to sort an array using selection sort

**INTRODUCTION-**The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning. The algorithm maintains two subarrays in a given array.

1) The subarray which is already sorted.  
2) Remaining subarray which is unsorted.

In every iteration of selection sort, the minimum element (considering ascending order) from the unsorted subarray is picked and moved to the sorted subarray.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

int a[100], n, i, j, position, swap;

printf("Enter number of elements-\n");

scanf("%d", &n);

printf("Enter %d Numbers- \n" , n);

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

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

for(i = 0; i < n - 1; i++)

{

position=i;

for(j = i + 1; j < n; j++)

{

if(a[position] > a[j])

position=j;

}

if(position != i)

{

swap=a[i];

a[i]=a[position];

a[position]=swap;

}

}

printf("Sorted Array: ");

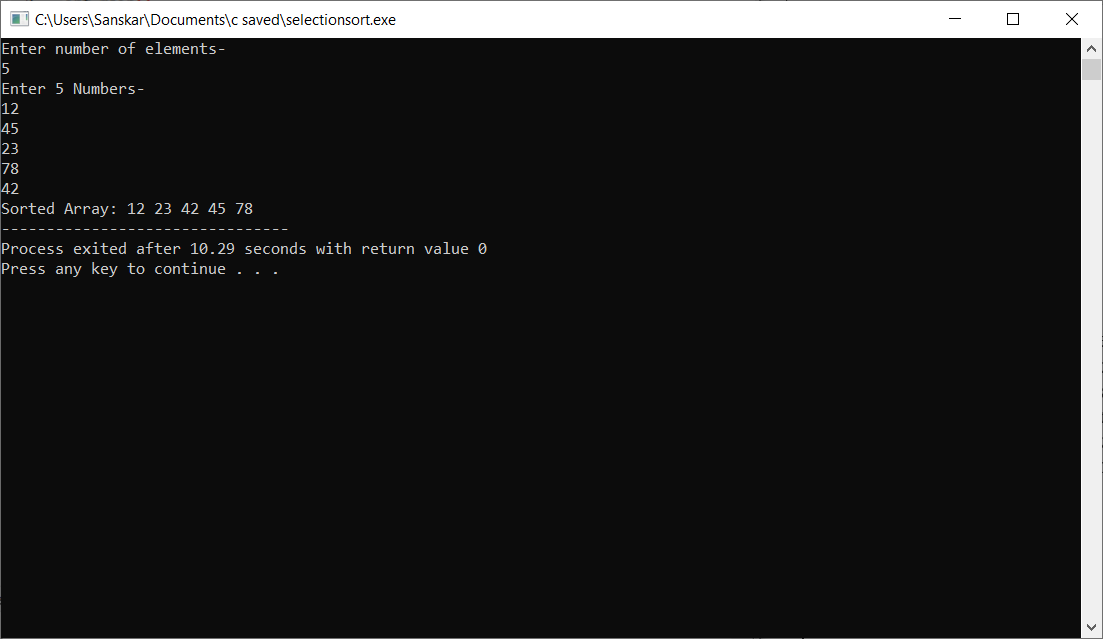
for(i = 0; i < n; i++)

printf("%d ", a[i]);

return 0;

}

**OUTPUT-**



**EXPERIMENT-16**

**OBJECTIVE –** Program to sort an array using insertion sort.

**INTRODUCTION -**Insertion sort algorithm picks elements one by one and places it to the right position where it belongs in the sorted list of elements.

**PROGRAM CODE –**

#include <stdio.h>

int main()

{

int n, i, j, temp;

int arr[64];

printf("Enter number of elements\n");

scanf("%d", &n);

printf("Enter %d integers\n", n);

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

{

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

}

for (i = 1 ; i <= n - 1; i++)

{

j = i;

while ( j > 0 && arr[j-1] > arr[j])

{

temp = arr[j];

arr[j] = arr[j-1];

arr[j-1] = temp;

j--;

}

}

printf("Sorted list in ascending order:\n");

for (i = 0; i <= n - 1; i++)

{

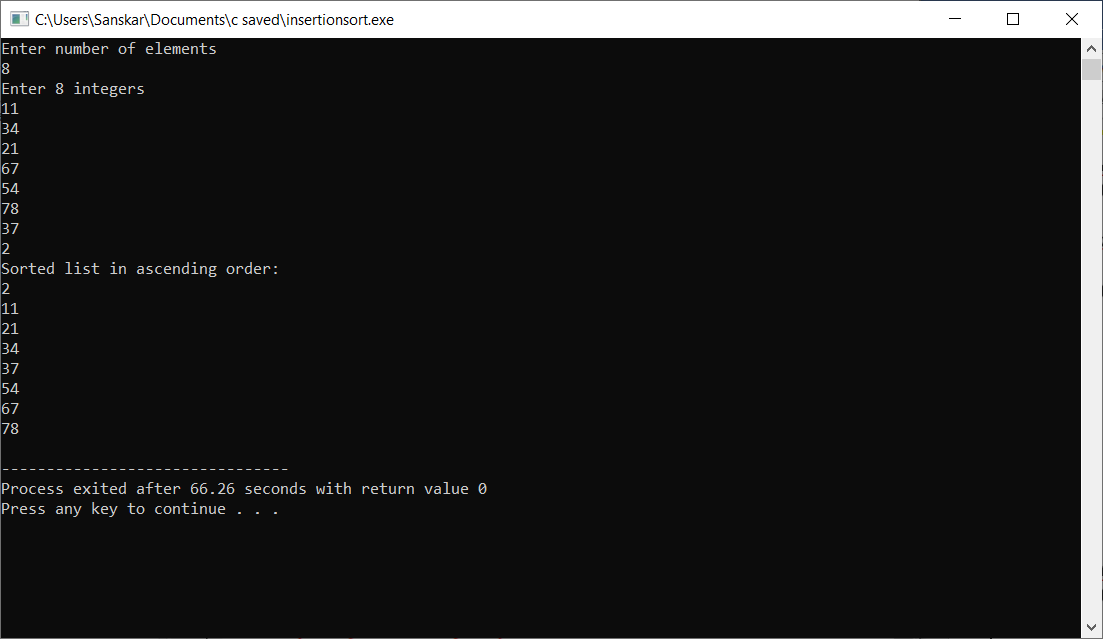
printf("%d\n", arr[i]);

}

return 0;

}

**OUTPUT –**



**EXPERIMENT-17**

**OBJECTIVE-** Program to find factorial of a number using recursion

**INTRODUCTION –** n! means the product of all integers from 1 to n. To find the factorial we will use the concept of recursion in this program.

**PROGRAM CODE -**

#include<stdio.h>

long int multiplyNumbers(int n); // function declaration

int main() {

int n;

printf("Enter a positive integer: ");

scanf("%d",&n);

printf("Factorial of %d = %ld", n, multiplyNumbers(n));

return 0;

}

long int multiplyNumbers(int n) // function defination

{

if (n>=1)

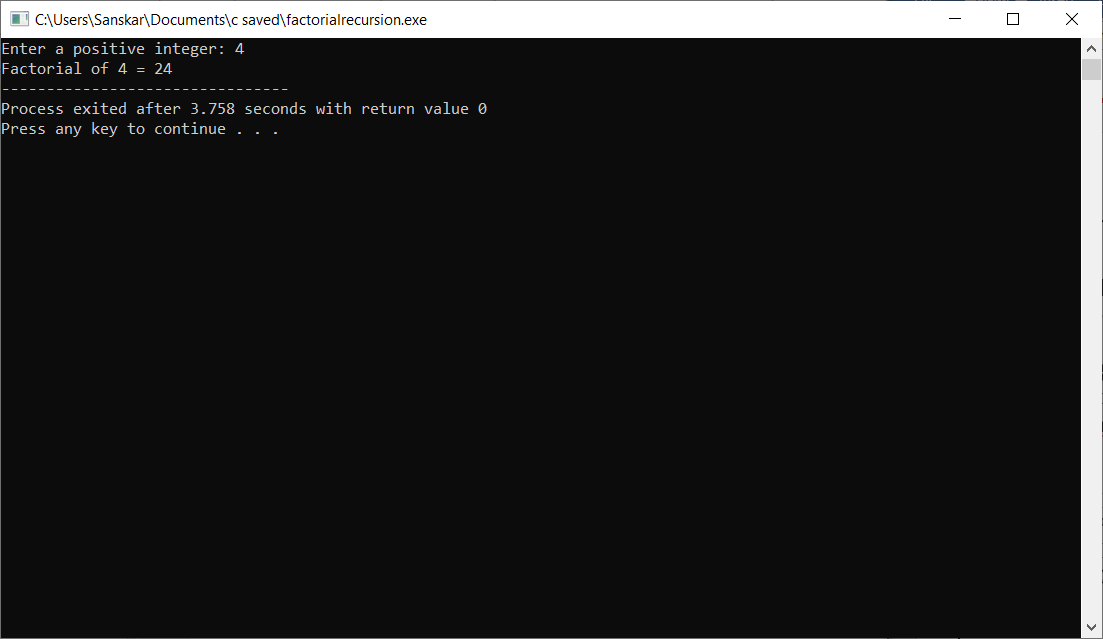
return n\*multiplyNumbers(n-1);

else

return 1;

}

**OUTPUT –**



**EXPERIMENT-18**

**OBJECTIVE -** Program to find the length of the string without using strlen and then pass the string to characters

**INTRODUCTION –** A string is a null terminated character array. We can find the length of a string using strlen function in C. However to do this without using the function , we need to use condition that a string ends with a null character.

**PROGRAM CODE –**

#include <stdio.h>

#include <string.h>

int main()

{

char Str[1000];

int i;

printf("Enter the String: ");

gets(Str) ;

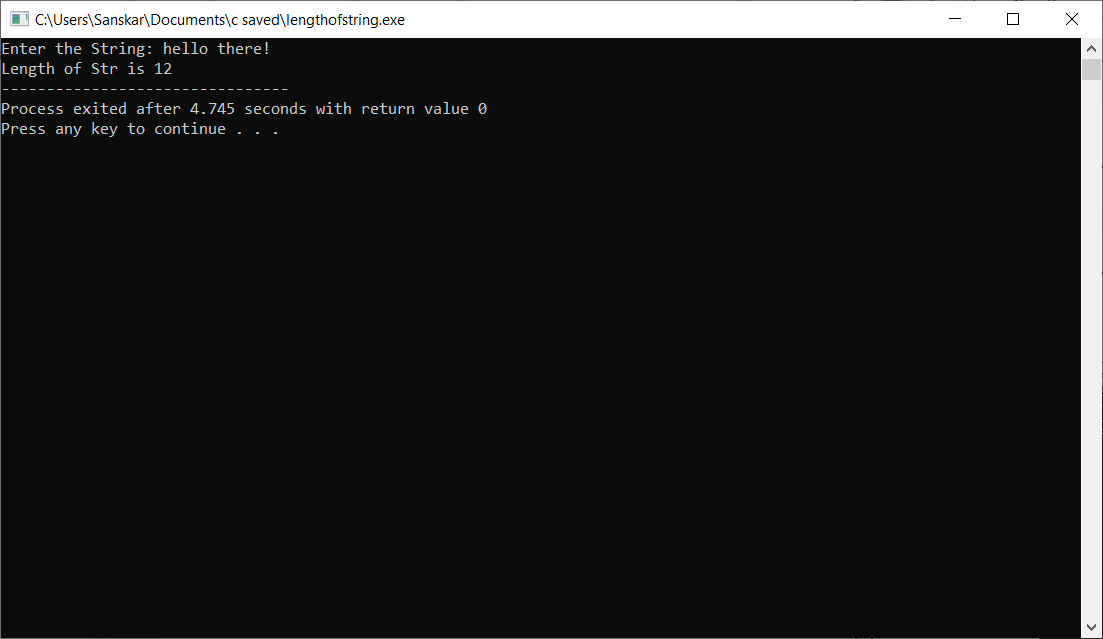
for (i = 0; Str[i] != '\0'; ++i);

printf("Length of Str is %d", i);

return 0;

}

**OUTPUT –**



**EXPERIMENT-19**

**OBJECTIVE -** Program to count the number of vowels in a given string.

**INTRODUCTION –** In English alphabets there are 5 vowels a , e, i , o u (A,E,I,O,U). In this program we will run a loop from starting of the string till the end and count all the vowels present. We will compare the characters with vowels and we will increment the count variable by 1 every time they match.

**PROGRAM CODE –**

#include <stdio.h>

int main()

{

int c = 0, count = 0;

char s[1000];

printf("Input a string\n");

gets(s); // using gets so that the spaces dont create a problem

while (s[c] != '\0') {

if (s[c] == 'a' || s[c] == 'A' || s[c] == 'e' || s[c] == 'E' || s[c] == 'i' || s[c] == 'I' || s[c] =='o' ||

s[c]=='O' || s[c] == 'u' || s[c] == 'U')

count++; //count variable keeps the number of vowels

c++; // this is the array index

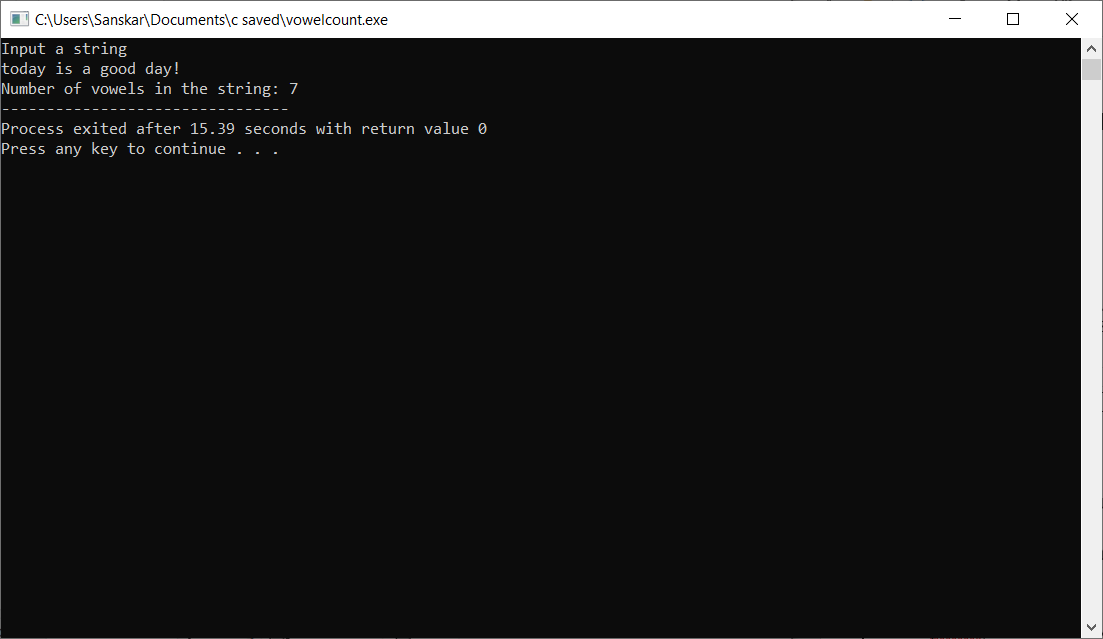
}

printf("Number of vowels in the string: %d", count);

return 0;

}

**OUTPUT –**



**EXPERIMENT-20**

**OBJECTIVE-** Program to check if the given string is a palindrome or not

**INTRODUCTION-** A palindrome is a sequence which reads the same if we read it from start to end or from end to start.

**PROGRAM CODE-**

#include <stdio.h>

#include <string.h>

int main()

{

char string1[20];

int i, length;

int flag = 0;

printf("Enter a string:");

scanf("%s", string1); // input the string to be checked

length = strlen(string1); // calculate string length

for(i=0;i < length ;i++){

if(string1[i] != string1[length-i-1])// check for palindrome , if we find anyone mismatch we exit the loop

{

flag = 1;

break;

}

}

if (flag) {

printf("%s is not a palindrome", string1);

}

else {

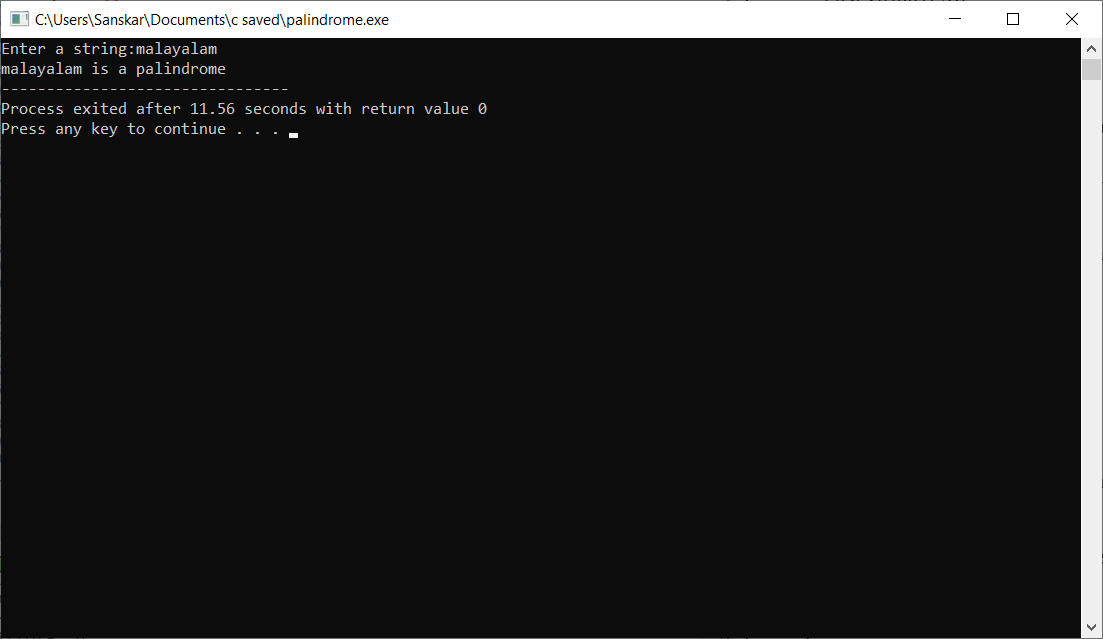
printf("%s is a palindrome", string1);

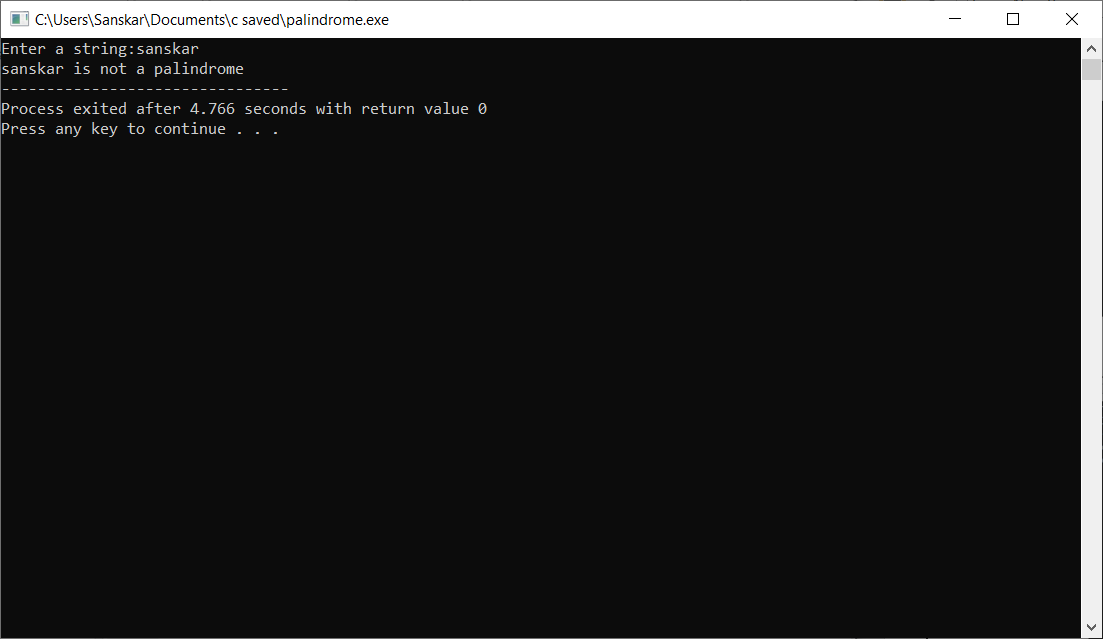
}

return 0;

}

**OUTPUT-**





**EXPERIMENT-21**

**OBJECTIVE-** Program to string concatenation

**INTRODUCTION-** String concatenation means joining 2 different strings to make a new string.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

char first\_string[20]; // declaration of char array variable

char second\_string[20]; // declaration of char array variable

int i, j; // integer variable declaration

printf("Enter the first string-\n");

gets(first\_string);

printf("\nEnter the second string-\n");

gets(second\_string);

for(i=0;first\_string[i]!='\0';i++);

{

for( j=0; second\_string[j]!='\0'; j++)

{

first\_string[i]=second\_string[j]; //joining the strings

i++;

}

}

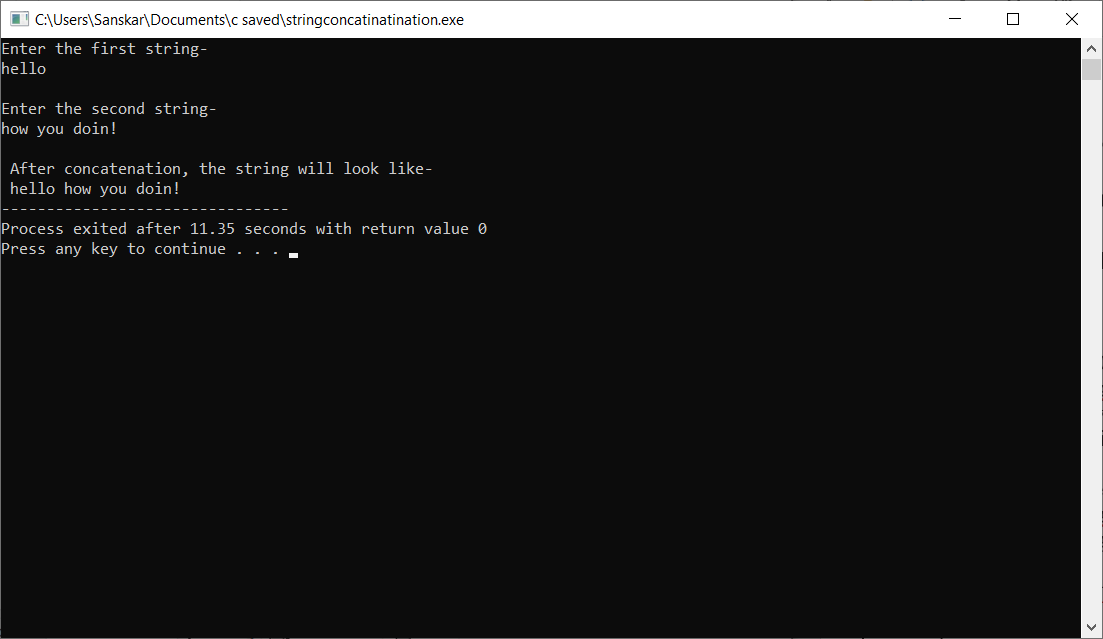
first\_string[i]='\0';

printf("\n After concatenation, the string will look like- \n %s", first\_string);

return 0;

}

**OUTPUT-**



**EXPERIMENT-22**

**OBJECTIVE-** Program to string comparison

**INTRODUCTION-** We compare two strings character by character and if we find a mis-match we exit the loop and print they aren’t the same.

**PROGRAM CODE-**

#include <stdio.h>

int compare(char[],char[]);

int main()

{

char str1[20]; // declaration of char array

char str2[20]; // declaration of char array

printf("Enter the first string : ");

gets(str1);

printf("Enter the second string : ");

gets(str2);

int c= compare(str1,str2); // calling compare() function

if(c==0)

printf("strings are same");

else

printf("strings are not same");

return 0;

}

// Comparing both the strings.

int compare(char a[],char b[])

{

int count=0,i=0; // integer variables declaration

while(a[i]!='\0' &&b[i]!='\0') // while loop

{

if(a[i]!=b[i])

{

count=1;

break;

}

i++;

}

if(count==0)

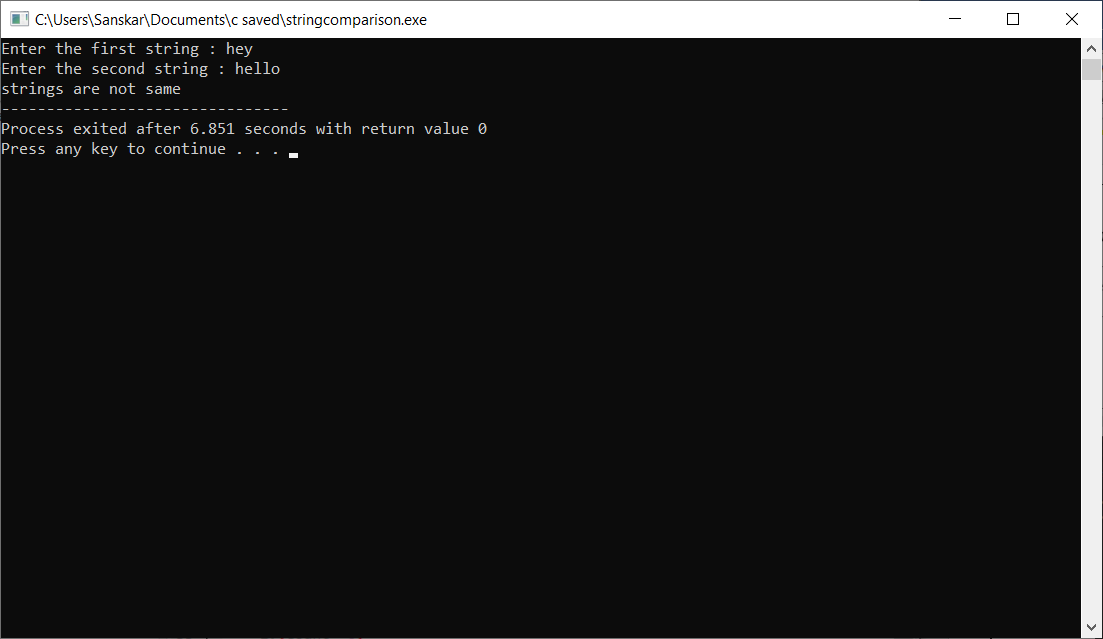
return 0;

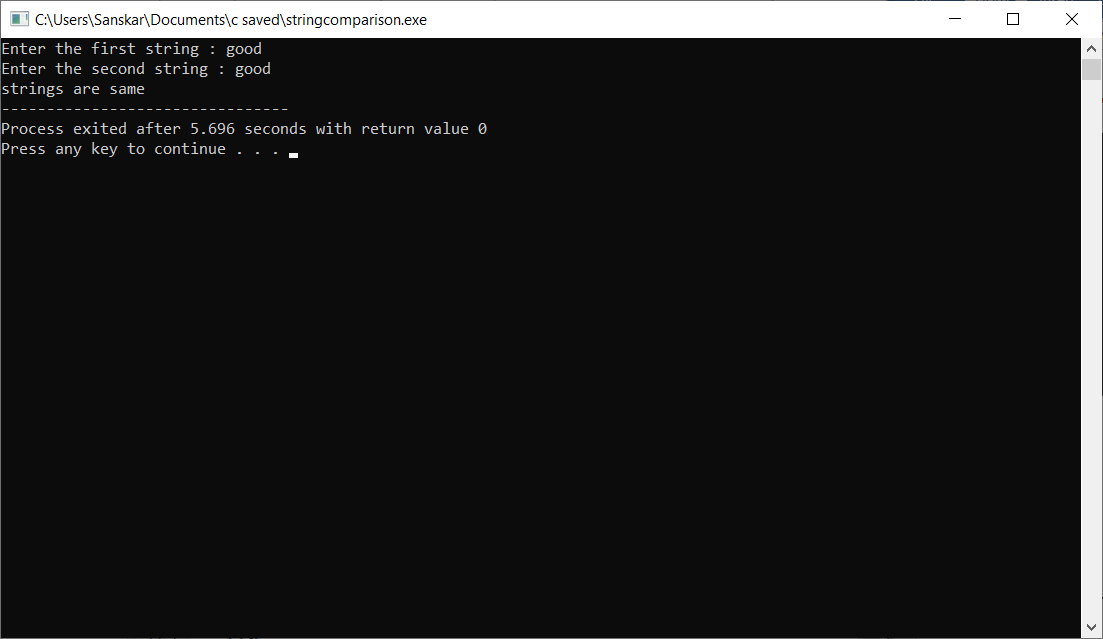
else

return 1;

}

**OUTPUT-**





**EXPERIMENT-23**

**OBJECTIVE-** Program to reverse a string

**INTRODUCTION-** In this program we reverse the input string. For this we use 2 arrays one

for the input string and one for the reverse.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

char s[1000], r[1000];

int begin, end, count = 0;

printf("Input a string\n"); // taking input string

gets(s);

// Calculating string length

while (s[count] != '\0')

count++;

end = count - 1;

for (begin = 0; begin < count; begin++) {

r[begin] = s[end];

end--;

}

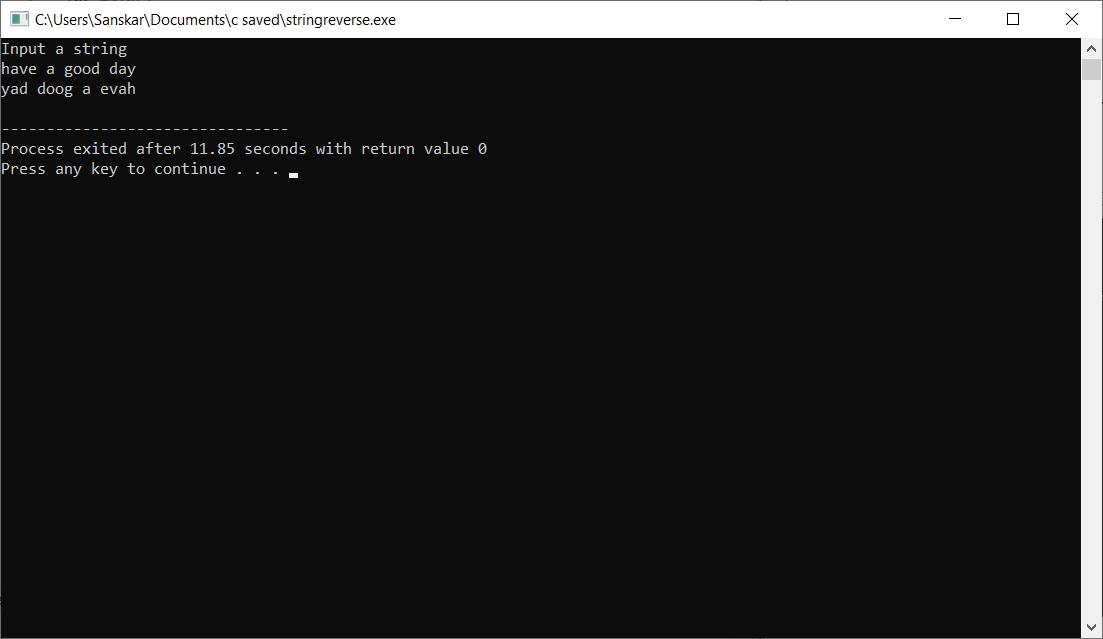
r[begin] = '\0';

printf("%s\n", r);

return 0;

}

**OUTPUT-**



**EXPERIMENT-24**

**OBJECTIVE-** Program to convert a string from lower case to upper case and vice versa

**INTRODUCTION –** The ASCII value of lower-case letters starts from a = 97 and so on. The ASCII value of upper-case letters starts from A= 65.

**PROGRAM CODE –**

**Upper case to lower case-**

#include <stdio.h>

#include <string.h>

int main() {

char s[100];

int i;

printf("\nEnter a string : "); // input string

gets(s);

for (i = 0; s[i]!='\0'; i++) {

if(s[i] >= 'A' && s[i] <= 'Z') {

s[i] = s[i] + 32; // adding 32 to ASCII values

}

}

printf("\nString in Lower Case = %s", s);

return 0;

}

**Lower case to upper case-**

#include <stdio.h>

#include <string.h>

int main() {

char s[100];

int i;

printf("\nEnter a string : "); // input string

gets(s);

for (i = 0; s[i]!='\0'; i++) {

if(s[i] >= 'A' && s[i] <= 'Z') {

s[i] = s[i] + 32; // adding 32 to ASCII values

}

}

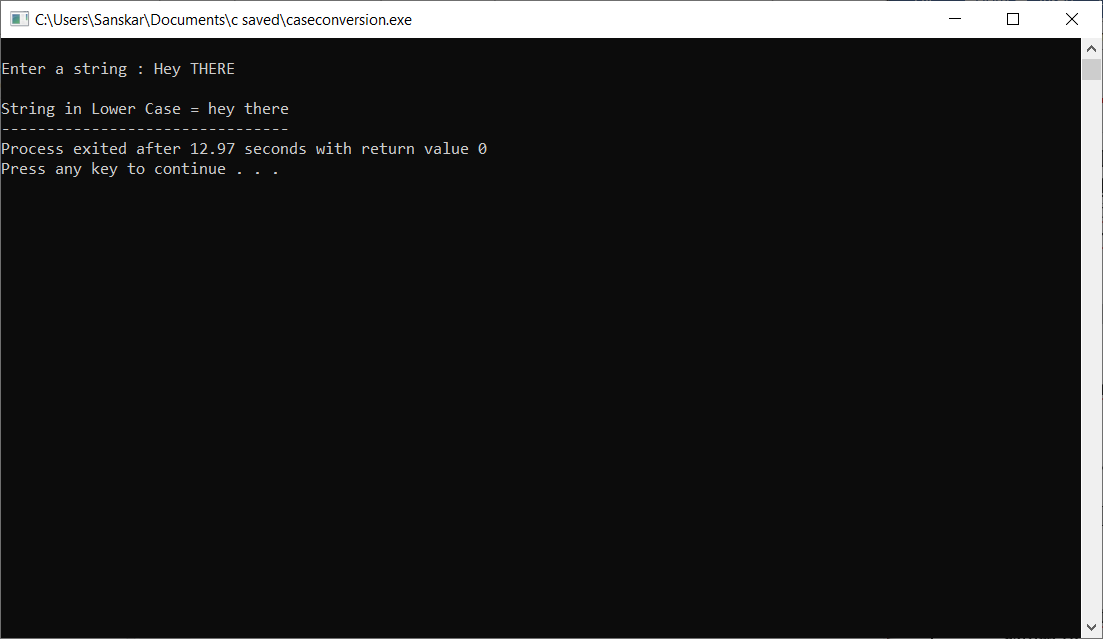
printf("\nString in Lower Case = %s", s);

return 0;

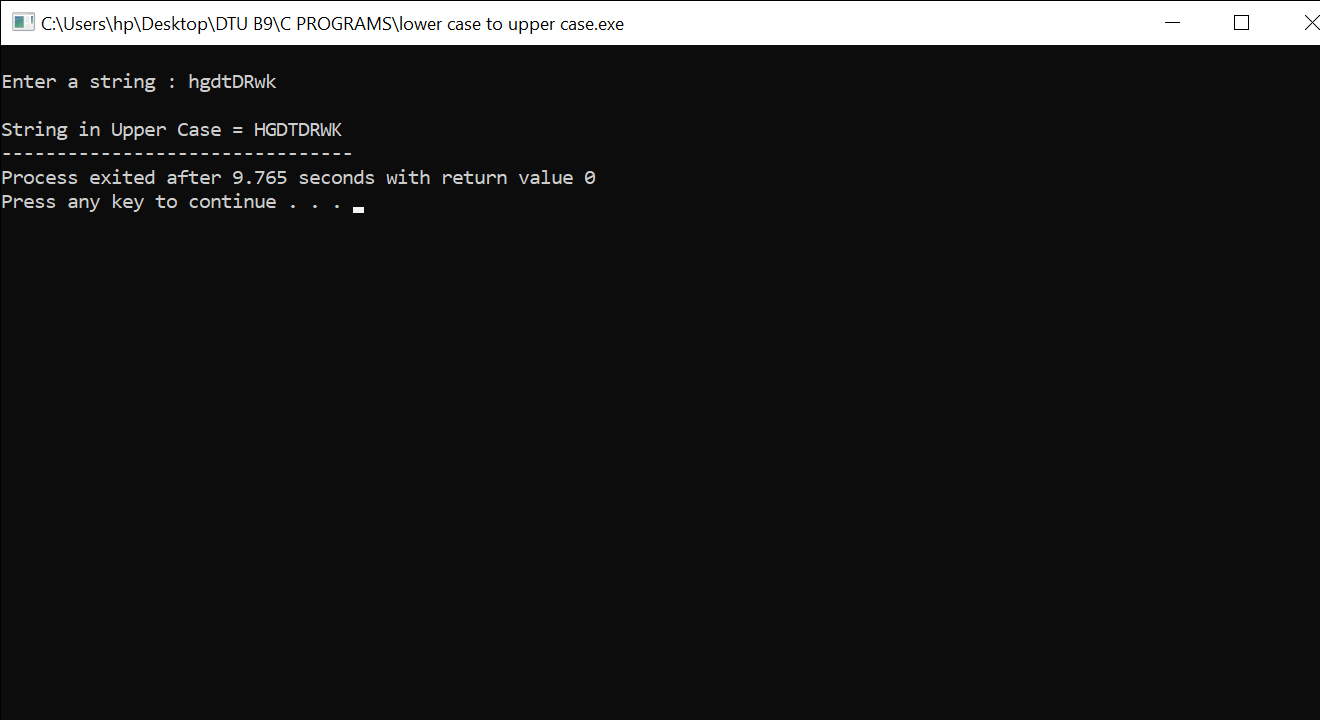
}

**OUTPUT-**

**Upper case to lower case**



**Lower-case to upper case-**

****

**EXPERIMENT-25**

**OBJECTIVE-** Program for addition of two 3x3 matrices

**INTRODUCTION-** To add 2 matrices we declare and initialize 3 matrices. Two of them for input from the user and the third one to store the sum.

**PROGRAM CODE-**

#include <stdio.h>

int main() {

int a[3][3], b[3][3], sum[3][3], i, j; // declaring two 3x3 matrices

printf("\nEnter elements of 1st matrix:\n");

for (i = 0; i < 3; ++i) // enter elements of 1st matrix

for (j = 0; j < 3; ++j) {

printf("Enter element a%d%d: ", i + 1, j + 1);

scanf("%d", &a[i][j]);

}

printf("Enter elements of 2nd matrix:\n");

for (i = 0; i < 3; ++i) // enter elements of 2nd matrix

for (j = 0; j < 3; ++j) {

printf("Enter element a%d%d: ", i + 1, j + 1);

scanf("%d", &b[i][j]);

}

// adding two matrices

for (i = 0; i < 3; ++i)

for (j = 0; j < 3; ++j) {

sum[i][j] = a[i][j] + b[i][j];

}

// printing the result

printf("\nSum of two matrices: \n");

for (i = 0; i < 3; ++i)

for (j = 0; j < 3; ++j)

{

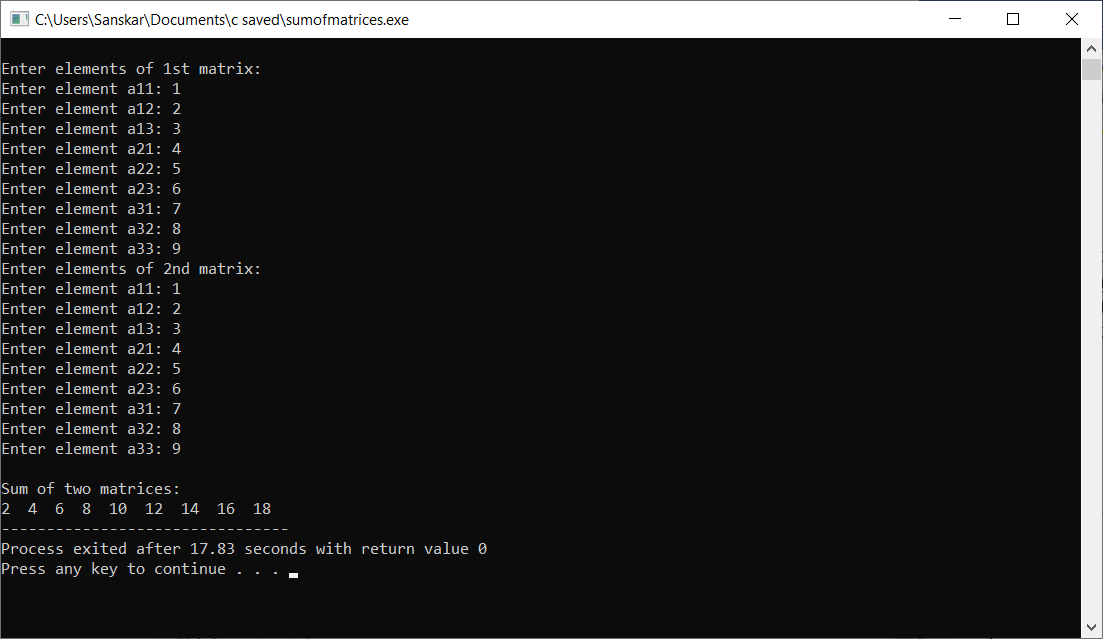
printf("%d ", sum[i][j]);

}

return 0;

}

**OUTPUT-**



**EXPERIMENT-26**

**OBJECTIVE-** Program to multiply two 3x3 matrices

**INTRODUCTION-** We are using 2-D arrays to multiply 2 arrays**.**

**PROGRAM CODE-**

#include<stdio.h>

int main()

{

//MATRIX 1 INPUT

int r1,c1;

printf("\nMatrix 1:\nEnter number of rows and column:\t");

scanf("%d%d",&r1,&c1);

int a[r1][c1];

printf("\nEnter the elements of matrix 1 rowwise :\t");

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

scanf("%d",&a[i][j]);

}

printf("MATRIX 1:\n");

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

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

printf("\n");

}

//MATRIX 2 INPUT

int r2,c2;

printf("\nMatrix 2:\nEnter number of rows and column:\t");

scanf("%d%d",&r2,&c2);

int b[r2][c2];

printf("\nEnter the elements of matrix 2 rowwise :\t");

for(int i=0;i<r2;i++)

{

for(int j=0;j<c2;j++)

scanf("%d",&b[i][j]);

}

printf("MATRIX 2:\n");

for(int i=0;i<r2;i++)

{

for(int j=0;j<c2;j++)

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

printf("\n");

}

if (r2!=c1)

{

printf("\nMULTIPLICATION NOT POSSIBLE");

}

else

{

//multiplying 2 matrices

int mult[r1][c2];

for(int i=0;i<r1;i++)

{

for(int j=0;j<c2;j++)

mult[i][j]=0;

}

for(int i=0;i<r1;i++)

{

for(int j=0;j<c2;j++)

{

for(int k=0;k<c1;k++)

mult[i][j]+=a[i][k]\*b[k][j];

}

}

//OUTPUT

printf("PRODUCT:\n");

for(int i=0;i<r1;i++)

{

for(int j=0;j<c2;j++)

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

printf("\n");

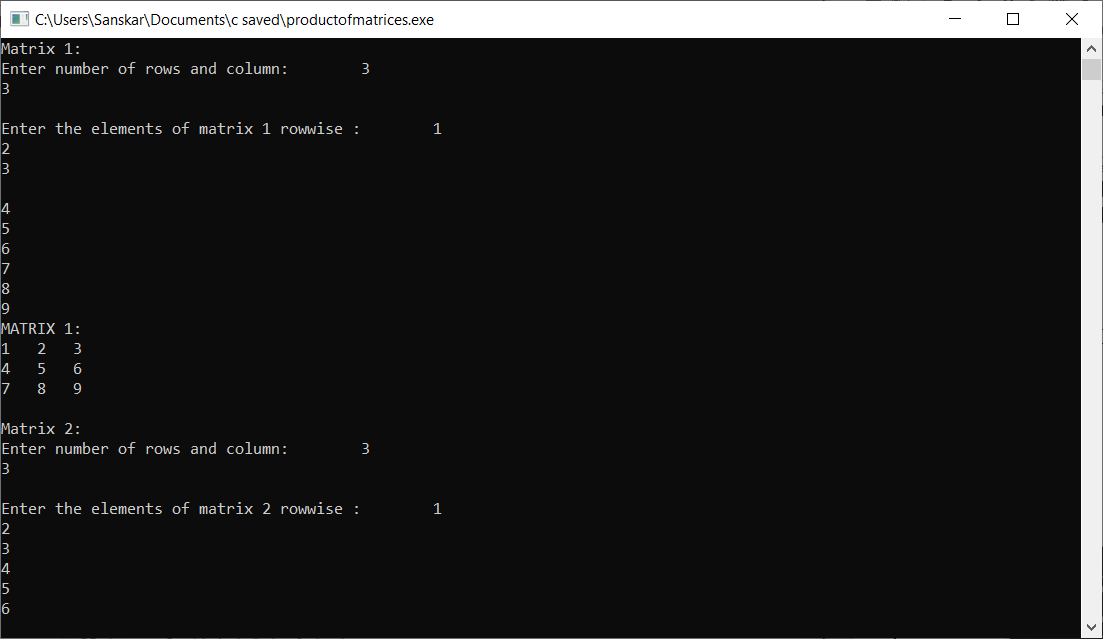
}

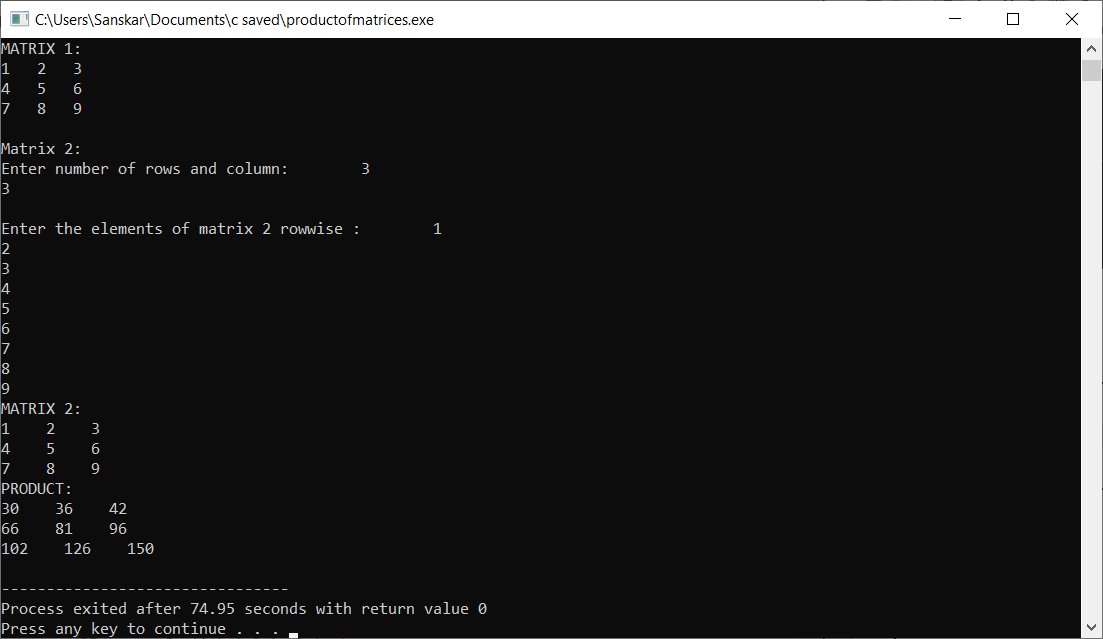
}

return 0;

}

**OUTPUT-**





**EXPERIMENT-27**

**OBJECTIVE-** Program to swap two numbers using pointers

**INTRODUCTION-** Two swap two numbers we have to swap the values at the address for that we use pointers.

**PROGRAM CODE-**

#include <stdio.h>

// function to swap the two numbers

void swap(int \*x,int \*y)

{

int t;

t = \*x;

\*x = \*y;

\*y = t;

}

int main()

{

int num1,num2;

printf("Enter value of num1: ");

scanf("%d",&num1);

printf("Enter value of num2: ");

scanf("%d",&num2);

//displaying numbers before swapping

printf("Before Swapping: num1 is: %d, num2 is: %d\n",num1,num2);

//calling by referance the user defined function swap()

swap(&num1,&num2);

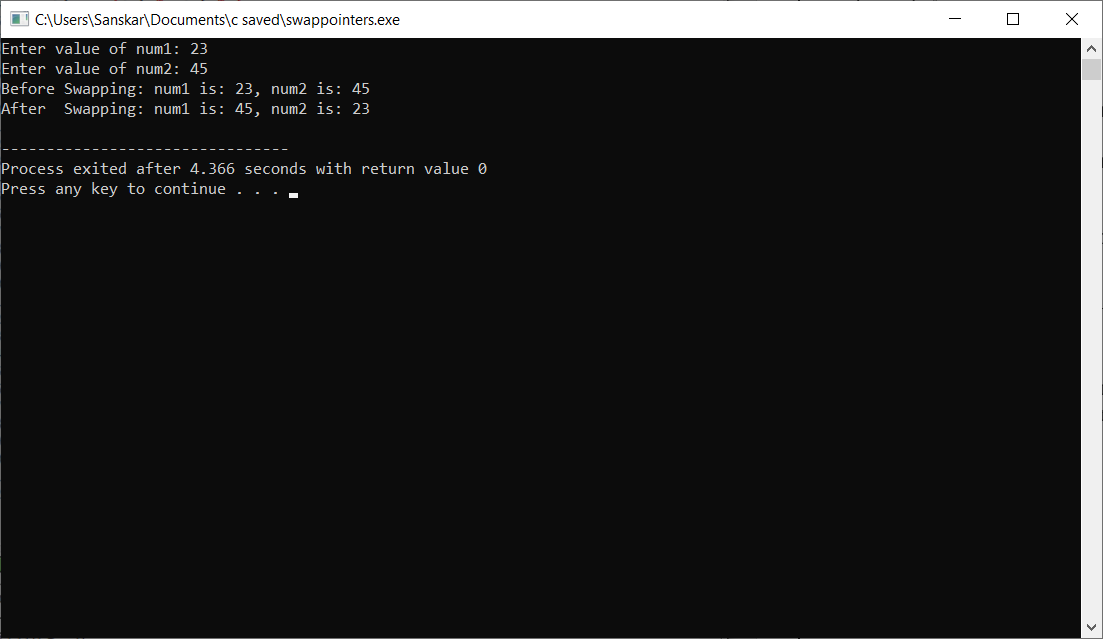
//displaying numbers after swapping

printf("After Swapping: num1 is: %d, num2 is: %d\n",num1,num2);

return 0;

}

**OUTPUT-**



**EXPERIMENT-28**

**OBJECTIVE-** Program to generate employee details using structures

**INTRODUCTION-** We are using the structure function to generate employee details.

**PROGRAM CODE-**

#include<stdio.h>

struct employee

{

char name[32];

int employee\_ID;

int salary;

}E;

int main()

{

//Enter your name

printf("\n Enter name- ");

gets(E.name);

//Enter your employee ID

printf("\n Enter employee ID- ");

scanf("%d",&E.employee\_ID);

getchar();

printf("\n Enter salary- ");

scanf("%d",&E.salary);

printf("\n Entered details are- ");

printf("\n Name- ");

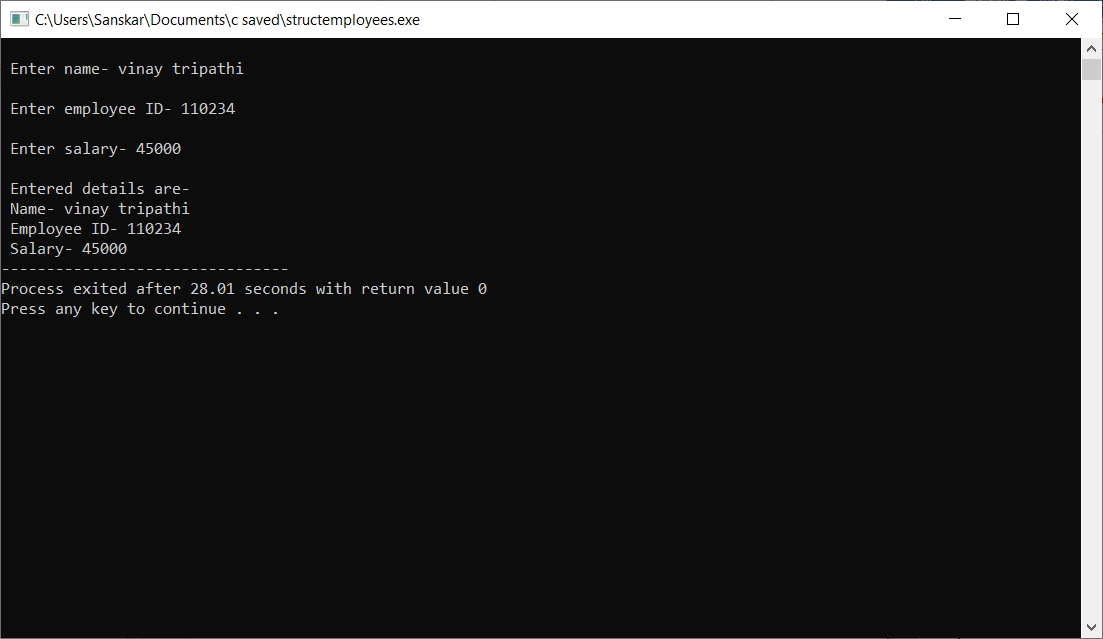
puts(E.name);

printf(" Employee ID- %d",E.employee\_ID);

printf("\n Salary- %d",E.salary);

}

**OUTPUT-**



**EXPERIMENT-29**

**OBJECTIVE-**Program to find the area and perimeter of a circle, square, rectangle and triangle using functions

**INTRODUCTION-** We will find the area and perimeter of circle, square, rectangle and triangle.

**PROGRAM CODE-**

**Area and perimeter of circle-**

#include<stdio.h>

#define PI 3.14

float circle\_area(int r)

{

float area;

area= PI\*r\*r;

return area;

}

float circle\_perimeter(int r)

{

float p;

p=2\*PI\*r;

return p;

}

int main()

{

float a,x,y;

printf("Enter the radius of the Circle : ");

scanf("%f",&a);

x=circle\_area(a); // calling function circle\_area

y=circle\_perimeter(a); // calling circle\_perimeter

printf("\nArea of circle = %f\n\nPerimeter of circle = %f",x,y);

return 0;

}

**Area and perimeter of Square-**

#include<stdio.h>

int sqr\_area(int l)

{

int area;

area=l\*l;

return area;

}

int sqr\_perimeter(int l)

{

int p;

p=4\*l;

return p;

}

int main()

{

int a,x,y;

printf("Enter Length of side of Square : ");

scanf("%d",&a);

x=sqr\_area(a); // calling function sqr\_area

y=sqr\_perimeter(a); // calling sqr\_perimeter

printf("\nArea of Square = %d\n\nPerimeter of Square = %d",x,y);

return 0;

}

**Area and perimeter of Rectangle-**

#include<stdio.h>

int rect\_area(int l,int w)

{

int area;

area=l\*w;

return area;

}

int rect\_perimeter(int l,int w)

{

int p;

p=2\*(l+w);

return p;

}

int main()

{

int a,b,x,y;

printf("Enter Length of Rectangle : ");

scanf("%d",&a);

printf("\nEnter Width of Rectangle : ");

scanf("%d",&b);

x=rect\_area(a,b); // calling function rect\_area

y=rect\_perimeter(a,b); // calling function perimeter

printf("\nArea of Rectangle = %d\n\nPerimeter of Rectangle = %d",x,y);

return 0;

}

**Area and perimeter if Triangle-**

#include <stdio.h>

#include <math.h>

double area\_of\_triangle(double, double, double);

int main()

{

double a, b, c, area, perimeter\_of\_triangle;

printf("Enter the lengths of sides of a triangle\n");

scanf("%lf%lf%lf", &a, &b, &c);

perimeter\_of\_triangle = a+b+c ;

printf("Perimeter of the triangle is- %lf \n ", perimeter\_of\_triangle);

area = area\_of\_triangle(a, b, c);

printf("Area of the triangle = %.2lf\n", area);

return 0;

}

double area\_of\_triangle(double a, double b, double c)

{

double s, area;

s = (a+b+c)/2;

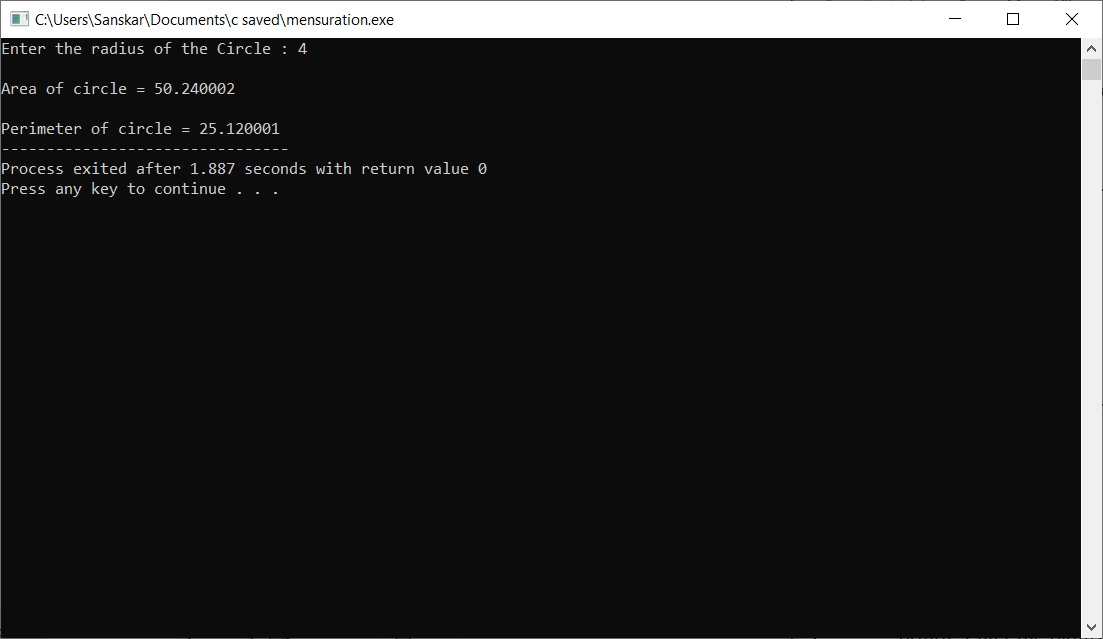
area = sqrt(s\*(s-a)\*(s-b)\*(s-c));

return area;

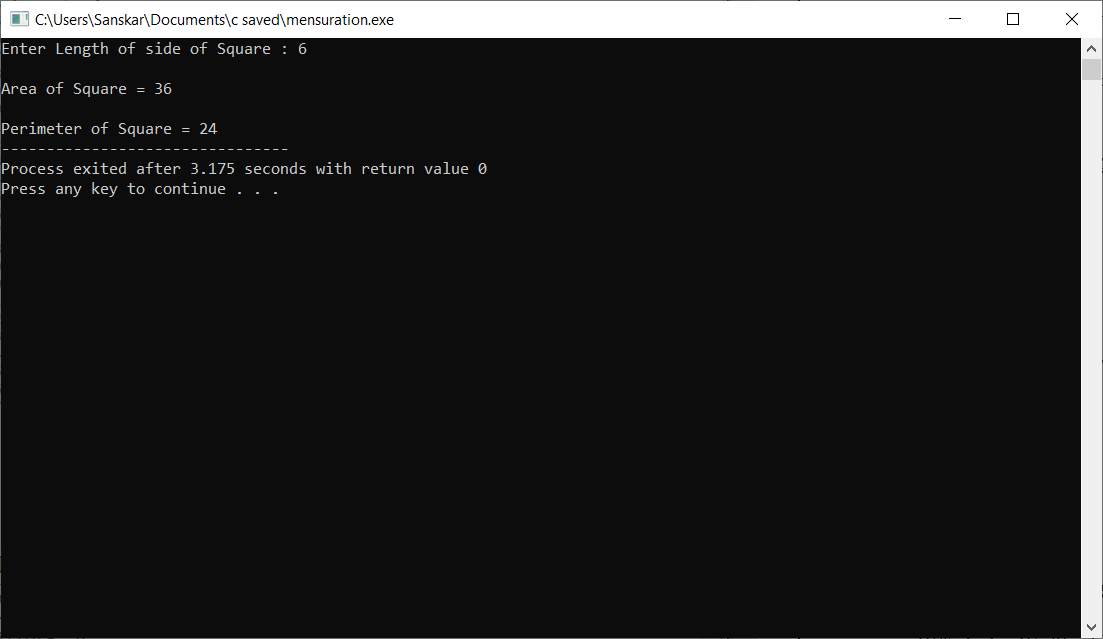
}

**OUTPUT-**

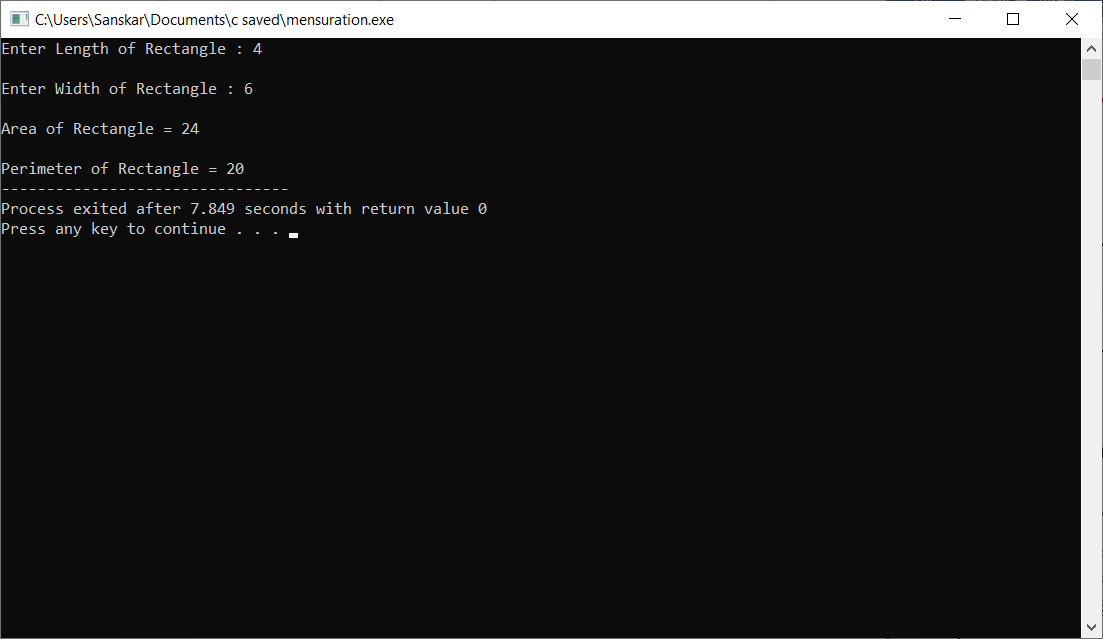
**Area and perimeter of circle-**



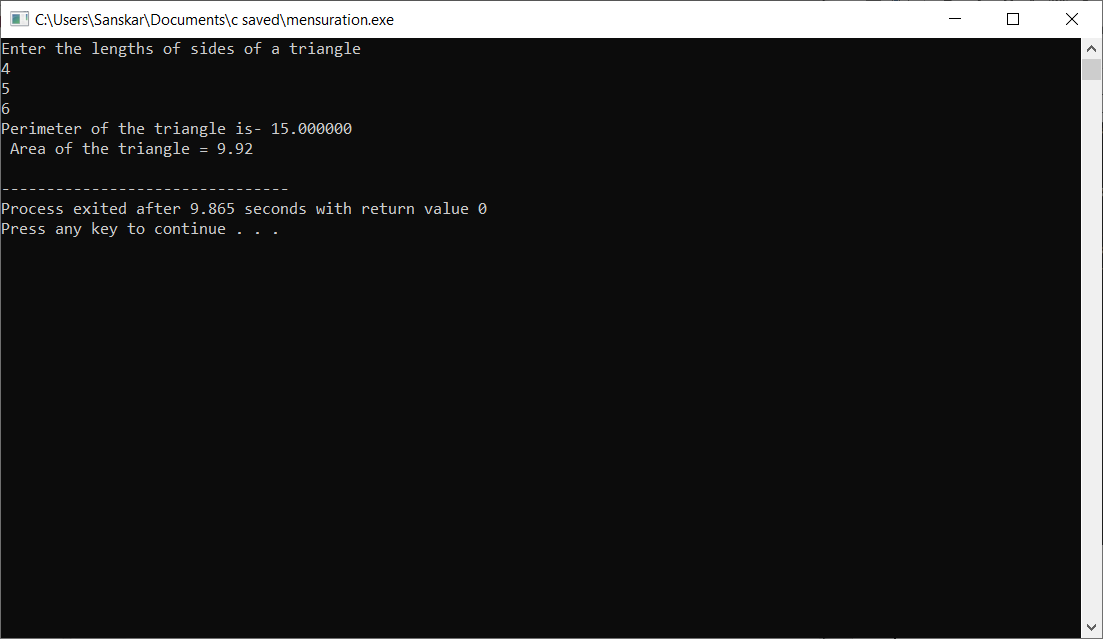
**Area and perimeter of square-**



**Area and perimeter of rectangle-**



**Area and Perimeter of triangle-**



**EXPERIMENT-30**

**OBJECTIVE-** Program to pass and return pointer to a function hence calculate the average of an array.

**INTRODUCTION-** We are using the concepts of arrays and pointers so as to find the average of the elements of the array.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

float a[50], sum = 0.0, avg;

int i,num ;

float \*pointera, \*pointersum, \*pointeravg;

pointera = &a[0]; // Or, px = &x;

pointersum = &sum, pointeravg = &avg;

printf("Enter the number of elements in the array-\n ");

scanf("%d", &num);

printf("Enter %d Elements- \n " , num );

for (i = 0; i < num; i++)

{

scanf("%f", (pointera + i));

\*pointersum += \*(pointera + i);

}

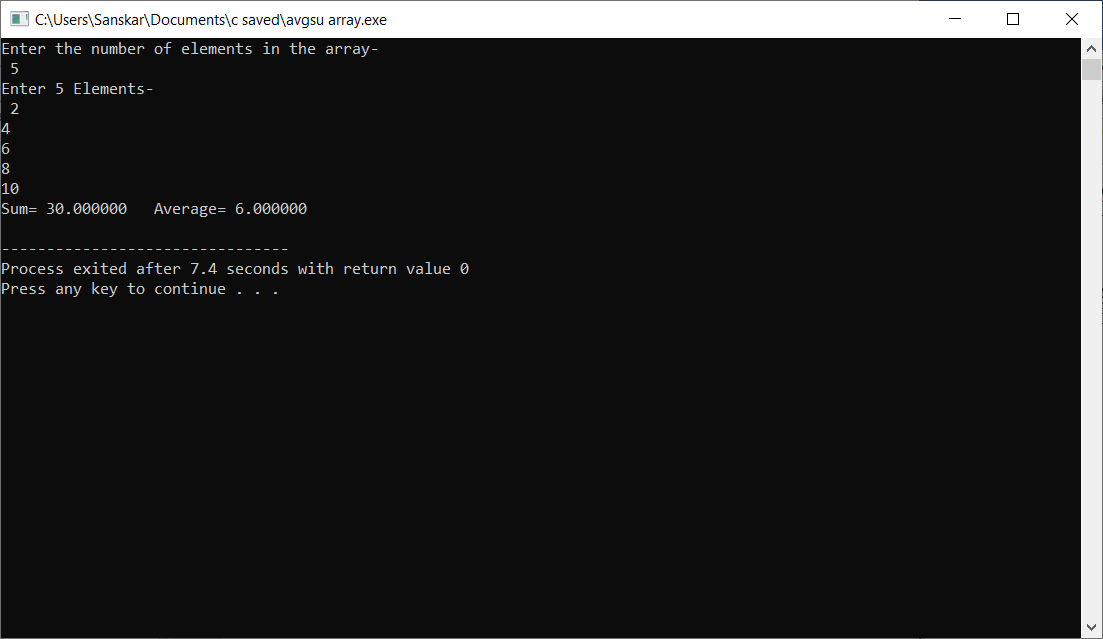
\*pointeravg = \*pointersum / 5;

printf("Sum= %f \t Average= %f\n", \*pointersum, \*pointeravg);

return 0;

}

**OUTPUT-**



**EXPERIMENT-31**

**OBJECTIVE-** Program to pass an array as pointer to a function that calculates the sum of all elements.

**INTRODUCTION-**Just like variables, array can also be passed to a function as an argument using pointers. The program calls a function to add all the element of an array and passes the array argument as a pointer. The program should dynamically allocate a piece of memory for that array and use a pointer to point to that array memory as well as traverse that array using a pointer.

**PROGRAM CODE-**

#include <stdio.h>

int sum\_of\_elements(int \*arr , int n) // sum function

{

int i=0,sum=0 ;

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

{

sum = sum + arr[i];

}

return sum;

}

int main()

{

int total = 0, num, j;

int array[50];

printf("Enter the number of elements in the array-\n ");

scanf("%d",&num);

printf("Enter the %d elements -\n " , num );

for (j = 0 ; j < num ; j++) // taking array elements input

{

scanf("%d", &array[j]);

}

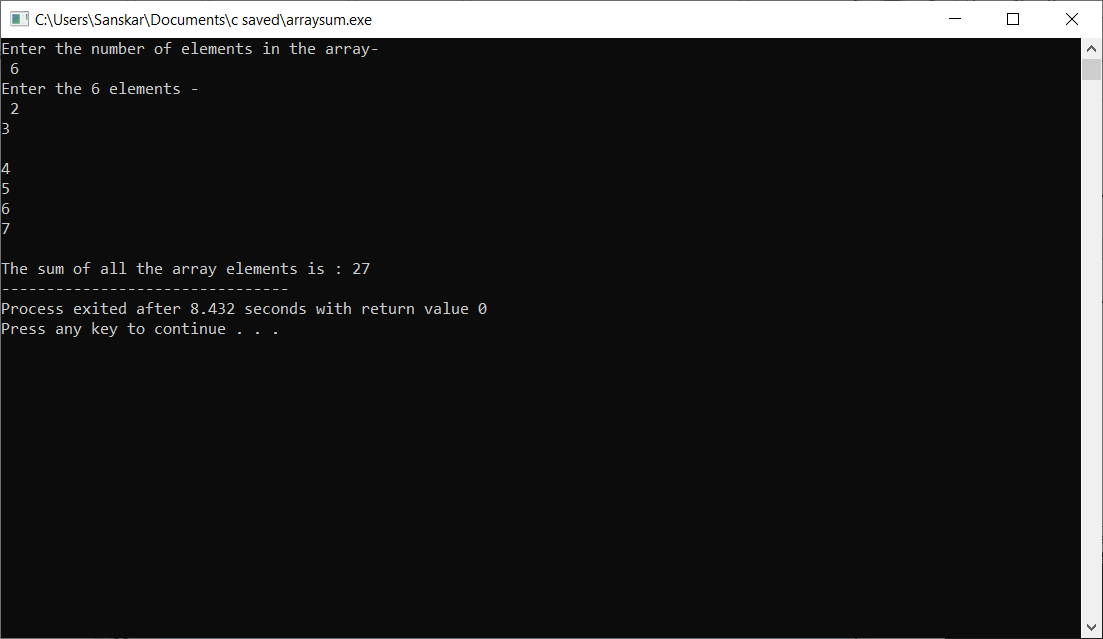
total = sum\_of\_elements(array,num);

printf("\nThe sum of all the array elements is : %d",total); // print sum as output

return 0;

}

**OUTPUT-**



**EXPERIMENT-32**

**OBJECTIVE-** Program to demonstrate the example of array of pointers.

**INTRODUCTION-** We can declare a pointer that can point to a whole array instead of only one element of the array. This pointer is useful when talking about multidimensional arrays.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

/\*declare same type of variables\*/

int a,b,c;

/\*we can create an integer pointer array to

store the address of these integer variables\*/

int \*ptr[3];

/\*assign the address of all integer variables to ptr\*/

ptr[0]= &a;

ptr[1]= &b;

ptr[2]= &c;

/\*assign the values to a,b,c\*/

a=10;

b=20;

c=30;

/\*print values using pointer variable\*/

printf("value of a: %d, b: %d, c: %d\n",\*ptr[0],\*ptr[1],\*ptr[2]);

/\*add 10 to all values using pointer\*/

\*ptr[0] +=5;

\*ptr[1] +=5;

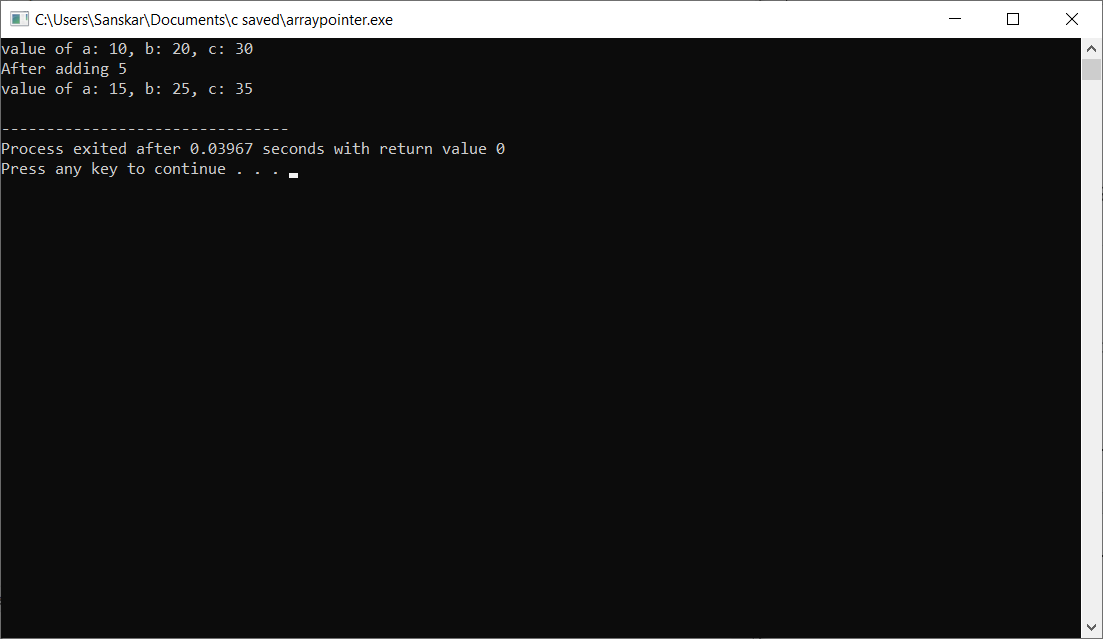
\*ptr[2] +=5;

printf("After adding 5\nvalue of a: %d, b: %d, c: %d\n",\*ptr[0],\*ptr[1],\*ptr[2]);

return 0;

}

**OUTPUT-**



**EXPERIMENT-33**

**OBJECTIVE-** Program which copies one file contents to other.

**INTRODUCTION-** We are doing this program with the help of fseek(), fopen(), fclose(), fputc() functions.

**PROGRAM CODE-**

#include <stdio.h>

#include <stdlib.h> // For exit()

int main()

{

FILE \*fptr1, \*fptr2;

char filename[100], c;

printf("Enter the filename to open for reading \n");

scanf("%s", filename);

// Open one file for reading

fptr1 = fopen(filename, "r");

if (fptr1 == NULL)

{

printf("Cannot open file %s \n", filename);

exit(0);

}

printf("Enter the filename to open for writing \n");

scanf("%s", filename);

// Open another file for writing

fptr2 = fopen(filename, "w");

if (fptr2 == NULL)

{

printf("Cannot open file %s \n", filename);

exit(0);

}

// Read contents from file

c = fgetc(fptr1);

while (c != EOF)

{

fputc(c, fptr2);

c = fgetc(fptr1);

}

printf("\nContents copied to %s", filename);

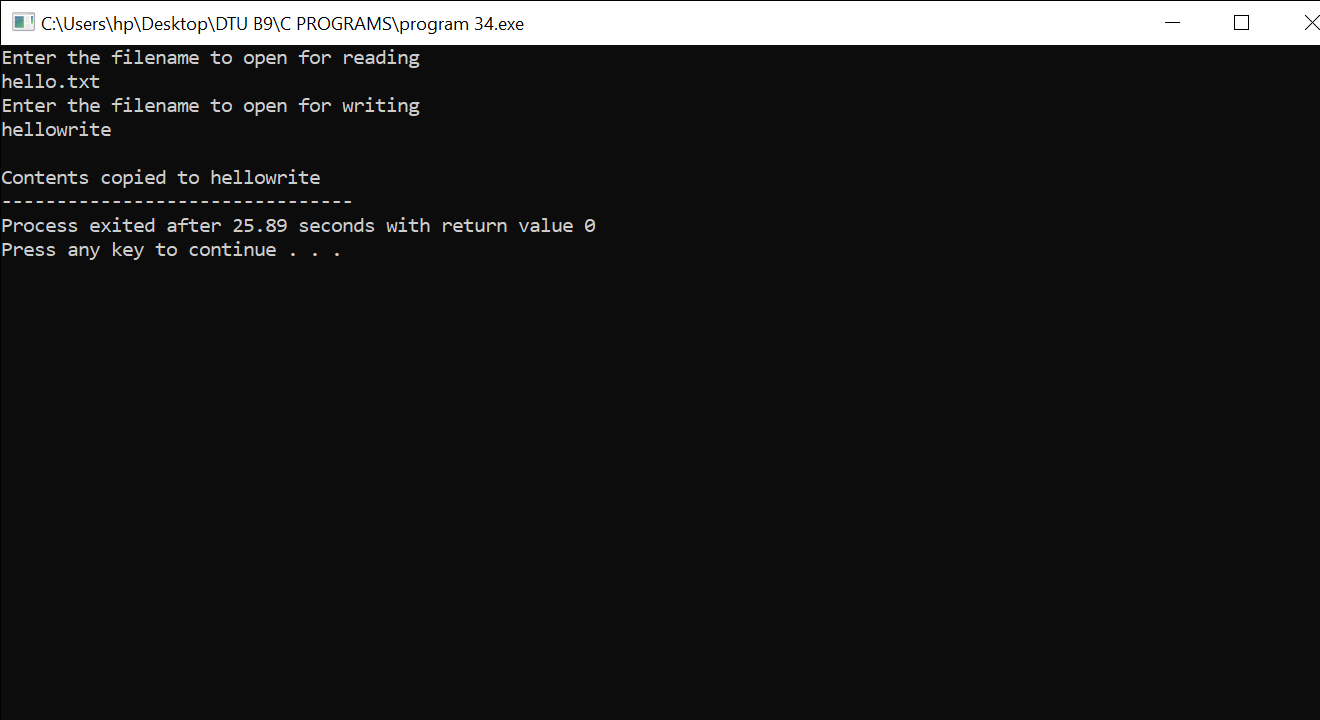
fclose(fptr1);

fclose(fptr2);

return 0;

}

**OUTPUT-**

****

**EXPERIMENT-34**

**OBJECTIVE-** Program to find size of a given file

**INTRODUCTION-** We are doing this program so as to find the size of a given file. We use fseek (), fopen (), fclose () functions.

**PROGRAM CODE-**

#include<stdio.h>

#include<process.h>

long file\_size(FILE \*fp)

{

long len;

//move file pointer to end of the file \*/

fseek(fp, 0L, 2);

//Obtain the current position of file pointer

len = ftell(fp);

return len;

}

void main(void)

{

FILE \*fp;

char filename[20];

printf("\nEnter the file name:\n");

scanf("%s",filename);

fp = fopen(filename, "r");

if(fp == NULL)

{

printf("Error: file not found!\n");

exit(0);

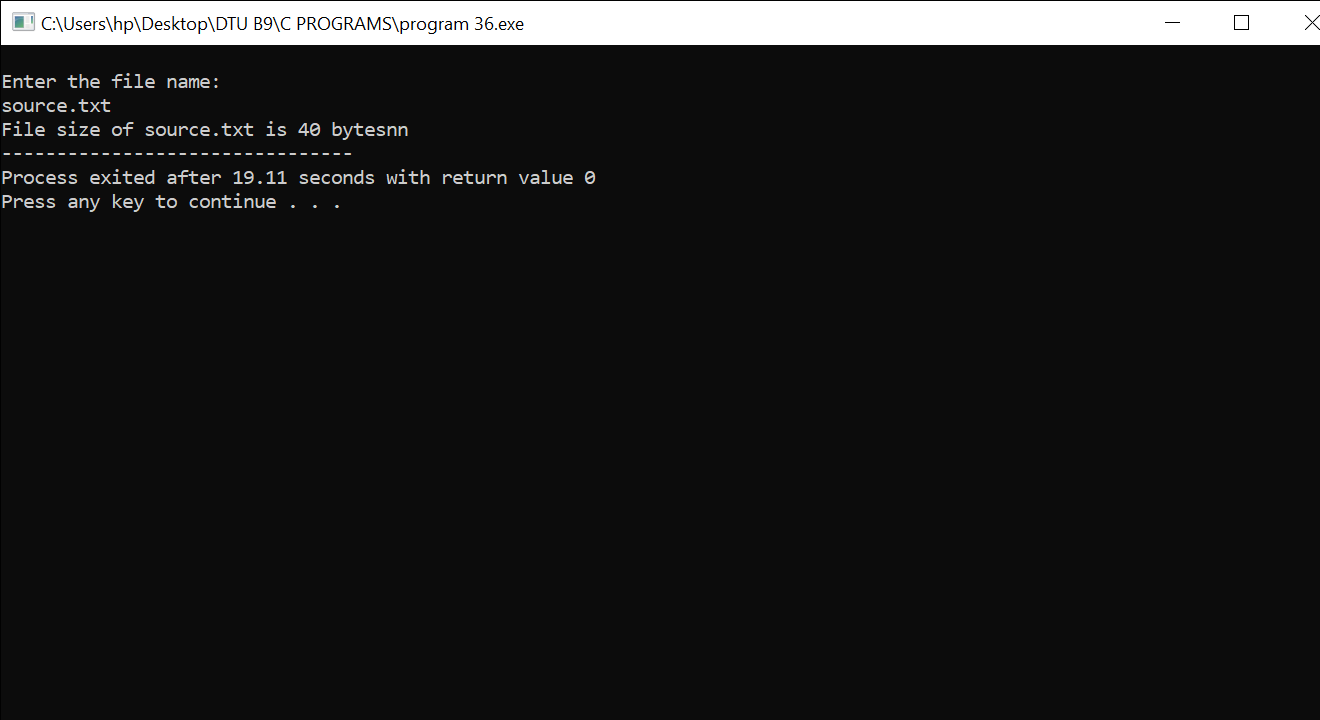
}

printf("File size of %s is %ld bytesnn",filename, file\_size(fp));

fclose(fp);

}

**OUTPUT-**

****

**EXPERIMENT-35**

**OBJECTIVE-** Program to read a file and after converting all lower case to upper case letters write it to another file.

**INTRODUCTION-** Program to read a file and after converting all lower case to upper case letters, write it to another file.

**PROGRAM CODE-**

#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

int main()

{

FILE \*fp1, \*fp2;

char ch;

fp1 = fopen("source.txt", "r");

if (fp1 == NULL)

{

puts("File does not exist..");

exit(1);

}

fp2 = fopen("target.txt", "w");

if (fp2 == NULL)

{

puts("File does not exist..");

fclose(fp1);

exit(1);

}

while((ch=fgetc(fp1))!=EOF)

{

ch = toupper(ch);

fputc(ch,fp2);

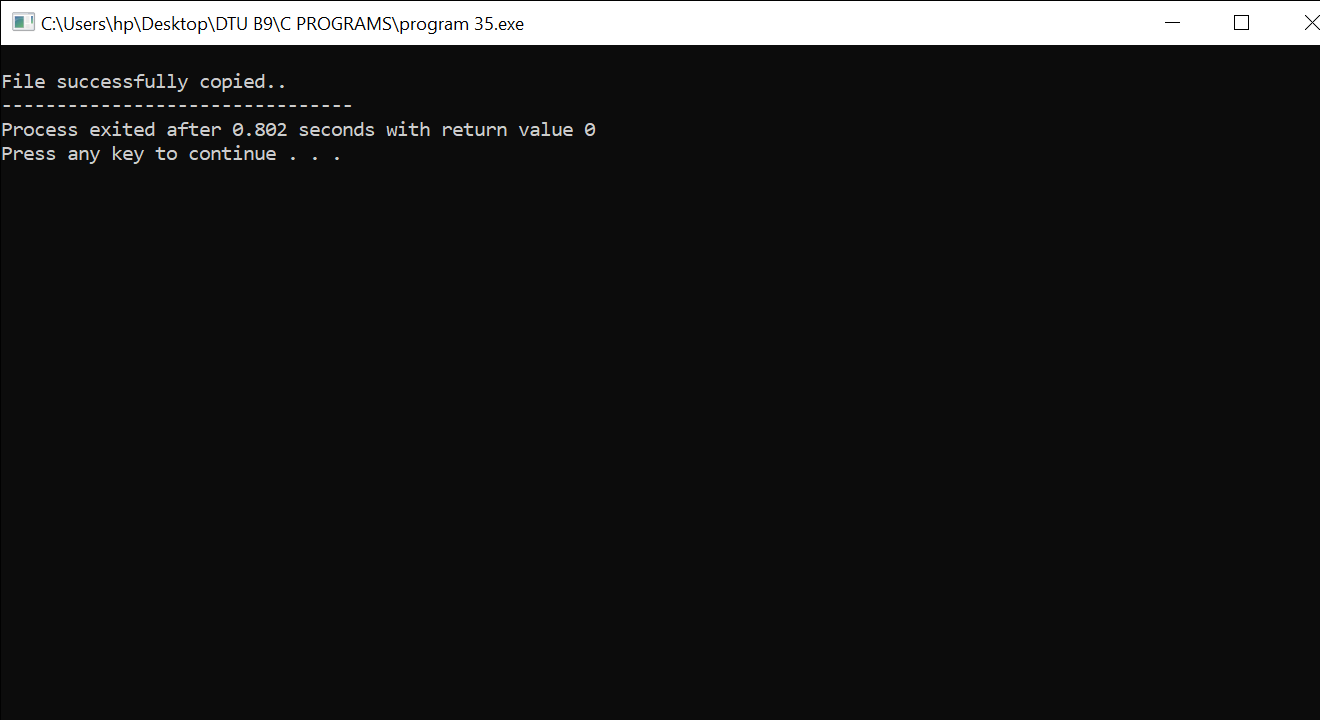
}

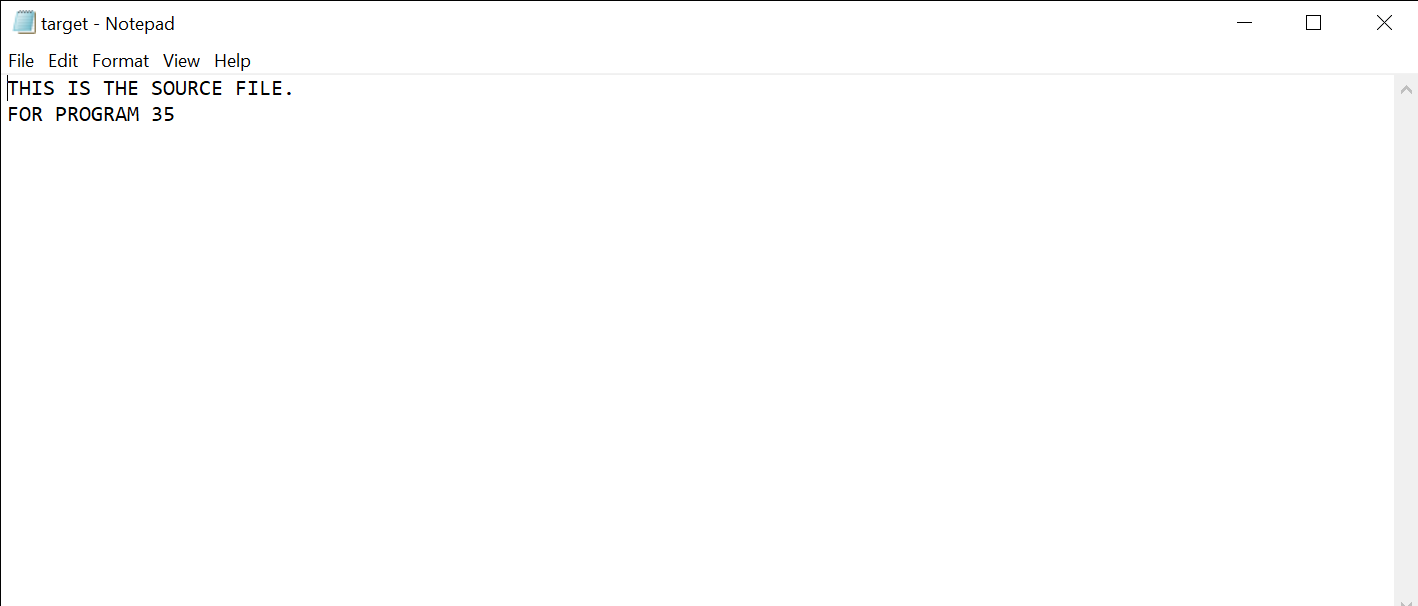
printf("\nFile successfully copied..");

return 0;

}

**OUTPUT-**

****

****

**EXPERIMENT-36**

**OBJECTIVE-** Program to find the size of a given file.

**INTRODUCTION-** Program to read a file and calculate its size.

**PROGRAM CODE-**

#include <stdio.h>

int main()

{

FILE \*fp;

char ch;

int size = 0;

char \* arrr = "C:\\Users\\acer\\Documents\\file4.txt";

fp = fopen(arrr, "r");

if (fp == NULL)

printf("\nFile unable to open ");

else

printf("\nFile opened ");

fseek(fp, 0, 2); /\* file pointer at the end of file \*/

size = ftell(fp); /\* take a position of file pointer un size variable \*/

printf("The size of given file is : %d\n", size);

fclose(fp);

return 0;

}

OUTPUT-

