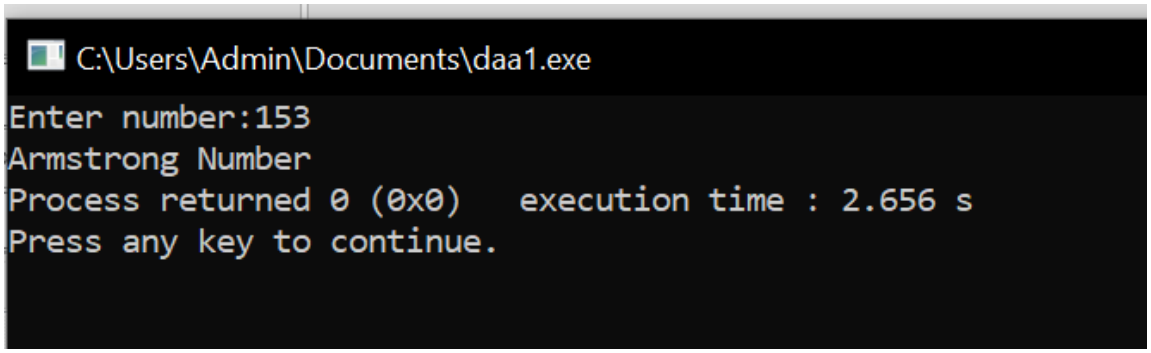
UBA0672 – DAA [DAY – 1]

K. Harish
192011134

1.Armstrong number

Program:

```
#include<stdio.h>
int main()
{
   int n,n1,sum=0,r;
   printf("Enter number:");
   scanf("%d",&n);
   n1=n;
   while (n1>0)
   {
      r=n1%10;
      sum=sum+(r*r*r);
      n1=n1/10;
   }
   if(sum==n)
   {
      printf("Armstrong Number");
   }
   else
   {
      printf("Not Armstrong Number");
   }
}
```



2.Time complexity 5 programs

Program-1:

Problem Statement 2:

Convert the following algorithm into a program and find its time complexity using the counter method.

```
void function (int n)
{
    int i= 1, s =1;
    while
(s <= n)
    {
                        i++;
                        s += i;
     }
}
```

Note: No need of counter increment for declarations and scanf() and count variable printf() st

Manually find the complexity using counter method and write the same in observation

Input:

A positive Integer n

Output:

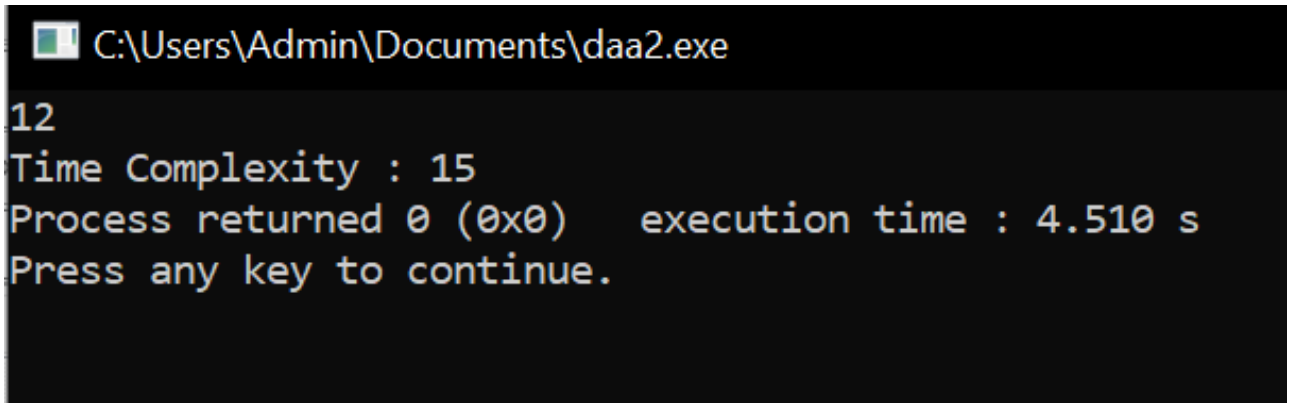
Print the value of the counter variable

For example:

Input	Result
9	12

Program:

```
#include<stdio.h>
int main()
{
    int n;
    scanf("%d",&n);
    function(n);
    return 0;
}

void function(int n)
{
    int c=0;
    int i=1,s=1;
    c++;
    c++;
    while(s<=n)
    {
        c++;
        i++;
        c++;
        s+=i;
        c++;
    }
    c++;
    printf("Time Complexity : %d",c);
}
```



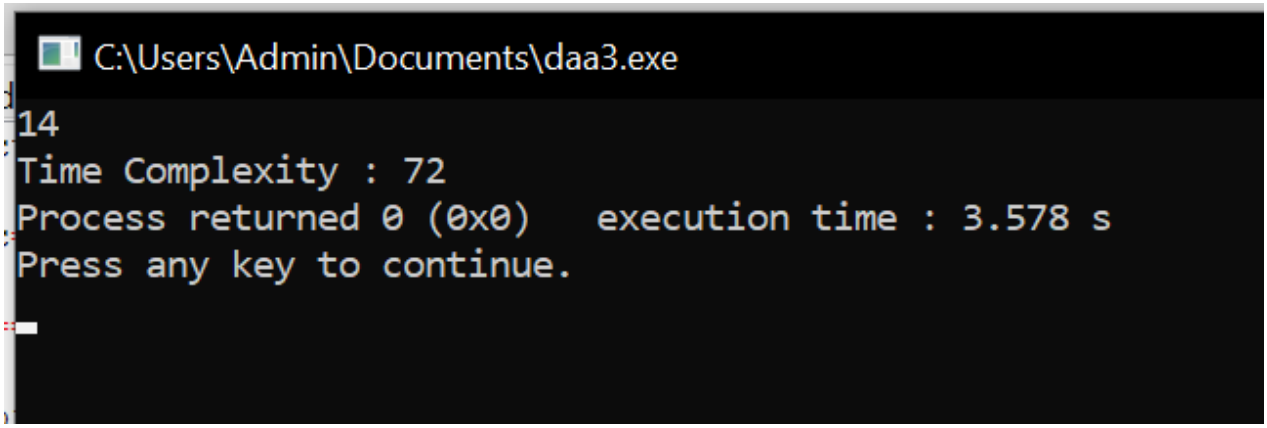
Program-2:

```
Problem
Statement 3:
 Convert the following algorithm into a program and find its time
complexity using the counter method.
void func(int n)
{
      if
(n==1)
            {
                        printf("");
            }
    else
            {
                        for
(int i=1; i<=n; i++)
                        {
                                    for
(int j=1; j<=n; j++)
                                    {
                                                printf
("");
                                                printf("");
                                                break;
                                    }
                        }
            }
  }
Note: No need of counter increment for declarations and scanf() and  count variable printf() st
Manually find the complexity using counter method and write the
same in observation
Input:
 A positive Integer n
Output:
Print the value of the counter variable
```

Program:

```
#include<stdio.h>
int main()
{
    int n;
    scanf("%d",&n);
    function(n);
```

```
    return 0;
}
void function(int n)
{
    int c=0;
    c++;
    if(n==1)
    {
        printf("");
        c++;
    }
    else
    {
        for(int i=1;i<=n;i++)
        {
            c++;
            for(int j=1;j<=1;j++)
            {
                c++;
                printf("");c++;
                printf("");c++;
                break;
            }
            c++;
        }
        c++;
    }
    printf("Time Complexity : %d",c);
}
```



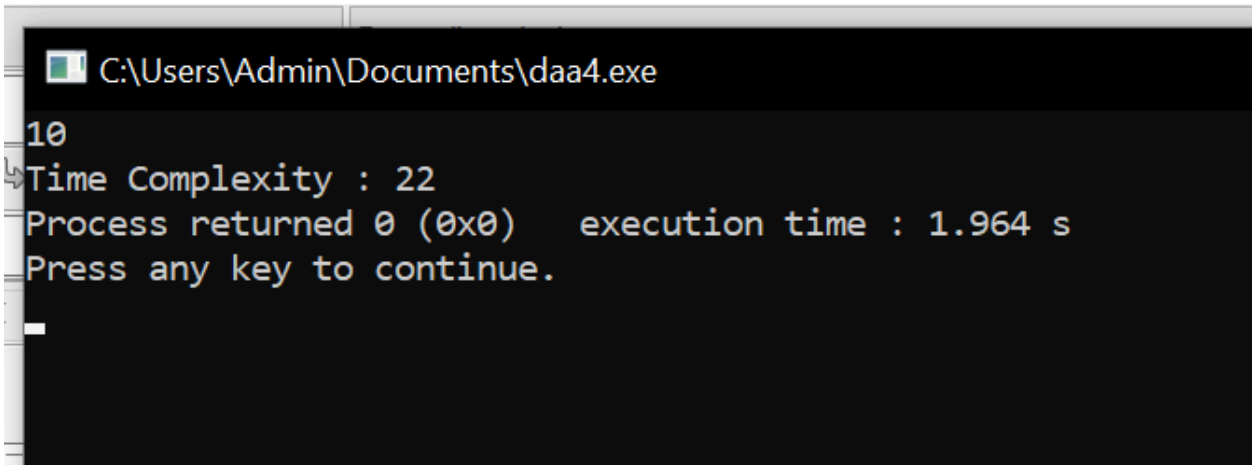
Program-3:

```
Problem Statement 4:

Convert the following algorithm into a program and find its time
complexity using counter method.


Factor(n) {
    {
    for (i = 1; i <= num;++i)
    {
                if (num % i== 0)
        {
                printf("%d ", i);
        }

    }
        return 0;
}


Note: No need of counter increment for declarations and scanf() and  printf() statements.
Manually find the complexity using counter method and write the
same in observation
Input:
 A positive Integer n
Output:
Print the value of the counter variable
```

Program:

```
#include<stdio.h>
int main()
{
   int n;
   scanf("%d",&n);
   Factor(n);
   return 0;
}

int Factor(int n)
{
```

```
    int c=0;
    int i=0;
    c++;
    for(int i=1;i<=n;i++)
    {
        c++;
        c++;
        if(n%i==0)
        {
            //printf("%d\n",i);

        }
    }
    c++;
    printf("Time Complexity : %d",c);
    return 0;
}
```



Program-4:

```
Problem Statement 5:


Convert the following algorithm into a program and find its time
complexity using counter method.

void function(int n)
{
    int c= 0;
    for(int i=n/2; i<n; i++)
        for(int j=1; j<n; j = 2 * j)
            for(int k=1; k<n; k = k * 2)
                c++;
}

Note: No need of counter increment for declarations and scanf() and  count variable printf() st
Manually find the complexity using counter method and write the
same in observation
Input:
 A positive Integer n
Output:
Print the value of the counter variable
```

Program:

```
#include<stdio.h>
int main()
{
    int n;
    scanf("%d",&n);
    function(n);
    return 0;
}

void function(int n)
{
    int c=0,cn=0;
    cn++;
    for(int i=n/2;i<n;i++)
```

```
    {
        cn++;
        for(int j=1;j<n;j=2*j)
        {
            cn++;
            for(int k=1;k<n;k=k*2)
            {
                cn++;
                c++;
                cn++;
            }
            cn++;
        }
        cn++;
    }
    cn++;
    printf("Time Complexity : %d",cn);
}
```

```
C:\Users\Admin\Documents\daa5.exe
15
Time Complexity : 338
Process returned 0 (0x0)   execution time : 2.648 s
Press any key to continue.
```

Program-5:

Problem Statement 6:

Convert the following algorithm into a program and find its time complexity using counter method.

```
void reverse(int n)
{
   int rev = 0, remainder;
   while (n != 0)
    {
        remainder = n % 10;
        rev = rev * 10 + remainder;
        n/= 10;

    }
print(rev);
}
```

Note: No need of counter increment for declarations and scanf() and count variable printf() st
Manually find the complexity using counter method and write the same in observation

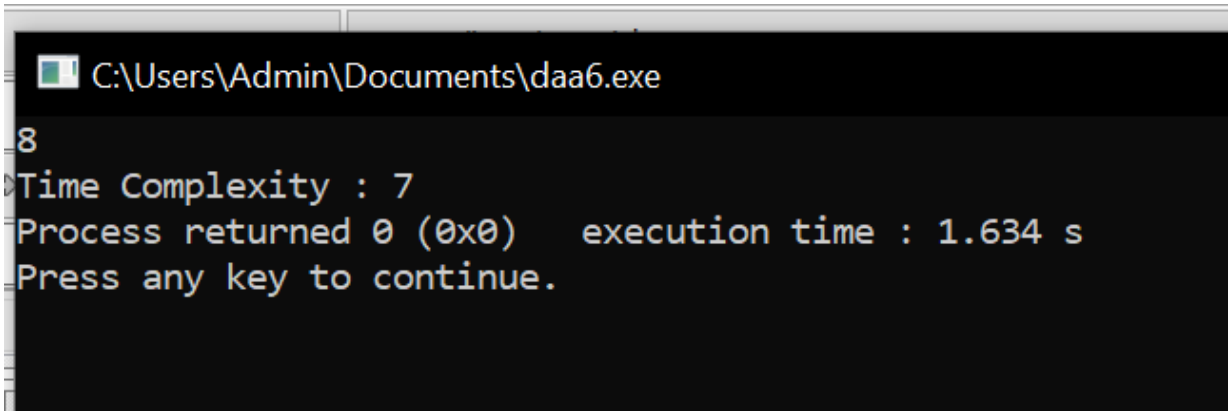
Input:
A positive Integer n

Output:
Print the value of the counter variable

Program:

```
#include<stdio.h>
int main()
{
   int n;
   scanf("%d",&n);
   reverse(n);
   return 0;
}

void reverse(int n)
{
```

```
    int c=0;
    int rev=0,remainder;
    c++;
    while(n!=0)
    {
        c++;
        remainder=n%10;
        c++;
        rev=rev*10+remainder;
        c++;
        n/=10;
        c++;
    }
    c++;
    //printf("%d",rev);
    c++;
    printf("Time Complexity : %d",c);
}
```

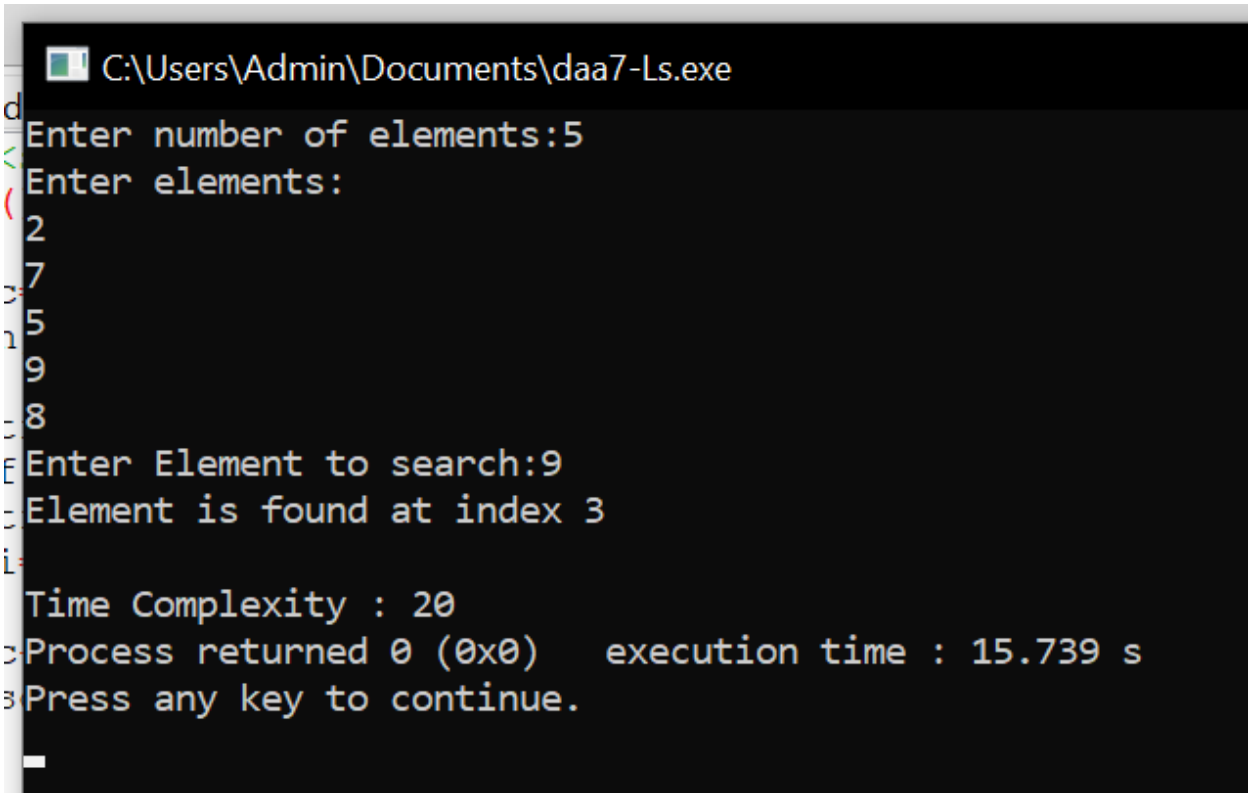


3.Write a program to search a number in a list using binary search and estimate time complexity

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);
}
```



4. Write a program to search a number in a list using linear search and estimate time complexity

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;
      }
```

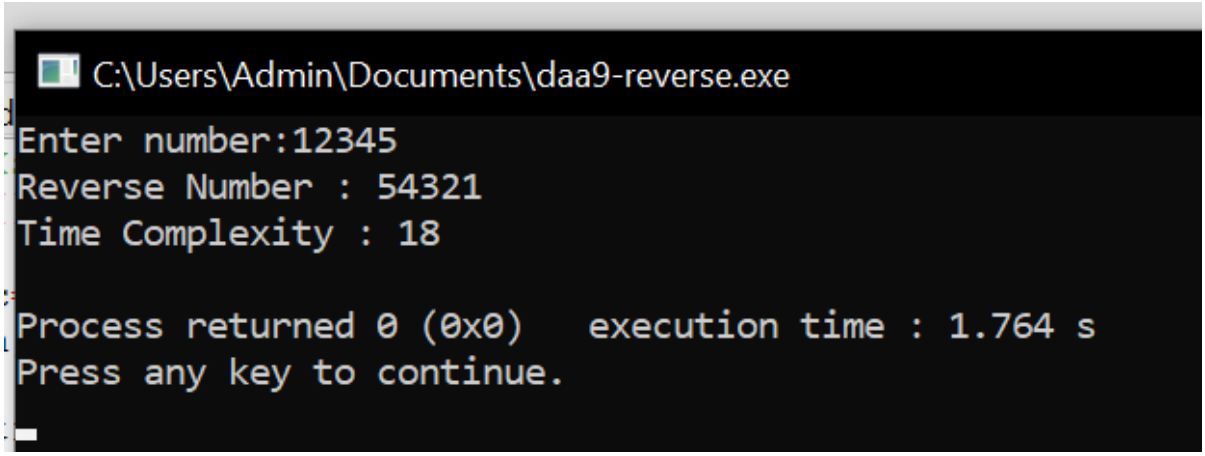
```
        else
        {
            high=mid-1;  c++;
        }
        mid=(low+high)/2;   c++;
    }
    c++;
    c++;
    if(low>high)
    {
        printf("Element is not found\n");
    }
    printf("\nTime Complexity : %d\n",c);
}
```

```
C:\Users\Admin\Documents\daa8-Bs.exe
Enter number of elements:6
Enter elements:
2
3
5
7
8
9
Enter Element to search:7

Element is found at index 3

Time Complexity : 23

Process returned 0 (0x0)   execution time : 6.250 s
Press any key to continue.
```

5. Write a program to find the reverse of a given number.

Program:

```
#include<stdio.h>
int main()
{
    int c=0;
    int n,r,rev=0;
    c++;
    printf("Enter number:");
    scanf("%d",&n);
    c++;
    while (n!=0)
    {
        r=n%10; c++;
        rev=(rev*10)+r; c++;
        n=n/10; c++;
    }
    c++;
    printf("Reverse Number : %d",rev);
    printf("\nTime Complexity : %d\n",c);
}
```



6. Write a C program to perform Strassen's Matrix Multiplication for the 2*2 matrix elements and Estimate time complexity.

Program:

```
#include<stdio.h>
int main()
{
    int a[2][2],b[2][2],c[2][2],i,j;
    printf("Enter elements of matrix A:\n");
    for(i=0;i<=1;i++)
    {
        for(j=0;j<=1;j++)
        {
            scanf("%d",&a[i][j]);
        }
    }
    printf("Enter elements of matrix B:\n");
    for(i=0;i<=1;i++)
    {
        for(j=0;j<=1;j++)
        {
            scanf("%d",&b[i][j]);
        }
    }
    c[0][0]=(a[0][0]*b[0][0])+(a[0][1]*b[1][0]);
    c[0][1]=(a[0][0]*b[0][1])+(a[0][1]*b[1][1]);
    c[1][0]=(a[1][0]*b[0][0])+(a[1][1]*b[1][0]);
    c[1][1]=(a[1][0]*b[0][1])+(a[1][1]*b[1][1]);
```

```
    printf("Matrix C : \n");
    for(i=0;i<=1;i++)
    {
        for(j=0;j<=1;j++)
        {
            printf("%d\t",c[i][j]);
        }
        printf("\n");
    }
}
```