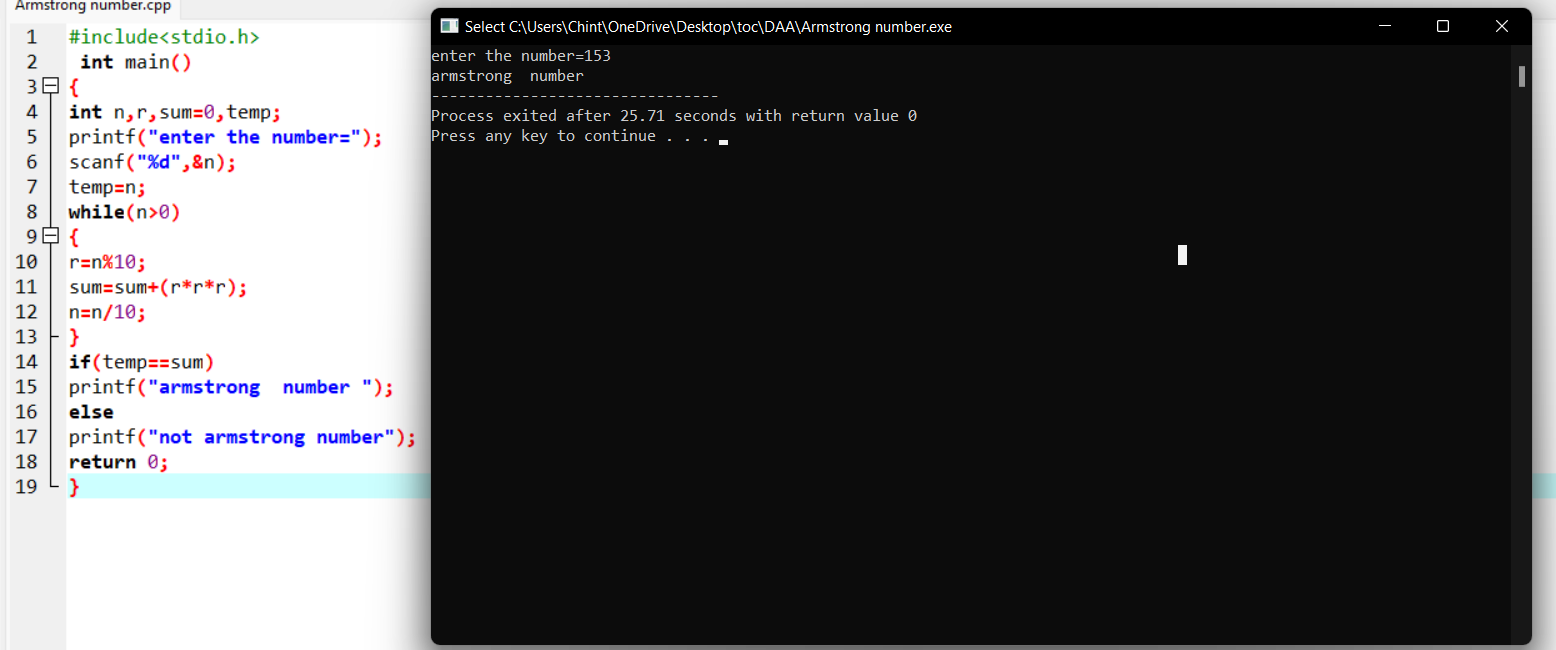
**DAA**

# 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:



# 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++;

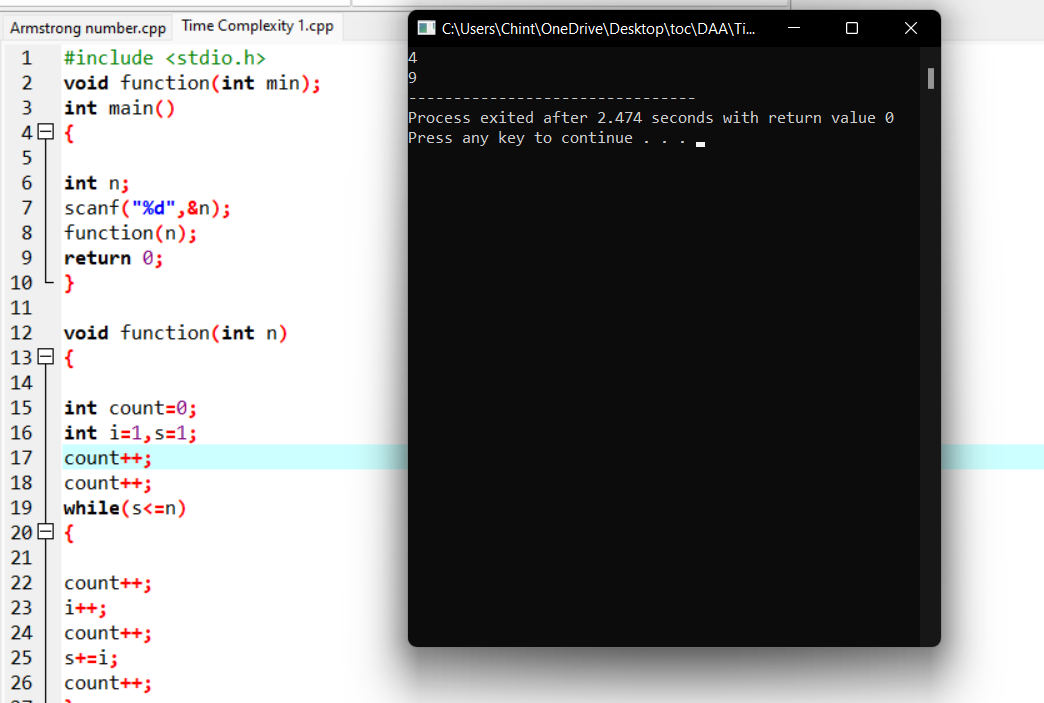
}

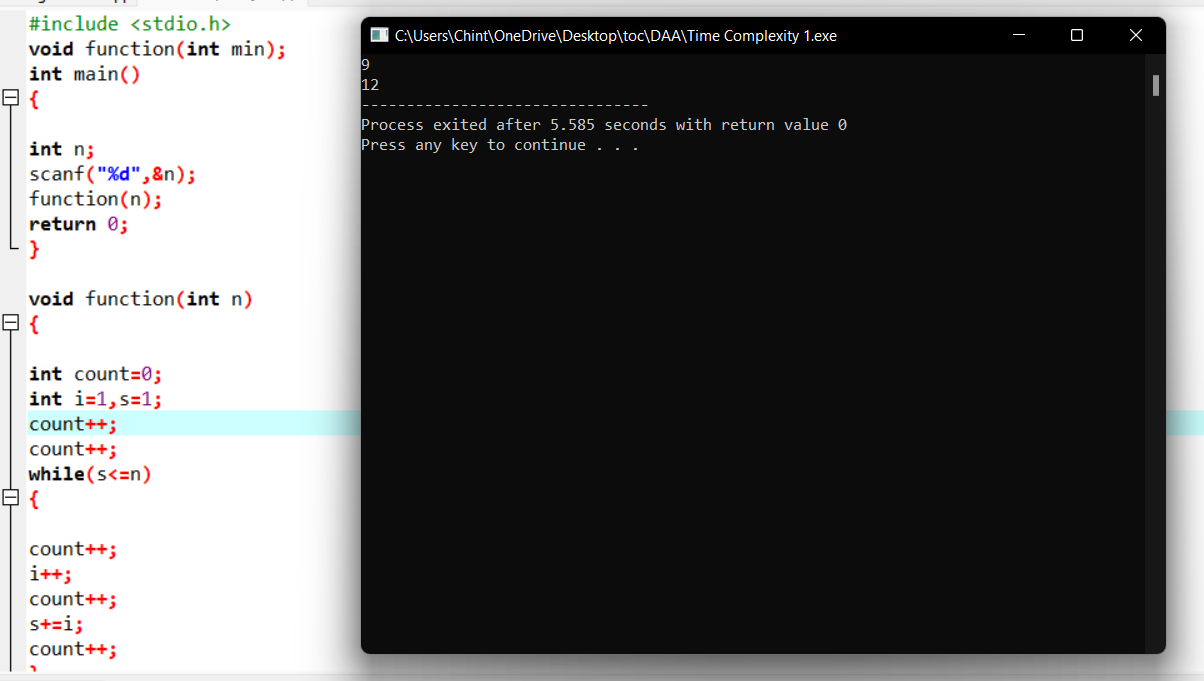
count++;

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;

}

}

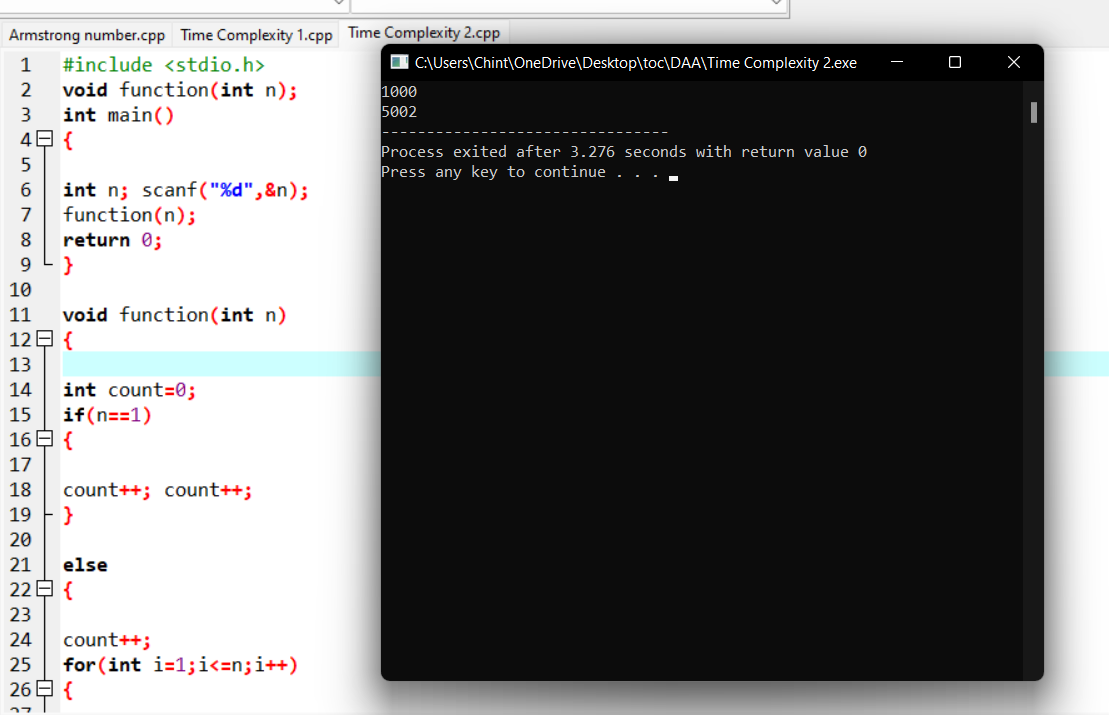
count++;

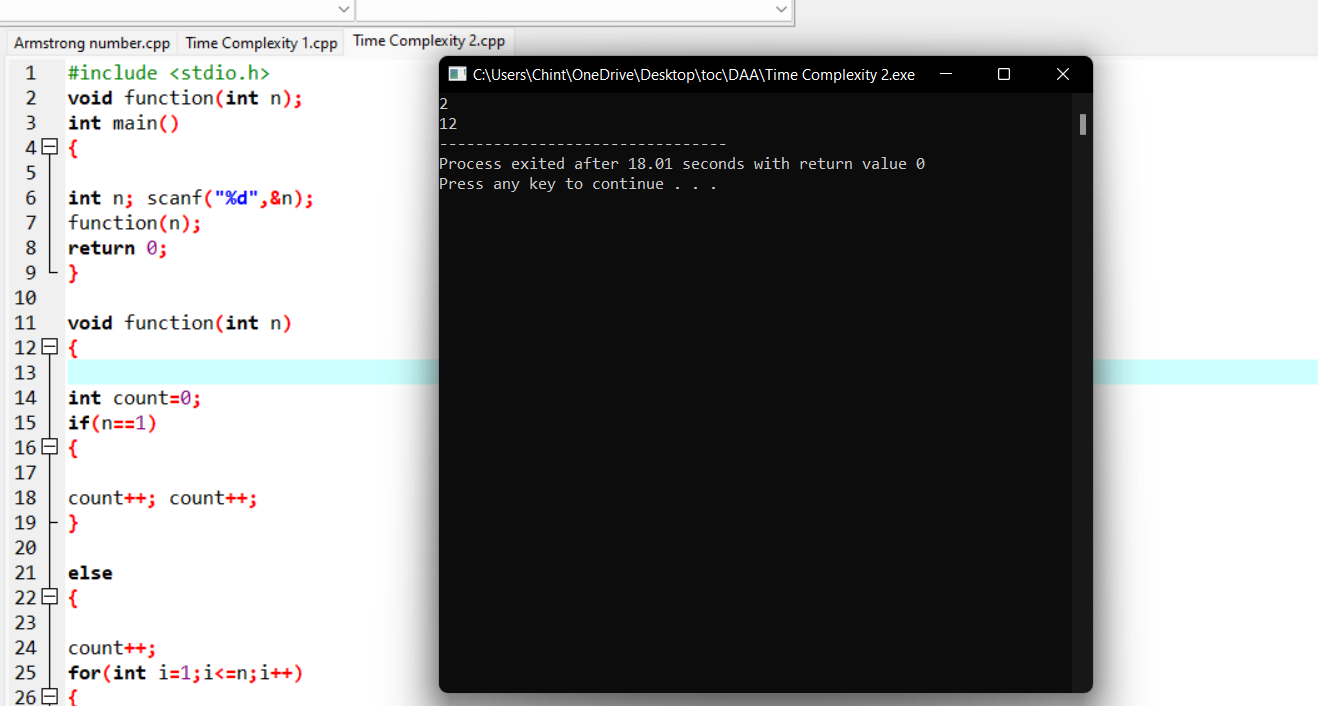
}

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++;

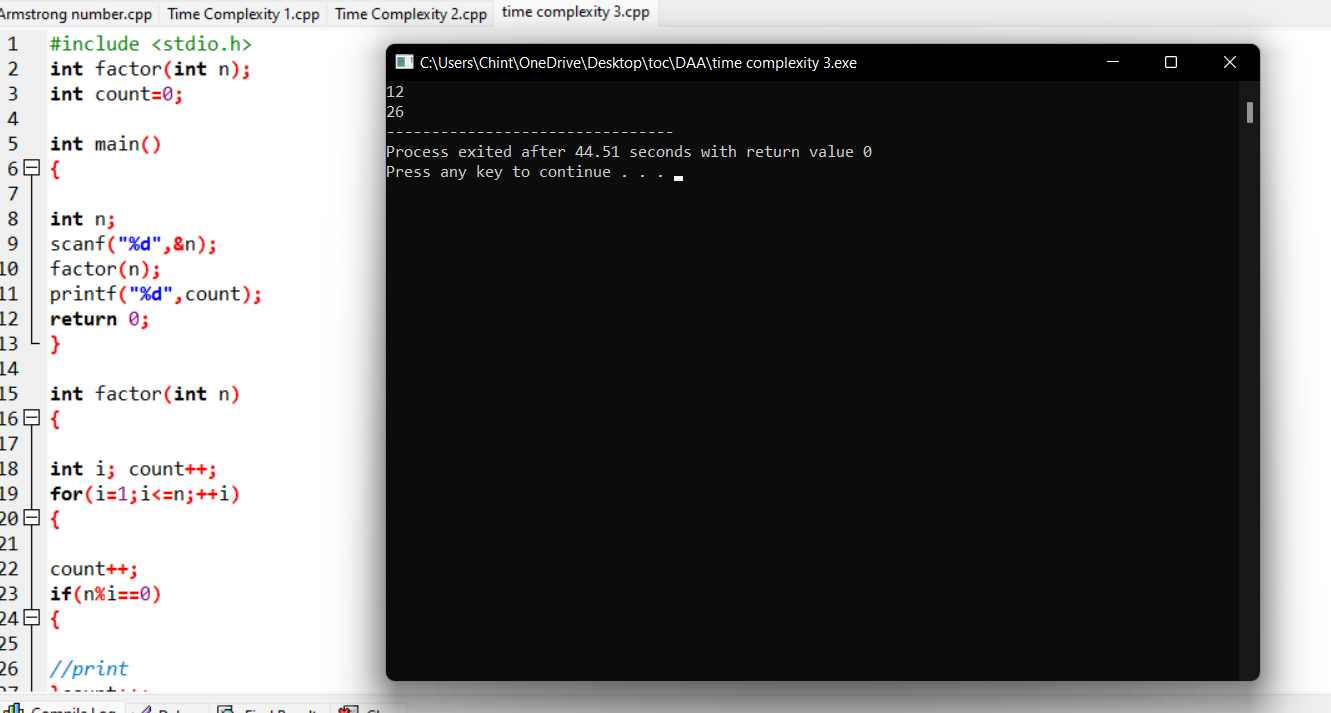
}

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++;

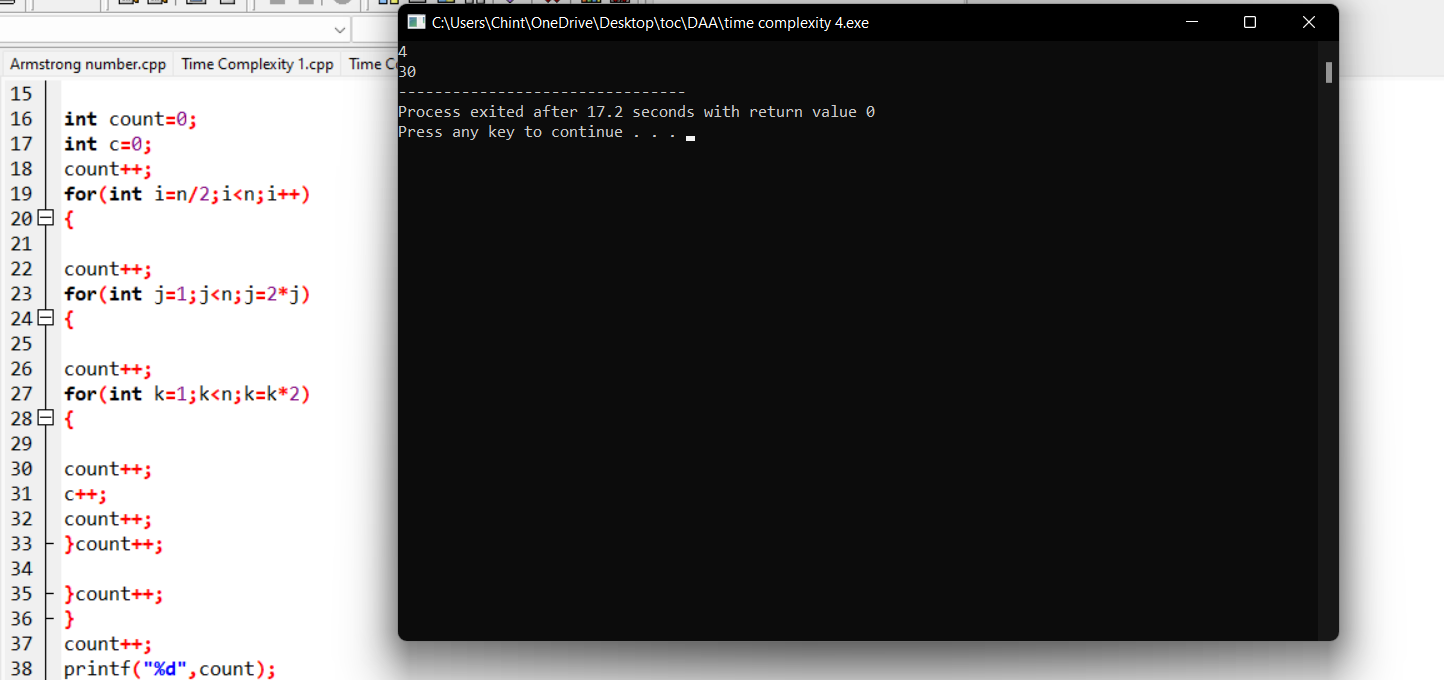
}

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:

# 

# Linear search:

## Program:

#include <stdio.h>

int main()

{

int array[100], search, c, n;

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

scanf("%d", &n);

printf("Enter %d integer(s)\n", n);

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

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

printf("Enter a number to search\n");

scanf("%d", &search);

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

{

if (array[c] == search) /\* If required element is found \*/

{

printf("%d is present at location %d.\n", search, c+1);

break;

}

}

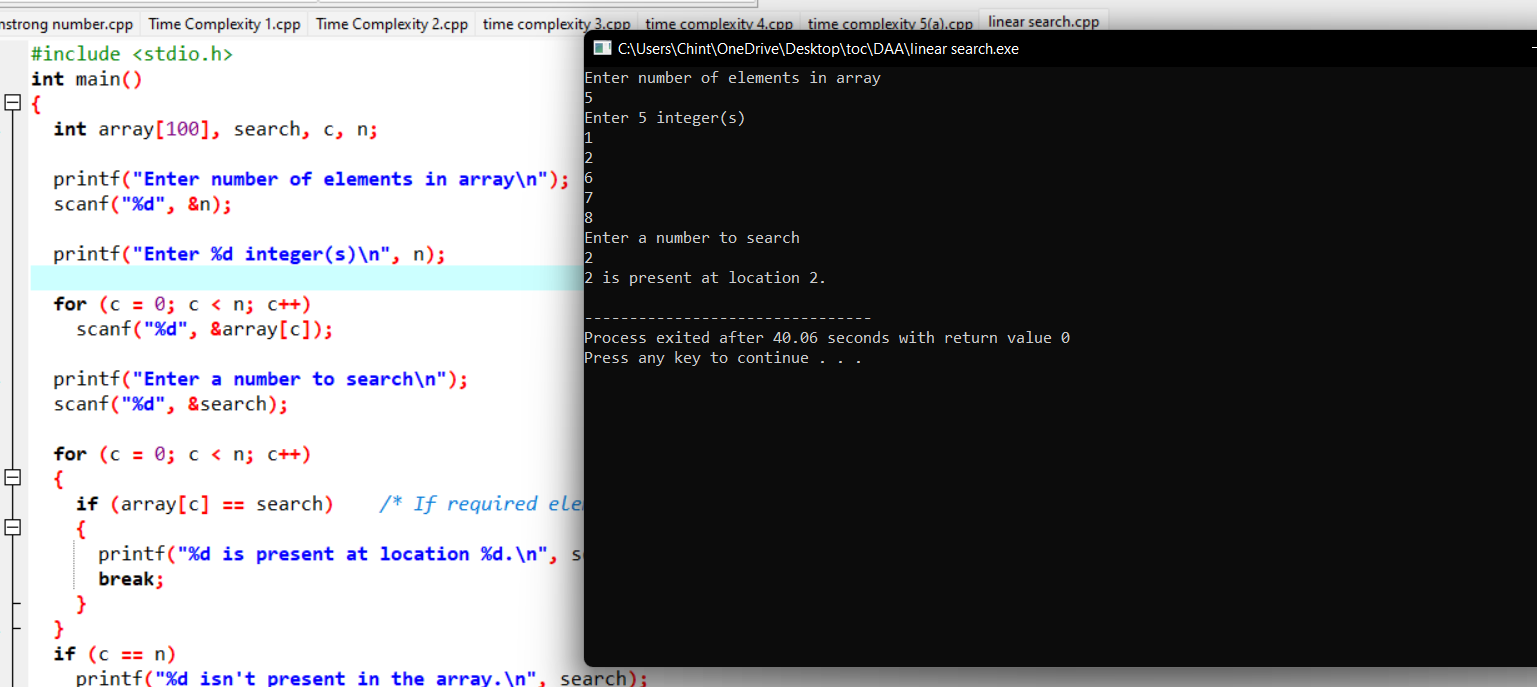
if (c == n)

printf("%d isn't present in the array.\n", search);

return 0;

}

## Output:



# 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;

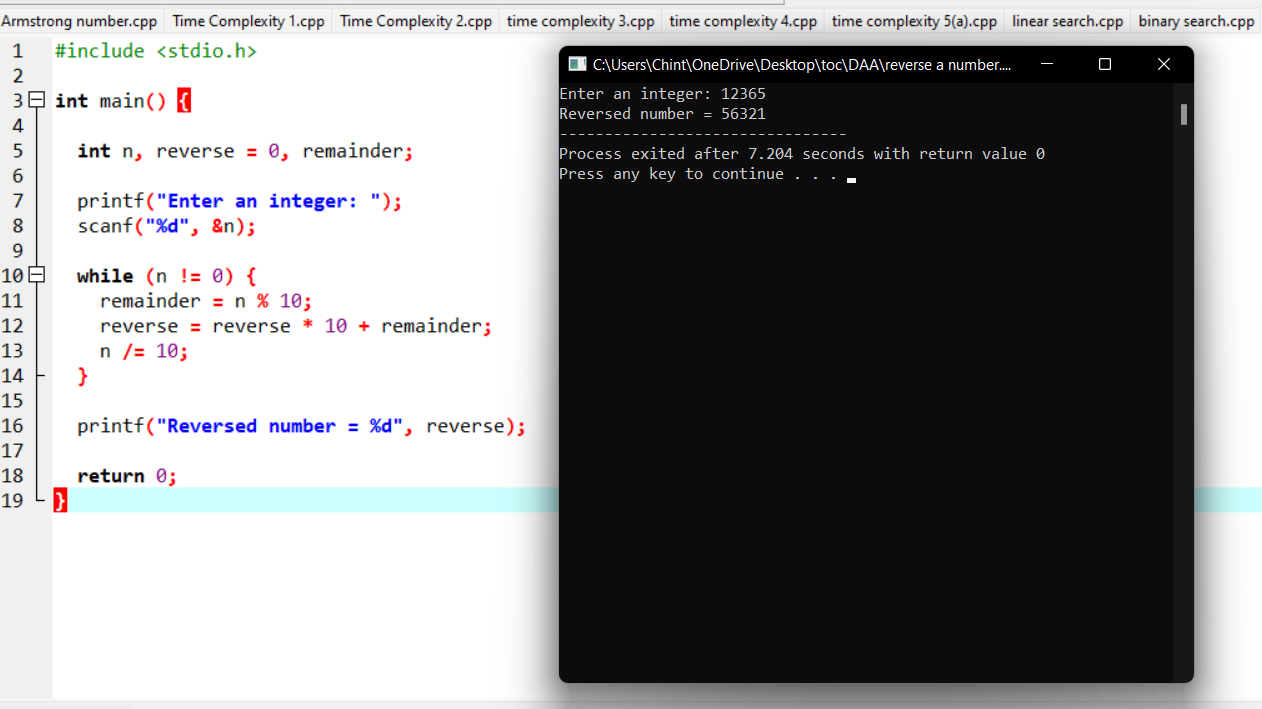
}

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

return 0;

}

# Output:



# Matrix multiplication:

# Program:

#include<stdio.h>

int main(){

int a[2][2],b[2][2],c[2][2],i,j;

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

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

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

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

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

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

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

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

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

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

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

printf("\n");

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

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

}

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

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

printf("\n");

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

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

}

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

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

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

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

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

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

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

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

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

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

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

printf("\nAfter multiplication using \n");

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

printf("\n");

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

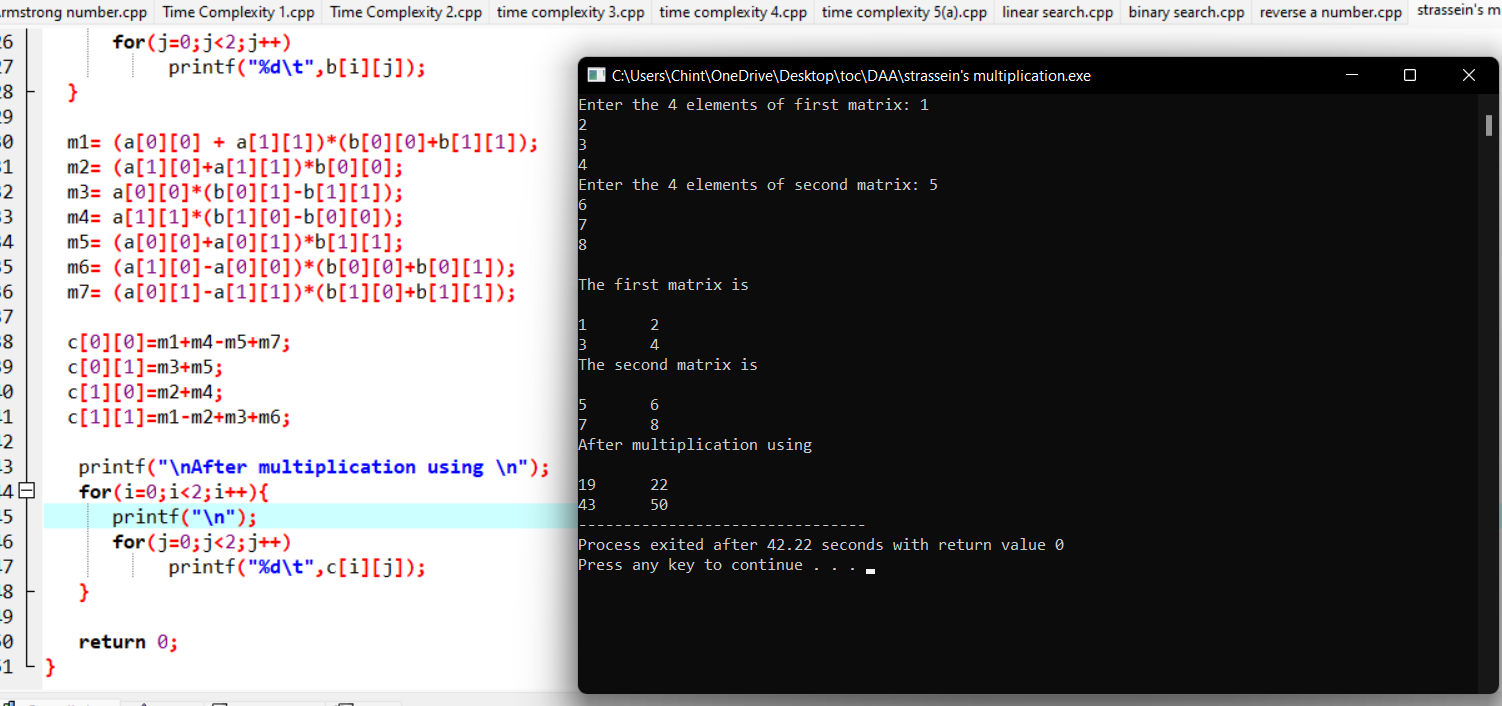
printf("%d\t",c[i][j]);

}

return 0;

}

## Output;



# Pascal triangle:

# Program:

#include <stdio.h>

int factorial(int n) {

int f;

for(f = 1; n > 1; n--)

f \*= n;

return f;

}

int ncr(int n,int r) {

return factorial(n) / ( factorial(n-r) \* factorial(r) );

}

int main() {

int n, i, j;

n = 5;

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

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

printf(" ");

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

printf(" %3d", ncr(i, j));

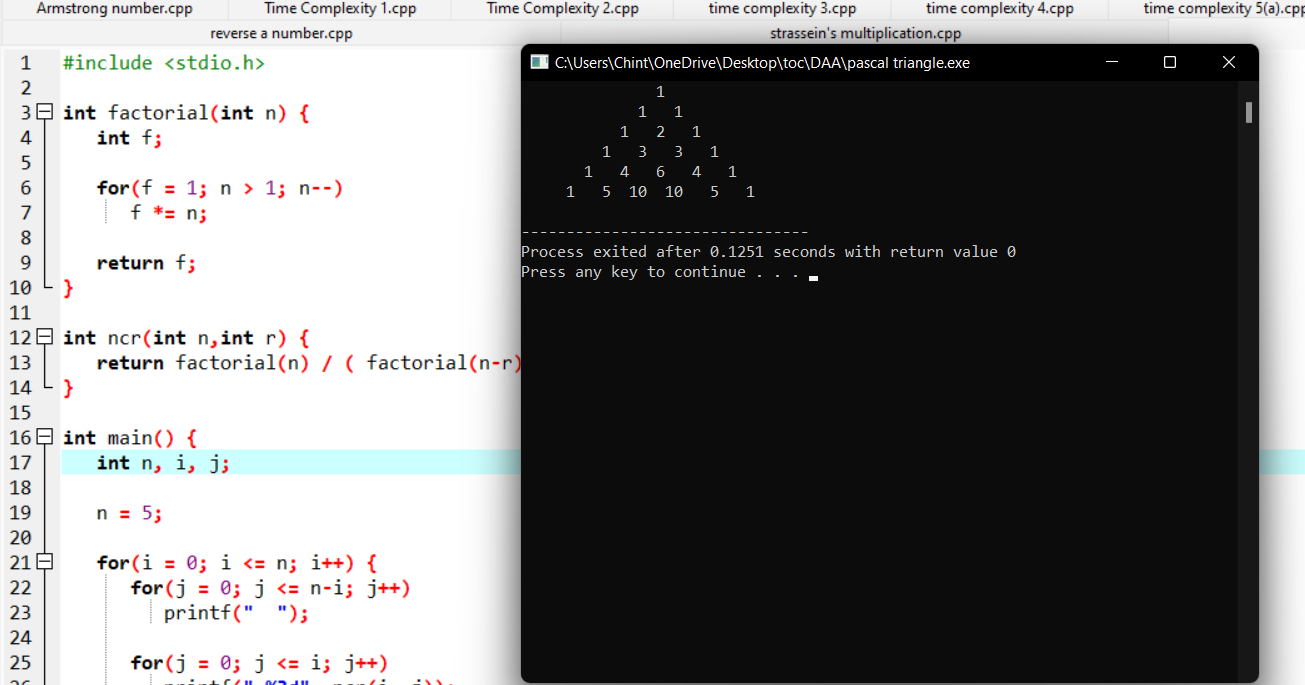
printf("\n");

}

return 0;

}

# Output:



# 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:

