Day 1:

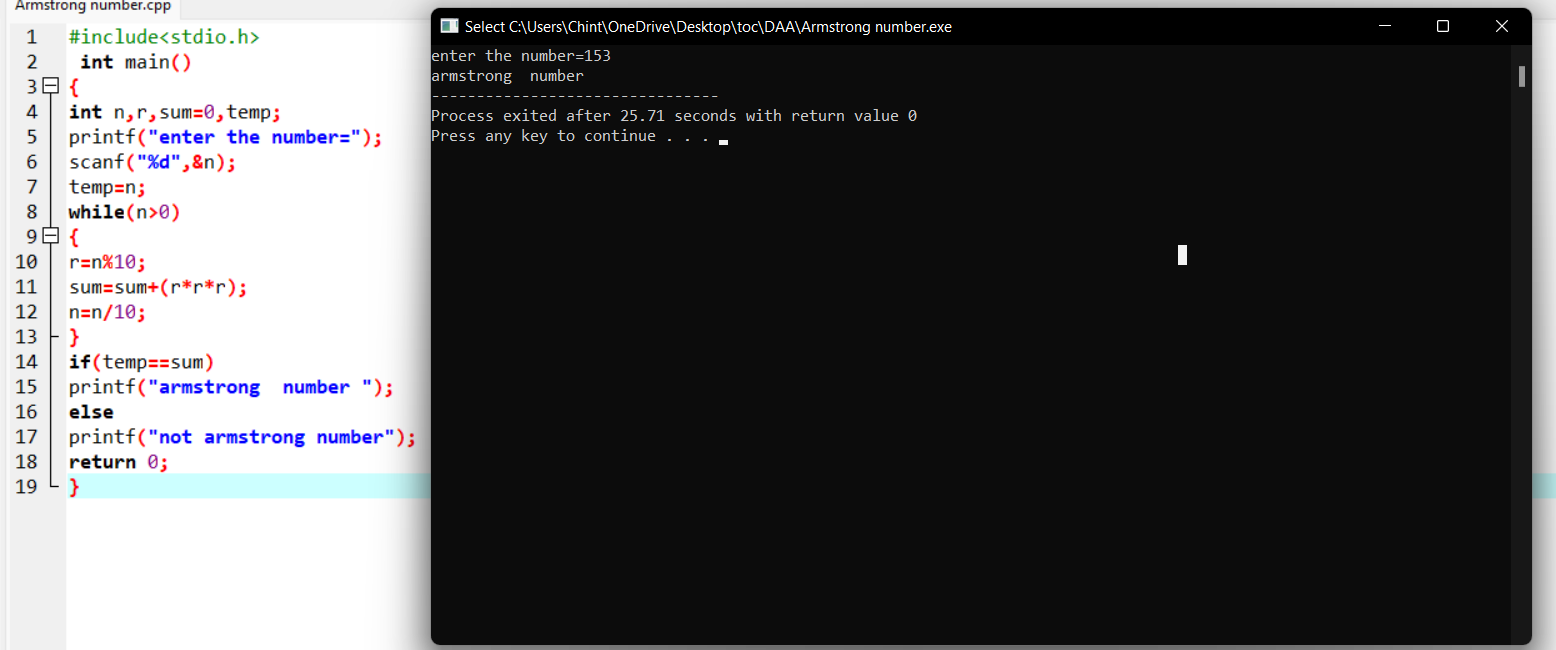
DAA

# 1.Armstrong number:

## Program:

1. #include<stdio.h>
2. **int** main()
3. {
4. **int** n,r,sum=0,temp;
5. printf("enter the number=");
6. scanf("%d",&n);
7. temp=n;
8. **while**(n>0)
9. {
10. r=n%10;
11. sum=sum+(r\*r\*r);
12. n=n/10;
13. }
14. **if**(temp==sum)
15. printf("armstrong  number ");
16. **else**
17. printf("not armstrong number");
18. **return** 0;
19. }

## Output:



# 2.Time complexity:

# i)

## program:

#include <stdio.h>

void function(int min);

int main()

{

int n;

scanf("%d",&n);

function(n);

return 0;

}

void function(int n)

{

int count=0;

int i=1,s=1;

count++;

count++;

while(s<=n)

{

count++;

i++;

count++;

s+=i;

count++;

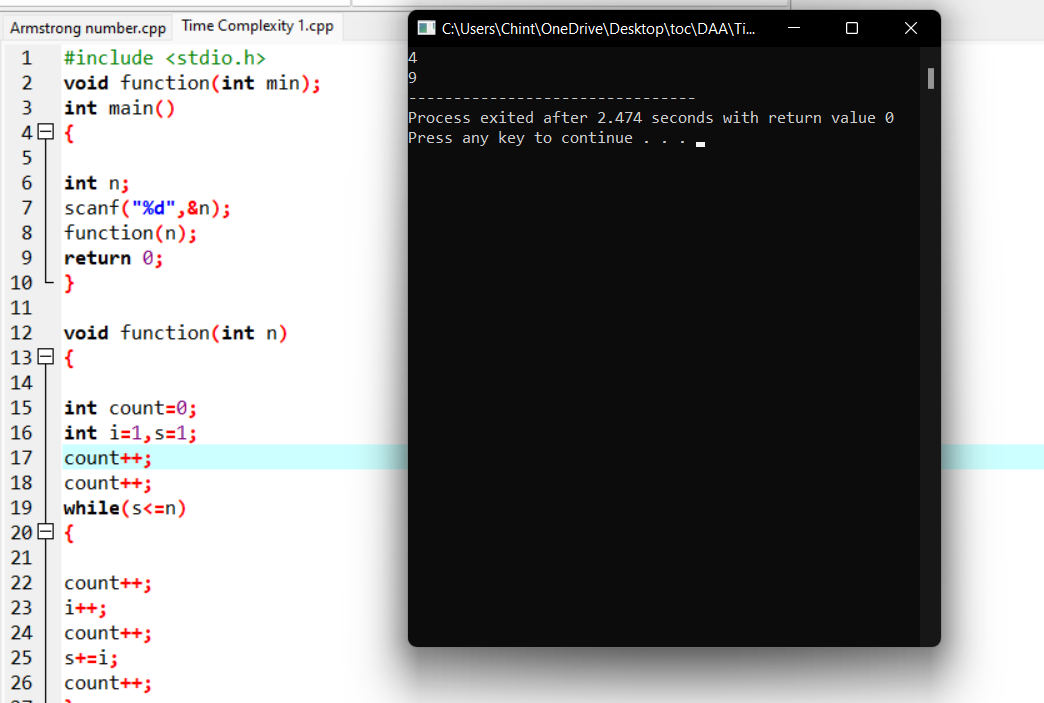
}

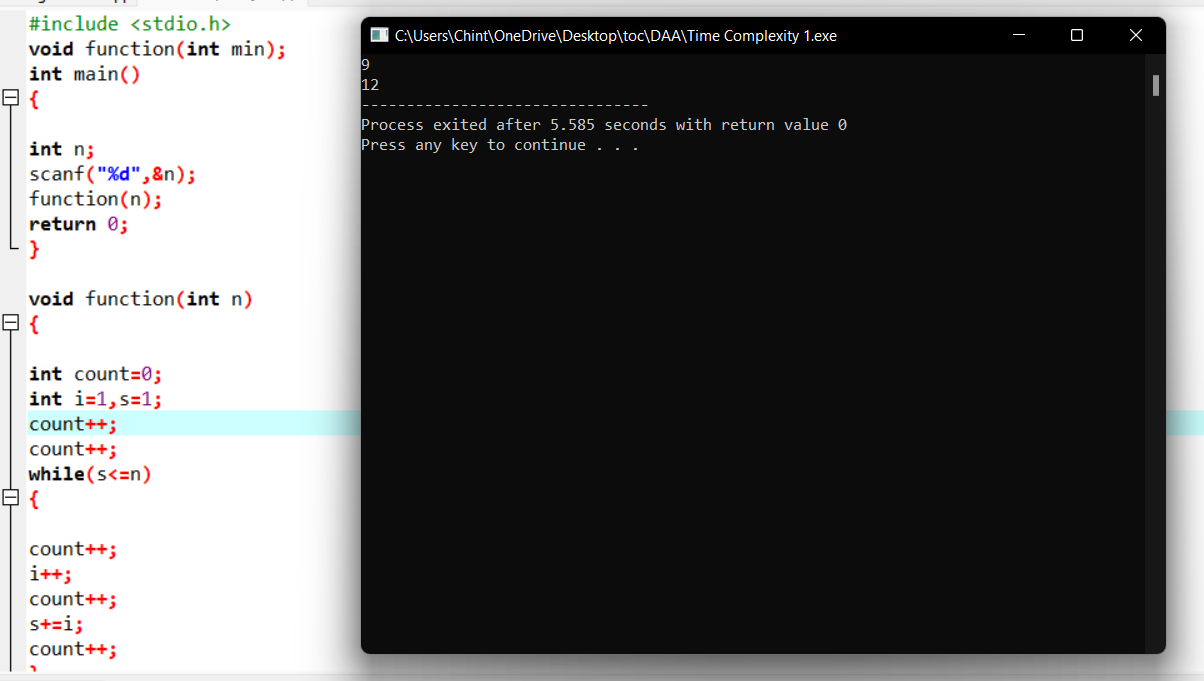
count++;

printf("%d",count);

}

## Output:





# ii)

## program:

#include <stdio.h>

void function(int n);

int main()

{

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

function(n);

return 0;

}

void function(int n)

{

int count=0;

if(n==1)

{

count++; count++;

}

else

{

count++;

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

{

count++;

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

{

count++;

count++;

count++;

count++;

break;

}

}

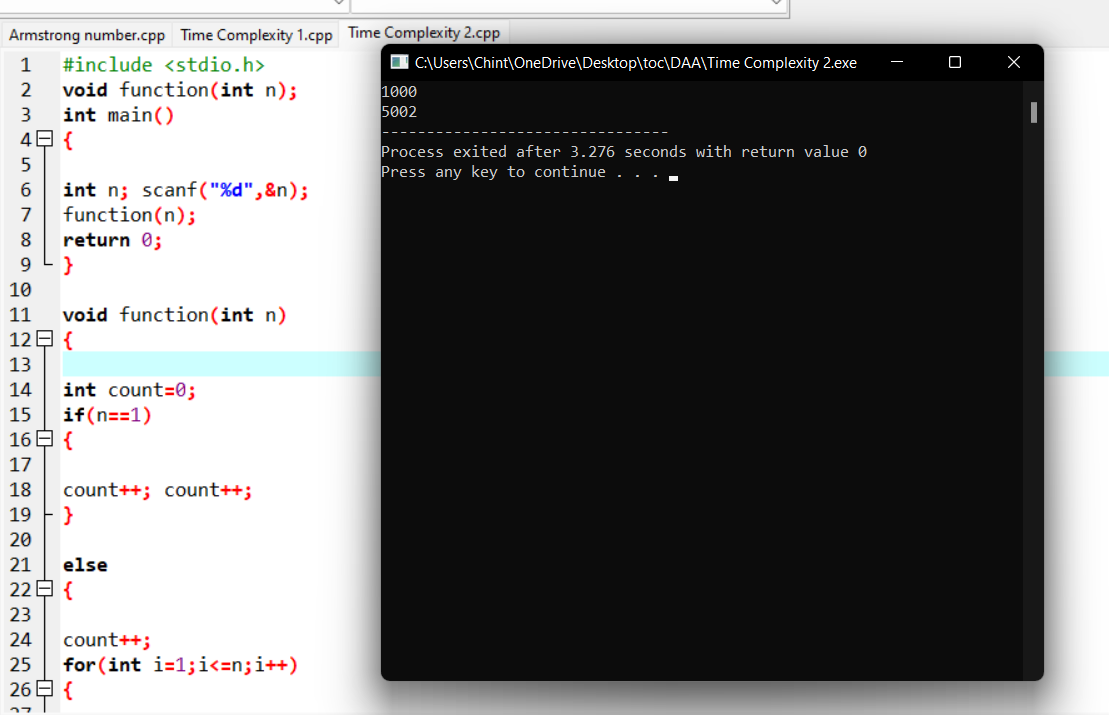
count++;

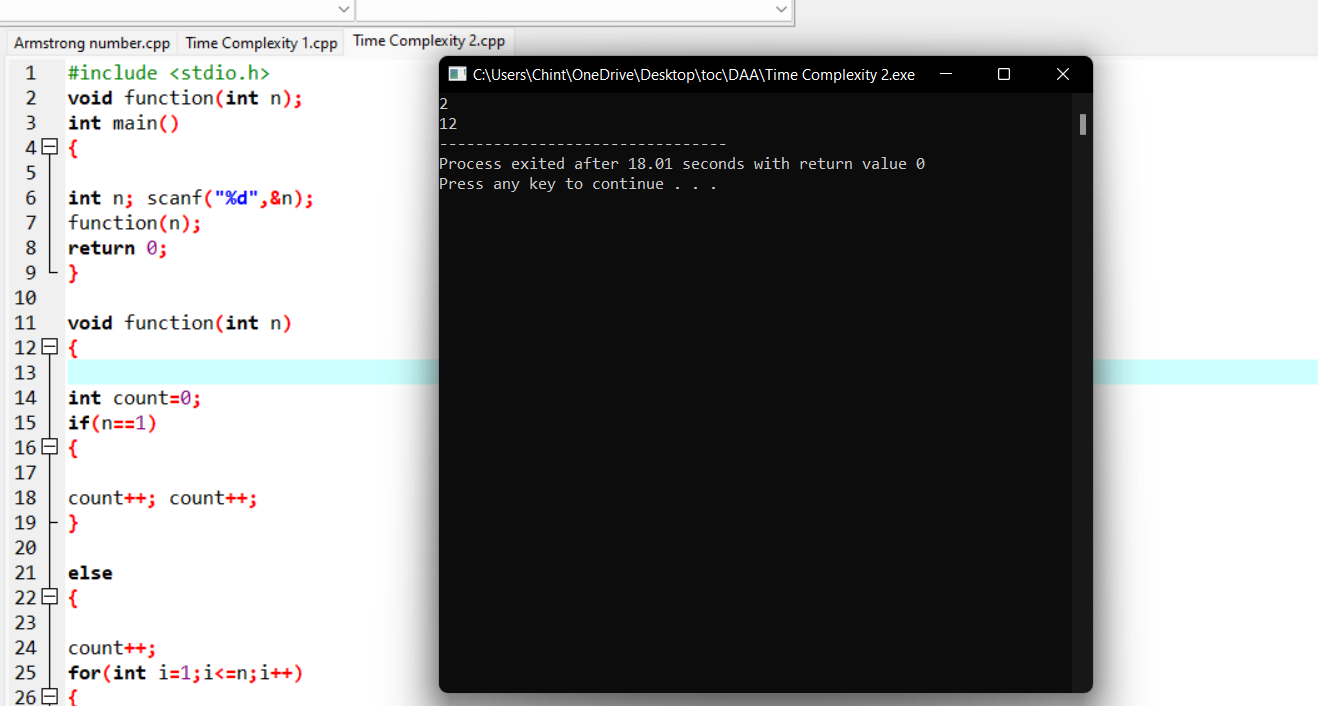
}

printf("%d",count);

}

## Output:





# 

# iii)

## program:

#include <stdio.h>

int factor(int n);

int count=0;

int main()

{

int n;

scanf("%d",&n);

factor(n);

printf("%d",count);

return 0;

}

int factor(int n)

{

int i; count++;

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

{

count++;

if(n%i==0)

{

//print

}count++;

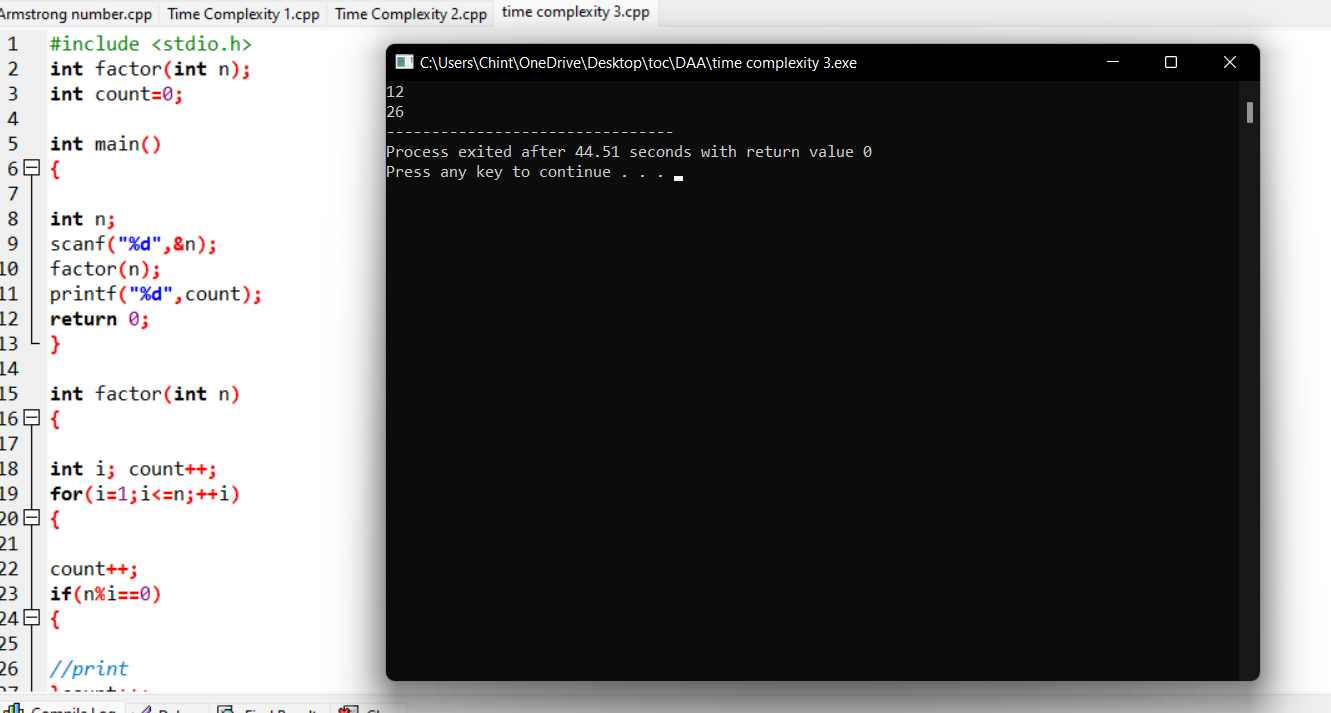
}

count++;

return 0;

}

## Output:



# iv)

## program:

#include <stdio.h>

void function(int n);

int main()

{

int n;

scanf("%d",&n);

function(n);

return 0;

}

void function(int n)

{

int count=0;

int c=0;

count++;

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

{

count++;

for(int j=1;j<n;j=2\*j)

{

count++;

for(int k=1;k<n;k=k\*2)

{

count++;

c++;

count++;

}count++;

}count++;

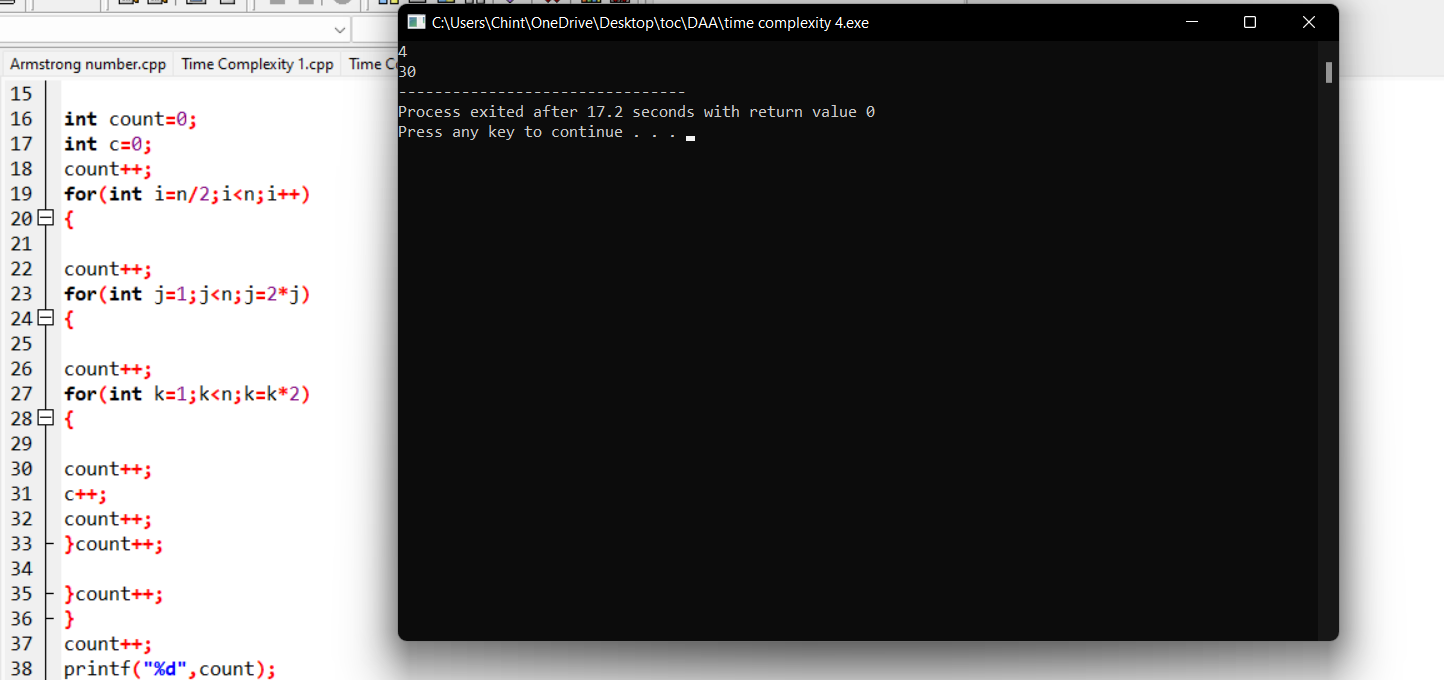
}

count++;

printf("%d",count);

}

## Output:



## v)

## program:

#include <stdio.h>

void reverse(int n);

int main()

{

int n;

scanf("%d",&n);

reverse(n);

return 0;

}

void reverse(int n)

{

int count=0;

int rev=0,

remainder;

count++;

while(n!=0)

{

count++;

remainder=n%10;

count++;

rev=rev\*10+remainder;

count++;

n=n/10;

count++;

}

count++;

count++;

printf("%d",count);

}

## Output:

# 

# 3.Binary search:

## Program:

#include<stdio.h>

int main()

{

int c=0;

int n,k,i,low,high,mid,a[50],temp;

printf("Enter number of elements:");

scanf("%d",&n);

printf("Enter elements:\n");

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

{

c++;

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

}

c++;

printf("Enter Element to search:");

scanf("%d",&k);

low=0; c++;

high=n-1; c++;

mid=low+high/2; c++;

c++;

while(low<=high)

{

c++;

c++;

if(a[mid]<k)

{

low=mid+1; c++;

}

else if(a[mid]==k)

{

printf("\nElement is found at index %d\n",mid);

break;

}

else

{

high=mid-1; c++;

}

mid=(low+high)/2; c++;

}

c++;

c++;

if(low>high)

{

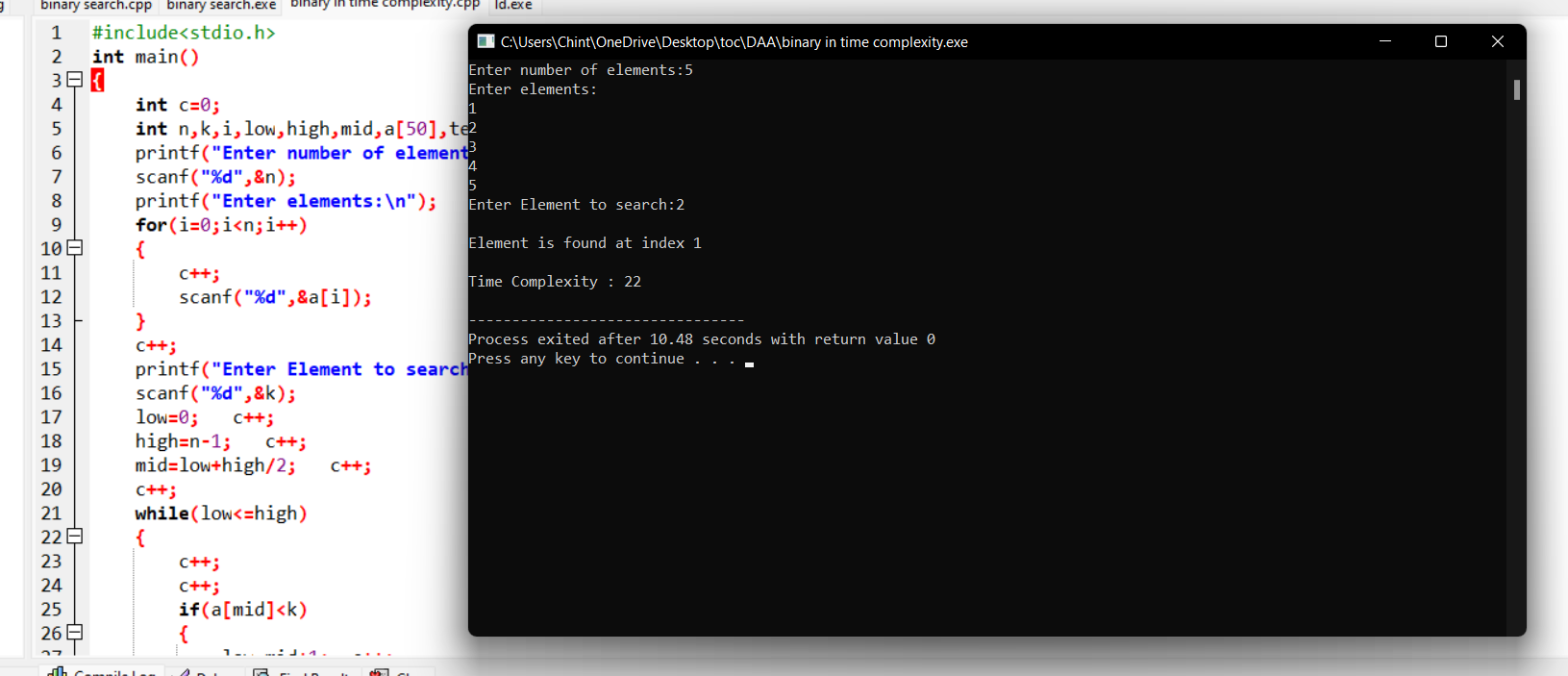
printf("Element is not found\n");

}

printf("\nTime Complexity : %d\n",c);

}

# Output:



# 4.Linear search:

## Program:

#include<stdio.h>

int main()

{

int c=0;

int n,k,i,j,f=0,a[50];

c++;

printf("Enter number of elements:");

scanf("%d",&n);

printf("Enter elements:\n");

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

{

c++;

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

}

c++;

printf("Enter Element to search:");

scanf("%d",&k);

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

{

c++;

c++;

if(k==a[i])

{

printf("Element is found at index %d\n",i);

f=1;

c++;

}

}

c++;

c++;

if(f==0)

{

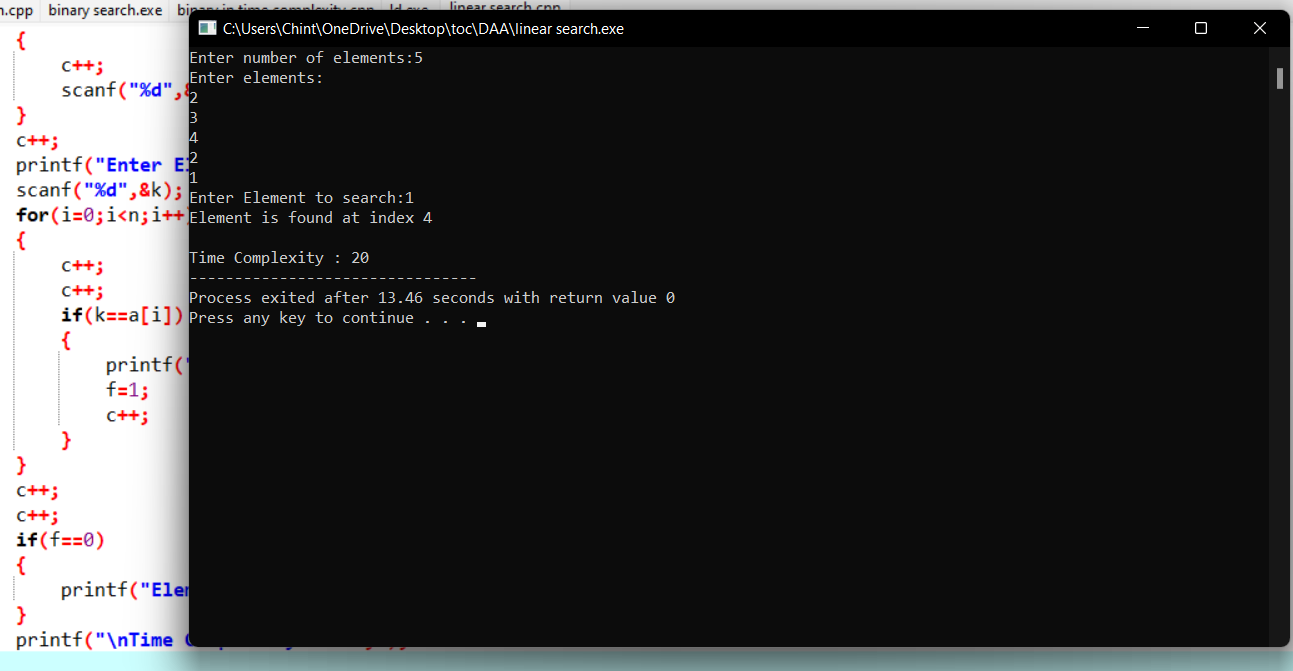
printf("Element is not found");

}

printf("\nTime Complexity : %d",c);

}

## Output:



# 5.Reverse a number:

## Program:

#include <stdio.h>

int main() {

int n, reverse = 0, remainder;

printf("Enter an integer: ");

scanf("%d", &n);

while (n != 0) {

remainder = n % 10;

reverse = reverse \* 10 + remainder;

n /= 10;

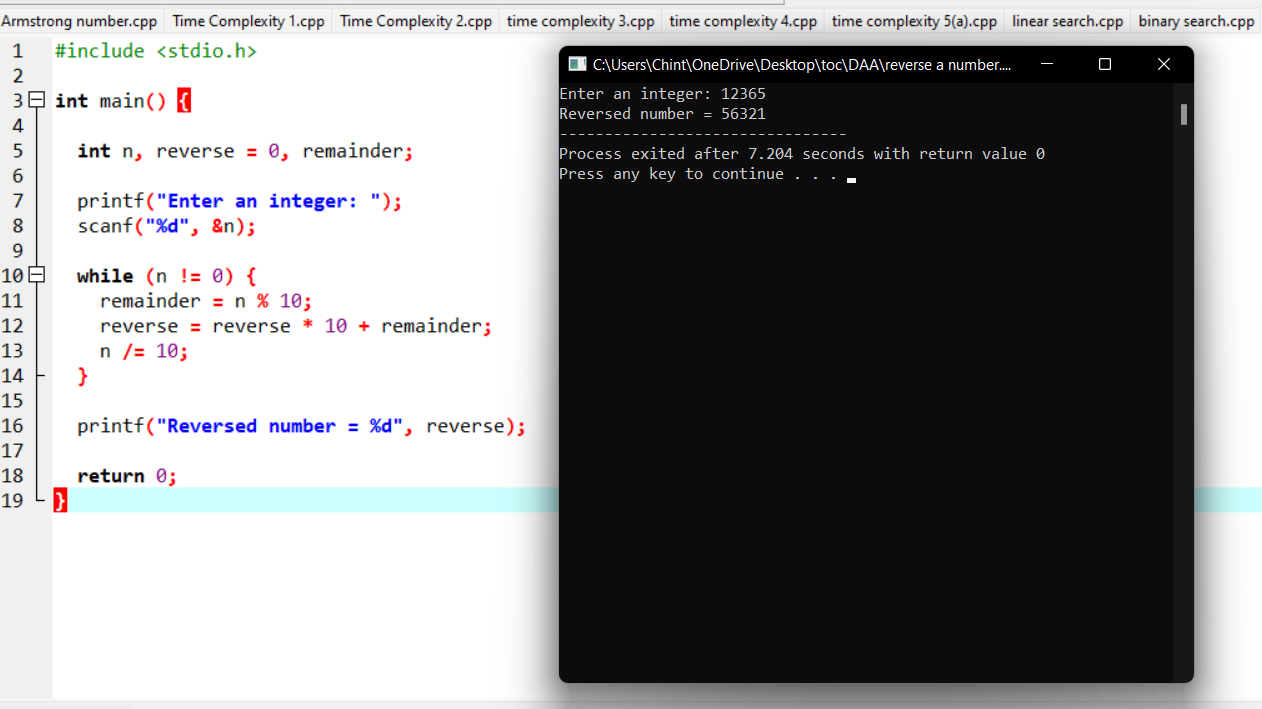
}

printf("Reversed number = %d", reverse);

return 0;

}

# Output:



# 6.Matrix multiplication:

# Program:

#include<stdio.h>

int main(){

int a[2][2], b[2][2], c[2][2], i, j,count=0;

int m1, m2, m3, m4 , m5, m6, m7;

printf("Enter the 4 elements of first matrix:");

count++;

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

{

count++;

for(j = 0;j < 2; j++)

{

count++;

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

}

}

count++;

count++;

printf("Enter the 4 elements of second matrix: ");

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

{

count++;

for(j = 0;j < 2; j++)

{

count++;

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

}

}

count++;

count++;

printf("\nThe first matrix is\n");

for(i = 0; i < 2; i++){

count++;

printf("\n");

for(j = 0; j < 2; j++){

count++;

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

}

}

count++;

count++;

printf("\nThe second matrix is\n");

for(i = 0;i < 2; i++){

count++;

printf("\n");

for(j = 0;j < 2; j++){

count++;

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

}

}

count++;

count++;

m1= (a[0][0] + a[1][1]) \* (b[0][0] + b[1][1]);

count++;

m2= (a[1][0] + a[1][1]) \* b[0][0];

count++;

m3= a[0][0] \* (b[0][1] - b[1][1]);

count++;

m4= a[1][1] \* (b[1][0] - b[0][0]);

count++;

m5= (a[0][0] + a[0][1]) \* b[1][1];

count++;

m6= (a[1][0] - a[0][0]) \* (b[0][0]+b[0][1]);

count++;

m7= (a[0][1] - a[1][1]) \* (b[1][0]+b[1][1]);

count++;

c[0][0] = m1 + m4- m5 + m7;

count++;

c[0][1] = m3 + m5;

count++;

c[1][0] = m2 + m4;

count++;

c[1][1] = m1 - m2 + m3 + m6;

count++;

printf("\nAfter multiplication using Strassen's algorithm \n");

for(i = 0; i < 2 ; i++){

count++;

printf("\n");

for(j = 0;j < 2; j++){

count++;

printf("%d\t", c[i][j]);

}

}

count++;

count++;

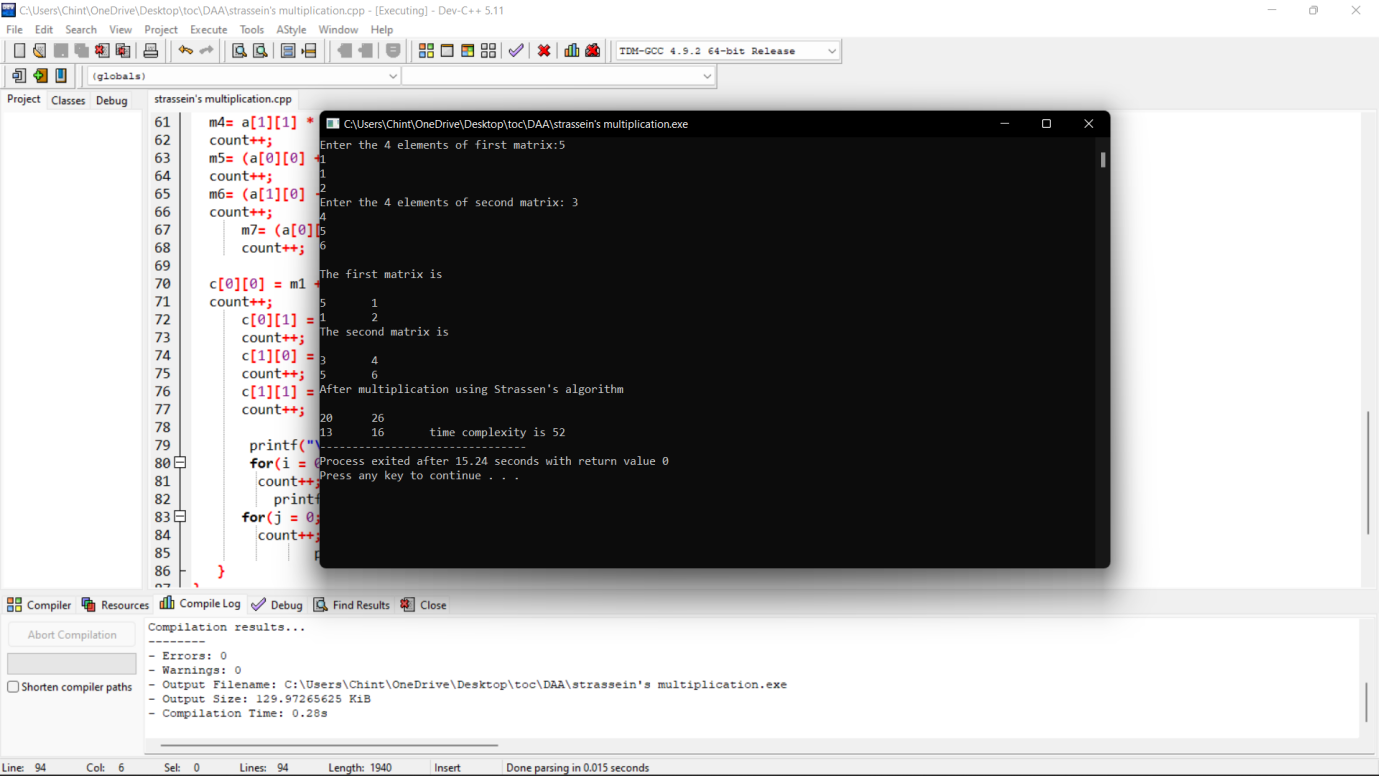
printf(" time complexity is %d",count);

return 0;

}

}

## Output;



# GCD:

# Program:

#include <stdio.h>

int main()

{

int n1, n2, i, GCD\_Num;

int count=0;

printf ( " Enter any two numbers: \n ");

scanf ( "%d %d", &n1, &n2);

for( i = 1; i <= n1 && i <= n2; ++i)

{

count++;

if (n1 % i ==0 && n2 % i == 0)

GCD\_Num = i;

count++;

}

count++;

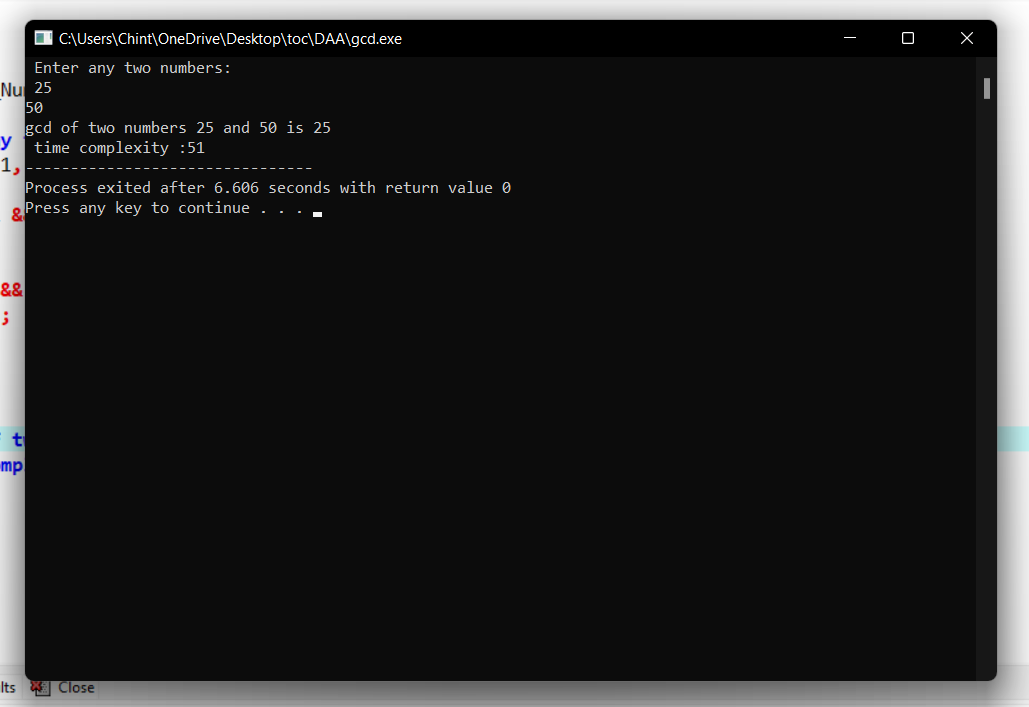
printf ("gcd of two numbers %d and %d is %d \n ", n1, n2, GCD\_Num);

printf("time complexity :%d ",count);

return 0;

}

# Output:



# 9.Pascal triangle:

# Program:

#include<stdio.h>

int main()

{

int rows, coef = 1, space, i, j;

int count=0;

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

scanf("%d", &rows);

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

{

count++;

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

printf(" ");

count++;

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

{

count++;

if(j == 0 || i == 0){

coef = 1;

count++;

}

else

{

coef = coef \* (i - j + 1) / j;

}

count++;

printf("%4d", coef);

}

printf("\n");

count++;

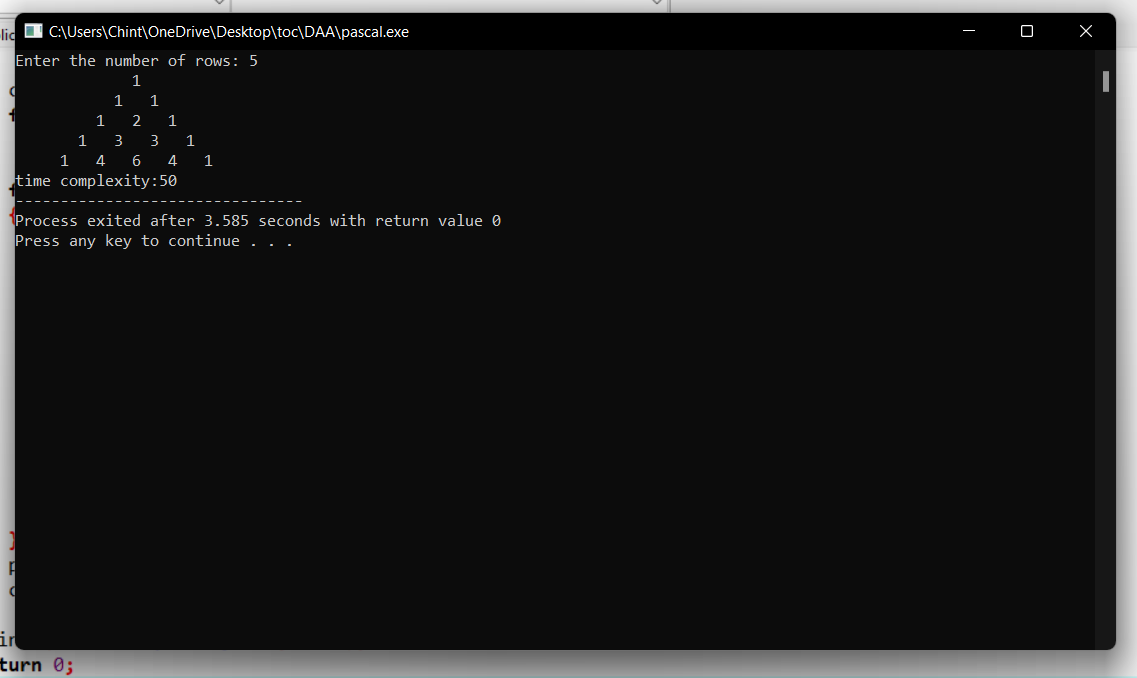
}

printf("time complexity:%d",count);

return 0;

}

# Output:



# 10.Largest number:

# Program:

#include <stdio.h>

int main() {

int n;

int count=0;

double arr[100];

printf("Enter the number of elements (1 to 100): ");

scanf("%d", &n);

count++;

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

count++;

printf("Enter number%d: ", i + 1);

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

}

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

count++;

if (arr[0] < arr[i]) {

arr[0] = arr[i];

}

count++;

}

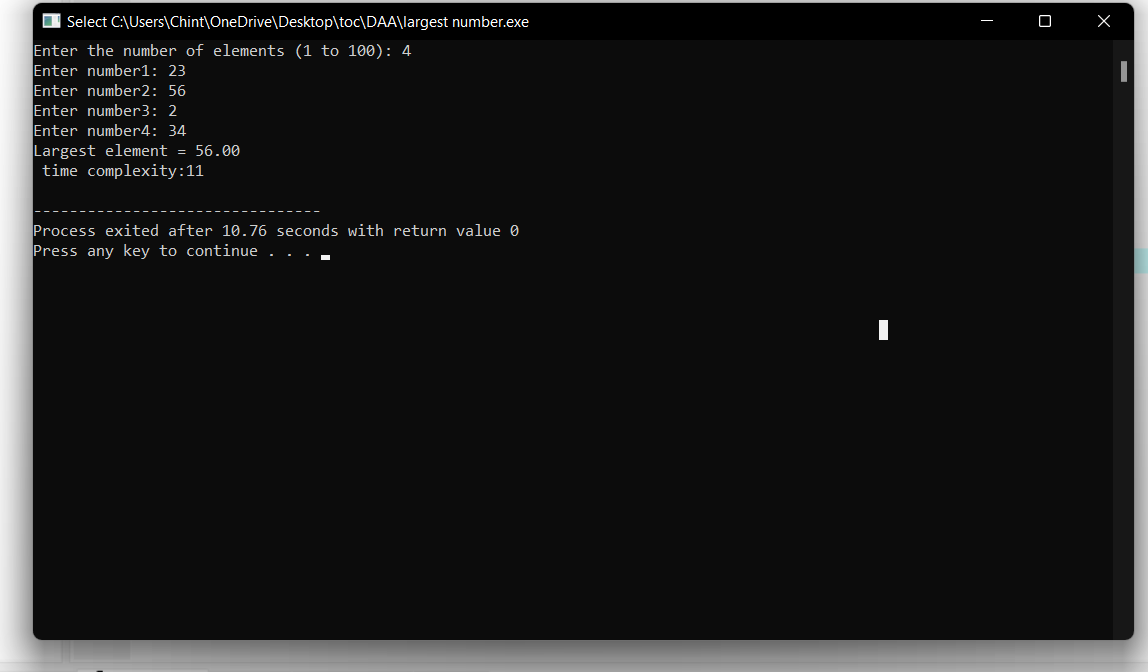
printf("Largest element = %.2lf \n ", arr[0]);

printf("time complexity:%d\n",count);

return 0;

}

# Output:



# Factorial:

# Program:

#include <stdio.h>

int main() {

int n, i;

int count=0;

unsigned long long fact = 1;

printf("Enter an integer: ");

scanf("%d", &n);

count++;

if (n < 0)

printf("Error! Factorial of a negative number doesn't exist.");

else {

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

fact \*= i;

count++;

}

printf("Factorial of %d = %llu \n ", n, fact);

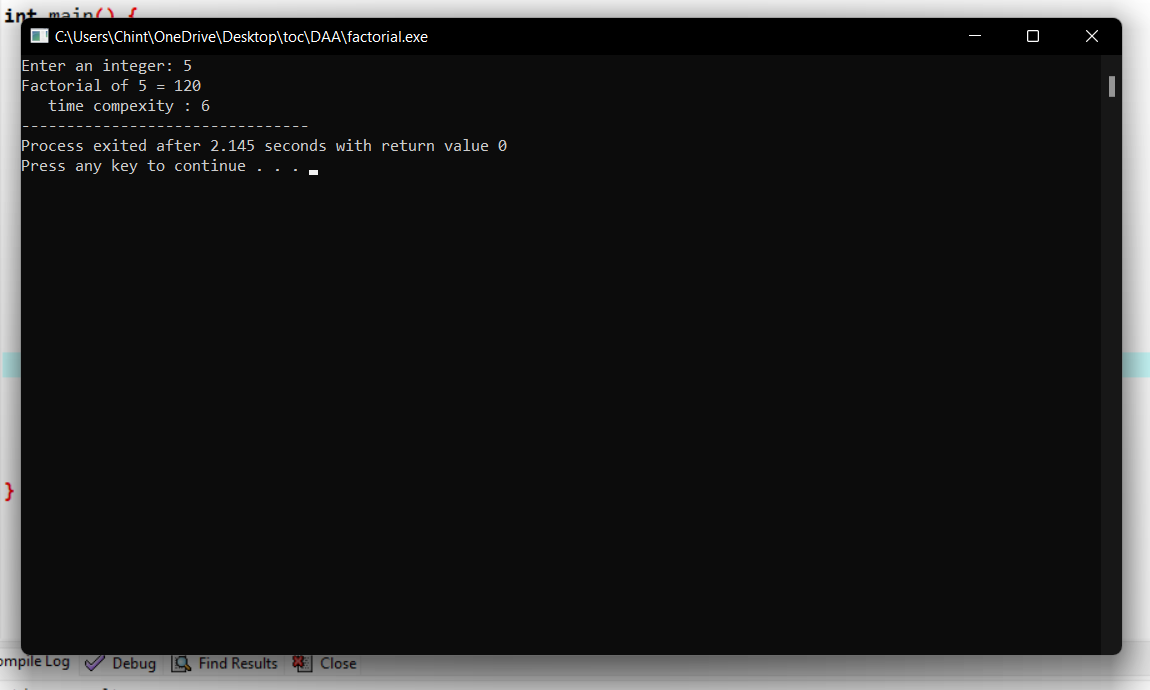
printf(" time compexity : %d ",count);

}

return 0;

}

# Output:



# 12.Perfect numbers:

## Program:

/\*\*

\* C program to print all Perfect numbers between 1 to n

\*/

#include <stdio.h>

int main()

{

int i, j, end, sum;

/\* Input upper limit to print perfect number \*/

printf("Enter upper limit: ");

scanf("%d", &end);

printf("All Perfect numbers between 1 to %d:\n", end);

/\* Iterate from 1 to end \*/

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

{

sum = 0;

/\* Check whether the current number i is Perfect number or not \*/

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

{

if(i % j == 0)

{

sum += j;

}

}

/\* If the current number i is Perfect number \*/

if(sum == i)

{

printf("%d, ", i);

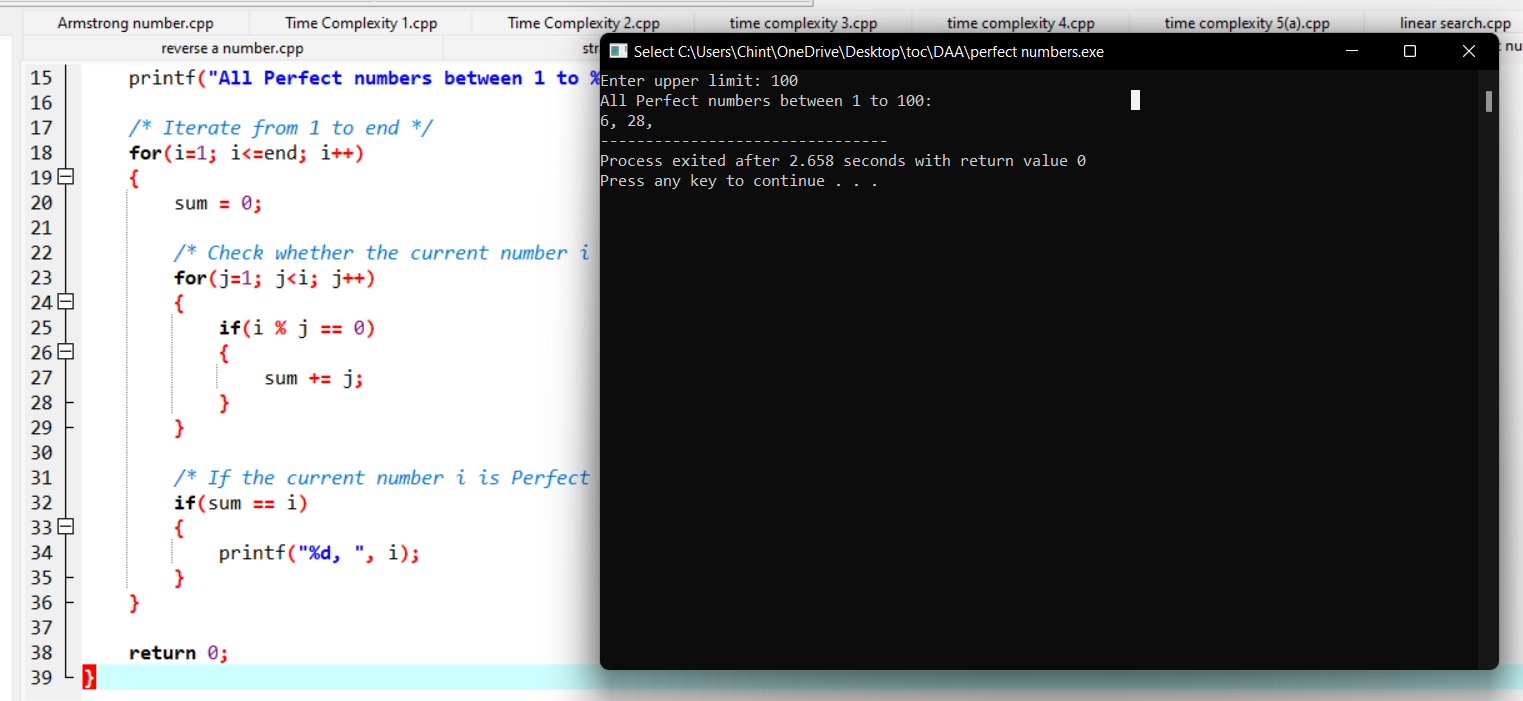
}

}

return 0;

}

## Output:



# 13.Palindrome:

# Program:

#include<stdio.h>

int main()

{

int i,n,r,s=0;

printf("\n Enter Integer Number:");

scanf("%d",&n);

//LOOP TO FIND REVERSE OF A NUMBER

for(i=n;i>0; )

{

r=i%10;

s=s\*10+r;

i=i/10;

}

/\* CHECKING IF THE NUMBER ENTERED AND THE REVERSE NUMBER IS EQUAL OR NOT \*/

if(s==n)

{

printf("\n %d is a Palindrome Number",n);

}

else

{

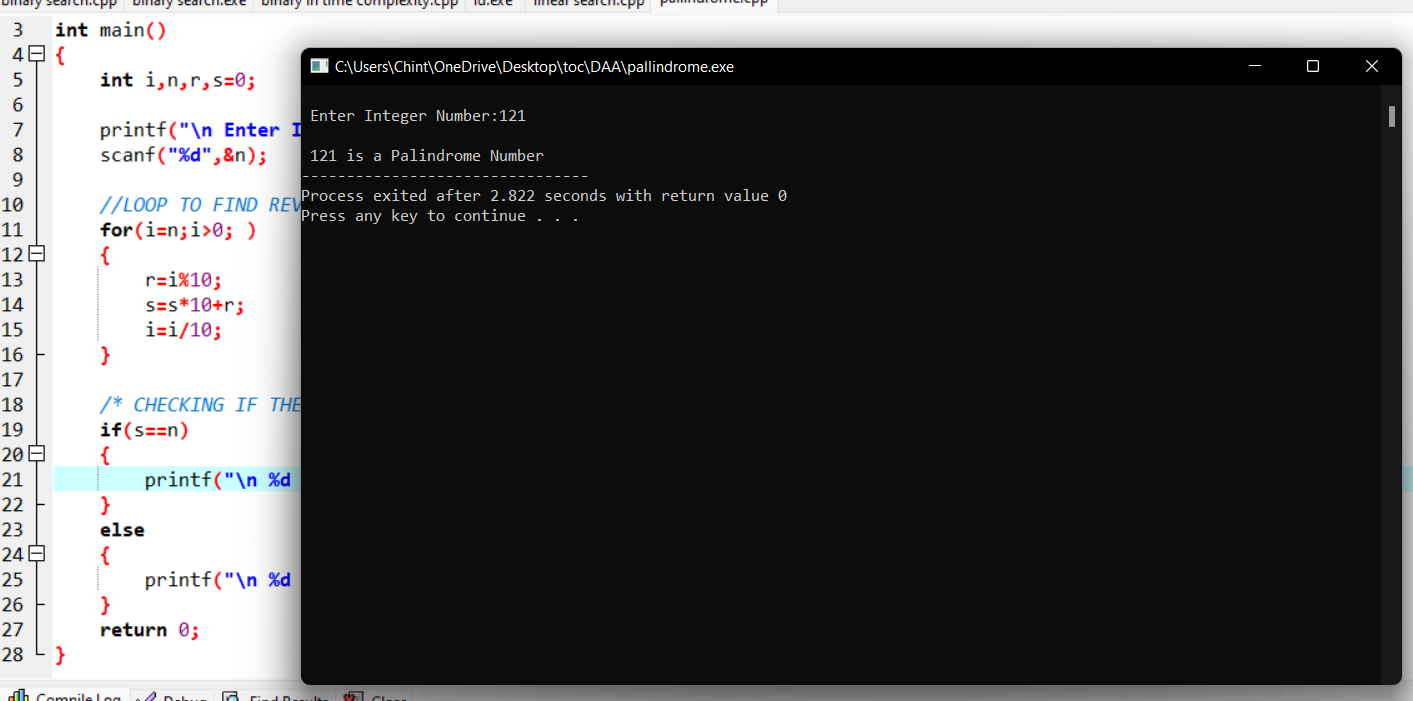
printf("\n %d is not a Palindrome Number",n);

}

return 0;

}

# Output:



# 14.Bubble sort:

## Program;

#include<stdio.h>

int main(){

int ele,count=0;

printf("Enter total element: ");

scanf("%d",&ele);

int arr[ele];

printf("Enter the elements: ");

for (int i = 0; i < ele; i++){

count++;

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

}count++;

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

{

count++;

for (int j =i+1; j < ele; j++)

{

count++;

if (arr[i]>arr[j])

{

count++;

int temp=arr[i];

count++;

arr[i]=arr[j];

count++;

arr[j]=temp;

count++;

}

}count++;

}count++;

printf("sorted array: ");

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

{count++;

count++;

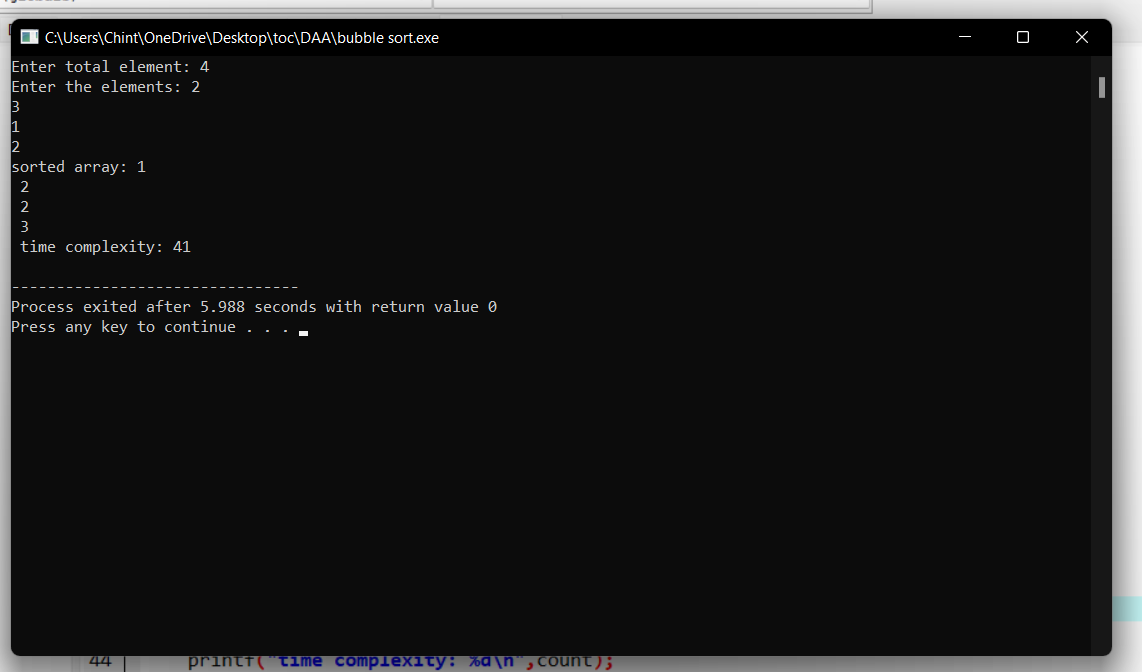
printf("%d \n ",arr[i]);

}count++;

printf("time complexity: %d\n",count);

}

# Output:



# 15.Reverse string:

# Program:

#include<stdio.h>

int main(){

char val[25];

printf("enter the value: ");

scanf("%s",&val);

int count=0,c=0;

while (val[count]!='\0'){

count++;

c++;

}c++;

for(int i=count-1;i>=0;i--){

c++;

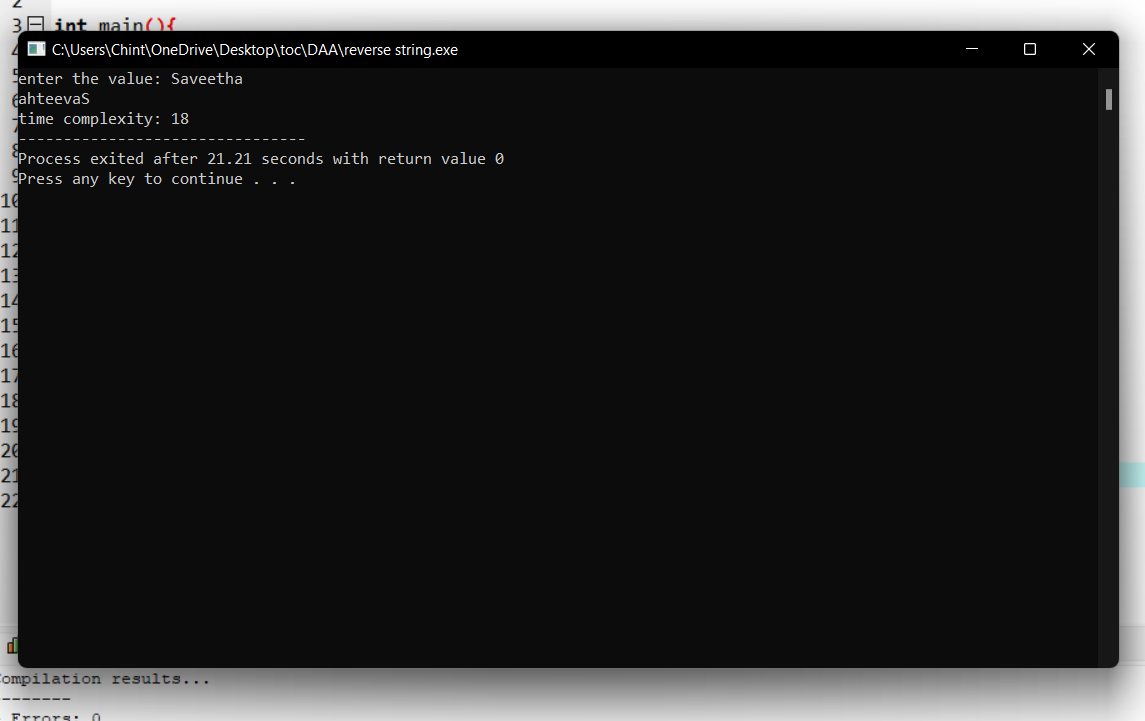
printf("%c",val[i]);

}c++;

printf("\ntime complexity: %d",c);

}

# Output:



# 16.substring:

# Program:

#include<stdio.h>

int main()

{

char str[80], search[10];

int count1 = 0, count2 = 0, i, j, flag;

printf("Enter a string:");

gets(str);

printf("Enter search substring:");

gets(search);

while (str[count1] != '\0')

count1++;

while (search[count2] != '\0')

count2++;

for (i = 0; i <= count1 - count2; i++)

{

for (j = i; j < i + count2; j++)

{

flag = 1;

if (str[j] != search[j - i])

{

flag = 0;

break;

}

}

if (flag == 1)

break;

}

if (flag == 1)

printf("SEARCH SUCCESSFUL!");

else

printf("SEARCH UNSUCCESSFUL!");

return 0;

}

# Output:

