Program for Fibonacci series using iterative approach::

#include <stdio.h>

void main()

{

int n,f1=0,f2=1,th;

printf("enter number of terms in the series");

scanf("%d",&n);

if(n<0)

printf("invalid");

else if(n==0)

printf("%d",f1);

else if(n==1)

printf("%d ",f2);

else

{

while(n-2)

{

th=f1+f2;

f1=f2;

f2=th;

n--;

}

printf("%d",th);

}

}

Fibonacci Series using recursion approach::

#include <stdio.h>

int fibonacci(int n);

int f1=0,f2=1;

void main()

{

int n,th,x;

printf("enter number of terms in the series");

scanf("%d",&n);

if(n<0)

printf("invalid");

else

{

n=n-1;

x=fibonacci(n);

printf("%d",x);

}

}

int fibonacci(int n)

{

int th;

if(n==0)

return f1;

else if(n==1)

return f2;

else

th=fibonacci(n-1)+fibonacci(n-2);

return th;

}

To find pow(x,n)%M using iterative approach::

#include <stdio.h>

int main()

{

int n,mod,i,res=1;

unsigned int x;

scanf("%d %d %d",&n,&x,&mod);

while(x!=0)

{

res=res\*n;

x--;

}

res=res%mod;

printf("%d",res);

return 1;

}

To find anumber is prime or not using recursive approach::

#include <stdio.h>

int prime(int num,int i);

void main()

{

int num,i=2;

scanf("%d",&num);

if(num<=0)

printf("enter valid number");

else if(num==1)

printf("neithr prime nor composite");

else if(1==prime(num,i))

printf("prime");

else

printf("not prime");

}

int prime(int num,int i)

{

if(num%i==0 && i==num)

return 1;

else if(num%i==0 && i!=num)

return 0;

else

{

i++;

prime(num,i);

}

}

To find whether a number is prime or not using iterative approach:::

#include <stdio.h>

int main()

{

int n,i,x;

scanf("%d",&n);

if(n<=0)

{

printf("enter valid number");

return 1;

}

else if(n==1)

{

printf("1 is neither prime nor composite");

return 1;

}

else

{

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

{

if(n%i==0)

x++;

}

if(x==0)

printf("prime");

else

printf("not prime");

}

return 1;

}

To find pow(x,n} using recursive approach::

#include <stdio.h>

float power(int n , int x);

void main()

{

int n,i,x;

float result=1;

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

result=power(n,x);

printf("%f",result);

}

float power(int n,int x)

{

float res=1;

if(x==0)

return 1;

else if(x<0)

{

res=res/n\*power(n,x=x+1);

return res;

}

else

res=n\*power(n,x=x-1);

return res;

}

To find pow(x,n) using iterative approach:

#include <stdio.h>

void main()

{

int n,i,x;

float result=1;

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

if(x>0)

{

while(x!=0)

{

result=result\*n;

x--;

}

}

else

{

while(x!=0)

{

result=result/n;

x++;

}

}

printf("%f",result);

}

To find factorial of a number –Recursive approach::

#include <stdio.h>

unsigned long long factorial(int n);

int main()

{

unsigned long long fact=1;

int i,n;

scanf("%d",&n);

if(n<0)

{

printf("enter valid number");

return 1;

}

else

fact=factorial(n);

printf("%llu",fact);

}

unsigned long long factorial(int n)

{

unsigned long long f;

if(n==0 || n==1)

return 1;

else

f=n\*factorial(n-1);

return f;

}

To find factorial of a number—iterative approach

#include <stdio.h>

void main()

{

unsigned long long fact=1;

int i,n;

scanf("%d",&n);

if(n<0)

printf("enter valid number");

else if(n==0)

printf("%d",1);

else

{

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

{

fact=fact\*i;

}

printf("%llu",fact);

}

}

Find the contiguous subarray within an array(containing atleast one number) which has largest sum::

#include <stdio.h>

#include<stdlib.h>

void main()

{

int a[100],i,j,n,\*s,sum=0,largest=0,t[10],k,l=0,m=0,b=0,x=0;

scanf("%d",&n);

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

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

while(b!=n)

{

i=b;

while(i!=n)

{

t[l]=0;

sum=0;

x=b;

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

{

sum=sum+(a[j]);

t[l]=t[l]+1;

}

if(sum>largest)

{

largest=sum;

m=t[l];

s=(int\*) malloc(m\*sizeof(int));

for(k=0;k<m;k++)

{

s[k]=a[x];

x++;

}

l++;

}

i++;

}

b++;

}

for(k=0;k<m;k++)

printf("%d ",s[k]);

}

Given an unsorted array,find the first repetitive element.

#include <stdio.h>

#include<stdlib.h>

void main()

{

int n,i,\*v,\*a,largest=0;

scanf("%d",&n);

v=(int\*) malloc(n\*sizeof(int));

a=(int\*) malloc(10000\*sizeof(int));

if(a==NULL||v==NULL)

{

printf("not-allocated");

exit(0);

}

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

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

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

{

if(a[v[i]]==0)

a[v[i]]=1;

else

{

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

break;

}

}

}

Given a read only array of n integers from 1 to n.one element is repeated and one element is missing.Find both repeating and missing element::

#include <stdio.h>

#include<stdlib.h>

void main()

{

int n,i,\*v,\*temp;

scanf("%d",&n);

temp=(int\*) malloc((n+1) \* sizeof(int));

v=(int\*) malloc(n \* sizeof(int));

if(temp == NULL || v==NULL)

{

printf("Error! memory not allocated.");

exit(0);

}

else

{

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

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

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

{

if(temp[v[i]]==0)

temp[v[i]]=1;

else

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

}

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

{

if(temp[i]==0)

printf("%d",i);

}

}

}

To find the first repeating integer in an array between range of numbers between1 to n+1::

#include <stdio.h>

void main()

{

int n,i,j,a[100];

scanf("%d",&n);

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

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

i=a[0];

j=a[a[0]];

while(i!=j)

{

i=a[i];

j=a[a[j]];

}

i = 0;

while(i!=j){

i=a[i];

j=a[j];

}

printf("%d",i); }