```
for( 1 ; 2 ; 4 ){
        // 3
}
```

Q => Given a number N, print all the factors of N.

N= 12   [1, 2, 3, 4 5, 6 7 8 9 10 11 12]

divide N completly

st

varying variable.

N stp.

```
12 <= N % 1 == 0
12 <= N % 2 == 0   -> take loop
      N % 3 == 0           1 -> N
       .
       .
      N % 12          i == 0
loop (           )
  if( N % i == 0 )
         print i
```

N=48 → 1 2 3 4 6 8 12 16 24 48.

```
int N= sc.nextInt();        1 -> N
for(int i =1 ; i<=N; i++) {
        if (N % i ==0) {
              sop(i);
}       }
```

Q2) Given a number N, → count all the factor of N.

N= 12 → 1 2 3 4 6 12

→ 1 2 3 4 5 6 → 6//

+1 +1 +1 +1 +1

0+1 => 1

0+2 => 2

0+3 = 3 →

int count=0;

1 → N

loop

to check factor of N

if(   )

count = count + 1 → count++

Algorithm ←
↓
step by step process.

1) variable count = 0:
2) loop from 1 → N. → while
   → for
3) check if N % i == 0   only if
   true ↳ 2
4) count ++.
5) Print count

```
int count = 0;
for (int i = 1; i <= N; i++) {
        if (N % i == 0) {
                count ++;
        }
}
sop(count);
```

N = 19 → 1 . . . . 19 => 2.

N = 23 → 1 23 => 2

N = 13 → 1 13 => 2
↓
Prime numbers

Prime number will have only Two factor.

Q => Give a number N check if its a prime number

if prime → "Prime."

if not prime → "Not Prime."

Prime Number => divisible by 1 and itself.

Prime Number no of factors == 2.

1) repeat steps of above question.
2) check if count == 2
   ↳ true ("Prime")
   else ↳ fal. ("Not Prime").

```
1) variable count = 0:
                                i →
2) loop from 1 → N.
3) check if N % i == 0
        true └→
   4) count ++.
   5) check if count == 2
                   └→ "Prime"
               else └→ "Not Prime".
```

```
int count = 0;
for (int i = 1; i <= N; i++) {
        if (N % i == 0) {
             count ++;
        }
    }
    if (count == 2)
          sop("Prime");
       else sop ("Not prime");
```

x x
0 1 2 → is smallest

Break statement

↳ break the current loop

→ exit the loop.

```
int count = 0;
for (int i = 1; i <= N; i++) {
        if (N % i == 0) {
            count ++;
        }
        if (count > 2) {
            break;
        }
}
                3 == 2
if (count == 2)
        sop("Prime");
else sop("Not prime");
```

N = 1,20,000.

(120000)

Count	i	i <= N	N % i == 0	count++	count>2	break	i++
0	1	T	T	0→1	1 > 2	×	1 → 2
1	2	T	T	1 → 2	2 > 2	×	2→3
2	3	T	T	2→3	3 > 2 T	break ↓ exit loop	

count == 2

3 → = 2

O/p Not prime.

1 == 1 ⇒ T

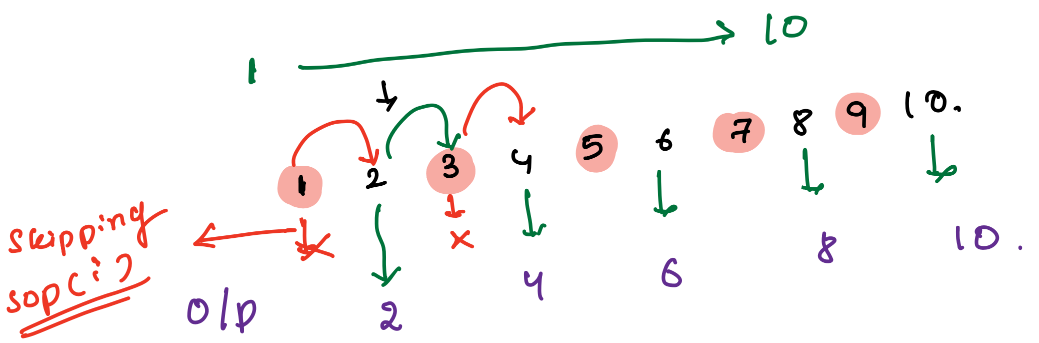
// **Bhoomika**

→ Refreshment cold, hot beverage.

11:11 → 11:16.

```
→ for(int i=1; i<=10; i++){
      if (i%2==1){
          continue;
      }
      sop(i+" ");
  }
```

→ skip below code.



i	i<=6	i%2==1	→ continue.	o/p	i++
1	T	T	✓	skip	1→2
2	T	F	✗	2	2→3
3	T	T	✓	skip	3→4
4	T	F	✗	4	4→5
5	T	T	✓	skip	5→6
6	T	F	✗	6	6→7
7	F → exit loop;				

Break :- exit the current loop.

Continue :- exit the current iteration.

/ will continue to next iteration.

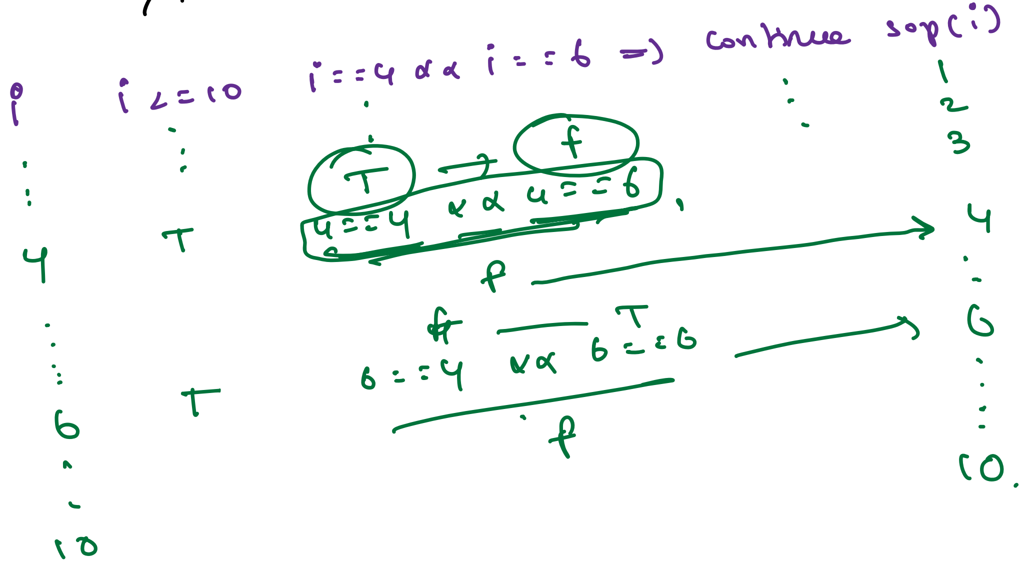
i → 1 - ~~2~~ - 3 - ~~4~~ - 5

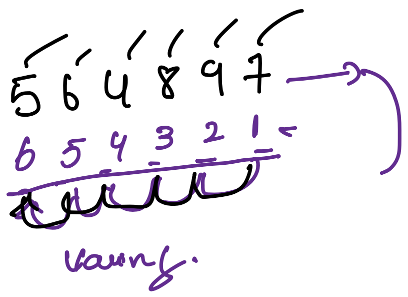
C C

```
for( int i = 1 ; i <= 10 ; i++) {
  → if ( i == 4 && i == 6 ) {  → false always
       continue;               i→ 1-2-3->4.
  }
  sopln(i);
}.
```

i == 4

i == 6.





sum_even_i;
sum_odd_i;

```
int index = 1;

while (num > 0) {
    int d = num % 10;
     if (index == 1) {
            sum_even_i += d;
       }
      else {
            sum_odd_i += d;
          }
    index++;
    num = num/10
 }

   sop(sum_even_i);
   sop(sum_odd_i);
```

$153 = 1^3 + 5^3 + 3^3$

```
i = 1
while ( i <= N) {
     int num = N;
```

$N = 143$

```
sum= 0;
while(num> 0 ) { → num/10
    d= n%10.                      1
    sum += ( d*d*d);              1
                                  0
    num= num/10
}
if( N == sum) {
    print(" Armstrong" );
}
i++;
}
1 → N
```

https://www.interviewbit.com/snippet/bd48d023f144cdaa6ac8/