# CSA0672 – DAA – DAY 2

**K. Harish 192011134**

# Write a program to generate all the reverse of a prime should be prime

**( for example 907 is prime and reverse 709 is also prime ) Generate all the no’s upto N and estimate time complexity.**

**Program:** #include<stdio.h> int main()

{

int c=0;

int n,n1,f,i,j,k,r,p[100],f1; int sum=0,b=0,rev=0; c++; c++; c++;

printf("Enter number:"); scanf("%d",&n); for(j=3;j<=n;j++)

{

c++;

f=0; c++;

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

{

c++; c++;

if(j%i==0)

{

f=f+1; c++;

}

}

c++; c++;

if(f==0)

{

n1=j; c++; rev=0; c++; while (n1!=0)

{

c++; r=n1%10; c++;

rev=(rev\*10)+r; c++; n1=n1/10; c++;

}

c++;

f1=0; c++;

for(k=2;k<rev;k++)

{

c++; c++;

if(rev%k==0)

{

f1++; c++;

}

}

c++; c++;

if(f1==0)

{

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

}

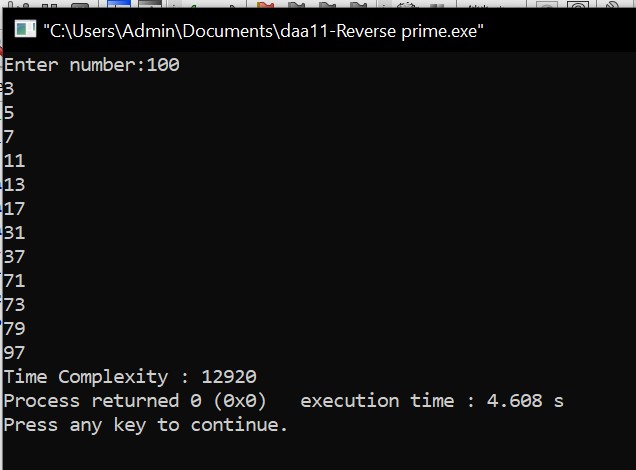
}

}

c++;

printf("Time Complexity : %d",c);

}



# Compute the program to find the GCD of two numbers. And also find the finf of time Recursion used to estimate time complexity.

**Program:** #include<stdio.h> int main()

{

int c=0;

int a,b,af[100],bf[100],cf[100],a1,b1,c1,i,j,g; printf("Enter 1st number : "); scanf("%d",&a);

printf("Enter 2nd number : "); scanf("%d",&b);

a1=-1; c++;

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

{

c++; c++;

if(a%i==0)

{

a1=a1+1; c++;

af[a1]=i; c++;

}

}

c++;

b1=-1; c++;

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

{

c++; c++;

if(b%i==0)

{

b1=b1+1; c++;

bf[b1]=i; c++;

}

}

c++;

c1=-1; c++;

for(i=0;i<a1+1;i++)

{

c++;

for(j=0;j<b1+1;j++)

{

c++; c++;

if(af[i]==bf[j])

{

g=af[i]; c++;

}

}

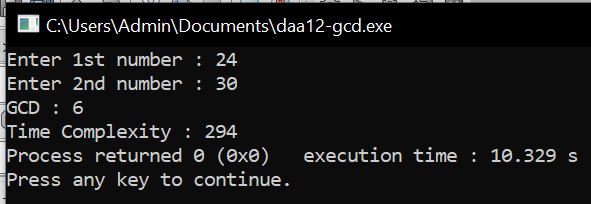
c++;

}

c++;

printf("GCD : %d\n",g); printf("Time Complexity : %d",c);

}



# Generate a program for Pascal triangle. Estimate the time complexity for the row=5

**1**

# 1 1

**1 2 1**

# 1 3 3 1

**1 4 6 4 1**

**Program:** #include<stdio.h> int main()

{

int c=0;

int n,i,j,k,s,c1;

printf("Enter no of rows :"); scanf("%d",&n);

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

{

c++;

k=k-1; c++; for(s=0;s<k;s++)

{

c++;

printf(" ");

}

c++;

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

{

c++; c++;

if(j==0)

{

c1=1; c++;

}

else

{

c1=c1\*(i-j+1)/j; c++;

}

printf("%d ",c1);

}

c++;

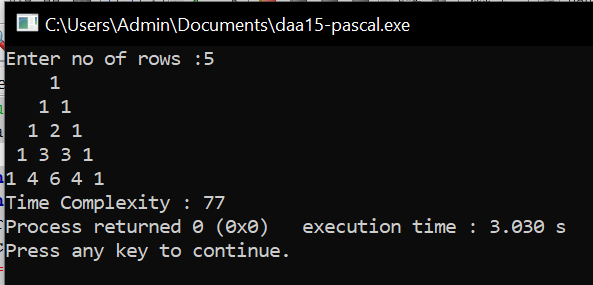
printf("\n");

}

c++;

printf("Time Complexity : %d",c);

}



# Write a program to find the largest element value in an array. Estimate the time complexity and no of comparison for the given set of values.

**Program:** #include<stdio.h> int main()

{

int c=0;

int com=0,i,j,k,a[100],n; c++;

printf("Enter no of elements:"); scanf("%d",&n);

printf("Enter elements :\n"); for(i=0;i<n;i++)

{

c++;

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

}

c++;

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

{

c++;

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

{

c++;

com++; c++; c++;

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

{

k=a[i]; c++; a[i]=a[j]; c++; a[j]=k; c++;

}

}

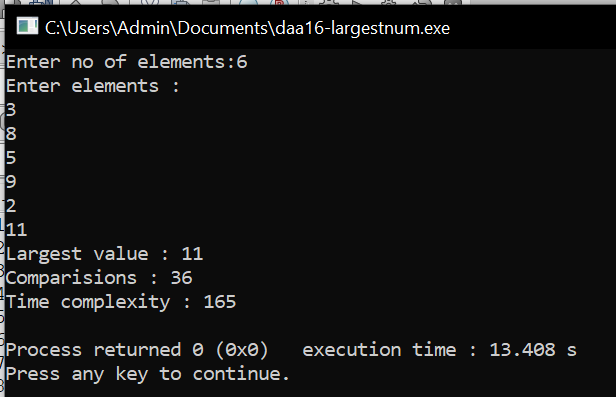
c++;

}

c++;

printf("Largest value : %d\n",a[0]); printf("Comparisions : %d\n",com); printf("Time complexity : %d\n",c);

}



# Write a program to find the factorial (fact)of a number and to estimate time complexity.

**Condition such as i. n=0, return 1 otherwise fact (n-1) \* n Program:**

#include<stdio.h> int fact(int n);

int c=0; int main()

{

int n;

printf("Enter Number : "); scanf("%d",&n);

fact(n);

printf("Factorial : %d\n",fact(n)); printf("Time Complexity : %d\n",c); return 0;

}

int fact(int n)

{

int f; c++;

if(n<=1)

{

f= 1; c++;

}

else

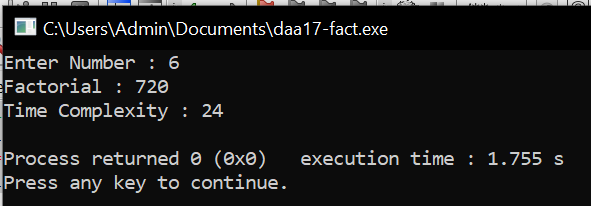
{

f= n\*fact(n-1); c++;

}

return f;

}



# Write a program to print the first n perfect numbers. (Hint Perfect number means a positive integer that is equal to the sum of its proper divisors)

**Sample Input:**

# N = 3

**Sample Output:**

# First 3 perfect numbers are: 6 , 28 , 496 Test Cases:

**1. N = 0**

# 2. N = 5

**3. N = -2**

# 4. N = -5

**N = 0.2**

**Program:** #include<stdio.h> int main()

{

int c=0;

int i,j,sum,n,a[20],k=0; c++;

printf("Enter N:");

scanf("%d",&n); c++;

if(n<1)

{

printf("Invalid Input");

}

else

{

for(i=6;i<10000;i++)

{

c++;

sum=0; c++; for(j=1;j<i-1;j++)

{

c++; c++;

if(i%j==0)

{

sum=sum+j; c++;

}

}

c++; c++;

if(i==sum)

{

a[k++]=i; c++;

}

}c++;

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

{

c++;

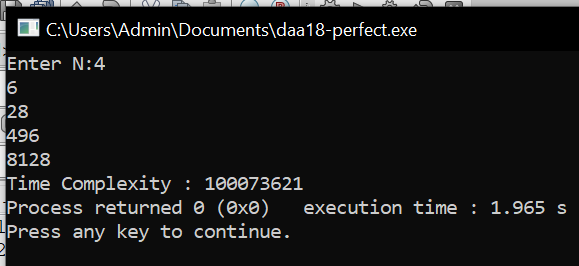
printf("%d\n",a[i]);

}c++;

}

printf("Time Complexity : %d",c);

}



# Write a C program to check whether is a given input is a palindrome

**Program:** #include<stdio.h> int main()

{

int c=0;

int n,r,rev=0,a; c++;

printf("Enter number:"); scanf("%d",&n);

a=n; c++; while (n!=0)

{

c++; r=n%10; c++;

rev=(rev\*10)+r; c++; n=n/10; c++;

}

c++; c++;

if(rev==a)

{

printf("Palindrome Number");

}

else

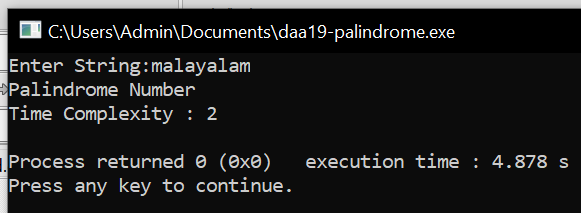
{

printf("Not Palindrome Number");

}

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

}



# Write a program to perform Bubble sort and estimate time Complexity

**Program:** #include<stdio.h> int main()

{

int c=0;

int com=0,i,j,k,a[100],n; c++;

printf("Enter no of elements:"); scanf("%d",&n);

printf("Enter elements :\n"); for(i=0;i<n;i++)

{

c++;

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

}

c++;

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

{

c++;

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

{

c++;

com++; c++; c++;

if(a[i]<a[j])

{

k=a[i]; c++; a[i]=a[j]; c++; a[j]=k; c++;

}

}

c++;

}

c++;

printf("Bubble Sort :\n"); for(i=0;i<n;i++)

{

c++;

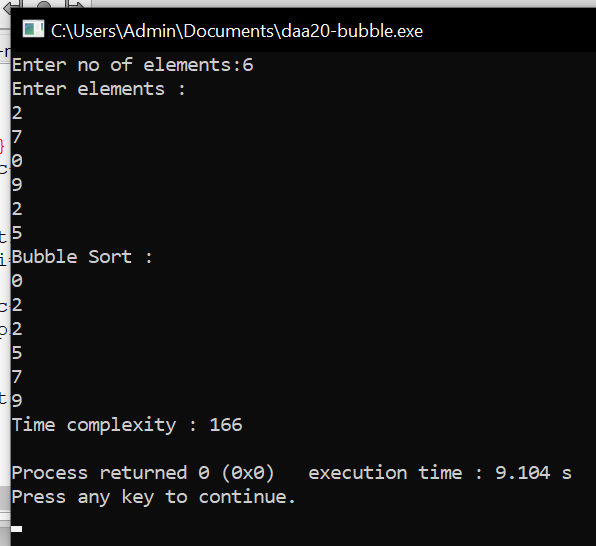
printf("%d\n",a[i]);

}

c++;

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

}



# Write a program to print the reverse of a string. And estimate the time complexity

**Program:** #include<stdio.h> int main()

{

int c=0,l,i; char s[20];

printf("Enter String:"); scanf("%s",&s); l=strlen(s); c++; printf("Reverse String : "); for(i=l-1;i>-1;i--)

{

c++;

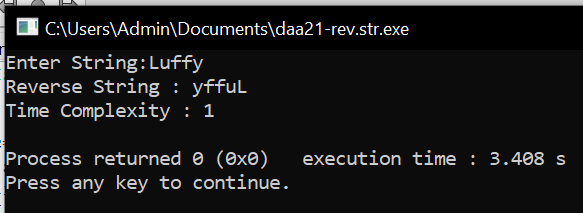
printf("%c",s[i]);

}

c++;

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

}



# Write a program to check sub string is there in a string or not. Program:

#include<stdio.h> int main()

{

int c=0,l1,l2,i,cnt=0;

char s[100],sub[20],os[20],at='@',sub1[20]; c++;

printf("Enter String:"); scanf("%s",&s); printf("Enter Sub String:"); scanf("%s",&sub); l1=strlen(s); c++; l2=strlen(sub); c++; strncat(sub,&at,1); c++; for(i=0;i<=l1+1-l2;i++)

{

c++;

strncpy(os,s+i,l2); c++; c++;

if(strcmp(sub,os)==0)

{

cnt++; c++;

}

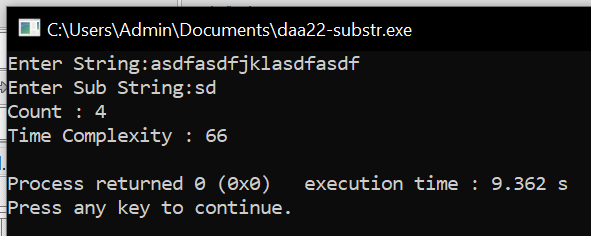
}

c++;

printf("Count : %d",cnt);

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

}



# Write a C program to merge sort using divide and Conquer Program:

#include<stdio.h>

void mergesort(int a[],int i,int j);

void merge(int a[],int i1,int j1,int i2,int j2); int main()

{

int a[30],n,i;

printf("Enter no of elements:"); scanf("%d",&n);

printf("Enter array elements:\n"); for(i=0;i<n;i++)

{

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

}

mergesort(a,0,n-1); printf("Merge Sort : \n"); for(i=0;i<n;i++)

{

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

}

return 0;

}

void mergesort(int a[],int i,int j)

{

int mid;

if(i<j)

{

mid=(i+j)/2; mergesort(a,i,mid); mergesort(a,mid+1,j); merge(a,i,mid,mid+1,j);

}

}

void merge(int a[],int i1,int j1,int i2,int j2)

{

int temp[50]; int i,j,k; i=i1;

j=i2; k=0;

while(i<=j1 && j<=j2)

{

if(a[i]<a[j])

{

temp[k++]=a[i++];

}

else

{

temp[k++]=a[j++];

}

}

while(i<=j1)

{

temp[k++]=a[i++];

}

while(j<=j2)

{

temp[k++]=a[j++];

}

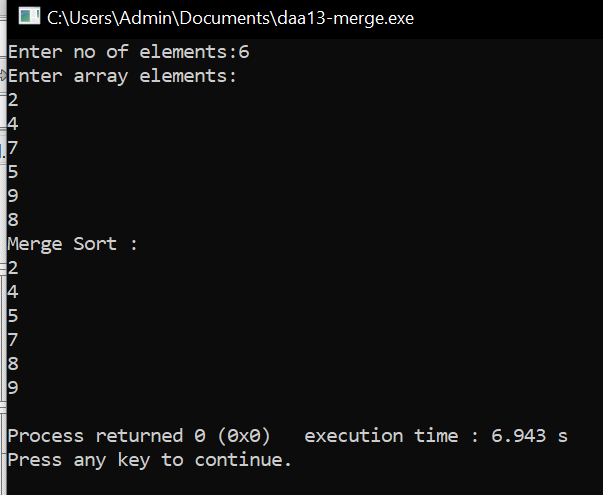
for(i=i1,j=0;i<=j2;i++,j++)

{

a[i]=temp[j];

}

}



# Write a C program to find max-min using divide and Conquer

**Program:**

#include<stdio.h>

void mergesort(int a[],int i,int j);

void merge(int a[],int i1,int j1,int i2,int j2); int main()

{

int a[30],n,i;

printf("Enter no of elements:"); scanf("%d",&n);

printf("Enter array elements:\n"); for(i=0;i<n;i++)

{

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

}

mergesort(a,0,n-1); printf("\nMin : %d",a[0]);

printf("\nMax : %d",a[n-1]); return 0;

}

void mergesort(int a[],int i,int j)

{

int mid; if(i<j)

{

mid=(i+j)/2; mergesort(a,i,mid); mergesort(a,mid+1,j); merge(a,i,mid,mid+1,j);

}

}

void merge(int a[],int i1,int j1,int i2,int j2)

{

int temp[50]; int i,j,k; i=i1;

j=i2;

k=0;

while(i<=j1 && j<=j2)

{

if(a[i]<a[j])

{

temp[k++]=a[i++];

}

else

{

temp[k++]=a[j++];

}

}

while(i<=j1)

{

temp[k++]=a[i++];

}

while(j<=j2)

{

temp[k++]=a[j++];

}

for(i=i1,j=0;i<=j2;i++,j++)

{

a[i]=temp[j];

}

}

