**WEEK-7**

**1.write a C function for the following tasks**

**i)calculating factorial**

**pseudocode:**

begin

declare int x

take user input x

to find factorial call function fact(x)

control jumps to function defnition

declare int i,fact=1

run loop from i=1 to i=n

fact=fact\*i

print(fact)

end

**c program:**

#include<stdio.h>

void fact(int n);

main()

{

int x;

printf("enter x value:");

scanf("%d",&x);

fact(x);

}

void fact(int n)

{

int i,fact=1;

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

{

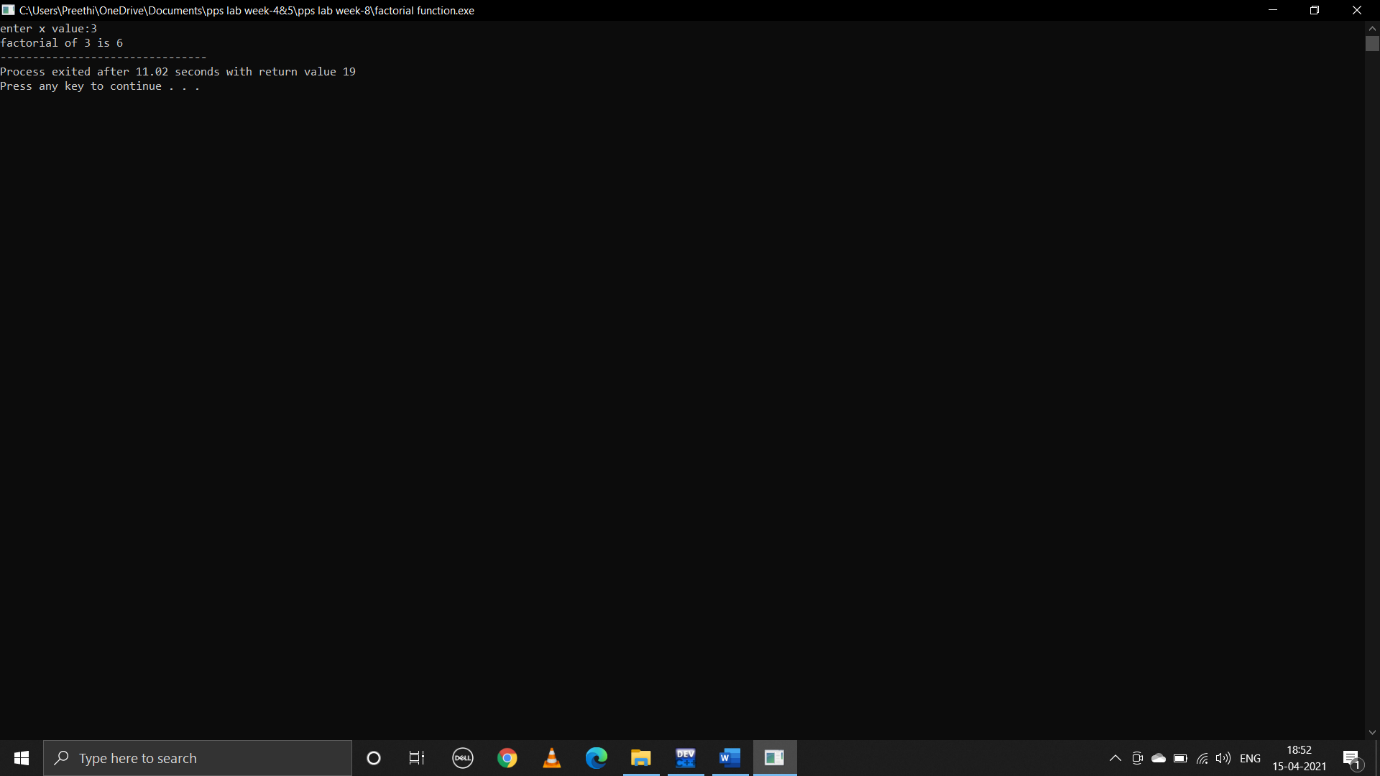
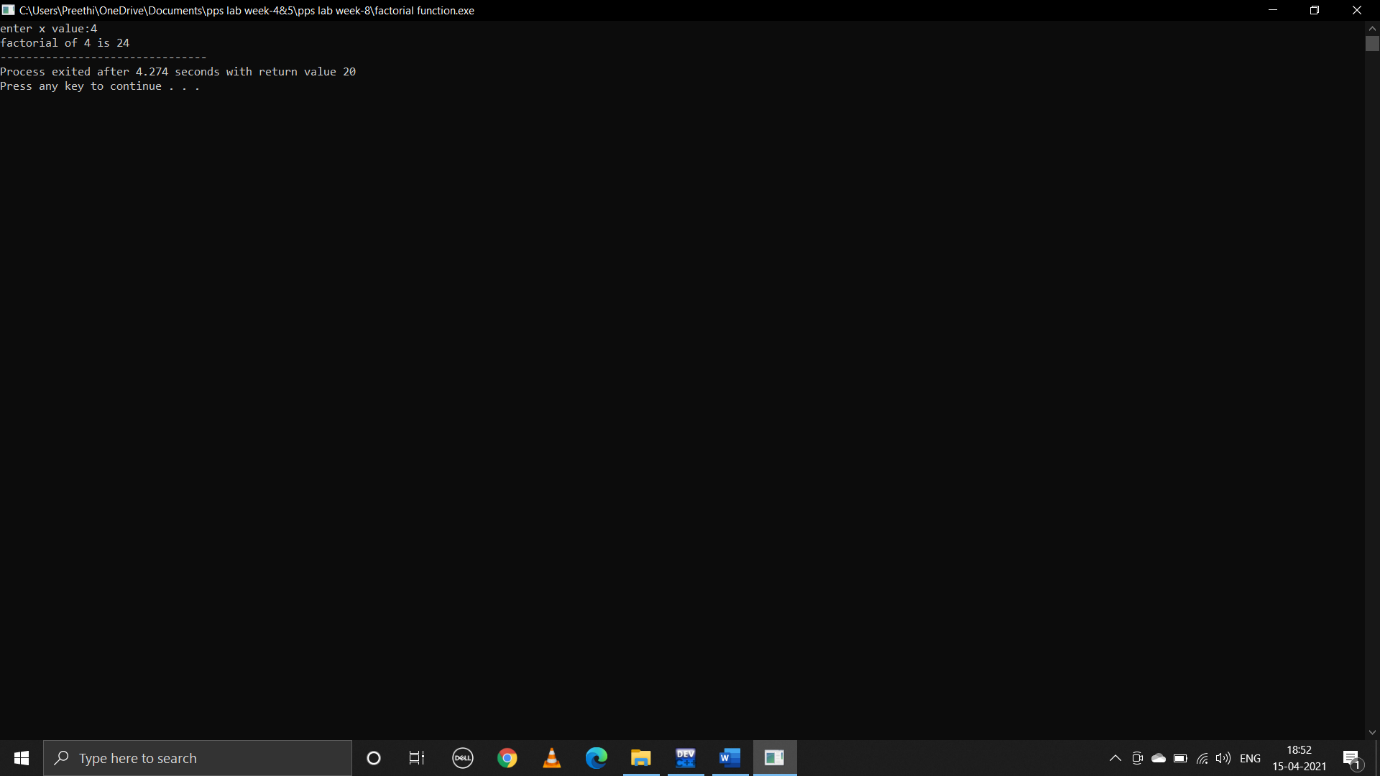
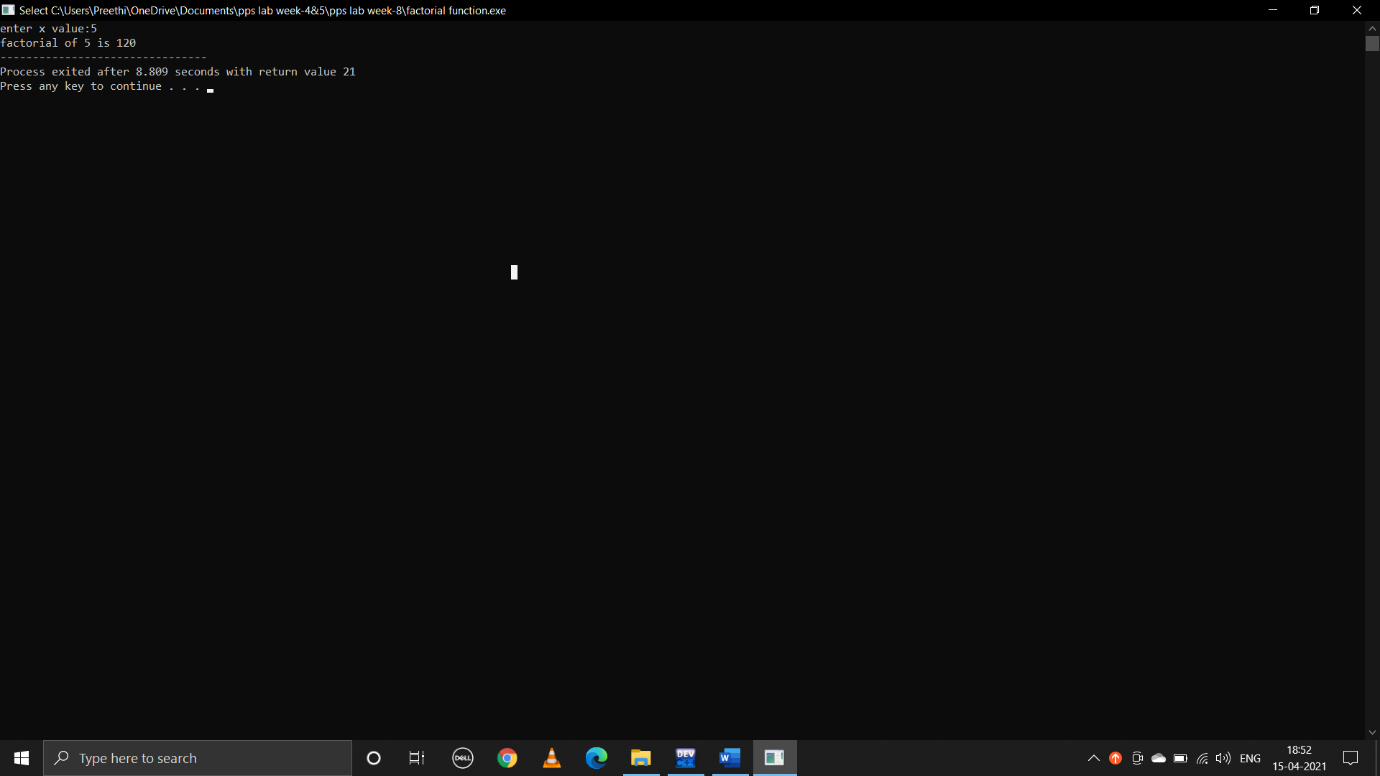
fact=fact\*i;

}

printf("factorial of %d is %d",n,fact);

}

**output:**

****

**ii).Find value of a given fibonacci term**

**pseudocode:**

begin

declare int x

take user input x to print number of terms

call the function fib(x) to print fibonacci series upto x

control jumps to function definition

declare int I,n1=0,n2=1,nth

print(n1 ,n2)

run loop from i=0 to i=n-2

nth=n1+n2

print(nth)

update n1 and n2

n1=n2

n2=nth

end

**c program:**

#include<stdio.h>

void main()

{

int x;

printf("enter x:");

scanf("%d",&x);

fib(x);

}

void fib(n)

{

int i,n1=0,n2=1,nth;

printf("%d\n%d\n",n1,n2);

for(i=0;i<n-2;i++)

{

nth=n1+n2;

printf("%d\n",nth);

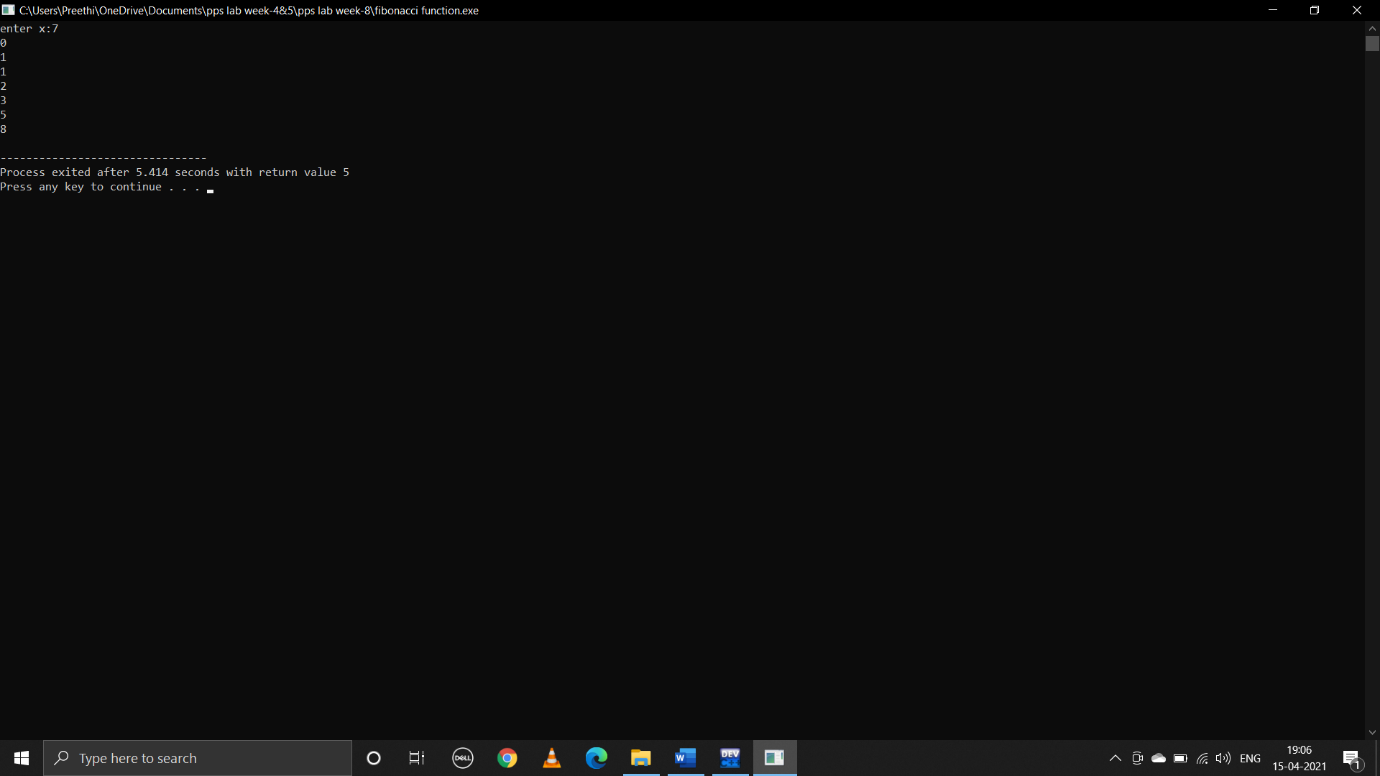
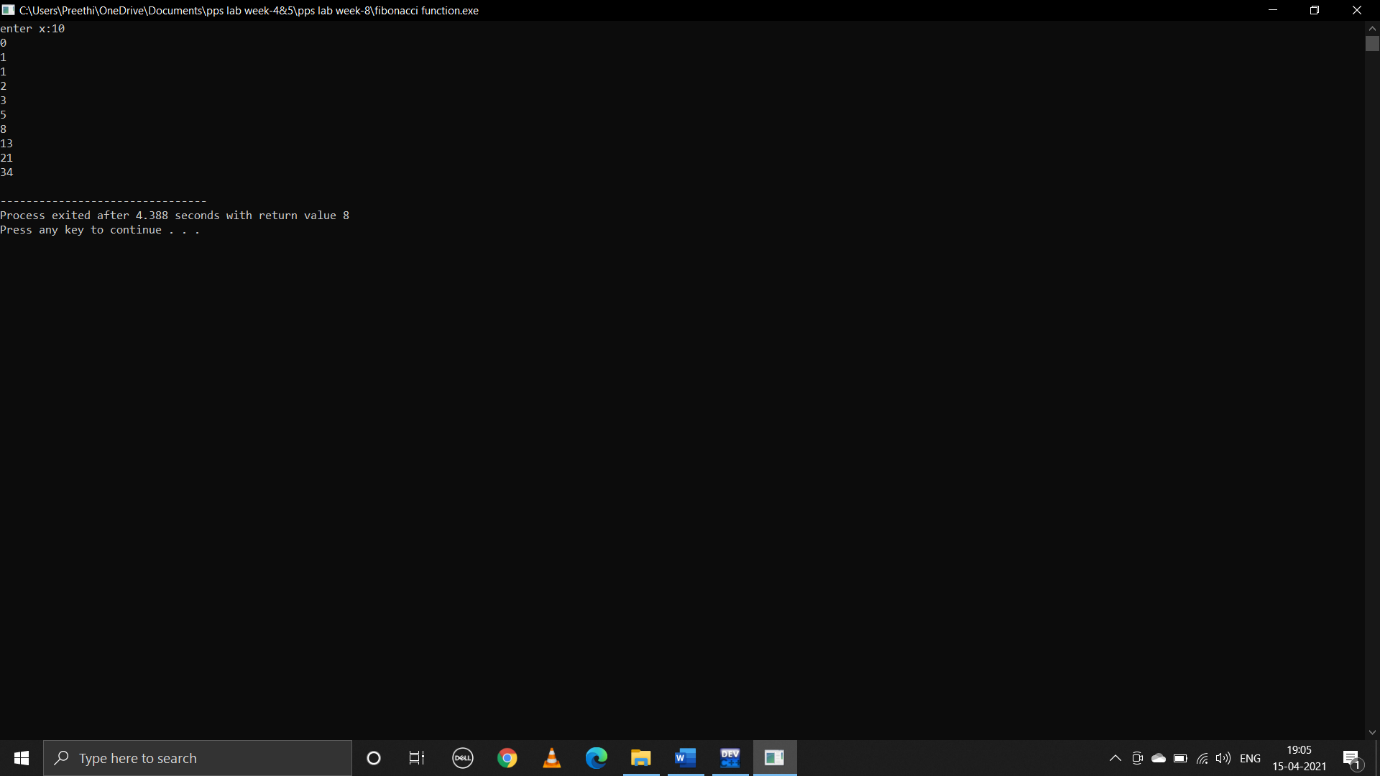
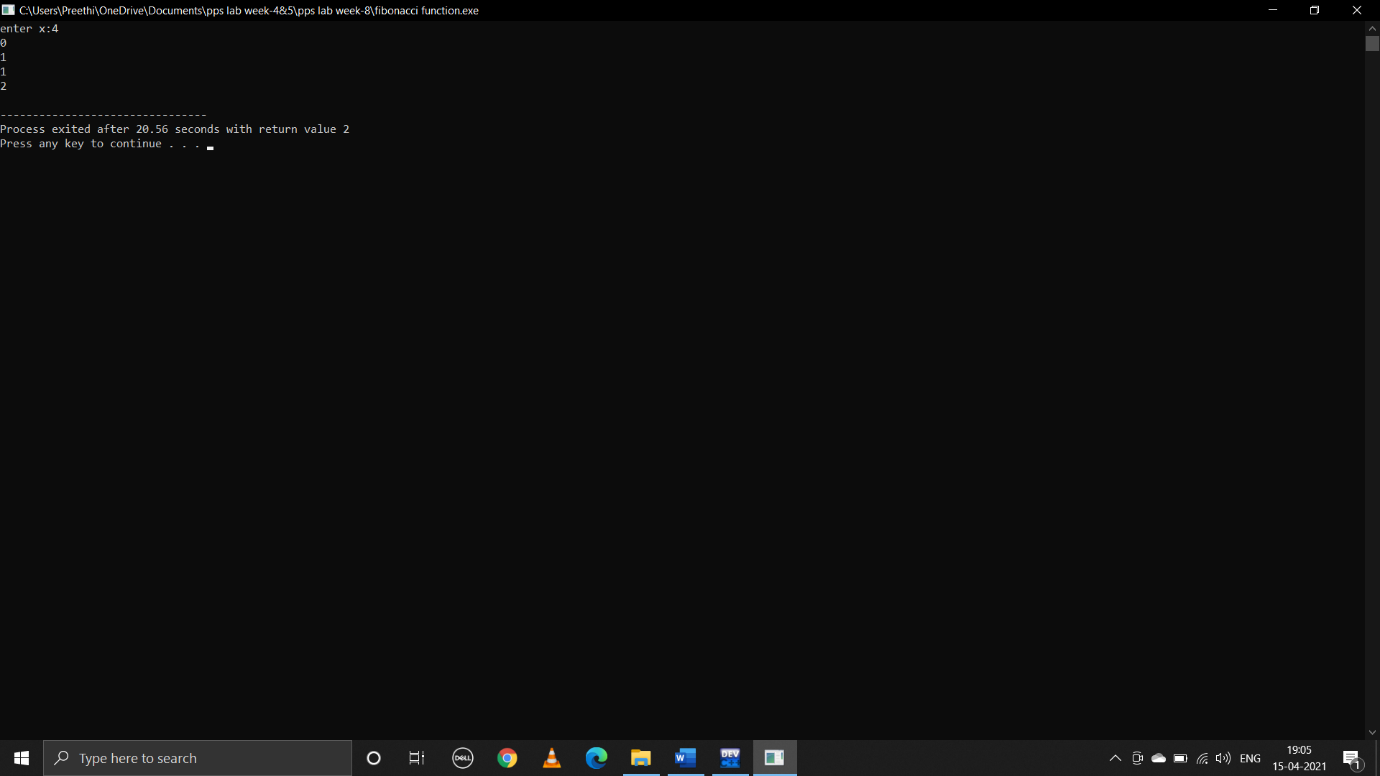
n1=n2;

n2=nth;

}

}

**output:**

****

**iii).swapping the values of two variable**

**pseudocode:**

begin

take user input values of a and b

to swap a and b call the function swap(a,b)

control jumps to function definition

declare int temp

temp=x

x=y

y=temp

print(x,y)

end

**c program:**

#include<stdio.h>

void swap(int,int);

main()

{

int a,b;

printf("enter a:");

scanf("%d",&a);

printf("enter b:");

scanf("%d",&b);

swap(a,b);

}

void swap(int x,int y)

{

printf("before swapping:\n a is %d\n b is %d\n",x,y);

int temp;

temp=x;

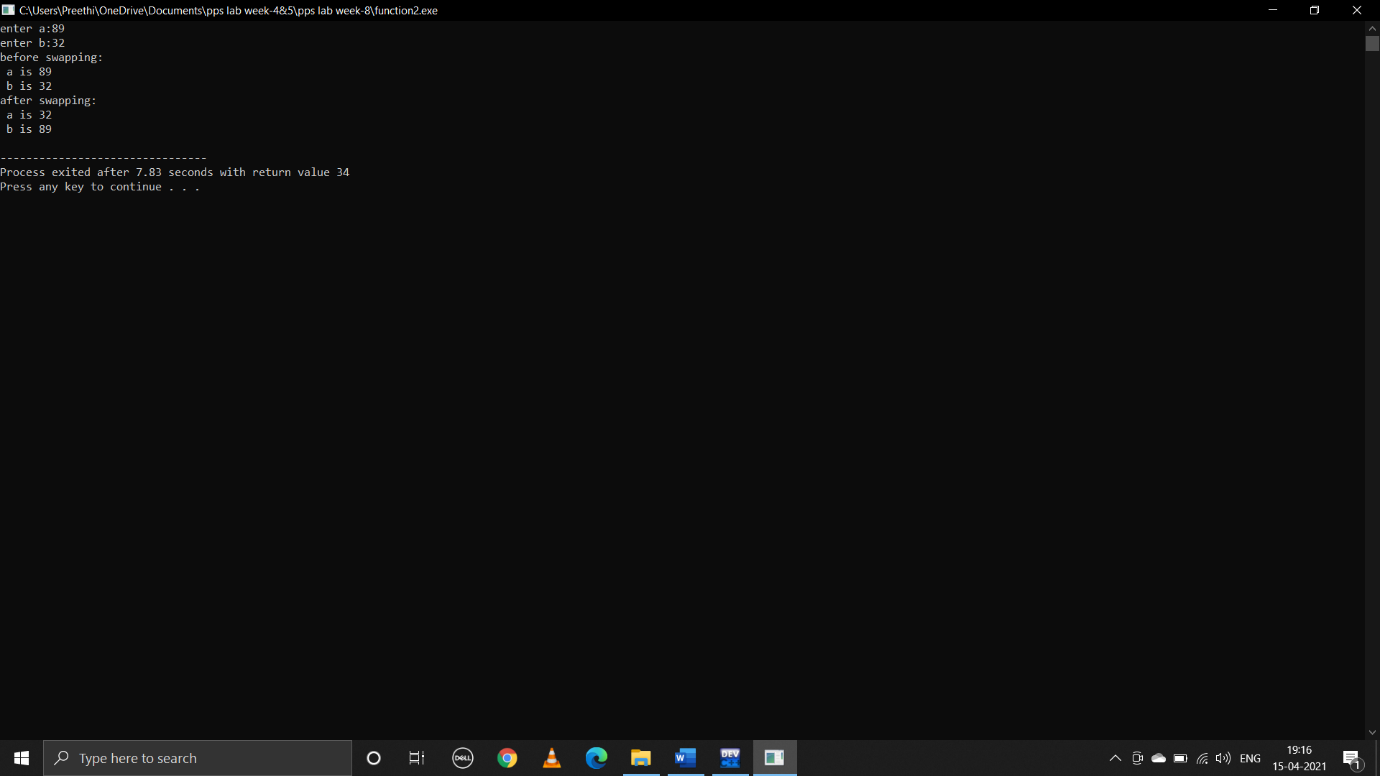
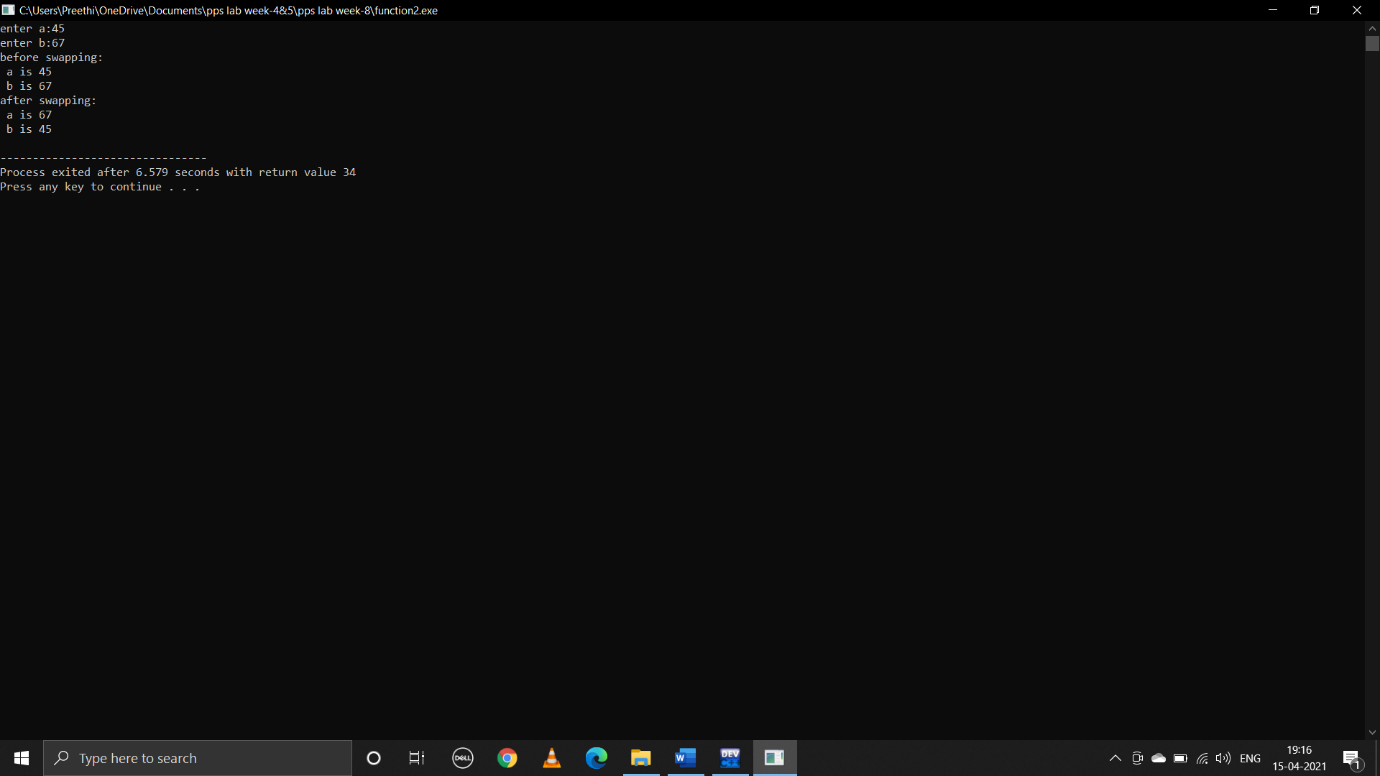
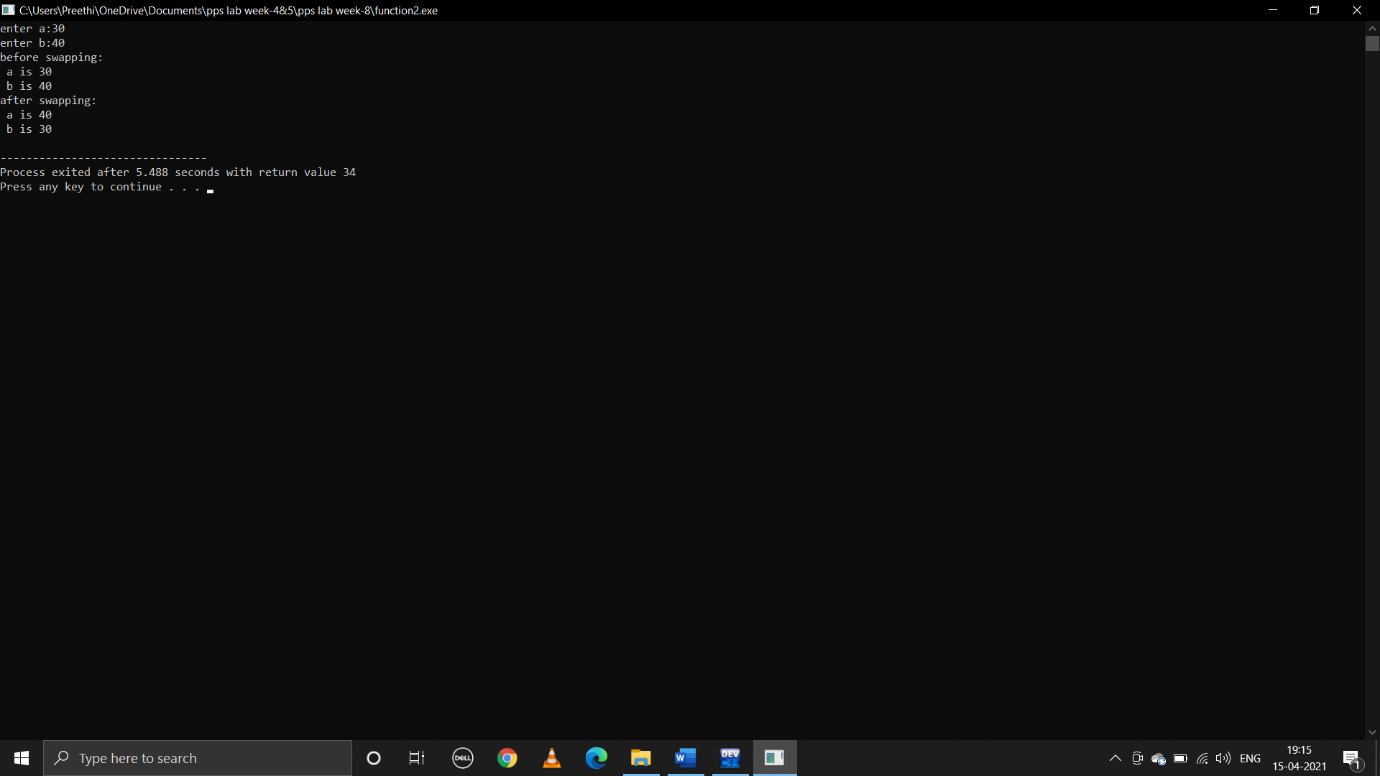
x=y;

y=temp;

printf("after swapping:\n a is %d\n b is %d\n",x,y);

}

**output:**

****

**2.write a c program that uses functions to perform the following operations**

**i) To insert a sub-string in to a given main string from a given postion**

**pseudocode:**

begin

take user input mainstring,substring and position to insert substring into main string

call the function insert(mainstr,substr,p)

control jumps to function defnition

decalre int i,m,n

m=strlen(mainstr)

n=strlen(substr)

create a blank space in mainstring at specified positon of size strlen(substr)

for(i=m-1;i>=p;i--)

mainstr[i+n]=mainstr[i]

now insert substring at that blank location

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

mainstr[i+p]=substr[i]

print(mainstring)

end

**c program:**

#include<stdio.h>

#include<string.h>

void insert(char mainstr[100],char substr[100],int p);

main()

{

char mainstr[100];

char substr[100];

int p;

printf("enter main string:");

gets(mainstr);

printf("enter substring:");

gets(substr);

printf("enter position where to insert substring:");

scanf("%d",&p);

insert(mainstr,substr,p);

}

void insert(char mainstr[100],char substr[100],int p)

{

int i,m,n;

m=strlen(mainstr);

n=strlen(substr);

for(i=m-1;i>=p;i--)

{

mainstr[i+n]=mainstr[i];

}

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

{

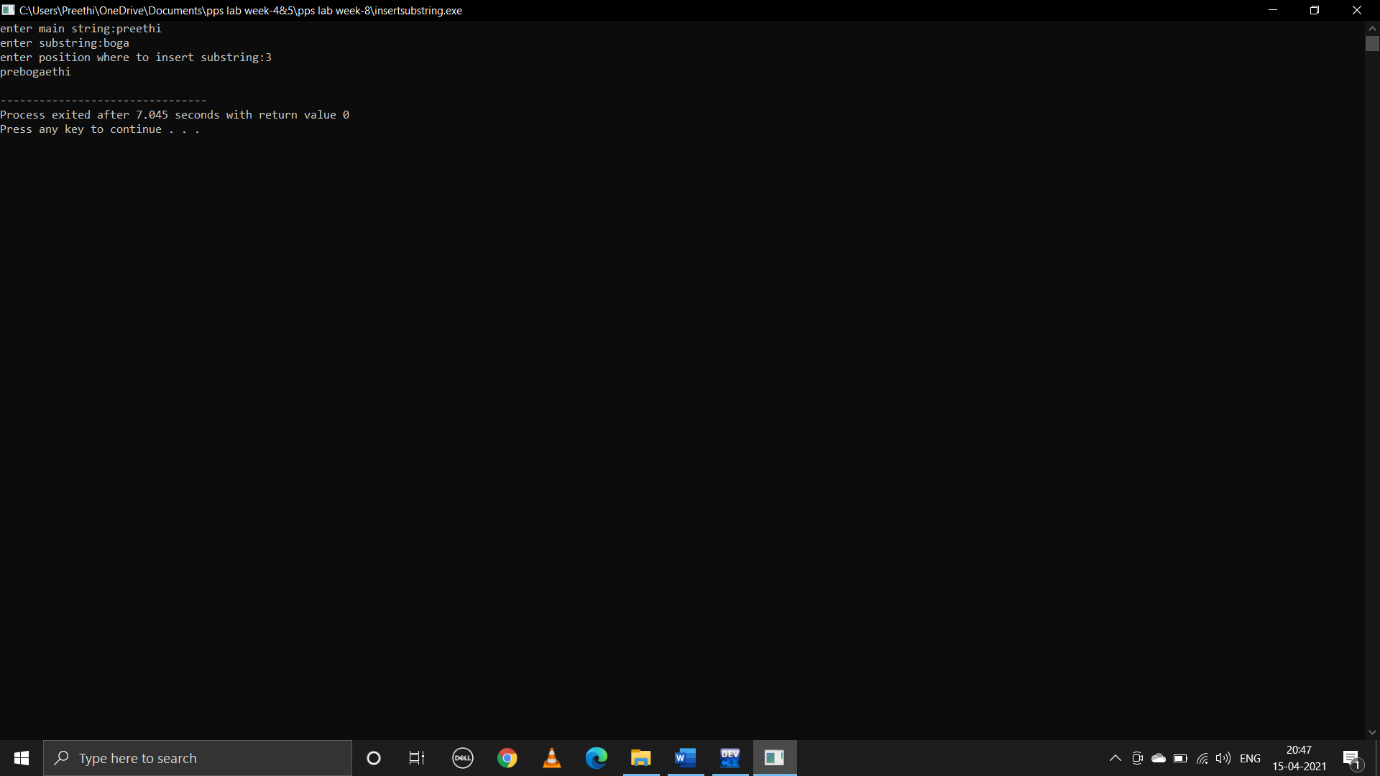
mainstr[i+p]=substr[i];

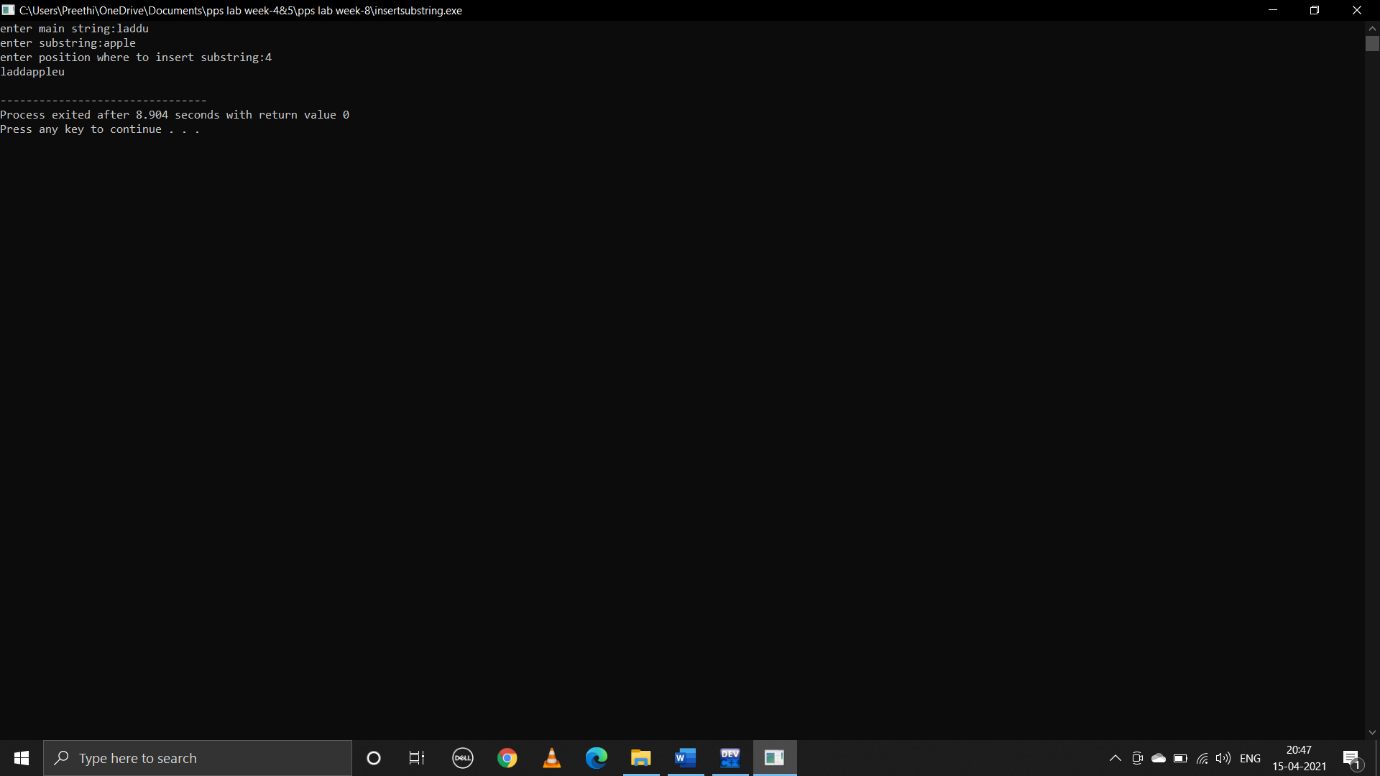
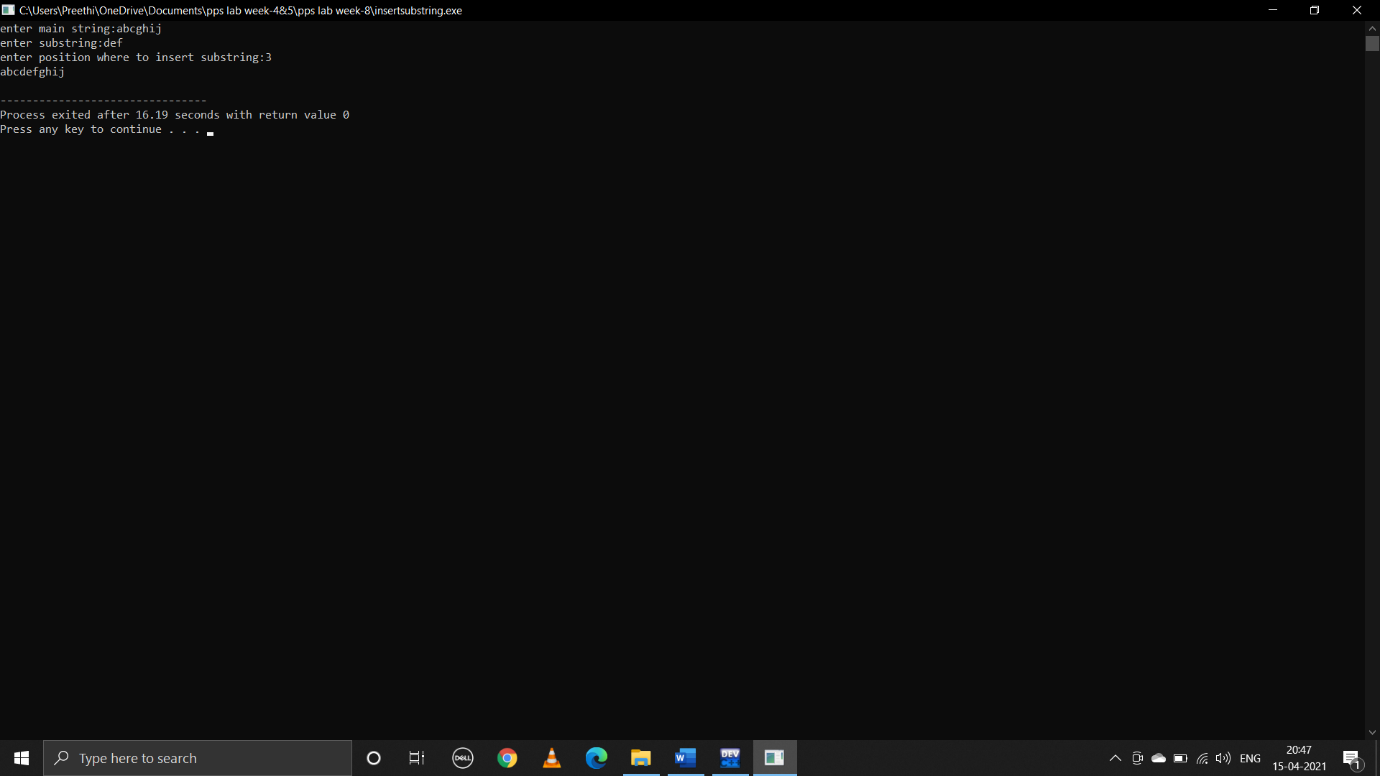
}

mainstr[m+n]=’\0’;

puts(mainstr);

}

**output: **

****

**ii) To delete n characters from a given position in a given string**

**pseudocode:**

begin

take user input string

take user input position (location)

take user input n to delete number of characters

len = strlen(s)

call the function del(s,loc,n,len)

control jumps to function defnition

declare int i,j

if(loc>len)

print(invalid location)

else

{

Declare char s2[100]

Copy characters of s1 upto loc to s2

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

s2[i]=s1[i]

take another loop starting from loc+n to len

copy remaining charcters to s2

for(j=loc+n;j<len;j++,i++)

s2[i]=s1[j]

end s2 with null character

print s2

end

**c program:**

#include<stdio.h>

#include<string.h>

char del(char s1[],int,int,int);

int main()

{

char s1[100];

int loc,n,len;

printf("enter string:\n");

gets(s1);

printf("enter location:\n");

scanf("%d",&loc);

printf("enter no.of characters to delete:\n");

scanf("%d",&n);

len=strlen(s1);

del(s1,loc,n,len);

}

char del(char s1[],int loc,int n,int len)

{

int i,j;

if(loc>len)

{

printf("invalid location\n");

}

else

{

char s2[100];

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

{

s2[i]=s1[i];

}

for(j=loc+n;j<len;j++,i++)

{

s2[i]=s1[j];

}

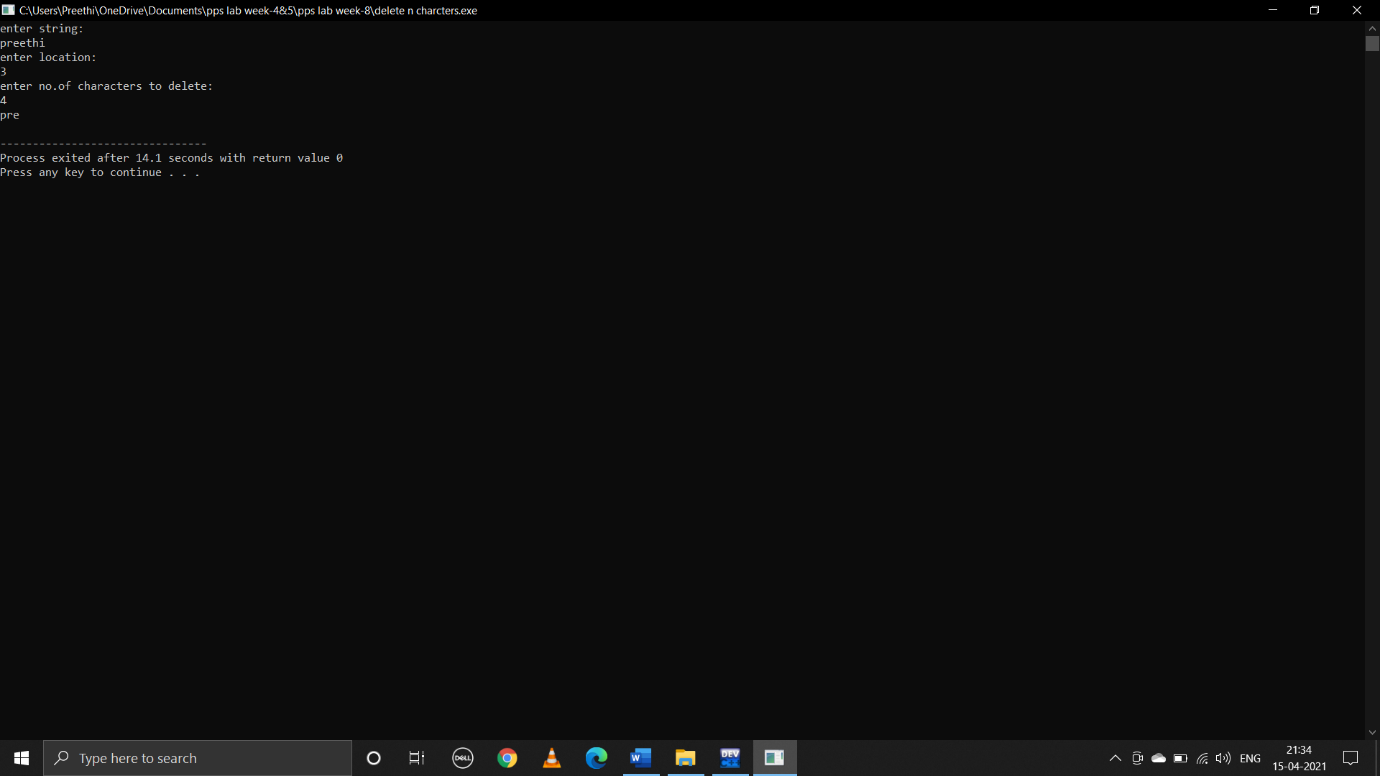
s2[i]='\0';

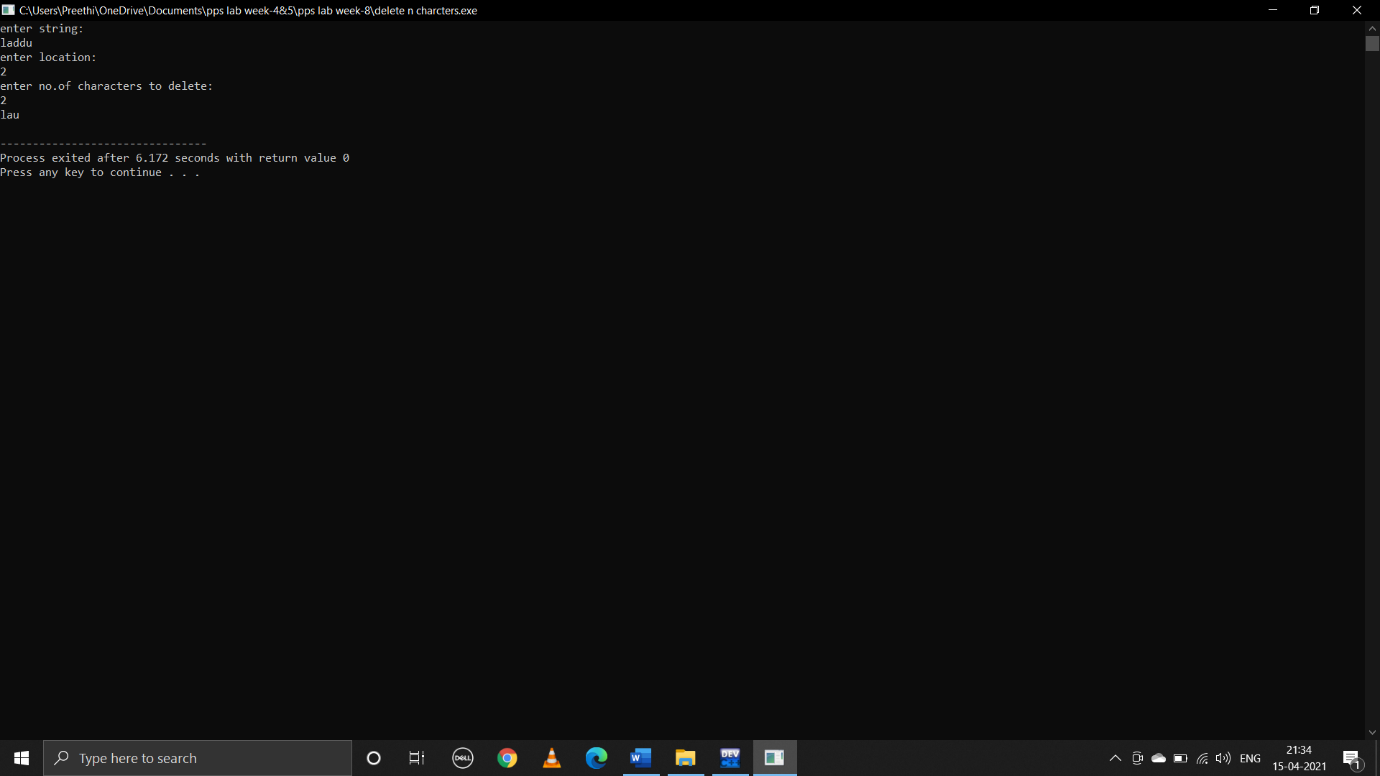
puts(s2);

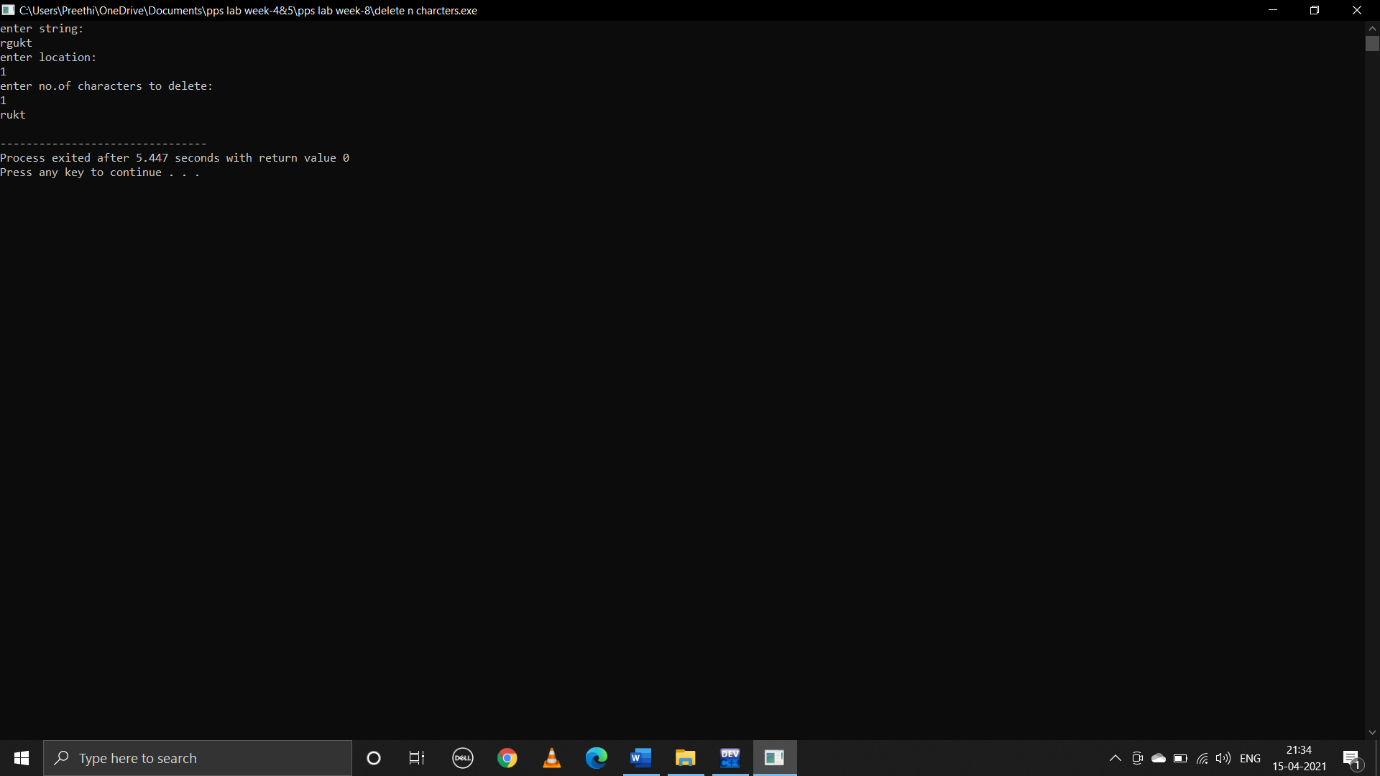
}

}

**output:**

****

****

****

**WEEK-8**

**1.C program that reads N integer numbers and searches for a given number using linear search**

**Pseudocode:**

Begin

Take user input size array arr[a]

Declare comp=0,i,key

Take user input key

Run loop from i=0;i<n

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

{

comp=comp+1

if (arr[i]==key)

print(found at postion i+1)

print(no.of comparisions comp)

break

if(comp>a)

print("key not found")

print("no.of comparisions %d",a)

}

end

**C program:**

#include<stdio.h>

void main()

{

int a;

printf("size of array:");

scanf("%d",&a);

int arr[a],i,key,comp=0;

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

{

printf("enter %d element:",i);

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

}

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

{

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

}

printf("\n");

printf("enter key:");

scanf("%d",&key);

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

{

comp=comp+1;

if (arr[i]==key)

{

printf("found at postion %d\n",i+1);

printf("no.of comparisions %d\n",comp);

break;

}

if(comp>a)

{

printf("key not found\n");

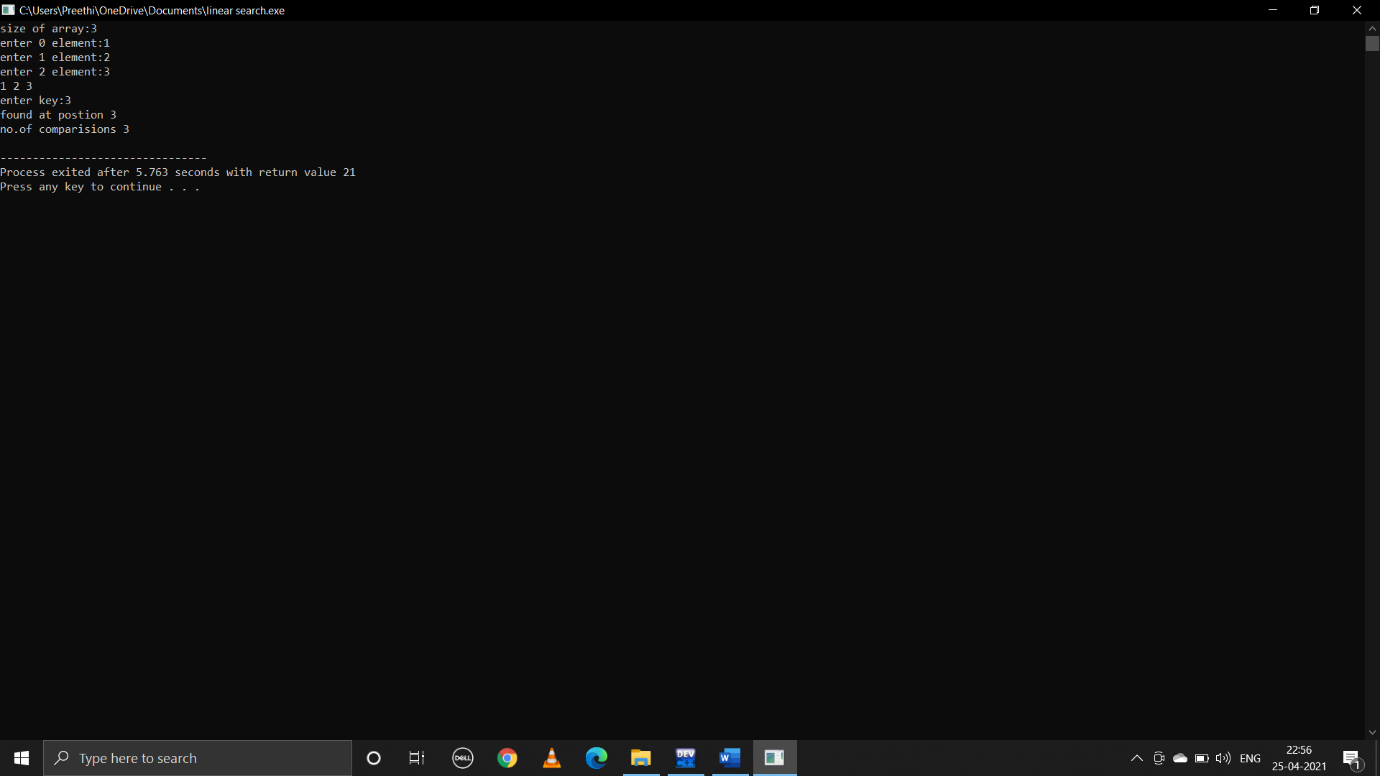
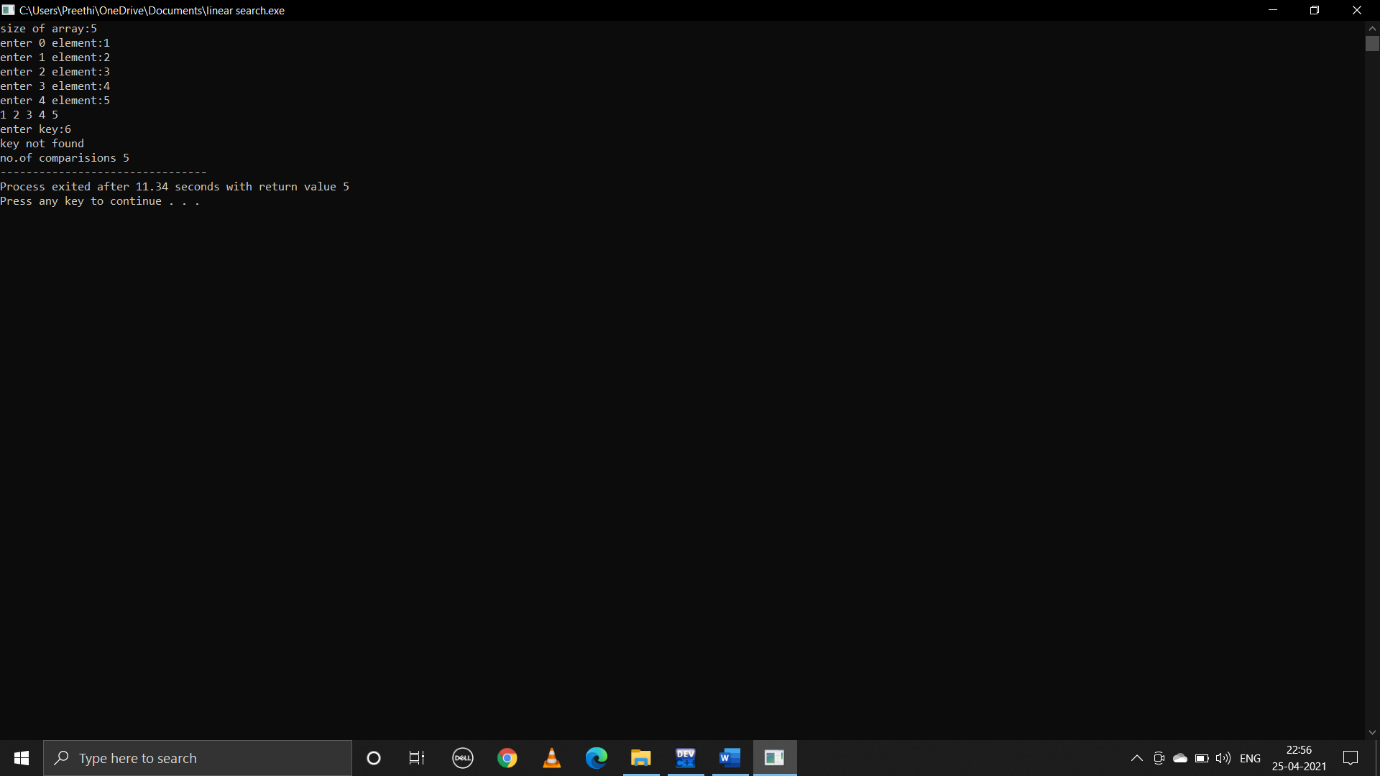
printf("no.of comparisions %d",a);

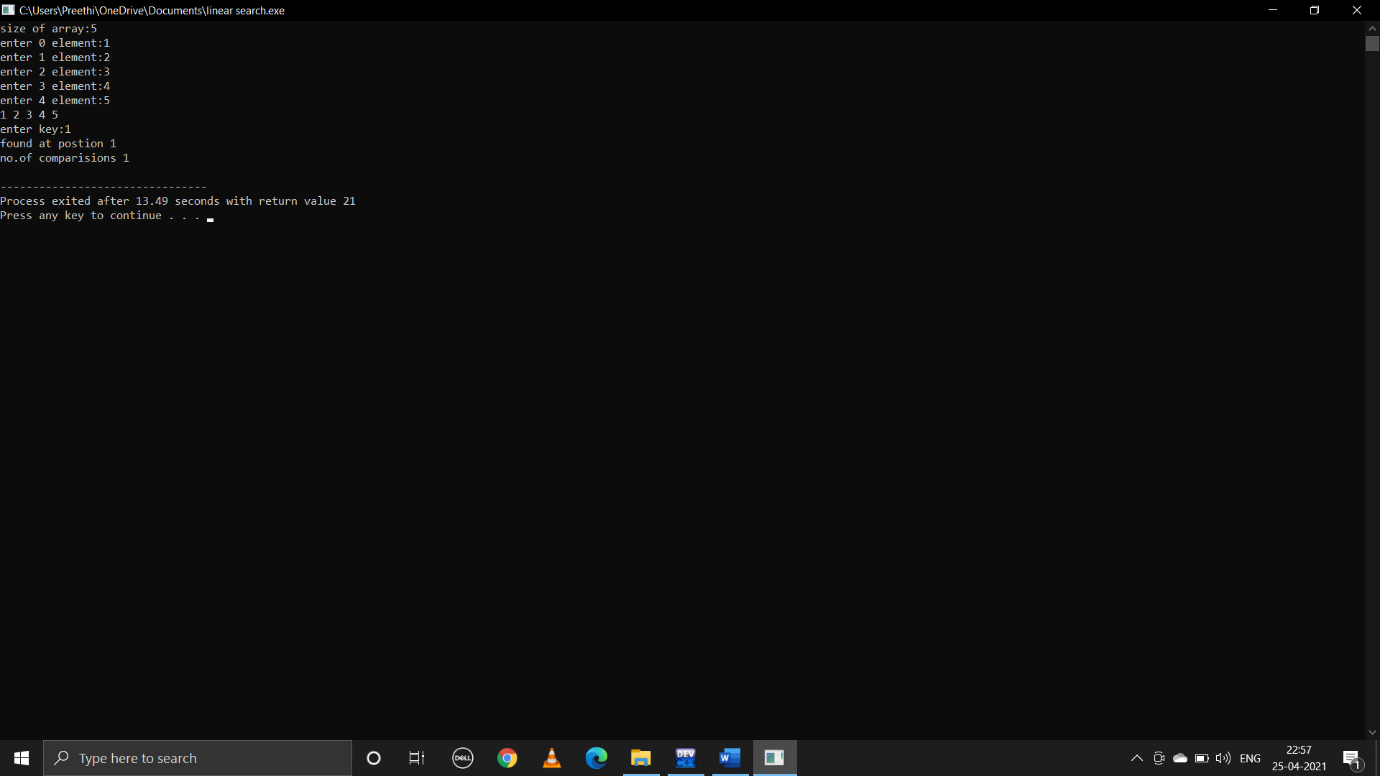
}

}

}

**Output:**

****

****

**2.C program that reads N integer numbers and searches for a given number using binary search**

**Pseudocode:**

Begin

Take user input size array arr[a],key

Decalre first=0,last=a-1;

middle=(first+last)/2;

while(first<=last)

if(arr[middle]<key)

first=middle+1

else if(arr[middle]==key)

print(key found at middle index)

break

else

last=middle-1

middle=(first+last)/2

if(first>last)

print(not found)

end

**C program:**

#include<stdio.h>

void main()

{

int a;

printf("enter size of array:");

scanf("%d",&a);

int arr[a],i,key,first,last,middle;

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

{

printf("enter %d element:",i);

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

}

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

{

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

}

printf("\n");

printf("enter key:");

scanf("%d",&key);

first=0;

last=a-1;

middle=(first+last)/2;

while(first<=last)

{

if(arr[middle]<key)

first=middle+1;

else if(arr[middle]==key)

{

printf("key found at index %d ",middle);

break;

}

else

last=middle-1;

middle=(first+last)/2;

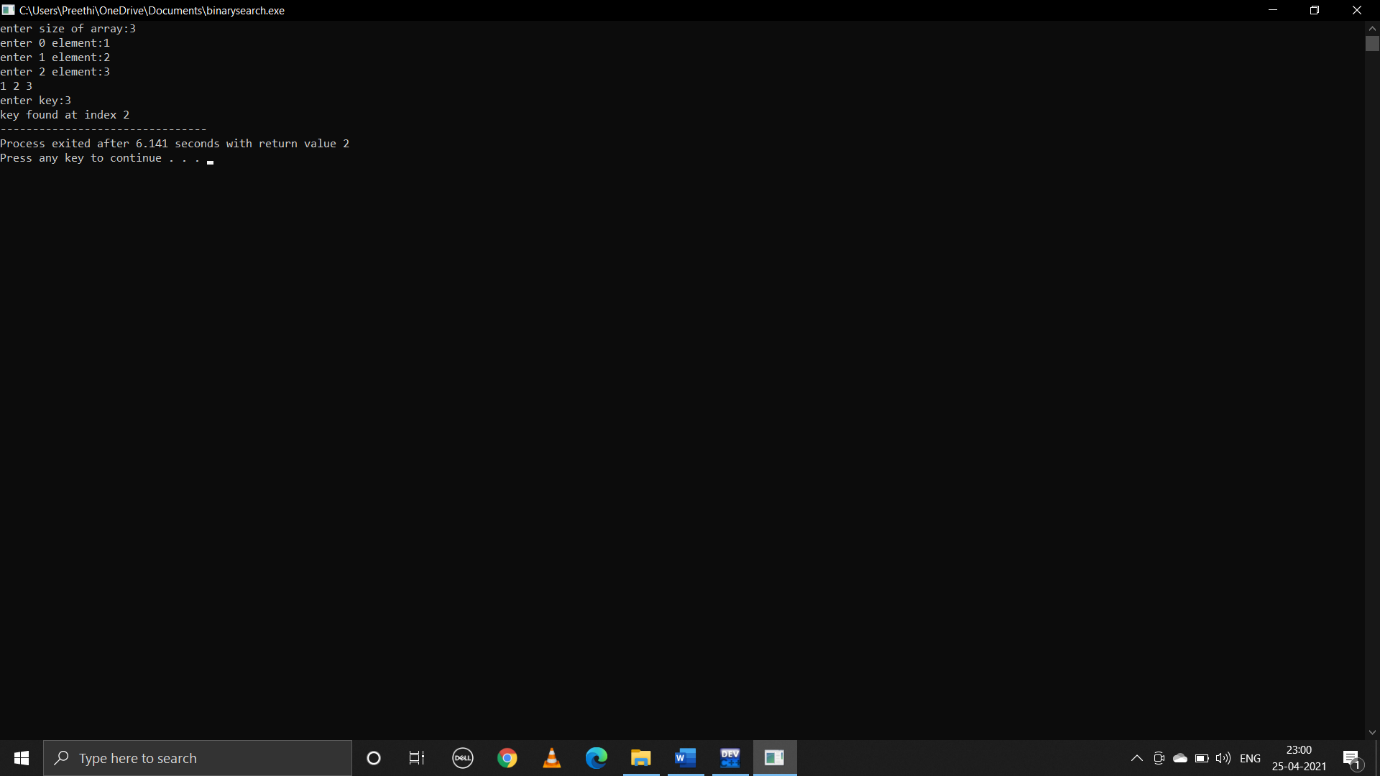
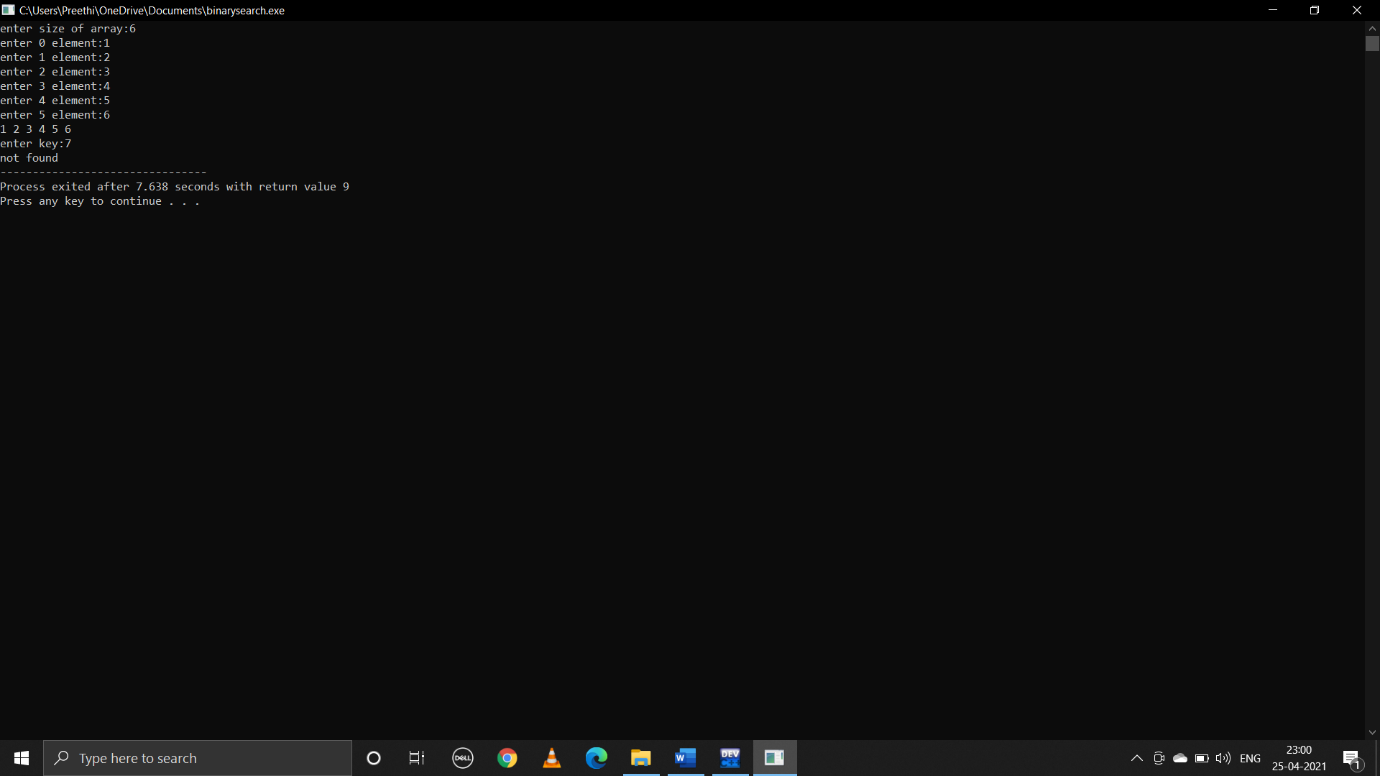
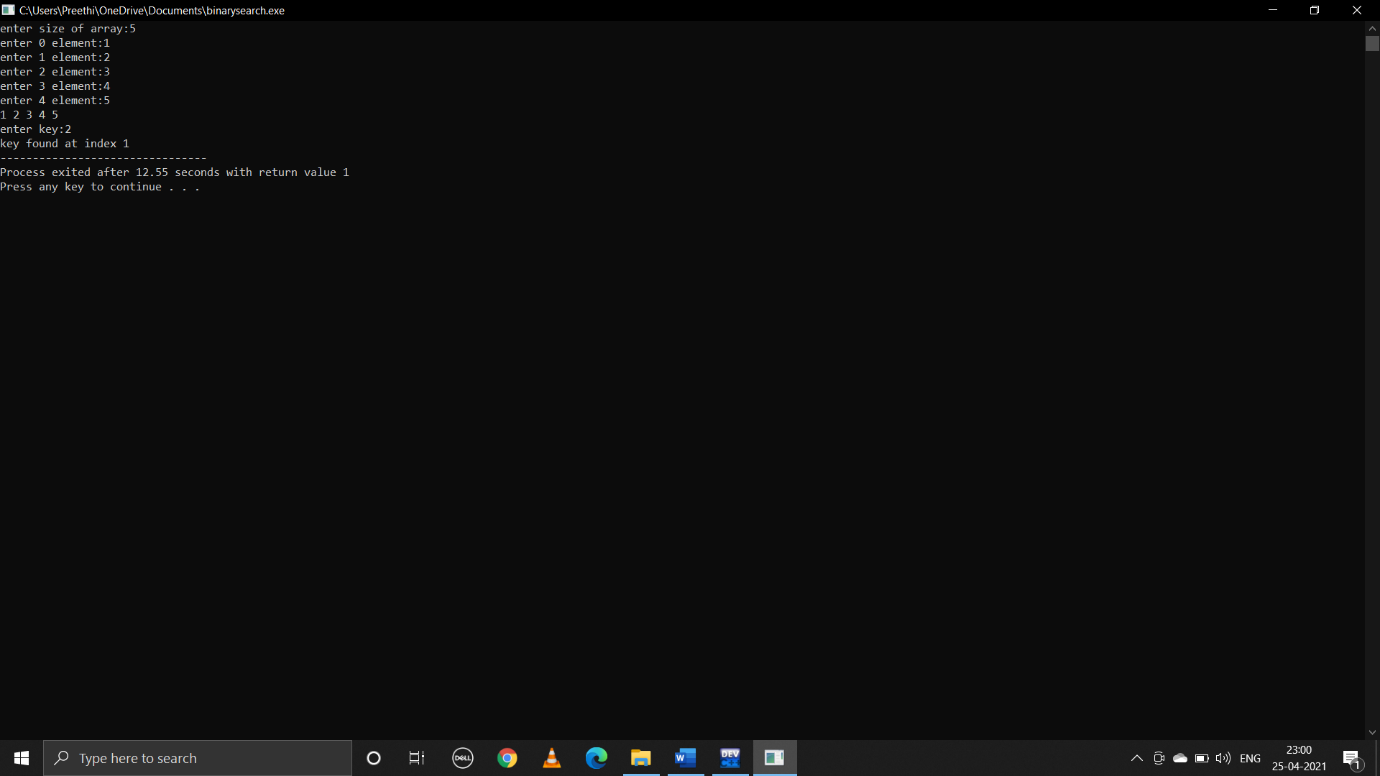
}

if(first>last)

printf("not found");

}

**Output:**

****

**3.C program that reads N integer numbers and finds the sum**

**Pseudocode:**

Begin

Take use input size array arr[a]

Declare sum=0

Run loop from i=0 to i<a

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

sum=sum+arr[i]

Print(sum)

end

**C program:**

#include <stdio.h>

void main()

{

int a;

printf("enter size:");

scanf("%d",&a);

int arr[a],sum=0,i;

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

{

printf("enter %d element:",i);

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

}

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

{

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

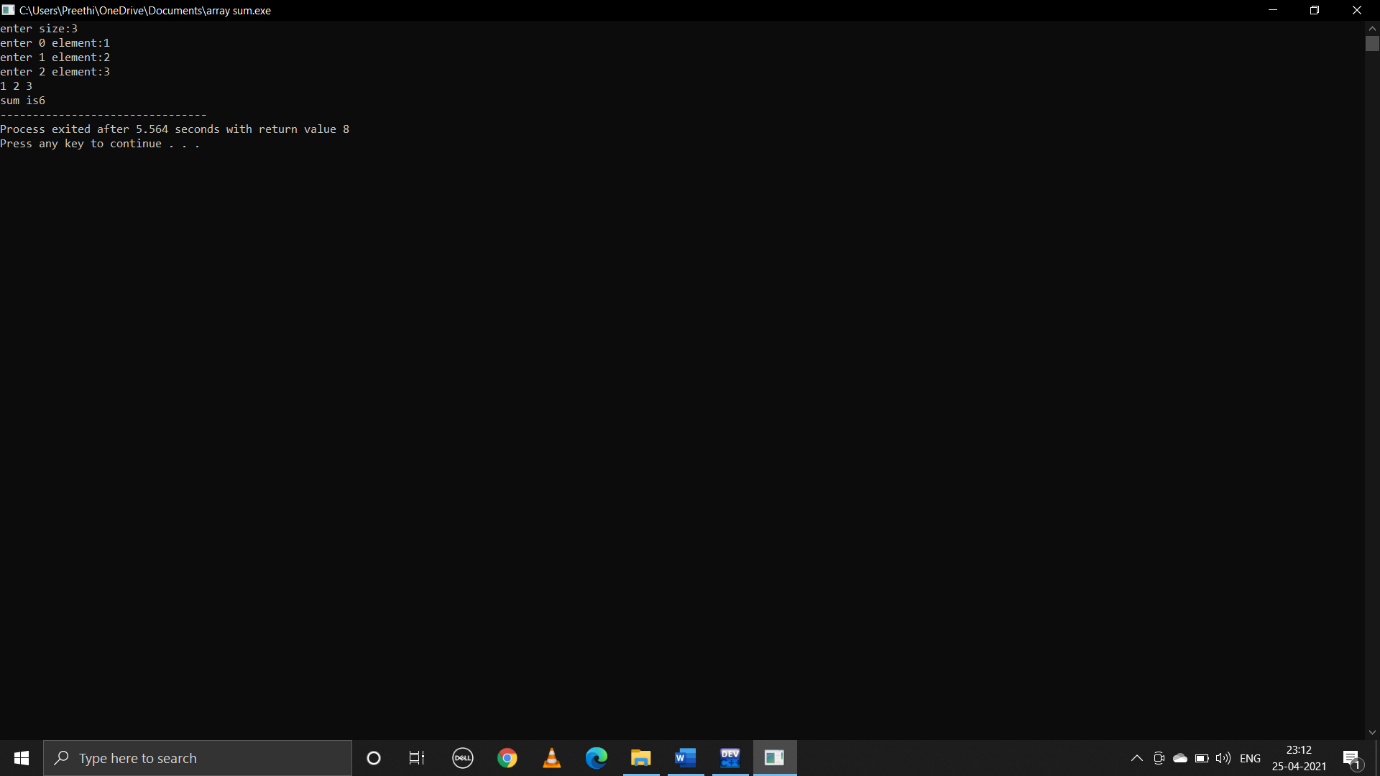
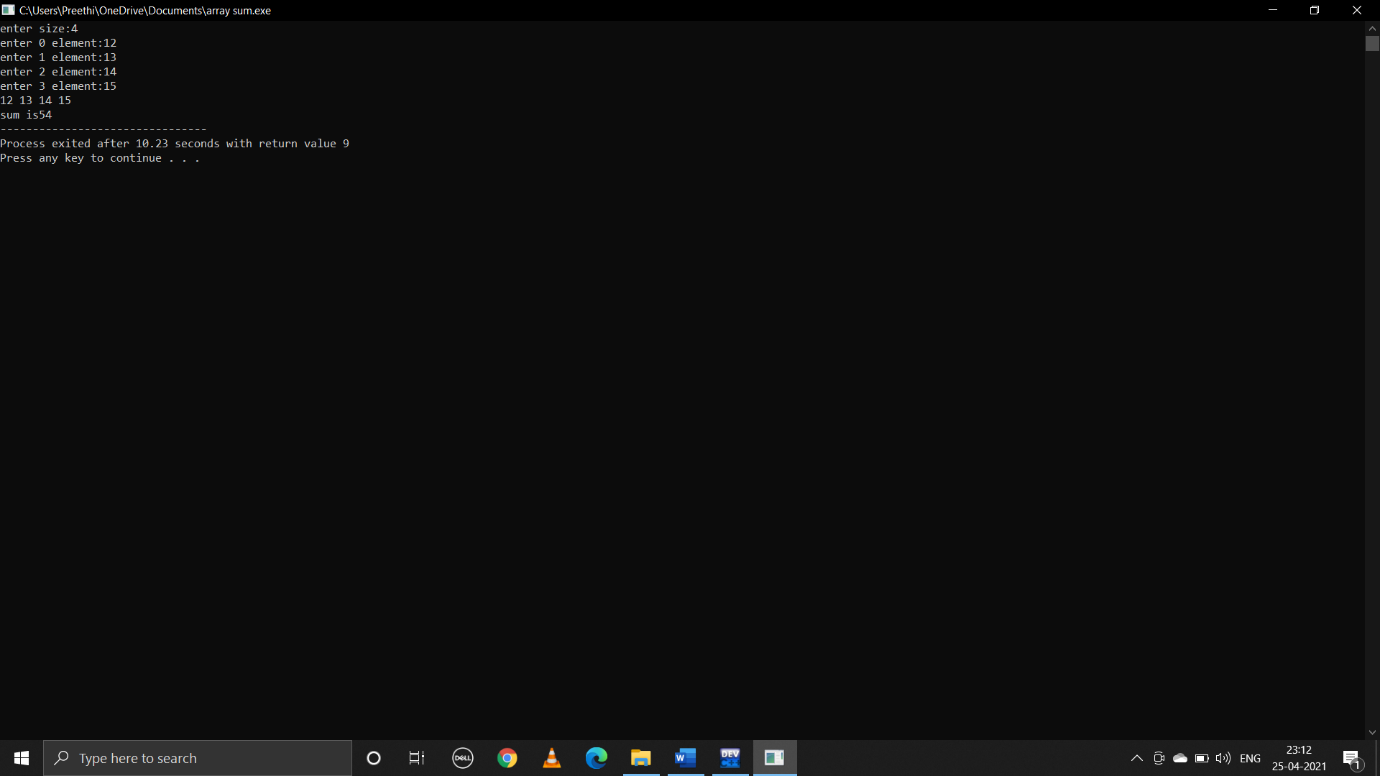
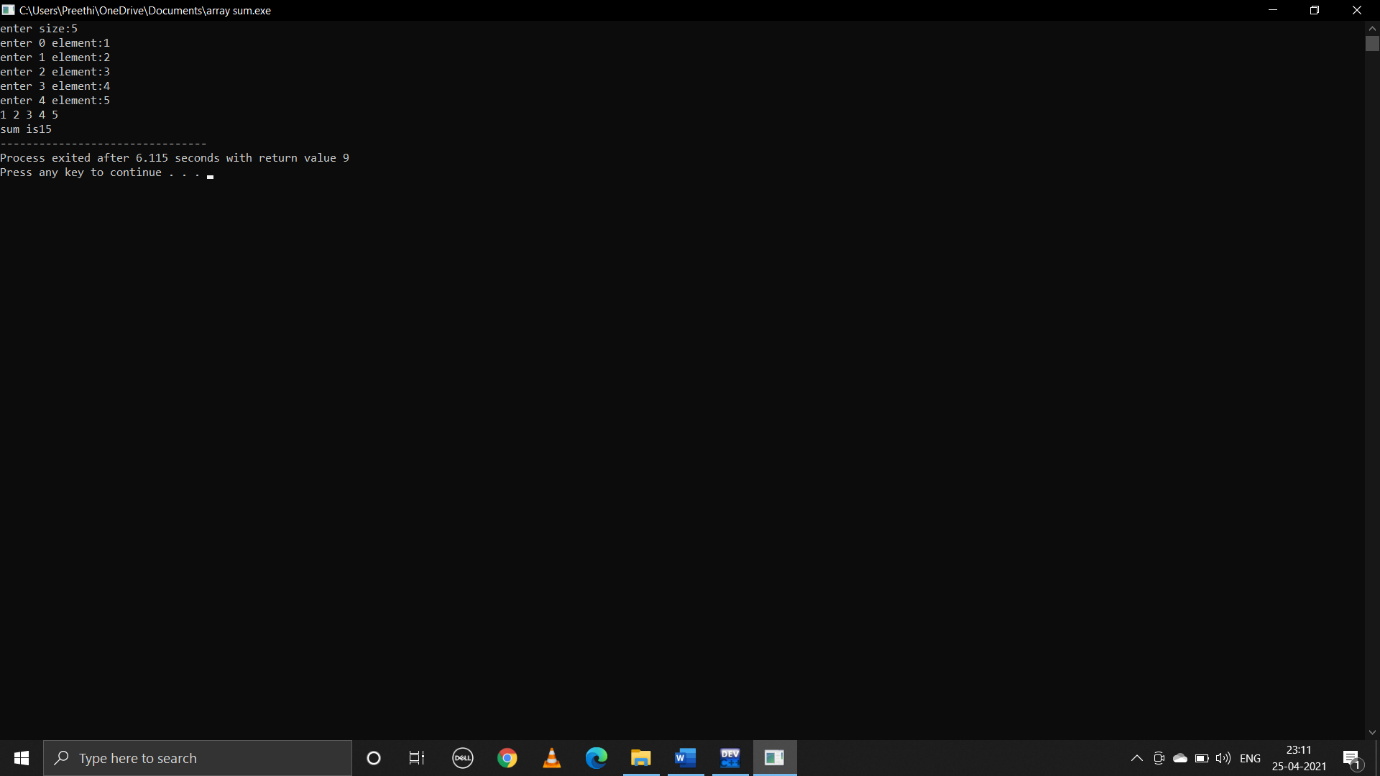
sum=sum+arr[i];

}

printf("\nsum is%d",sum);

}

**Output:**

****

**4.C program that reads N integers numbers and arrange them in ascending order using Bubble sort**

**Pseudocode:**

begin

take user input size of an array n

int a[n]

run loop from i=0 to i<size to print and enter elements of an array

now,to sort this user input array call the function bubblesort(a,n)

declare functions: bubblesort(int arr[], int size) and printarray(int arr[], int size)

bubblesort function:

int i,j,temp

run loops two times:one to run through the array and other is for comparision

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

for(j=0;j<size-1;j++)

if(arr[j]>arr[j+1])

swap j+1 and j

to print array status after each pass call the function printarray which prints array

end

**C program:**

#include<stdio.h>

void bubblesort(int arr[],int size);

main()

{

int n,i;

printf("enter size of an array:");

scanf("%d",&n);

int a[n];

printf("enter elements in array:\n");

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

{

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

}

printf("unsorted array is:\n");

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

{

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

}

printf("\n");

bubblesort(a,n);

}

void bubblesort(int arr[],int size)

{

int temp,i,j;

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

{

for(j=0;j<size-1;j++)

{

if(arr[j]>arr[j+1])

{

temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

printf("array status after pass %d\n",i);

printarray(arr,size);

printf("\n");

}

}

void printarray(int arr[],int size)

{

int i;

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

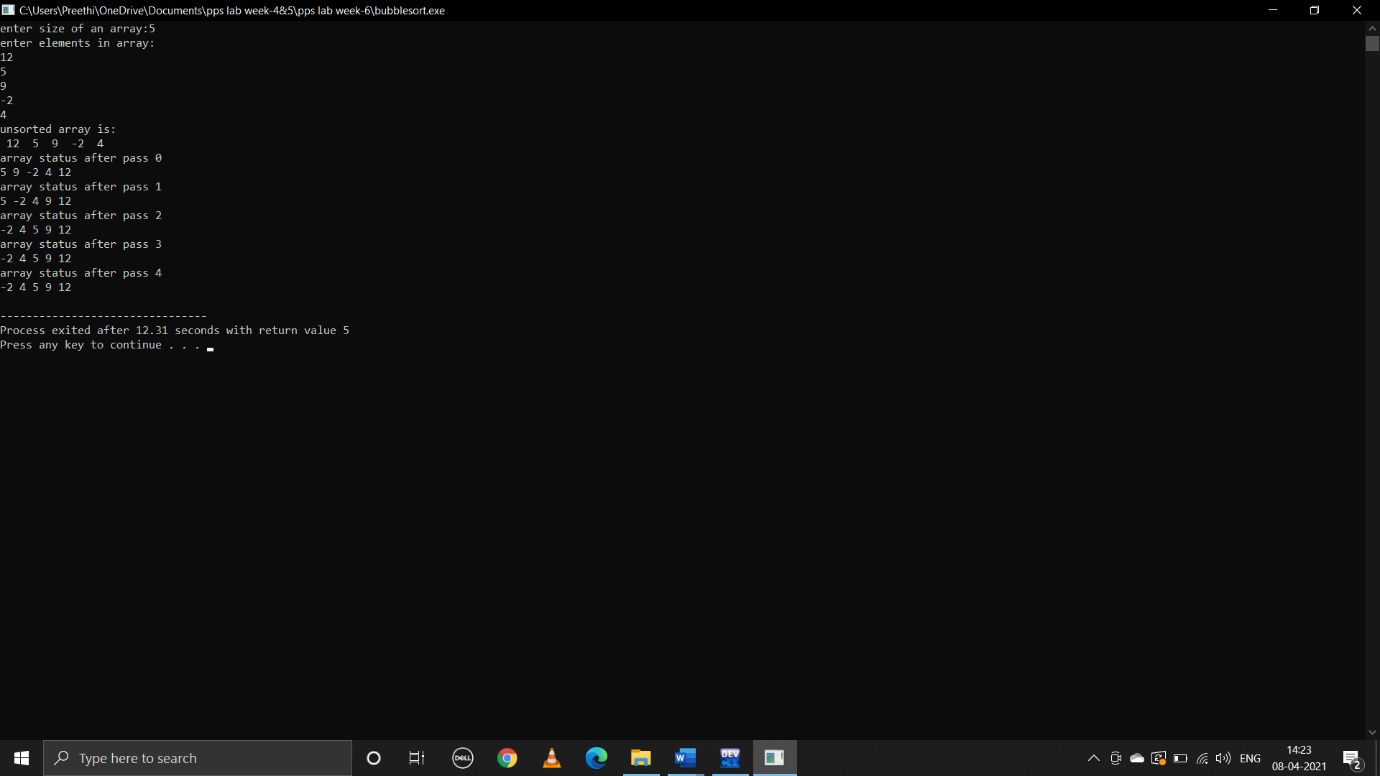
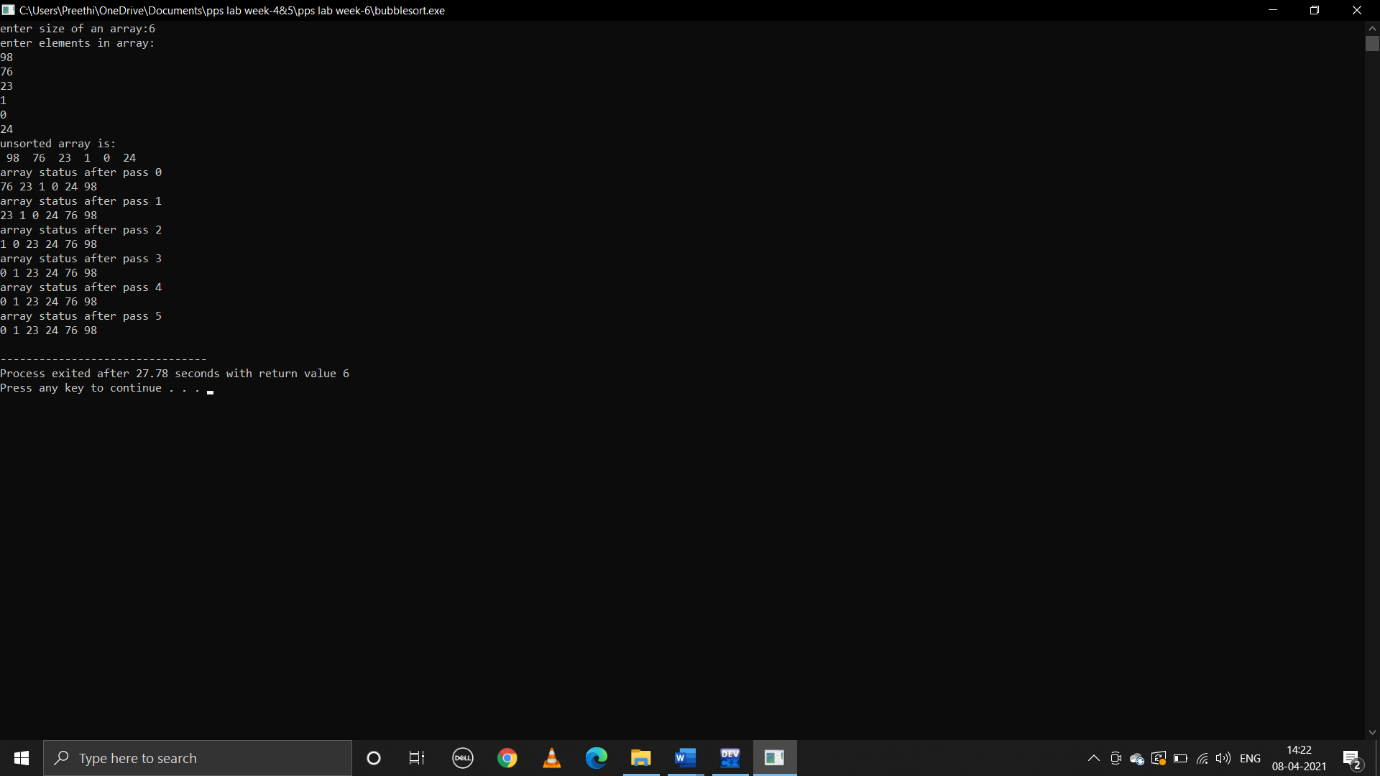
{

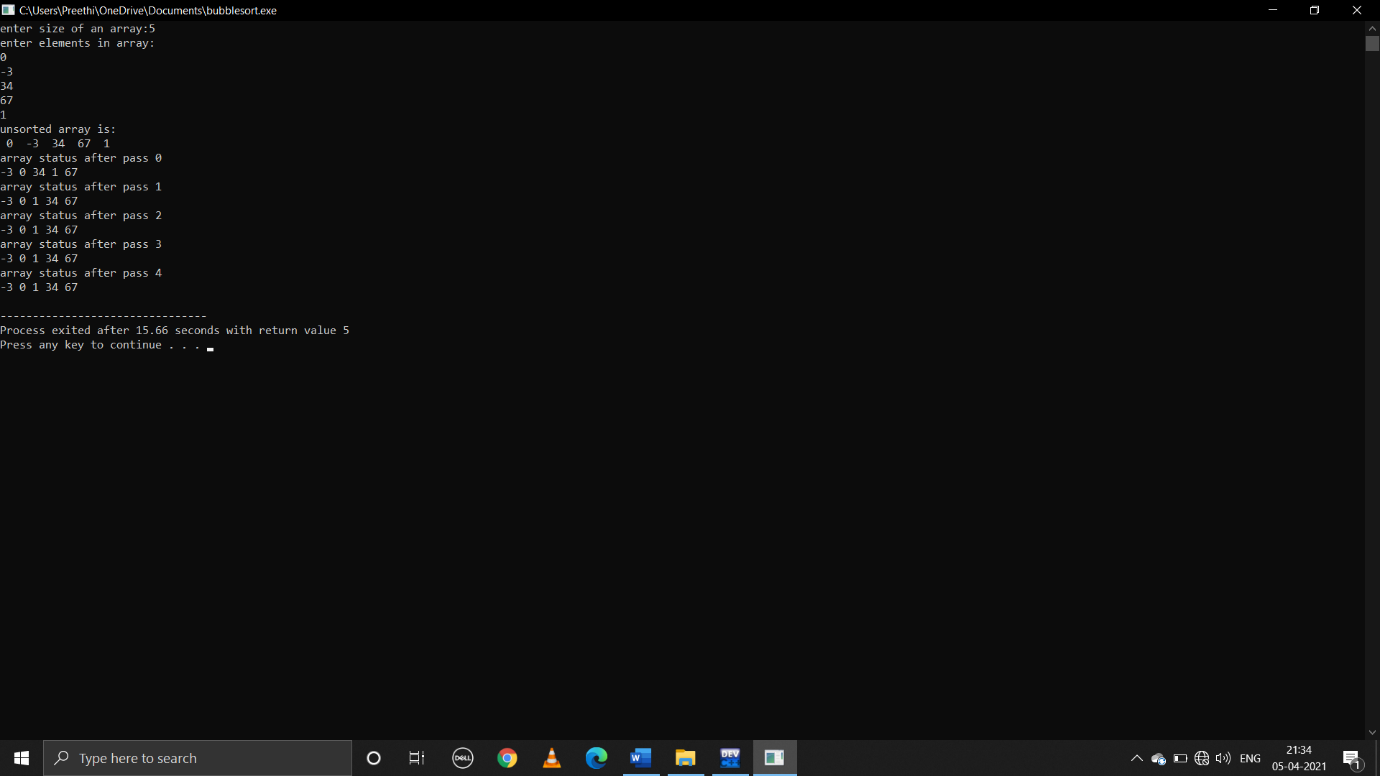
printf("%d ",arr[i]);

}

}

**Output:**

****

****

**5.C program that reads N integer numbers and arrange them in ascending order using merge sort**

**Pseudocode:**

Begin

Take user input array of size n

Declare left and right variables which will mark the extreme indices of the array

Left will be assigned to 0 right will be assigned to n-1

For sorting this call the function mergesort function

Now control jumps to the mergesort function

Find mid=(left+right)/2

Call mergesort on(left,mid,n) and(mid+1,last,n)

Above will continue till left<right

Then we will call merge on the 2 subproblems

merge(arr,p,q,r,n)

n1=[p-q+1]

n2=[r-q]

int l[n1]

int m[n2]

for(i=0;i<n1;i++)

l[i]=arr[p+i]

for(j=0;j<n2;j++)

m[j]=arr[q+1+j]

k=p

l and m have elements

while(i<n1 && j<n2)

{

if(l[i]<=m[j])

we are merging elements in ascending to arr[k]

add l[i] to arr[k]

i++

else add m[j] to arr[k]

j++

k++

}

While l has elements

While(i<n1)

{

arr[k]=l[i]

i++

k++

}

While m has elements

While(j<n2)

{

arr[k]=m[j]

j++

k++

}

Call the function print array which prints array after each pass

Printarray(arr,size)

{

Declare static int i=0 to store value of last execution

Print i

i++

run loop from int j=0 to j<size

{

Print arr[i]

}

End

**C program:**

#include<stdio.h>

void merge(int arr[],int p,int q,int r,int n)

{

int n1=q-p+1;

int n2=r-q;

int l[n1],m[n2];

int i,j,k;

for(i=0;i<n1;i++)

{

l[i]=arr[p+i];

}

for(j=0;j<n2;j++)

{

m[j]=arr[q+1+j];

}

i=0;

j=0;

k=p;

while(i<n1 && j<n2)

{

if(l[i]<=m[j])

{

arr[k]=l[i];

i++;

}

else

{

arr[k]=m[j];

j++;

}

k++;

}

while(i<n1)

{

arr[k]=l[i];

i++;

k++;

}

while(j<n2)

{

arr[k]=m[j];

j++;

k++;

}

printarray(arr,n);

}

void mergesort(int arr[],int l,int r,int n)

{

if(l<r)

{

int m=(l+r)/2;

mergesort(arr,l,m,n);

mergesort(arr,m+1,r,n);

merge(arr,l,m,r,n);

}

}

void printarray(int arr[],int size)

{

static int i=0;

printf("array status after pass %d:",i);

i++;

int j;

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

{

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

}

printf("\n");

}

int main()

{

int n;

printf("enter size of an array:");

scanf("%d",&n);

int i,arr[n];

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

{

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

}

printf("unsorted array is:\n");

printarray(arr,n);

mergesort(arr,0,n-1,n);

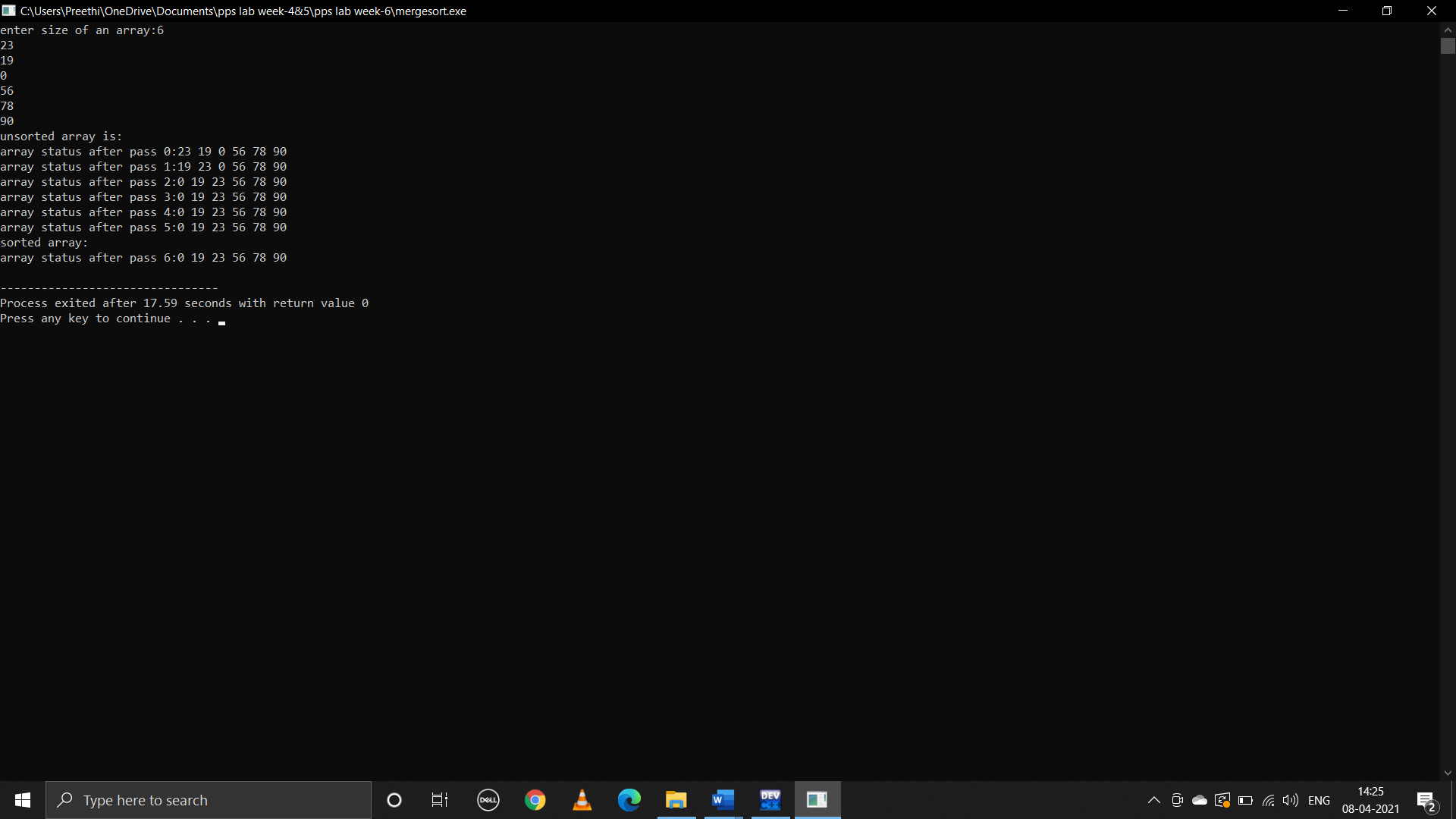
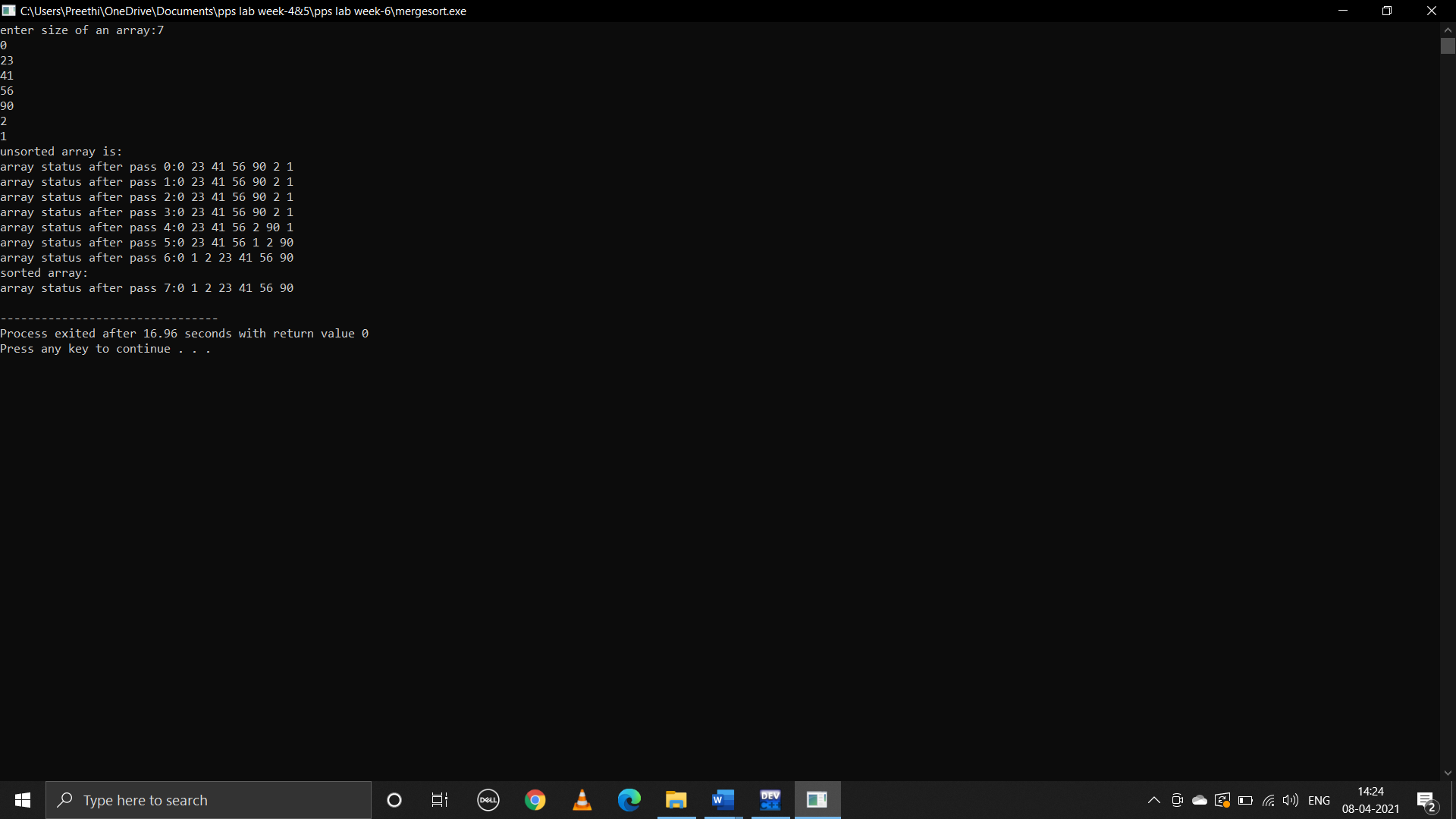
printf("sorted array:\n");

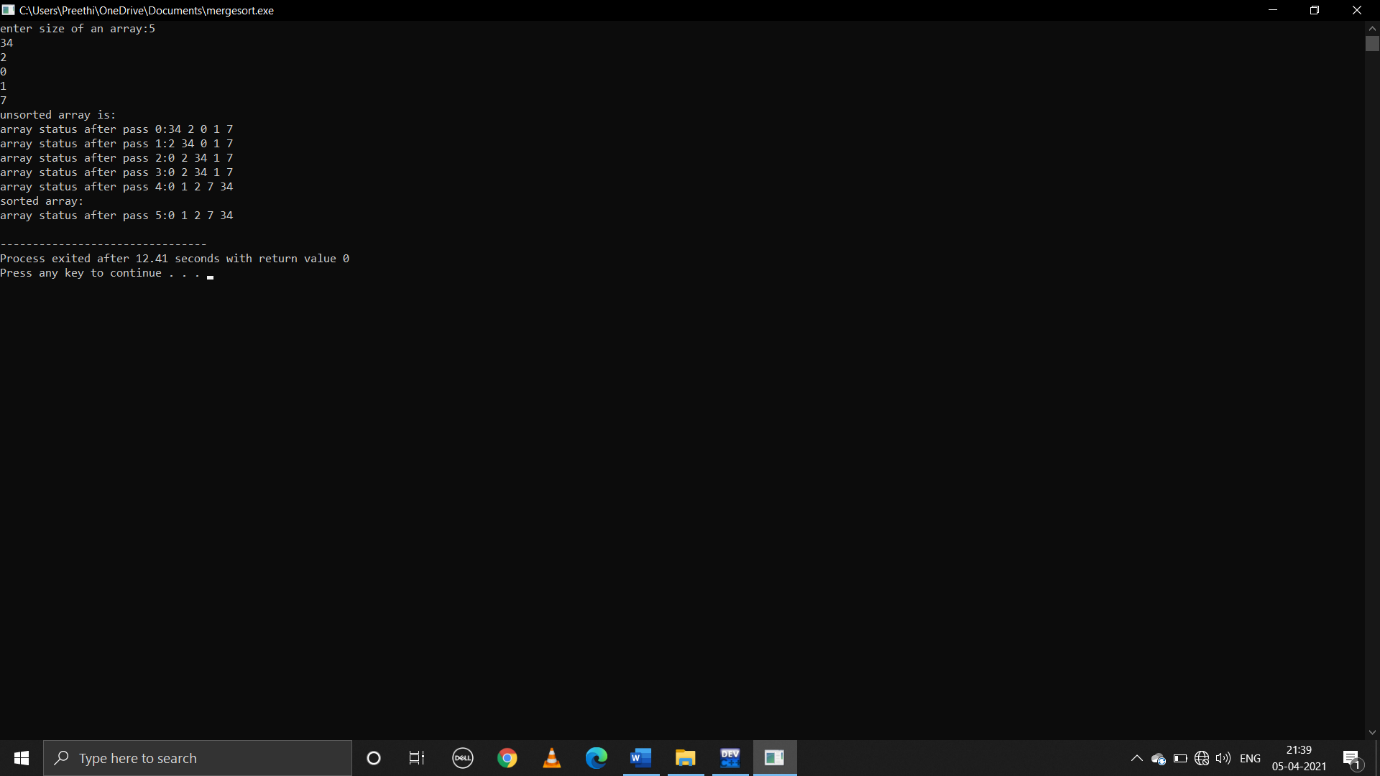
printarray(arr,n);

return 0;

}

**Output:**

****

****

**6.C program that reads N integer numbers and arrange them in ascending order using Quick Sort**

**Pseudocode:**

begin

declare int i,n,a[]

take user input array of size n

call the function quicksort

control jumps to the function definition quicksort

in this function declare int i,j,pivot ,temp

first element =pivot

i=first

j=last

while(i<j)

while(a[i]<=a[pivot]&&i<last)

{ i++;

while(a[j]>a[pivot])

j--;

if(i<j)

swap a[i] and a[j]

}

Swap a[j] and a[pivot]

Now pivot element is in its correct position

Call printarray function which prints array to print array status after each pass

Printarray(arr,size)

{

Declare static int i=0 to store value of last execution

Declare static int i=0 to store value of last execution

Print i

i++

run loop from int j=0 to j<size

{

Print arr[i]

}

Call quicksort function to sort elements on right and left side of pivot

quicksort(a,0,j-1,n)

quicksort(a,j+1,last,n)

end

**C program:**

#include<stdio.h>

void quicksort(int a[],int first,int last,int n)

{

int i,j,pivot,temp;

if(first<last)

{

pivot=first;

i=first;

j=last;

while(i<j)

{

while(a[i]<=a[pivot]&&i<last)

i++;

while(a[j]>a[pivot])

j--;

if(i<j)

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

temp=a[pivot];

a[pivot]=a[j];

a[j]=temp;

printarray(a,n);

printf("\n");

quicksort(a,0,j-1,n);

quicksort(a,j+1,last,n);

}

}

int main()

{

int n,i;

printf("enter size of array:");

scanf("%d",&n);

int a[n];

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

{

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

}

printf("before sorted:\n");

printarray(a,n);

quicksort(a,0,n-1,n);

printf("after sorted:\n");

printarray(a,n);

return 0;

}

void printarray(int a[],int last)

{

static int i=0;

printf("array status after pass %d:",i);

i++;

int j;

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

{

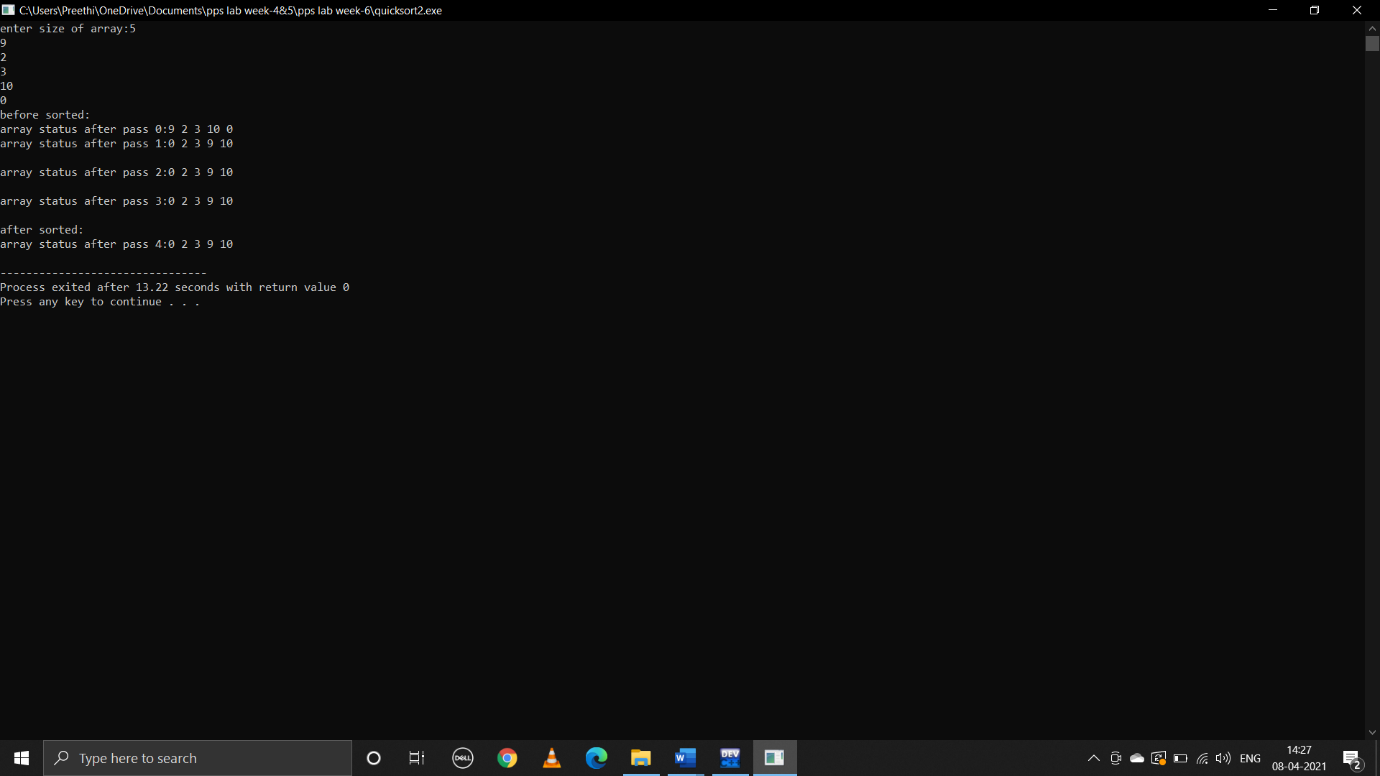
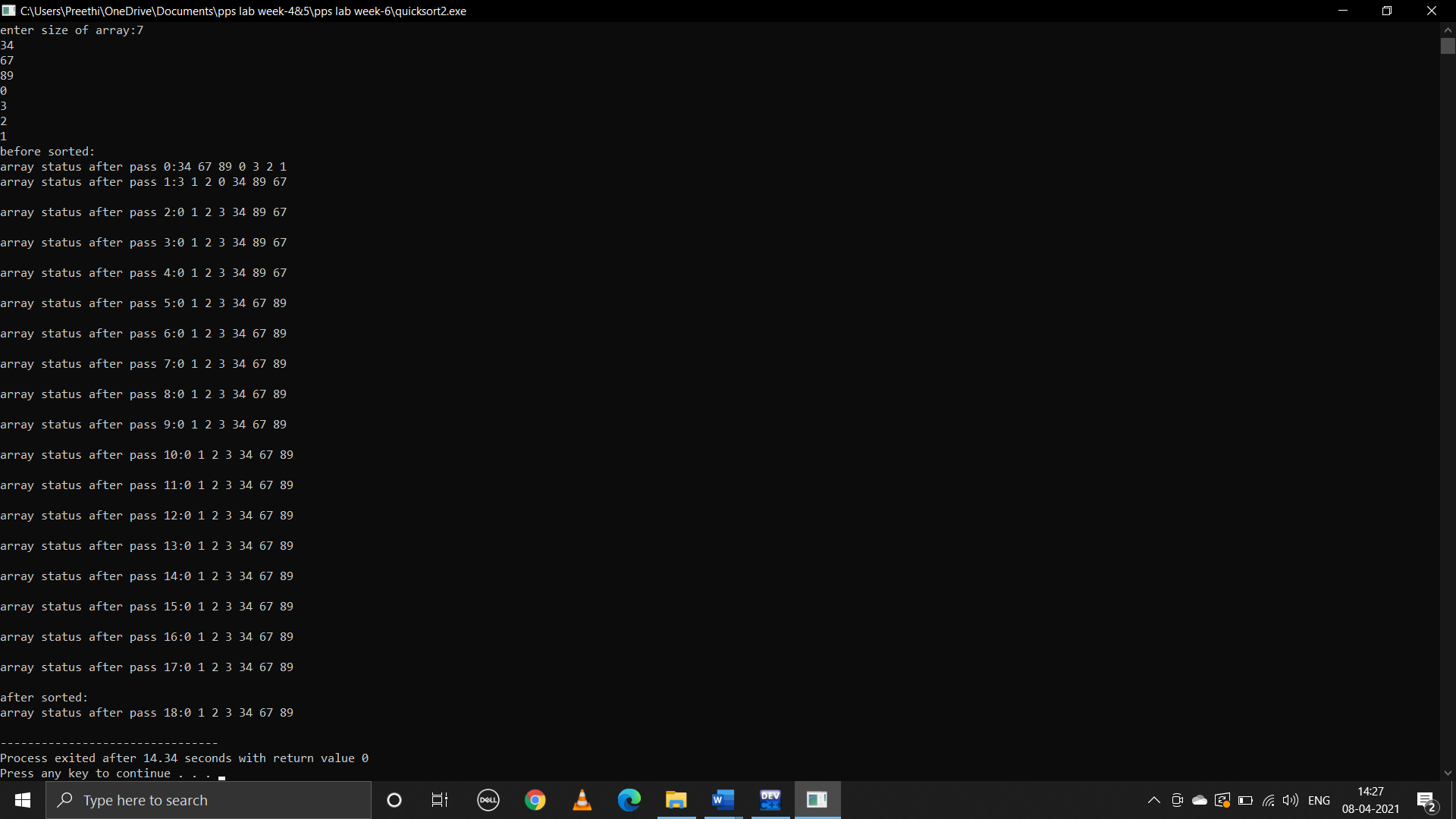
printf("%d ",a[j]);

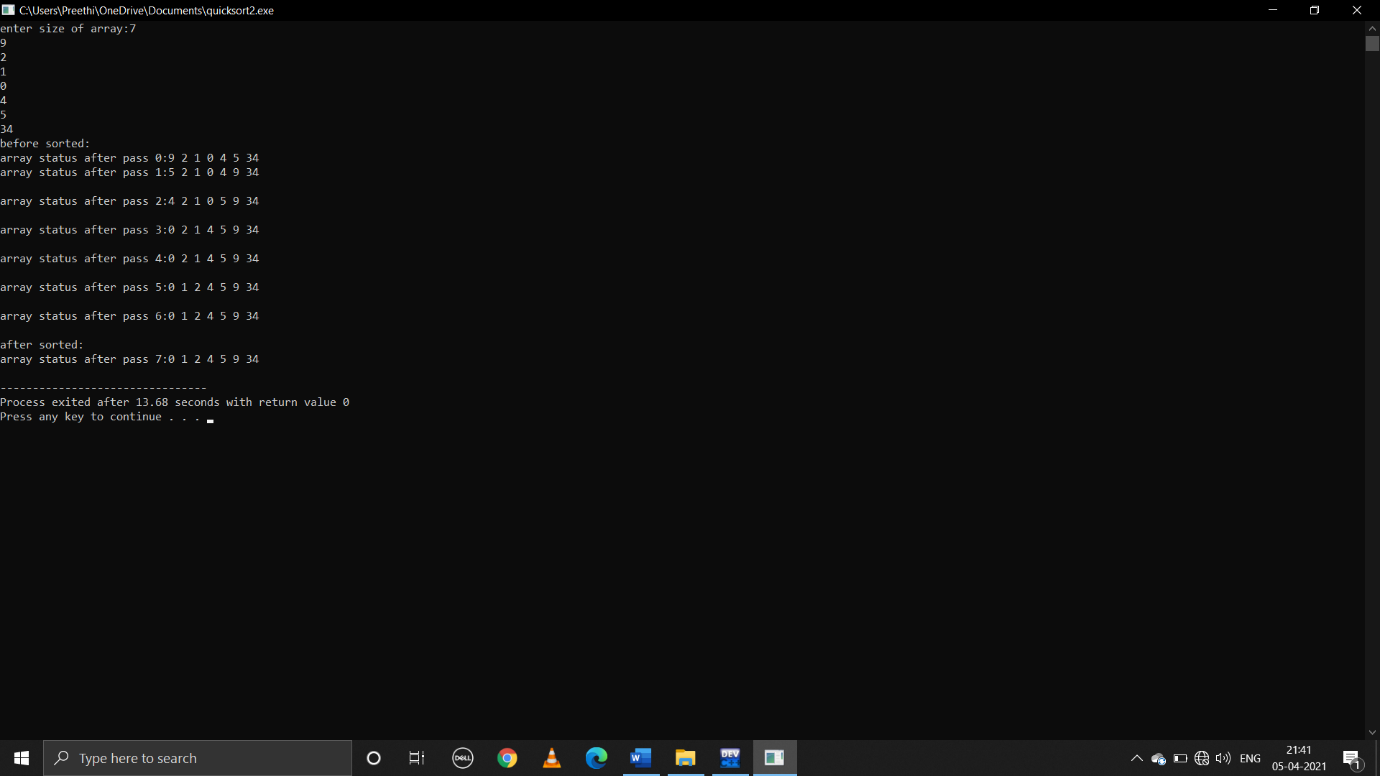
}

printf("\n");

}

**Output:**

****

****

**WEEK-9**

**1.write the following recursive c function**

**i)Factorial of a given number**

**pseudocode:**

begin

declare int x

take user input value of x

call the function fact(x) to find the factorial of x

now control jumps to function defnition

if n>=1

return n\*fact(n-1)

else

return 1

end

**c program:**

#include<stdio.h>

int fact(int n);

int main()

{

int x;

printf("enter number:");

scanf("%d",&x);

fact(x);

printf("factorial of number is:%d",fact(x));

return 0;

}

int fact(int n)

{

if(n>=1)

{

return n\*fact(n-1);

}

else

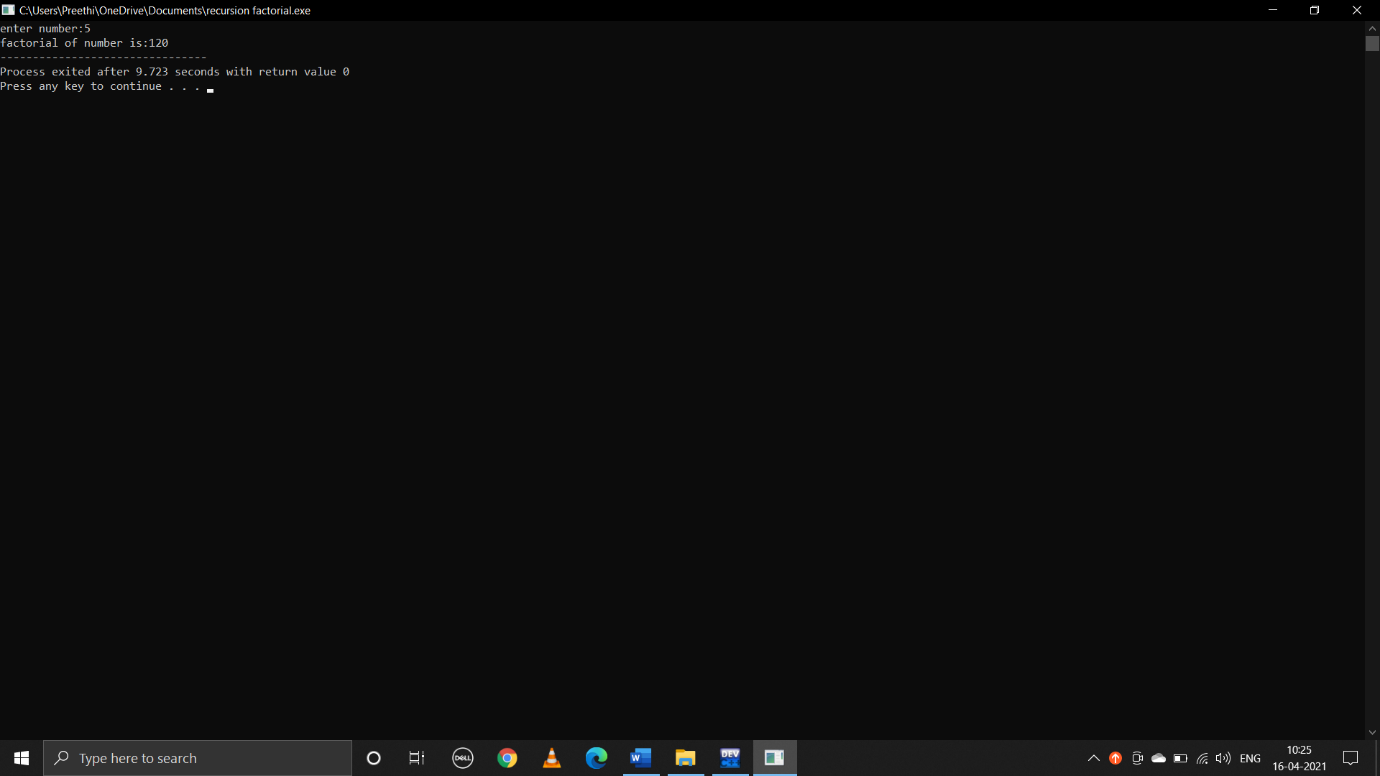
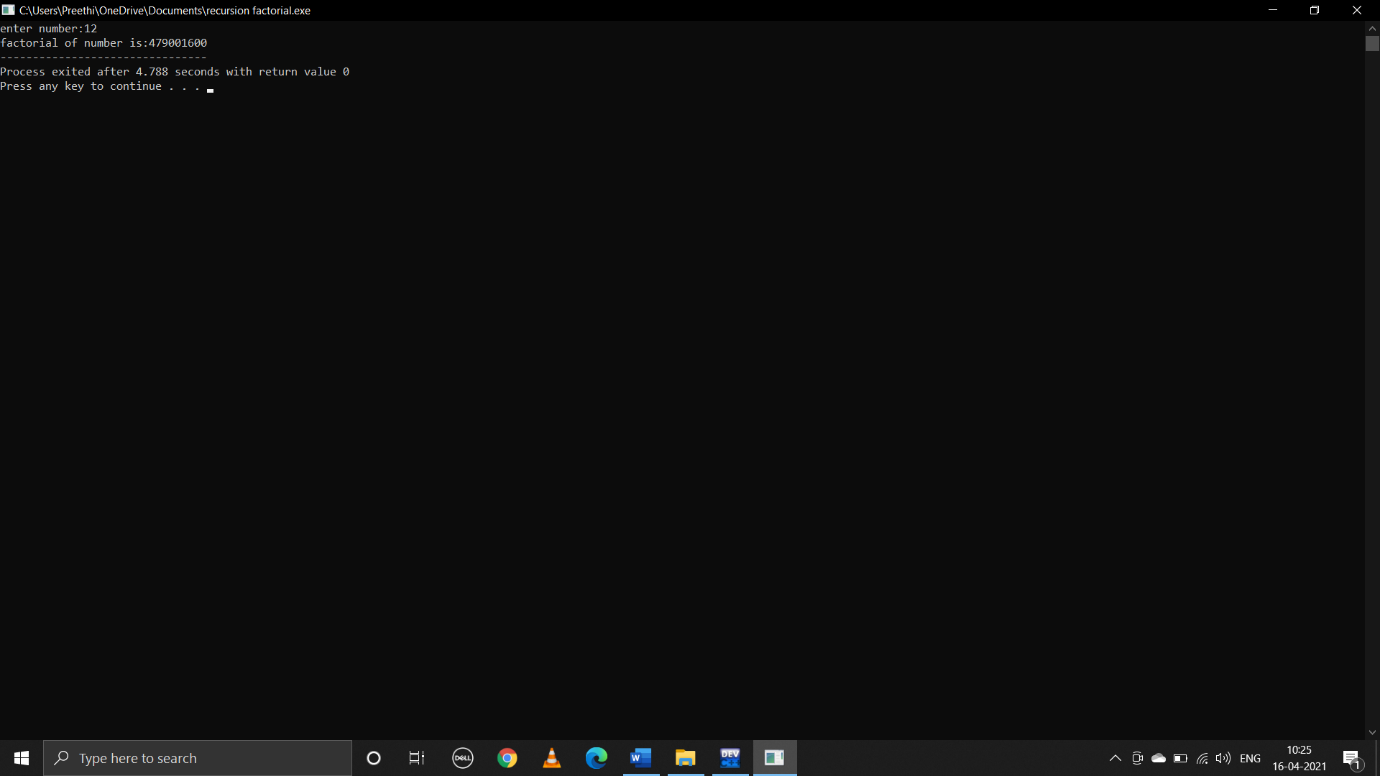
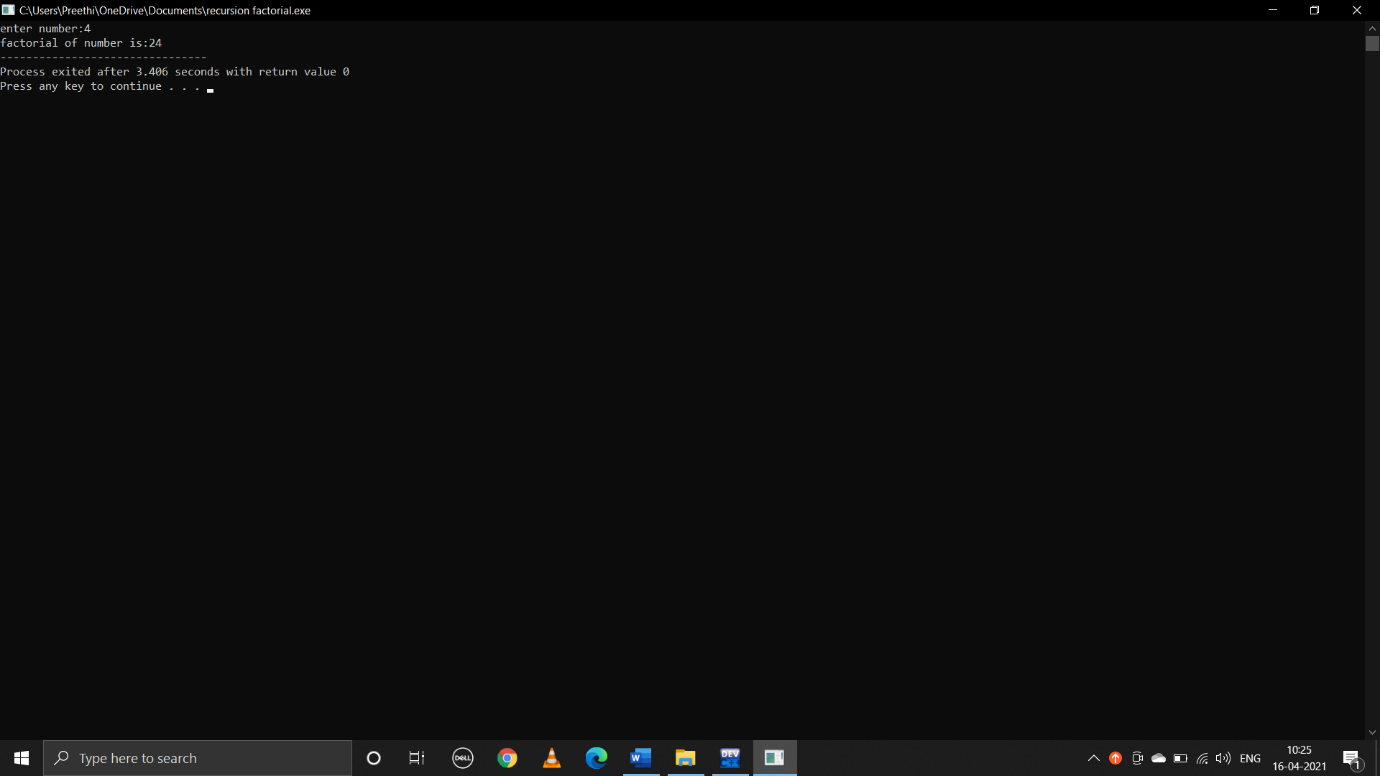
{

return 1;

}

}

**output:**

****

**ii)Nth fibonacci number**

**pseudocode:**

begin

declare int x,i

take user input x to print number of terms

run loop from i=0 to i<x by increment i by 1

print(fib(i))

control jumps to function defnition

if n==0

return 0

if n==1

return 1

else

return(fib(n-1)+fib(n-2))

end

**c program:**

#include<stdio.h>

int fib(int n);

main()

{

int x,i;

printf("enter no.of terms:");

scanf("%d",&x);

printf("fibonacci series is:\n");

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

{

printf("%d\n",fib(i));

}

return 0;

}

int fib(int n)

{

if(n==0)

{

return 0;

}

else if(n==1)

{

return 1;

}

else

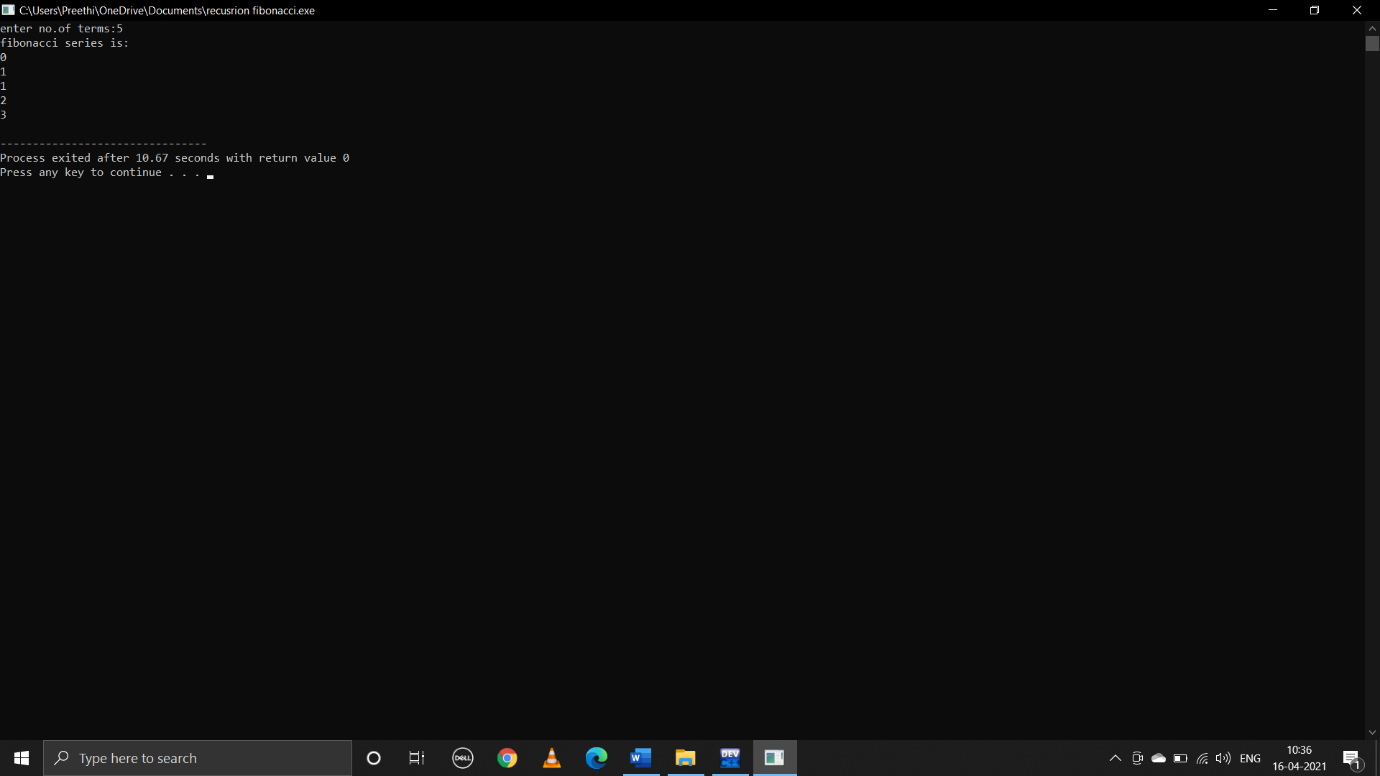
{

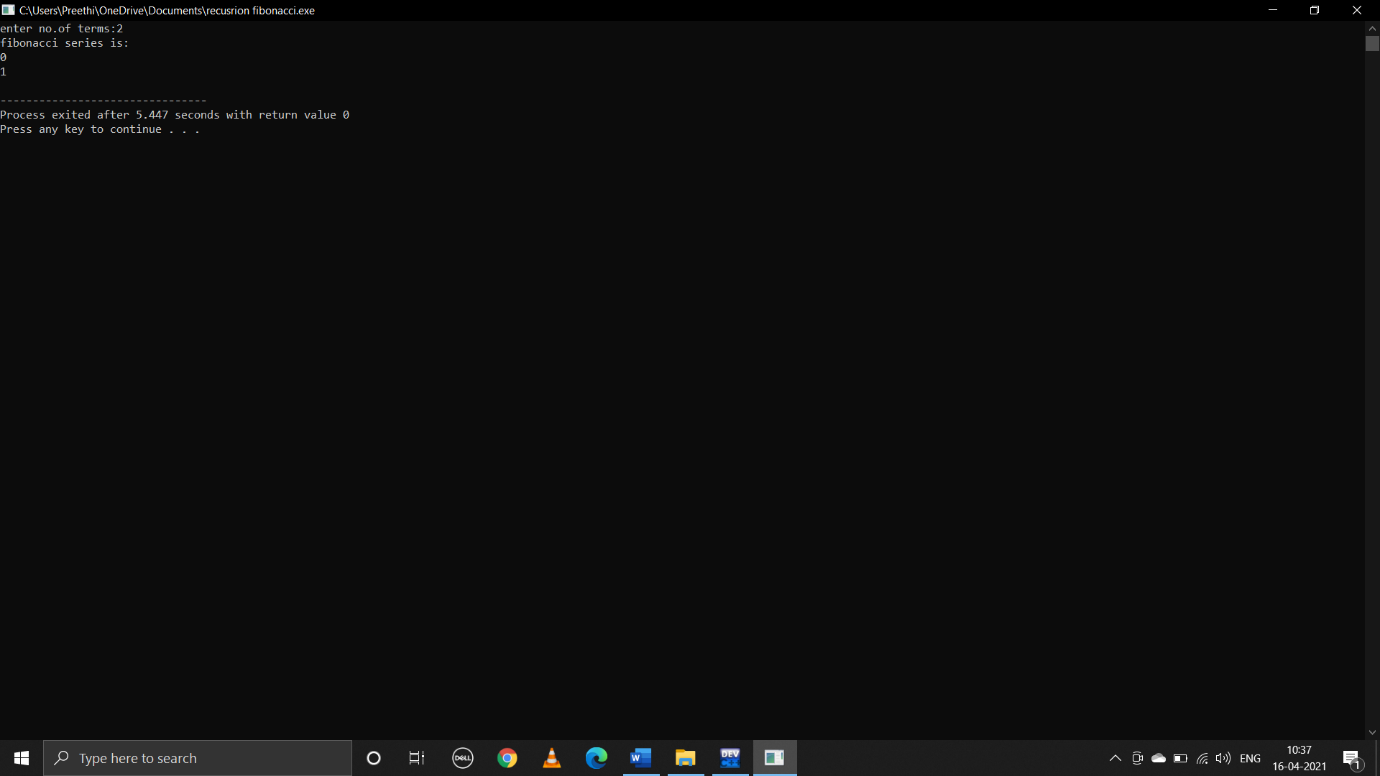
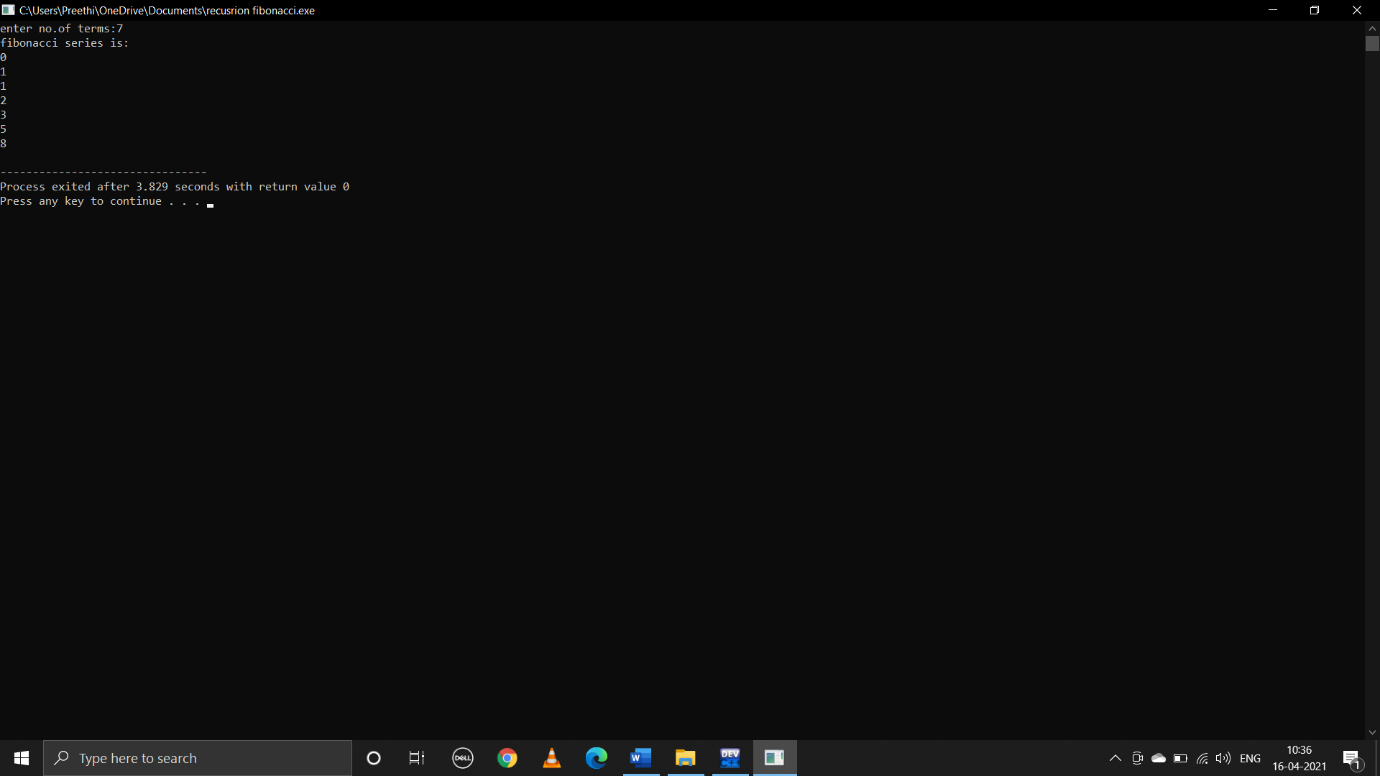
return(fib(n-1)+fib(n-2));

}

}

**output:**

****

****

**iii)Reverse of a given string**

**pseudocode:**

begin

take user input string s to reverse s call the recursive function reverse(s)

now control jumps to function defnition reverse(char\*str)

if(\*str!=’\0’)

reverse(str+1)

print \*str

end

**c program:**

#include<stdio.h>

void reverse(char \*str);

main()

{

char s[100];

printf("enter string:");

gets(s);

puts(s);

reverse(s);

return 0;

}

void reverse(char\*str)

{

if((\*str)!='\0')

{

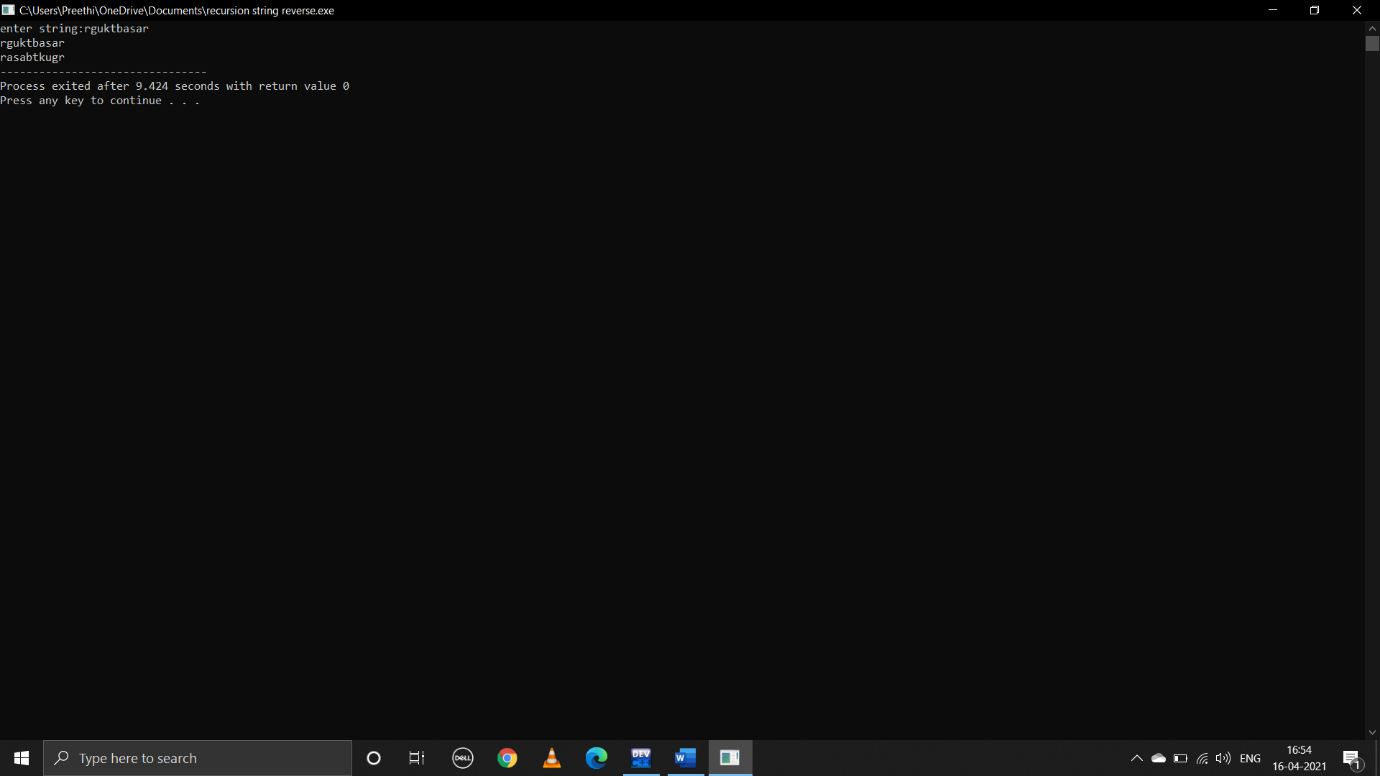
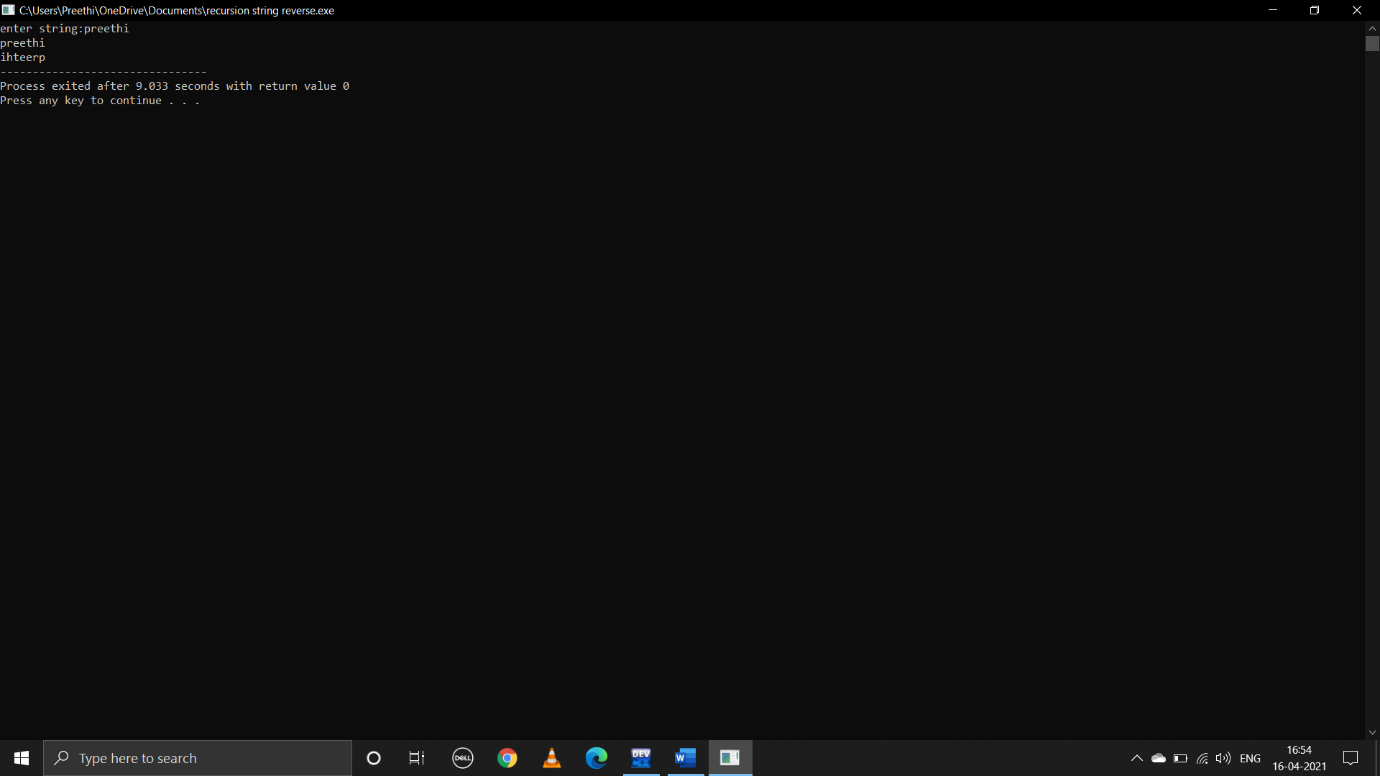
reverse(str+1);

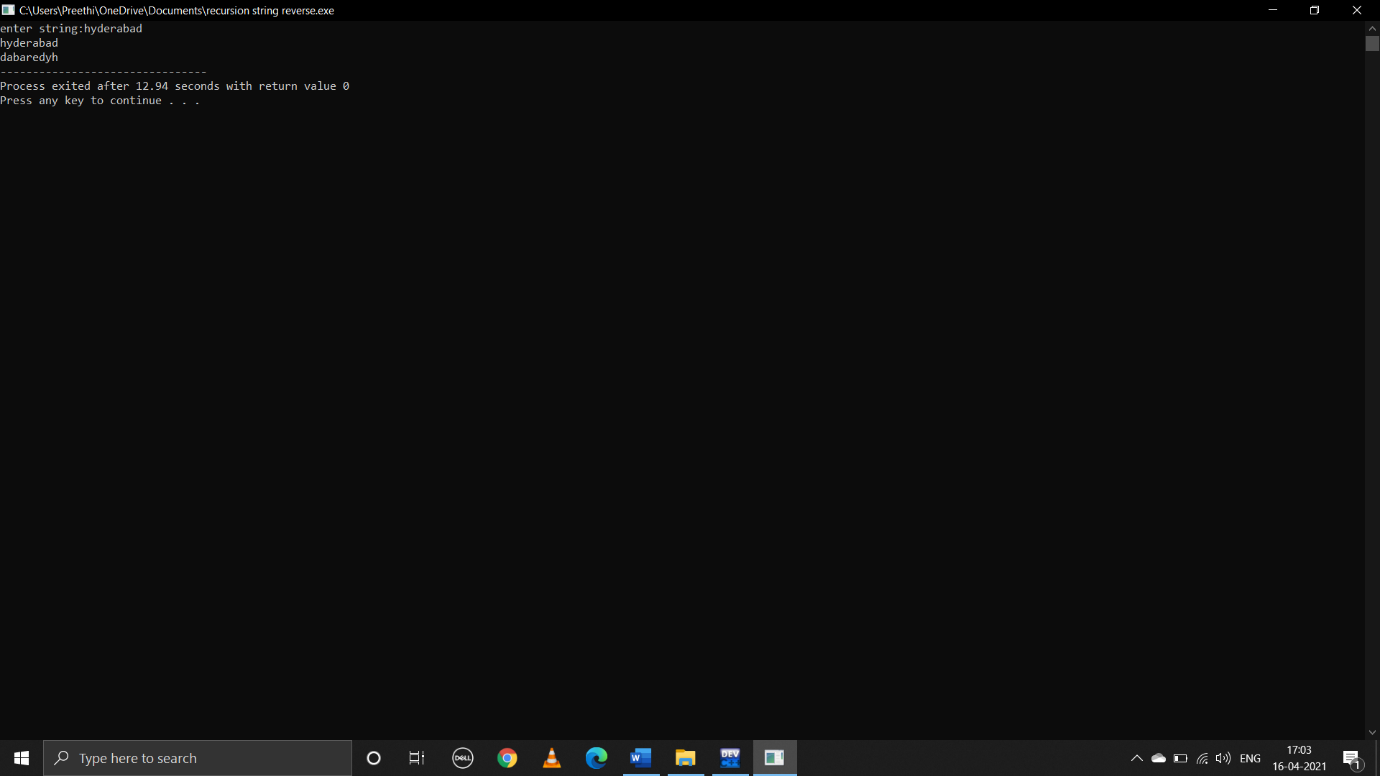
printf("%c",\*str);

}

}

**output:**

****

****

**iv)Reverse of a given number**

**pseudocode:**

begin

declare int x

take user input value of x and call the recursive function rev(x) to reverse the x

control jumps to function defnition

rem=a%10

print rem

recursive call rev(a/10)

if a==0

return

end

**c program:**

#include<stdio.h>

main()

{

int x;

printf("enter x value:");

scanf("%d",&x);

rev(x);

return 0;

}

void rev(int a)

{

int rem;

if(a==0)

{

return;

}

else

{

rem=a%10;

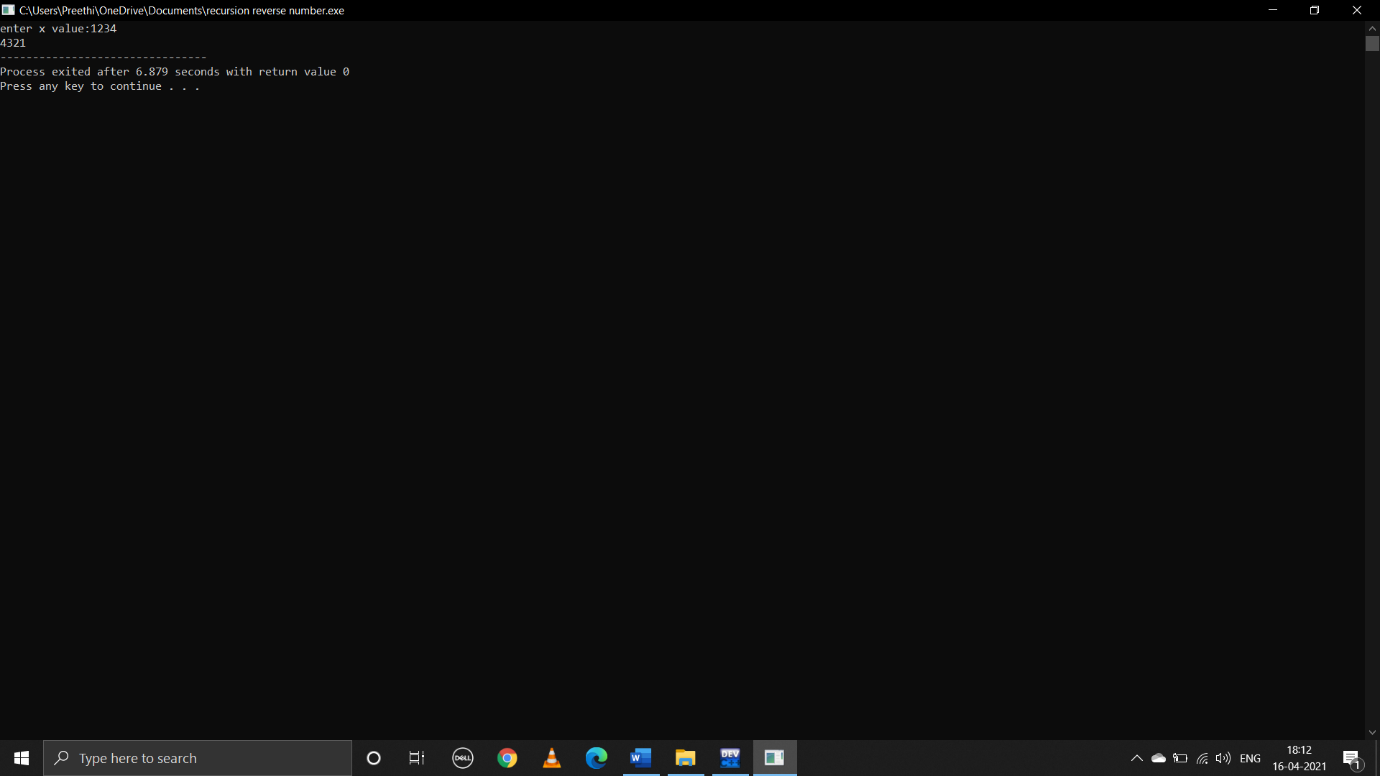
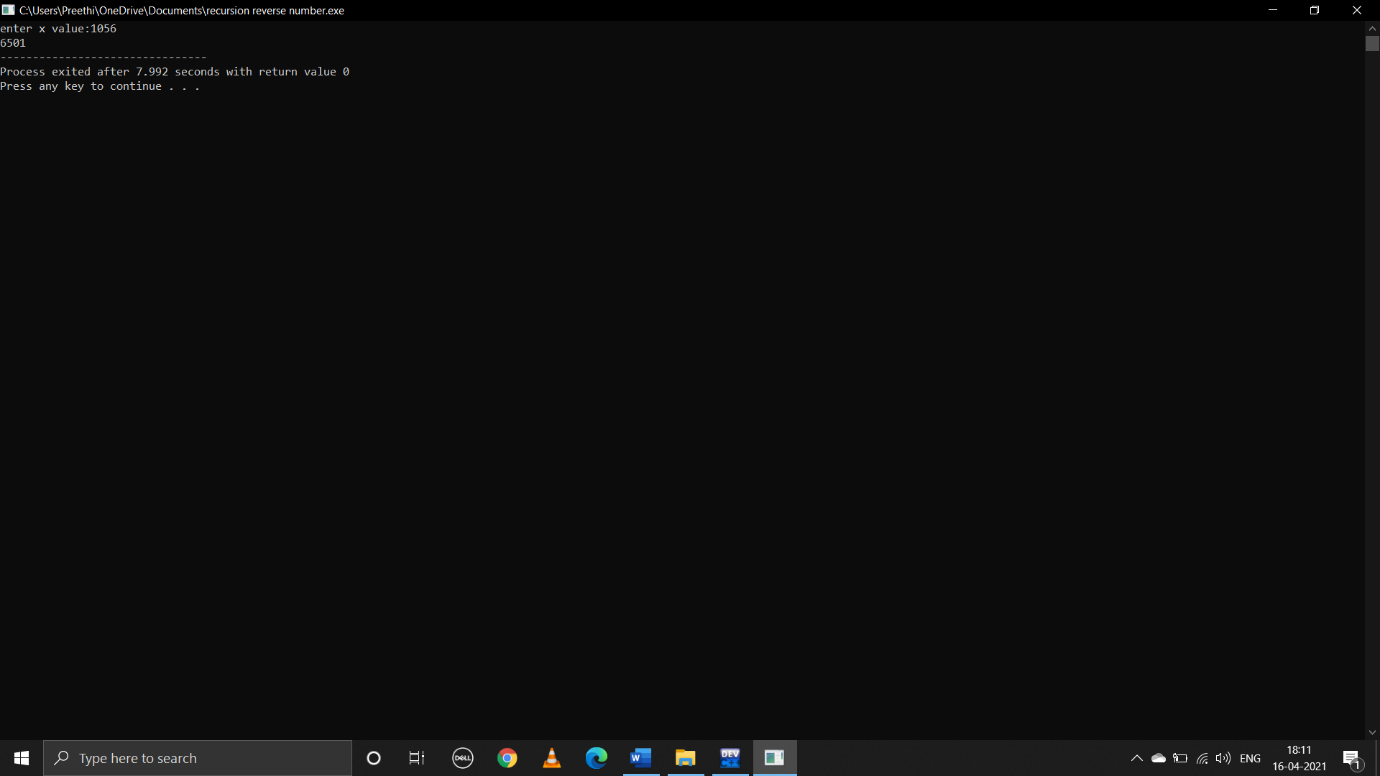
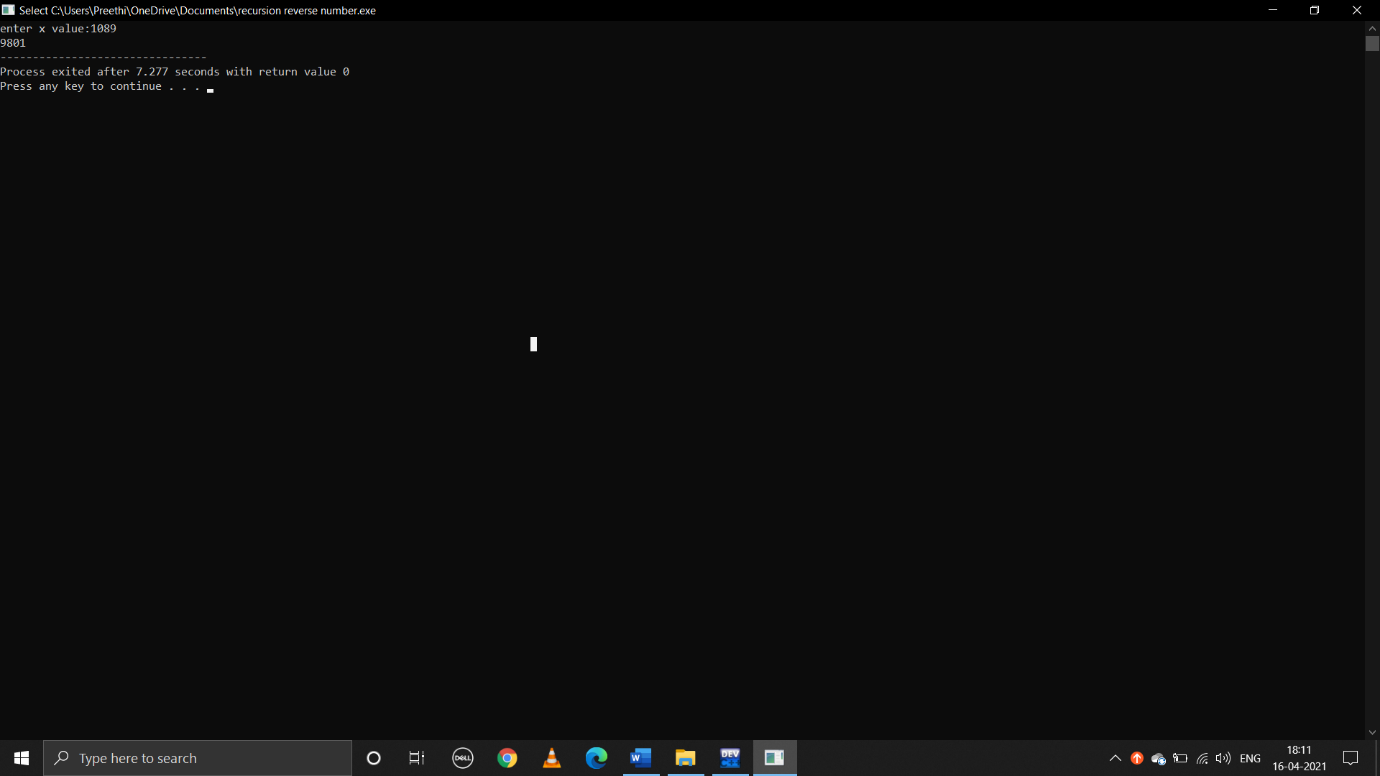
printf("%d",rem);

rev(a/10);

}

}

**output:**

****

Done by:

B.preethi

ID:B181356

Class:E1-CSE,AB-2,305