Take N as input. Print the sum of its odd placed digits and sum of its even placed digits.

Input Format

Constraints

0 < N <= 1000000000

Output Format

Sample Input

2635

Sample Output

11

5

Explanation

5 is present at 1st position, 3 is present at 2nd position, 6 is present at 3rd position and 2 is present at 4th position.

Sum of odd placed digits on first line. 5 and 6 are placed at odd position. Hence odd place sum is 5+6=11

Sum of even placed digits on second line. 3 and 2 are placed at even position. Hence even place sum is 3+2=5

#include<iostream>

using namespace std;

int main() {

    int n;

    cin>>n;

    int flag=1;

    int rem;

    int odd=0,even=0;

    for(int x=n;x!=0;x/=10){

        rem=x%10;

        if(flag%2==0){

            even+=rem;

        }

        else{

            odd+=rem;

        }

        flag++;

    }

    cout<<odd<<endl<<even;

    return 0;

}

Take the following as input.  
A number  
A digit  
Write a function that returns the number of times digit is found in the number. Print the value returned.

Input Format

Integer (A number) Integer (A digit)

Constraints

0 <= N <= 1000000000 0 <= Digit <= 9

Output Format

Integer (count of times digit occurs in the number)

Sample Input

5433231

3

Sample Output

3

Explanation

The digit can be from 0 to 9. Assume decimal numbers.In the given case digit 3 is occurring 3 times in the given number.

#include<bits/stdc++.h>

using namespace std;

int main() {

    string x;

    cin>>x;

    int k;

    cin>>k;

int s = stoi(x);

    int cnt=0,last;

    while(s>0){

        last = s%10;

        if(last==k){

            cnt++;

        }

        s/=10;

    }

    cout<<cnt;

    return 0;

}

Take N as input, Calculate it's reverse also Print the reverse.

Input Format

Constraints

0 <= N <= 1000000000

Output Format

Sample Input

123456789

Sample Output

987654321

Explanation

You've to calculate the reverse in a number, not just print the reverse.

#include<iostream>

using namespace std;

int main() {

    int x;

    cin>>x;

    int rem,rev=0;

    while(x>0){

        rem=x%10;

        rev=rev\*10+rem;

        x/=10;

    }

    cout<<rev;

    return 0;

}

Take N (number in binary format). Write a function that converts it to decimal format and Print the value returned.

Input Format

Constraints

0 < N <= 1000000000

Output Format

Sample Input

101010

Sample Output

42

Explanation

For binary number fedcba , Decimal number = f \* 25 + e \* 24 + d \* 23 + …..+ a \* 20.

#include<bits/stdc++.h>

using namespace std;

int main() {

    int cnt=0,no,rem;

    cin>>no;

    int decimal=0;

    while(no>0){

        rem=no%10;

        decimal+=rem\*pow(2,cnt);

        cnt++;

        no/=10;

    }

    cout<<decimal;

    return 0;

}

Take the following as input.

A number (N1)  
A number (N2)  
Write a function which prints first N1 terms of the series 3n + 2 which are not multiples of N2.

Input Format

Constraints

0 < N1 < 100 0 < N2 < 100

Output Format

Sample Input

10

4

Sample Output

5

11

14

17

23

26

29

35

38

41

Explanation

The output will've N1 terms which are not divisible by N2.

#include<bits/stdc++.h>

using namespace std;

void print(int n1,int n2){

    int cnt=0;

    int x=1;

    while (cnt < n1) {

int term = (3 \* x) + 2;

if (term % n2 != 0) {

cout << term << endl;

cnt++;

}

x++;

}

}

int main() {

    int n1,n2;

    cin>>n1>>n2;

    print(n1,n2);

    return 0;

}

Take the following as input.

A number (N1)  
A number (N2)  
Write a function which returns the LCM of N1 and N2. Print the value returned.

Input Format

Constraints

0 < N1 < 1000000000  
0 < N2 < 1000000000

Output Format

Sample Input

4

6

Sample Output

12

Explanation

The smallest number that is divisible by both N1 and N2 is called the LCM of N1 and N2.

#include<iostream>

using namespace std;

int main() {

int x,y,lcm;

cin>>x>>y;

lcm=(x>y)?x:y;

while(1){

if(lcm%x==0&&lcm%y==0){

break;

}

lcm++;

}

cout<<lcm;

return 0;

}

Take N as input. Print Nth Fibonacci Number, given that the first two numbers in the Fibonacci Series are 0 and 1.

Input Format

Constraints

0 <= N <= 1000

Output Format

Sample Input

10

Sample Output

55

Explanation

The 0th fibonnaci is 0 and 1st fibonnaci is 1.

#include<iostream>

using namespace std;

int main() {

    int n;

    cin>>n;

    int a=0,b=1,c;

    for(int i=3;i<=n+1;i++){

        c=a+b;

        a=b;

        b=c;

    }

    cout<<c;

    return 0;

}

Take the following as input.

Minimum Fahrenheit value  
Maximum Fahrenheit value  
Step

Print as output the Celsius conversions. Use the formula C = (5/9)(F – 32) E.g. for an input of 0, 100 and 20 the output is  
0 -17  
20 -6  
40 4  
60 15  
80 26  
100 37

Input Format

The first line of the input contains an integer denoting the Minimum Fahrenheit value. The second line of the input contains an integer denoting the Maximum Fahrenheit value. The third line of the input contains an integer denoting the Step.

Constraints

0 < Min < 100  
Min < Max < 500  
0 < Step<150

Output Format

Print Fahrenheit and Celsius values separated by a tab. Each step should be printed in a new line.

Sample Input

0

100

20

Sample Output

0 -17

20 -6

40 4

60 15

80 26

100 37

Explanation

First number in every output line is fahrenheit, second number is celsius. The two numbers are separated by a tab.

#include<iostream>

using namespace std;

int main() {

    int x,y,z;

    cin>>x>>y>>z;

    int cel;

    for(int xx=x;xx<=y;xx+=z){

        cel=((xx-32)\*5)/9;

        cout<<xx<<"\t"<<cel<<endl;

    }

    return 0;

}

Take as input a number N, print "Prime" if it is prime if not Print "Not Prime".

Input Format

Constraints

2 < N <= 1000000000

Output Format

Sample Input

3

Sample Output

Prime

Explanation

The output is case specific

#include<iostream>

using namespace std;

int main() {

    int n;

    cin>>n;

    int flag=1;

    for(int i=2;i<=n/2;i++){

        if(n%i==0){

            flag=0;

            break;

        }

    }

    if(flag==1){

        cout<<"Prime";

    }

    else{

        cout<<"Not Prime";

    }

    return 0;

}

Take the following as input.

A number (N1)  
A number (N2)  
Write a function which returns the GCD of N1 and N2. Print the value returned.

Input Format

Two integers seperated by a new line.

Constraints

0 < N1 < 1000000000  
0 < N2 < 1000000000

Output Format

Output a single integer which is the GCD of the given integers.

Sample Input

16

24

Sample Output

8

Explanation

The largest number that divides both N1 and N2 is called the GCD of N1 and N2.

#include<iostream>

using namespace std;

int main() {

int x,y,gcd;

cin>>x>>y;

gcd=(x>y)?y:x;

while(1){

if(x%gcd==0&&y%gcd==0){

break;

}

gcd--;

}

cout<<gcd;

return 0;

}