**#include <stdio.h>**

**int nat(int n){**

**return (n\*(n+1))/2;**

**}**

**float fact(float n){**

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

**return 1;**

**else**

**return n\*fact(n-1);**

**}**

**int po(int x, int n){**

**int l=1;**

**for(int i=n;i>0;i--){**

**l=l\*x;**

**}**

**return l;**

**}**

**unsigned long int count(long int n){**

**short c=0;**

**for(;n;n=n/10,c=c+1);**

**return c;**

**}**

**unsigned long int sum(long int n){**

**short s=0;**

**while(n){**

**int a = n%10;**

**n=n/10;**

**s = s + a;**

**}**

**return s;**

**}**

**unsigned long int zfill(long int n){**

**short a=1;**

**for(;n;a=a\*10,n=n-1);**

**return a;**

**}**

**unsigned long int reverse(int n){**

**short m=0;**

**if(n<0)**

**n=n\*-1;**

**while(n){**

**int k=n%10;**

**n = n/10;**

**m = m +(k\*zfill(count(n)));**

**}**

**return m;**

**}**

**unsigned long int hcf(long int a, long int b){**

**while(b!=0){**

**int k=a%b;**

**if(k==0)**

**return b;**

**a=b;**

**b=k;**

**}**

**}**

**unsigned long int lcm(long int a ,long int b){**

**long int l=(a\*b)/hcf(a,b);**

**return l;**

**}**

**int palindrome(int n){**

**if(n==reverse(n))**

**return 1;**

**else**

**return 0;**

**}**

**int main()**

**{**

**int n = 111;**

**int x = 2;**

**long int a = 16;**

**long int b = 32;**

**printf("%d\n",nat(n));**

**printf("%lf\n",1/fact(n));**

**printf("%lf\n",po(x,n)/fact(n));**

**printf("%lu\n",count(n));**

**printf("%lu\n",sum(n));**

**printf("%lu\n",reverse(n));**

**printf("%lu\n",hcf(a,b));**

**printf("%lu\n",lcm(a,b));**

**if(palindrome(n)){**

**printf("It is palindrome");**

**}**

**else**

**printf("Not a palindrome");**

**return 0;**

**}**

**12.** **#include<stdio.h>**

**unsigned long int prime(long n){**

**if (n<=1){**

**return 0;**

**}**

**else {**

**int count=0;**

**for(int i=2;i<n;i++){**

**if(n%i==0){**

**count++;**

**}**

**}**

**if(count==0)**

**return 1;**

**else**

**return 0;**

**}**

**}**

**int main(){**

**int n = 61;**

**if(prime(n))**

**printf("It is a Prime");**

**else**

**printf("Not a Prime");**

**return 0;**

**}**

**10.** a)

**#include<stdio.h>**

**unsigned long int prime(long n){**

**if (n<=1){**

**return 0;**

**}**

**else {**

**int count=0;**

**for(int i=2;i<n;i++){**

**if(n%i==0){**

**count++;**

**}**

**}**

**if(count==0)**

**return 1;**

**else**

**return 0;**

**}**

**}**

**int primefact(int n){**

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

**if(prime(i) && n%i==0){**

**printf("%d\t",i);**

**}**

**}**

**}**

**int main(){**

**int n = 12;**

**primefact(n);**

**return 0;**

**}**

**b)** **#include<stdio.h>**

**void primefact(int n){**

**for(int i=2;n>1;i++){**

**while(n%i==0){**

**printf("%d\t",i);**

**n=n/i;**

**}**

**}**

**}**

**int main(){**

**int n=210;**

**primefact(n);**

**return 0;**

**}**

**13.**

**#include <stdio.h>**

**int perfect(int n){**

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

**if(n == i\*i){**

**return 1;**

**}**

**}**

**return 0;**

**}**

**int main()**

**{**

**int n=20;**

**if(perfect(n))**

**printf("It is a Perfect square");**

**else**

**printf("It is not a Perfect square");**

**return 0;**

**}**

**14.**

**#include <stdio.h>**

**int Armstrong(int n){**

**int sum=0;**

**while(n){**

**int k=n%10;**

**sum += (k\*k\*k);**

**n=n/10;**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n=120;**

**if(n==Armstrong(n))**

**printf("It is a Armstrong number");**

**else**

**printf("It is not a Armstrong number");**

**return 0;**

**}**

**15.**

**#include<stdio.h>**

**int fact(int n){**

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

**return 1;**

**else**

**return n\*fact(n-1);**

**}**

**double strong(int n){**

**int sum=0;**

**while(n){**

**int k=n%10;**

**sum += fact(k);**

**n=n/10;**

**}**

**return sum;**

**}**

**int main(){**

**int n = 40585;**

**if(n==strong(n))**

**printf("It is a Strong Number");**

**else**

**printf("It is not a Strong Number");**

**return 0;**

**}**

**16.**

**#include<stdio.h>**

**unsigned long PerfectNum(long n){**

**int sum=0;**

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

**if(n%i==0){**

**sum += i;**

**}**

**}**

**return sum;**

**}**

**int main(){**

**int n = 6;**

**if(n==PerfectNum(n))**

**printf("It is a Perfect Number");**

**else**

**printf("It is not a Perfect Number");**

**return 0;**

**}**

**17.**

**#include<stdio.h>**

**unsigned long Harshad(long n){**

**int sum=0;**

**while(n){**

**int k = n%10;**

**sum += k;**

**if(n%sum==0){**

**return 1;**

**}**

**else**

**return 0;**

**}**

**}**

**int main(){**

**int n = 13;**

**if(Harshad(n))**

**printf("It is a Harshad Number");**

**else**

**printf("It is not a Harshad Number");**

**return 0;**

**}**

**18.**

**#include<stdio.h>**

**unsigned long Abundant(long n){**

**int sum=0;**

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

**if(n%i==0){**

**sum += i;**

**}**

**}**

**return sum;**

**}**

**int main(){**

**int n = 12;**

**if(Abundant(n)>n)**

**printf("It is a Abundant Number");**

**else**

**printf("It is not a Abundant Number");**

**return 0;**

**}**

**19.**

**#include<stdio.h>**

**unsigned long Automorphic(long n){**

**int sq=n\*n;**

**while(n){**

**if(sq%10==n%10)**

**return 1;**

**n=n/10;**

**sq=sq/10;**

**}**

**return 0;**

**}**

**int main(){**

**int n = 6;**

**if(Automorphic(n))**

**printf("It is a Automorphic Number");**

**else**

**printf("It is not a Automorphic Number");**

**return 0;**

**}**

**20.**

**int reverse(int n){**

**int rev=0;**

**while(n){**

**rev = rev\*10 + (n%10);**

**n=n/10;**

**}**

**return rev;**

**}**

**int magic(int n){**

**int sum=0;**

**while(n){**

**int k=n%10;**

**sum += k;**

**n/=10;**

**}**

**int sq = sum \* reverse(sum);**

**return sq;**

**}**

**#include <stdio.h>**

**int main()**

**{**

**int n=1729;**

**if(magic(n)==n)**

**printf("It is a Magic Number");**

**else**

**printf("It is not a Magic number");**

**return 0;**

**}**

**22.**

**#include <stdio.h>**

**int Neon(int n){**

**int sq=n\*n,sum=0;**

**while(sq){**

**int k=sq%10;**

**sum += k;**

**sq=sq/10;**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n=45;**

**if(n==Neon(n))**

**printf("It is a Neon Number");**

**else**

**printf("It is not a Neon Number");**

**return 0;**

**}**

**23.**

**#include <stdio.h>**

**int Spy(int n){**

**int prod=1,sum=0;**

**while(n){**

**int k=n%10;**

**sum += k;**

**prod \*= k;**

**n=n/10;**

**}**

**if(sum==prod){**

**return 1;**

**}**

**return 0;**

**}**

**int main()**

**{**

**int n=123;**

**if(Spy(n))**

**printf("It is a Spy Number");**

**else**

**printf("It is not a Spy Number");**

**return 0;**

**}**

**24.**

**#include <stdio.h>**

**int Happy(int n){**

**int sum=0;**

**while(n>0 || sum>9){**

**if(n==0){**

**n=sum;**

**sum=0;**

**}**

**int k=n%10;**

**sum += k;**

**n=n/10;**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n=23;**

**if(Happy(n)==1)**

**printf("It is a Happy Number");**

**else**

**printf("It is not a Happy Number");**

**return 0;**

**}**

**25.**

**#include <stdio.h>**

**int Sunny(int n){**

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

**if((n+1) == i\*i){**

**return 1;**

**}**

**}**

**return 0;**

**}**

**int main()**

**{**

**int n=26;**

**if(Sunny(n))**

**printf("It is a Sunny Number");**

**else**

**printf("It is not a Sunny Number");**

**return 0;**

**}**

**26.**

**#include <stdio.h>**

**int power(int x,int n){**

**int l=1;**

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

**l=l\*x;**

**return l;**

**}**

**int count(int n){**

**int c=0;**

**while(n){**

**int k=n%10;**

**c+=1;**

**n=n/10;**

**}**

**return c;**

**}**

**int Disarium(int n){**

**int sum=0;**

**int c=count(n);**

**while(n){**

**int k=n%10;**

**sum += power(k,c--);**

**n=n/10;**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n=135;**

**if(Disarium(n)==n)**

**printf("It is a Disarium Number");**

**else**

**printf("It is not a Disarium number");**

**return 0;**

**//printf("%d\n",count(25));**

**}**

**27.**

**#include <stdio.h>**

**int Pronic(int n){**

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

**if(n==(i\*(i+1))){**

**return 1;**

**}**

**}**

**return 0;**

**}**

**int main()**

**{**

**int n=240;**

**if(Pronic(n))**

**printf("It is a Pronic Number");**

**else**

**printf("It is not a Pronic number");**

**return 0;**

**}**

**28.**

**#include <stdio.h>**

**int count(int n){**

**int c=0;**

**while(n){**

**int k=n%10;**

**c+=1;**

**n/=10;**

**}**

**return c;**

**}**

**int Trimorphic(int n){**

**int cu=n\*n\*n;**

**if(n%10==cu%10){**

**return 1;**

**}**

**n=n/10;**

**cu=cu/10;**

**return 0;**

**}**

**int main()**

**{**

**int n=24;**

**if(Trimorphic(n))**

**printf("It is a Trimorphic Number");**

**else**

**printf("It is not a Trimorphic number");**

**return 0;**

**}**

**29.**

**30.a)**

**#include<stdio.h>**

**unsigned long int count(long int n){**

**short c=0;**

**for(;n;n=n/10,c=c+1);**

**return c;**

**}**

**unsigned long int zfill(long int n){**

**short a=1;**

**for(;n;a=a\*10,n=n-1);**

**return a;**

**}**

**unsigned long int reverse(int n){**

**short m=0;**

**if(n<0)**

**n=n\*-1;**

**while(n){**

**int k=n%10;**

**n = n/10;**

**m = m +(k\*zfill(count(n)));**

**}**

**return m;**

**}**

**int palindrome(int n){**

**if(n==reverse(n))**

**return 1;**

**else**

**return 0;**

**}**

**int main()**

**{**

**int n1=1,n2=100;**

**for(int i=n1;i<=n2;i++){**

**if(palindrome(i)){**

**printf("%d\t",i);**

**}**

**}**

**return 0;**

**}**

**b)** **int reverse(int n){**

**int rev=0;**

**while(n){**

**rev = rev\*10 + (n%10);**

**n=n/10;**

**}**

**return rev;**

**}**

**#include <stdio.h>**

**int main()**

**{**

**int n1=1,n2=100;**

**for(int i=n1;i<=n2;i++){**

**if(reverse(i)==i)**

**printf("%d\t",i);**

**}**

**return 0;**

**}**

**31.**

**for(int i=n1;i<=n2;i++){**

**if(prime(i))printf("%d\t",i);**

**}**

**32. for(int i=n1;i<=n2;i++){**

**if(perfectsquare(i))printf("%d\t",i);**

**}**

**33. for(int i=n1;i<=n2;i++){**

**if(Armstrong(i))printf("%d\t",i);**

**}**

**34. for(int i=n1;i<=n2;i++){**

**if(Strong(i))printf("%d\t",i);**

**}**

**35. for(int i=n1;i<=n2;i++){**

**if(Perfect(i))printf("%d\t",i);**

**}**

**36. for(int i=n1;i<=n2;i++){**

**if(Harshad(i))printf("%d\t",i);**

**}**

**37. for(int i=n1;i<=n2;i++){**

**if(Abundant(i))printf("%d\t",i);**

**}**

**38. for(int i=n1;i<=n2;i++){**

**if(Automorphic(i))printf("%d\t",i);**

**}**

**39. for(int i=n1;i<=n2;i++){**

**if(Magic(i))printf("%d\t",i);**

**}**

**40. for(int i=n1;i<=n2;i++){**

**if(Neon(i))printf("%d\t",i);**

**}**

**41. for(int i=n1;i<=n2;i++){**

**if(Spy(i))printf("%d\t",i);**

**}**

**42. for(int i=n1;i<=n2;i++){**

**if(Happy(i))printf("%d\t",i);**

**}**

**43. for(int i=n1;i<=n2;i++){**

**if(Sunny(i))printf("%d\t",i);**

**}**

**44. for(int i=n1;i<=n2;i++){**

**if(Disarium(i))printf("%d\t",i);**

**}**

**45. for(int i=n1;i<=n2;i++){**

**if(Pronic(i))printf("%d\t",i);**

**}**

**46. for(int i=n1;i<=n2;i++){**

**if(Trimorphic(i))printf("%d\t",i);**

**}**

**47.**

**48.**

**49.** **long int n\_prime(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(prime(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**50.**

**long int n\_perfect(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(perfect(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**51.** **long int n\_arm(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Armstrong(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**52.** **long int n\_strong(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Strong(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**53.** **long int n\_perfect(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(perfect(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**54.** **long int n\_harshad(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Harshad(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**55.** **long int n\_abundant(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Abundant(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**56.** **long int n\_autmorphic(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Automorphic(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**57.** **long int n\_magic(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Magic(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**58.** **long int n\_neon(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Neon(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**59.** **long int n\_spy(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Spy(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**60.** **long int n\_happy(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Happy(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**61.** **long int n\_sunny(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Sunny(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**62.** **long int n\_disarium(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Disarium(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**63.** **long int n\_pronic(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Pronic(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**64.** **long int n\_trimorphic(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Trimorphic(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**65.** **long int n\_evil(int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(Evil(x)){**

**c++;**

**}**

**}**

**return x;**

**}**

**66.** **int rev\_rec(int n,int sum){**

**if(n==0){**

**return sum;**

**}**

**sum=sum\*10+(n%10);**

**return rev\_rec(n/10,sum);**

**}**

**int main(){**

**int n=1235,sum=0;**

**printf("%d\n",rev\_rec(n,sum));**

**return 0;**

**}**

**67.**

**#include<stdio.h>**

**int Genericroot(int n){**

**int sum=0;**

**while(n){**

**int k=n%10;**

**sum+=k;**

**n=n/10;**

**if(n==0 && sum>9){**

**n=sum;**

**sum=0;**

**}**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n=246;**

**printf("%d\n",Genericroot(n));**

**return 0;**

**}**

**68.**

**#include<stdio.h>**

**int zeroesandones(int n){**

**int c=0,t=0;**

**while(n){**

**int k=n%10;**

**if(k==0)**

**c=c+1;**

**printf("No of 0's->%d\n",c);**

**if(k==1)**

**t=t+1;**

**printf("No of 1's->%d\n",t);**

**n=n/10;**

**}**

**}**

**int main()**

**{**

**int n=101;**

**zeroesandones(n);**

**return 0;**

**}**

**69.**

**70.**

**#include<stdio.h>**

**int large(int n){**

**int sum=0,k;**

**while(n){**

**k=n%10;**

**if(k>sum){**

**sum=k;**

**}**

**n=n/10;**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n=143;**

**printf("%d\n",large(n));**

**return 0;**

**}**

**71.**

**#include<stdio.h>**

**int small(int n){**

**int sum=n%10,k;**

**while(n){**

**k=n%10;**

**if(k<sum){**

**sum=k;**

**}**

**n=n/10;**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n=523;**

**printf("%d\n",small(n));**

**return 0;**

**}**

**72.**

**#include<stdio.h>**

**int Amicable(int n, int m){**

**int sum=0;**

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

**if(m%i==0){**

**sum+=i;**

**}**

**}**

**if(n==sum)**

**return 1;**

**return 0;**

**}**

**int main()**

**{**

**int n=220,m=284;**

**if(Amicable(n,m))**

**printf("Amicable Pair");**

**else**

**printf("Not a Amicable Pair");**

**return 0;**

**}**

**73.**

**74.**

**75.**

**#include<stdio.h>**

**int evensandodds(int n){**

**int c=0,t=0;**

**while(n){**

**int k=n%10;**

**if(k%2==0)**

**c=c+1;**

**else**

**t=t+1;**

**n=n/10;**

**}**

**if(c!=0)**

**printf("No of evens->%d\n",c);**

**if(t!=0)**

**printf("No of odds->%d\n",t);**

**}**

**int main()**

**{**

**int n=1243;**

**evensandodds(n);**

**return 0;**

**}**

**76.**

**77.**

**#include<stdio.h>**

**int classifyADP(int n){**

**int sum=0;**

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

**if(n%i==0){**

**sum += i;**

**}**

**}**

**return sum;**

**}**

**int main()**

**{**

**int n1=1,n2=10000;**

**int a=0,d=0,p=0;**

**for(int i=n1;i<=n2;i++){**

**if(classifyADP(i)>i)a++;**

**if(classifyADP(i)<i)d++;**

**if(classifyADP(i)==i)p++;**

**}**

**if(a!=0)**

**printf("Abundant Count->%d\n",a);**

**if(d!=0)**

**printf("Dificient Count->%d\n",d);**

**if(p!=0)**

**printf("Perfect Count->%d\n",p);**

**return 0;**

**}**

**78.**

**79.**

**#include<stdio.h>**

**int count(int n){**

**int c=0;**

**while(n){**

**int k=n%10;**

**c++;**

**n=n/10;**

**}**

**return c;**

**}**

**int karprekar(int n){**

**int sq=n\*n;**

**int a=1,sum,k;**

**int c=count(n);**

**while(c){**

**a=a\*10;**

**c--;**

**}**

**k=sq%a;**

**sum = (sq/a)+k;**

**return sum;**

**}**

**int main()**

**{**

**int n=297,c=0;;**

**printf("%d\n",karprekar(n));**

**for(int i=1;i<1000;i++){**

**if(karprekar(i)==i){**

**printf("%d\t",i);**

**c=c+1;**

**}**

**}**

**if(c)**

**printf("\nCount->%d",c);**

**return 0;**

**}**

**82.**

**#include <stdio.h>**

**long int lucas\_numbers(long int n){**

**int f1=2,f2=1,c=0,f3;**

**printf("%d\t%d\t",f1,f2);**

**while(c<n){**

**f3=f1+f2;**

**f1=f2;**

**f2=f3;**

**printf("%d\t",f3);**

**c++;**

**}**

**return 0;**

**}**

**int main()**

**{**

**int n=10;**

**lucas\_numbers(n);**

**return 0;**

**}**

**83.**

**#include <stdio.h>**

**int fact(int n){**

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

**return 1;**

**else**

**return n\*fact(n-1);**

**}**

**long int catalan\_number(long int n){**

**long res;**

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

**res=fact(2\*i)/(fact(i+1)\*fact(i));**

**printf("%ld\t",res);**

**}**

**return 0;**

**}**

**int main()**

**{**

**int n=10;**

**catalan\_number(n);**

**return 0;**

**}**

**//84.print the first 10 happy numbers.**

**long int first\_happy\_numbers(long int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(happy(x)){**

**printf("%d\t",x);**

**c++;**

**}**

**}**

**return x;**

**}**

**//85.to check whether a given number is a happy number or unhappy number.**

**long int happy\_or\_unhappy(long int n){**

**if(happy(n)) printf("Happy");**

**else printf("unhappy");**

**return 0;**

**}**

**//86.Disarium number or unhappy number.**

**long int disarium\_or\_unhappy(long int n){**

**if(disarium(n)) printf("Disarium");**

**//else printf("NOt a Disarium");**

**else printf("unhappy");**

**return 0;**

**}**

**//87.Harshad Number or not.**

**long int harshad\_number(long int n){**

**if(harshad(n)) return 1;**

**return 0;**

**}**

**//88. Pronic Number or Heteromecic Number or not.**

**long int pronic\_number(long int n){**

**if(pronic(n)) return 1;**

**return 0;**

**}**

**//90.to check two numbers are Amicable numbers or not.**

**long int aamicable\_pairs(long int n,long int m){**

**if(n<0) n=n\*-1;**

**if(m<0) m=m\*-1;**

**long int sum1=0,sum2=0;**

**sum1=pdsum(n);**

**sum2=pdsum(m);**

**//printf("%d\t %d",sum1,sum2);**

**if(sum1==m && sum2==n) return 1;**

**return 0;**

**}**

**//91.to check if a given number is circular prime or not.**

**long int circular\_prime(long int n){**

**while(n>0){**

**int r=n%10;**

**if(r==2 || r==4 || r==6 || r==5 || r==8 || r==0) return 0;**

**n=n/10;**

**}**

**return 1;**

**}**

**//93. to check a number is a cyclic or not.**

**long int cyclic(long int n){**

**int c=1;**

**while(1){**

**int res=n\*c;**

**printf("%d\t",rev(res));**

**if(rev(res)==n) return 1;**

**c++;**

**if(c>=n) return 0;**

**}**

**return 0;**

**}**

**//94.to display first 10 Fermat numbers.**

**unsigned long long int fermat\_numbers( unsigned long long int n){**

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

**unsigned long long int res=power(2,power(2,i))+1;**

**printf("%llu\t",res);**

**}**

**return 0;**

**}**

**//96.to check if a number is Mersenne number or not.**

**long int mersenne\_number(long int n){**

**int c=1;**

**while(1){**

**if(n==(1<<c)-1) return 1;**

**c++;**

**if(c>=n) return 0;**

**}**

**return 0;**

**}**

**//97. all the narcissistic numbers between 1 and 1000.**

**long int narcissistic\_numbers\_range(long int n,long int m){**

**for(int i=n;i<=m;i++){**

**if(armstrong(i) || i<=9 && i>=2) printf("%d\t",i);**

**}**

**}**

**//98.to check whether a number is a Keith Number or not.**

**long int pell\_series(long int n){**

**int f1=0,f2=1,c=0,f3;**

**printf("%d\t%d\t",f1,f2);**

**while(c<n){**

**f3=f1+2\*f2;**

**f1=f2;**

**f2=f3;**

**printf("%d\t",f3);**

**c++;**

**}**

**return 0;**

**}**

**//100. to create the first twenty Hamming numbers.**

**long int hamming\_numbers(long int n){**

**int c=1,x=0;**

**while(c<=n){**

**x++;**

**if(ugly\_number(x)){**

**printf("%d\t",x);**

**c++;**

**}**

**}**

**return x;**

**}**

**//101. swap two number with using thrid vairable**

**long int swap(long int n,long int m){**

**n=n+m;**

**m=n-m;**

**n=n-m;**

**printf("%d\t%d\n",n,m);**

**return 0;**

**}**

**//102. power of number using recursion**

**long int rec\_power(int n,int m){**

**if(m==0) return 1;**

**else return n\*rec\_power(n,m-1);**

**}**

**//103 sum of digits of a number using recurssion.**

**long int rev\_sof(long int n){**

**if(n==0) return 0;**

**else return n%10+rev\_sof(n/10);**

**}**

**//104.to convert decimal number to binary using recurssion .**

**long int rec\_db(long int n,long int sum,long int a){**

**if(n==0) return sum;**

**else{**

**sum=sum+a\*(n%2);**

**a=a\*10;**

**return rec\_db(n/2,sum,a);**

**}**

**}**

**//105 A character is a vowel on consonant**

**char v\_or\_c(char ch){**

**char c,v;**

**if (ch=='a'|| ch =='e'|| ch =='i'|| ch =='o' || ch=='u') {**

**return 'v';**

**}**

**return 'c';**

**}**

**//106 A character is an alphabet or not**

**int is\_alphabet(char c) {**

**if ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z')) return 1;**

**return 0;**

**}**

**//106.1.ASCII value of a character**

**char ascii(char ch){**

**//print("%d\t",ch); // give the ascii number;**

**return ch;**

**}**

**//106.2 Uppercase lowercase or special character**

**char u\_or\_l\_or\_s(char c){**

**if (c >= 'A' && c<= 'Z') {**

**return 'U';**

**} else if (c>= 'a' && c<= 'z') {**

**return 'L';**

**} else if ((c>= '!' && c<= '/') || (c>= ':' && c<= '@') ||**

**(c>= '[' && c<='`') || (c>= '{' && c<= '~')) {**

**return 's';**

**}**

**return '0';**

**}**

**//106.3 Area of a circle**

**float Area\_of\_circle(long int n){**

**float res;**

**res= 3.14\*(n\*n);**

**return res;**

**}**

**//106.4.Friendly pair or not**

**long int friendly\_pair(long int n,long int m){**

**//printf("%f %f\n",fsum(n)/n,fsum(m)/m);**

**if(fsum(n)/n==fsum(m)/m) return 1;**

**return 0;**

**}**

**//106.5.Replace all zeros with 1 in a given integer**

**unsigned long long int zeros\_with\_1(long int n){**

**if(n<0)n=n\*-1;**

**int sum=0,a=1,r;**

**while(n>0){**

**r=n%10;**

**if(r==0) r=1;**

**sum=sum+a\*r;**

**a\*=10;**

**n=n/10;**

**}**

**return sum;**

**}**

**//106.6.Binary to decimal conversion**

**long int bd(long int n){**

**int sum=0,c=0;**

**while(n>0){**

**if(n%10 ==1) sum=sum+power(2,c);**

**c++;**

**n/=10;**

**}**

**return sum;**

**}**

**//106.7. Decimal to binary**

**long int db(long int n){**

**long sum=0,a=1;**

**for(;n;sum+=a\*(n%2),a\*=10,n/=2);**

**return sum;**

**}**

**int main(){**

**//int n=143256,m=1000;**

**int n=3;**

**printf("%d",bd(110));**

**}**

**Mathematical Algorithms:**

**#include<stdio.h>**

**#include <stdbool.h>**

**#include <string.h>**

**#include <math.h>**

**//41**

**long int factorial(long int n)**

**{**

**int m=1;**

**while(n>0)**

**{**

**m\*=n;**

**n=n-1; }**

**return m;**

**}**

**int trailing\_zeros(int n)**

**{**

**int a,b,c=0;**

**a=factorial(n);**

**while(a)**

**{**

**b=a%10;**

**if(b==0)**

**{**

**c=c+1;**

**a=a/10;**

**}**

**else**

**{**

**return c;**

**}**

**}**

**return c;**

**}**

**//42**

**unsigned long catalan(unsigned int n)**

**{**

**if (n <= 1)**

**{**

**return 1;**

**}**

**unsigned long result=0;**

**for (unsigned int i=0;i<n;i++)**

**{**

**result += catalan(i)\*catalan(n-1-i);**

**}**

**return result;**

**}**

**//43**

**int determineNumber(int input,double p1,double p2,double p3)**

**{**

**double cumulativeP1 = p1 \* 100;**

**double cumulativeP2 = (p1 + p2) \* 100;**

**int modInput = input % 100;**

**if (p1 + p2 + p3 != 1.0)**

**{**

**printf("Probabilities must sum to 1.\n");**

**return 1;**

**}**

**if (modInput < cumulativeP1)**

**{**

**return 1;**

**}**

**else if (modInput < cumulativeP2)**

**{**

**return 2;**

**}**

**else**

**{**

**return 3;**

**}**

**}**

**//44**

**int printExcelColumnName(unsigned int columnNumber)**

**{**

**if (columnNumber<=0)**

**{**

**return 0;**

**}**

**unsigned int remainder=(columnNumber-1)%26;**

**unsigned int nextColumnNumber=(columnNumber-1)/26;**

**printExcelColumnName(nextColumnNumber);**

**printf("%c",'A'+remainder);**

**}**

**//45**

**int strLength(const char\* str)**

**{**

**int length = 0;**

**while (str[length] != '\0')**

**{**

**length++;**

**}**

**return length;**

**}**

**void swap(char\* a, char\* b)**

**{**

**char temp = \*a;**

**\*a = \*b;**

**\*b = temp;**

**}**

**bool findNextGreater(char\* num)**

**{**

**int n = strLength(num);**

**int i;**

**for (i = n - 1; i > 0; i--)**

**{**

**if (num[i - 1] < num[i])**

**break;**

**}**

**if (i == 0)**

**return false;**

**int x = num[i - 1];**

**int smallest = i;**

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

**if (num[j] > x && num[j] < num[smallest])**

**smallest = j;**

**}**

**swap(&num[i - 1], &num[smallest]);**

**for (int j = i; j < n - 1; j++)**

**{**

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

**{**

**if (num[j] > num[k])**

**{**

**swap(&num[j], &num[k]);**

**}**

**}**

**}**

**return true;**

**}**

**//46**

**int countDecodings(const char \*digits)**

**{**

**int countPrev = 1;**

**int countCurrent = 1;**

**int i = 0;**

**while (digits[i] != '\0')**

**{**

**int count = 0;**

**if (digits[i] > '0')**

**{**

**count = countCurrent;**

**}**

**if (i > 0 && (digits[i - 1] == '1' || (digits[i - 1] == '2' && digits[i] < '7')))**

**{**

**count += countPrev;**

**}**

**countPrev = countCurrent;**

**countCurrent = count;**

**i++;**

**}**

**printf("%d\n",countCurrent);**

**}**

**//47**

**double calculateAngle(int hour, int minutes)**

**{**

**hour = hour % 12;**

**double hourAngle = (hour \* 30) + (minutes \* 0.5);**

**double minuteAngle = minutes \* 6;**

**double angle = fabs(hourAngle - minuteAngle);**

**return fmin(angle, 360 - angle);**

**}**

**//48**

**int countBinaryStrings(int n)**

**{**

**if (n == 1)**

**{**

**return 2;**

**}**

**int a = 1;**

**int b = 1;**

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

**{**

**int new\_a = a + b;**

**int new\_b = a;**

**a = new\_a;**

**b = new\_b;**

**}**

**return a + b;**

**}**

**//49**

**int printSmallestNumber(int n)**

**{**

**if (n < 10)**

**{**

**return n;**

**}**

**for (int i = 9; i > 1; i--)**

**{**

**if (n % i == 0)**

**{**

**int result = printSmallestNumber(n / i);**

**if (result != -1)**

**{**

**return result \* 10 + i;**

**}**

**}**

**}**

**}**

**int main()**

**{**

**long int a=872256,b=25;**

**double p1=0.5,p2=0.3,p3=0.2;**

**int input=42;**

**const char \*digits = "1234";**

**char num[] = "534976";printf("%d\n",trailing\_zeros(10));**

**printf("%d\n",catalan(5));**

**printf("%d\n",determineNumber(input,p1,p2,p3));**

**printf("%c\n",printExcelColumnName(28));**

**printf("%s\n",findNextGreater(num));**

**countDecodings(digits);**

**printf("%.2f\n",calculateAngle(3,30));**

**printf("%d\n",countBinaryStrings(3));**

**printf("%d\n",printSmallestNumber(100));**

**}**