Question01:

Write a function to take temperature and conversion unit such that it converts the temperature to the respective conversion unit from either Celsius or Kelvin. Assume temperature is in Celsius if conversion unit is Kelvin. Assume temperature is in Kelvin if conversion unit is Celsius. Input: 35, K Output: 308 Input: 308, C Output: 35

```
JS demo.js ×
JS demo.js > func
const prompt = require("prompt-sync")();

function func(temprature, unit){
    let value1 = unit === 'C';
    let value2 = value1 ? temprature - 273 : temprature + 273;
    // let value3 = value1 ? 'C' : 'K';
    // let value4 = value1 ? `${value1} - 273` : `${value1} + 273`;
    return `${value2}`;
}
console.log(func(35, 'K'));
console.log(func(308, 'C'));
```

Question02:

Given an integer “n” perform the following conditional actions. If n is odd print “weird”. If n is even and in this inclusive range of 2 to 5 print “not weird”. If n is even and in this inclusive range of 6 to 20 then print “weird”. If n is even and greater than 20 print or “not weird”.

```
JS demo.js ×
JS demo.js > ...
const prompt = require("prompt-sync")();
let n = parseInt(prompt("Enter Input:"));
if(n % 2 != 0){
    console.log("weird");
}else if(n >= 2 & n <= 5){
    console.log("not weird");
}else if(n >= 6 & n <= 20){
    console.log("weird");
}else{
    console.log("not weird");
}
```

```
powershell ×
PS E:\HTML_CSS_JS> node demo.js
Enter Input:12
weird
PS E:\HTML_CSS_JS> node demo.js
Enter Input:5
weird
PS E:\HTML_CSS_JS> node demo.js
Enter Input:36
not weird
PS E:\HTML_CSS_JS> node demo.js
Enter Input:16
weird
PS E:\HTML_CSS_JS> node demo.js
Enter Input:0
not weird
PS E:\HTML_CSS_JS>
```

Question03:

there are three line of output the first contain an integer, second contain a double, third contain a string. the output must be first line print string, second line print double, third line print int.

```js
const prompt = require("prompt-sync")();
let num1 = parseInt(prompt("Enter Num1 : "));
let num2 = parseFloat(prompt("Enter Num2 : "));
let str1 = prompt("Enter String Value : ");
console.log(`String : ${str1}`);
console.log(`Double : ${num2}`);
console.log(`Integer : ${num1}`);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Num1 : 152
Enter Num2 : 125.33
Enter String Value : Atish Kumar Sahu
String : Atish Kumar Sahu
Double : 125.33
Integer : 152
PS E:\HTML_CSS_JS> node demo.js
Enter Num1 : 1000
Enter Num2 : 9510.2465
Enter String Value : Lipun Kumar Sahu
String : Lipun Kumar Sahu
Double : 9510.2465
Integer : 1000
PS E:\HTML_CSS_JS>
```

Question04:

Input: Every line of input will contain a string followed by an integer. Each string will have a maximum of 10 alphabetic characters and each integer will be in the inclusive range from 0 to 999.

Output: In each line of output there should be two columns. The first column contain the string and left justified using exactly 15 characters. The second column contain the integer, expressed in exactly 3 digits. If the original input has less than 3 digit you must add your output leading digits with zeroes.

```js
const prompt = require("prompt-sync")();
let range = parseInt(prompt("Enter Range : "));
for (let i = 0; i <= range; i++) {
    let str = prompt("str : ");
    let num = parseInt(prompt("Num Value : "));
    let str1 = str;
    let temp = num.toString();
    for (let j = str.length; j <= 14; j++) {
        str1 += " ";
    }
    if (temp.length === 3) {
        console.log(str1 + temp);
    } else if (temp.length === 2) {
        console.log(str + "0" + temp);
    } else if (temp.length === 1) {
        console.log(str + "00" + temp);
    }
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Range : 2
str : Atish
Num Value : 9
Atish009
str : Lipun
Num Value : 15
Lipun015
str : Mritunjay
Num Value : 100
Mritunjay       100
PS E:\HTML_CSS_JS>
```

Question05:

You are given q queries in the form of a, b, and n for each query, print the series corresponding to the given a, b, and n values as a single line of n space-separated integers.

$$(a + 2^0 \cdot b), (a + 2^0 \cdot b + 2^1 \cdot b), \ldots, (a + 2^0 \cdot b + 2^1 \cdot b + \ldots + 2^{n-1} \cdot b)$$

```js
const prompt = require("prompt-sync")();
let range = parseInt(prompt("Enter Range : "));
for(let i = 0; i < range; i++){
    let a  = parseInt(prompt("Enter num1 : "));
    let b = parseInt(prompt("Enter num2 : "));
    let c = parseInt(prompt("Enter num3 : "));
    for(let j = 0; j < c; j++){
        a += b * Math.pow(2, j);
        console.log(a + " ");
    }
    console.log();
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Range : 1
Enter num1 : 200
Enter num2 : 100
Enter num3 : 5
300
500
900
1700
3300

PS E:\HTML_CSS_JS>
```

Question06:

Input an integer denoting the number of test cases. Each test cases is comprised of a single line with an integer which can be arbitrary large or small. For each input variable and appropriate primitive datatype. You must determine if the given primitives are capable of storing or not.

```js
const prompt = require("prompt-sync")();
let range = parseInt(prompt("Enter Range : "));
for(let i = 0; i < range; i++){
    try{
        let Long = prompt("Enter Input1 : ");
        console.log(Long+" can be fitted in : ");
        if(Long >= -128 && Long <= 127)
            console.log(" Byte");
        if(Long >= -32768 && Long <= 32767)
            console.log(" Short");
        if(Long >= -2147483648 && Long <= 2147483647)
            console.log(" Int");
    }catch(Exception ){
        console.log("Can't be Fitted AnyWhere");
    }
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Range : 5
Enter Input1 : 2000
2000 can be fitted in :
 Short
 Int
Enter Input1 :
```

Question07:

In a premier championship series of sports car racing initially the 1st car is ahead of the 2nd car by x meters. After that in every second the 1st car moves ahead by n1 meter and the 2nd car moves ahead n2 meter (in the same direction). the task is to print the number of seconds after which the 2nd car crosses the 1st car. if it is impossible to do so, then print -1.

Input:		Input:	
3	value of n1	5	Value of n1
4	value of n2	4	Value of n2
1	value of X	1	Value of X
Output: 2		Output: -1	

```js
const prompt = require("prompt-sync")();
let ip1 = parseInt(prompt("enter input1 : "));
let ip2 = parseInt(prompt("enter input2 : "));
let ip3 = parseInt(prompt("enter input3 : "));
if(ip2 <= ip1){
    console.log(-1);
}else{
    let time = Math.floor(ip3 / (ip2 - ip1));
    console.log(time + 1);
}
```

```
PS E:\HTML_CSS_JS> node demo.js
enter input1 : 4
enter input2 : 3
enter input3 : 2
-1
PS E:\HTML_CSS_JS> node demo.js
enter input1 : 4
enter input2 : 8
enter input3 : 2
1
PS E:\HTML_CSS_JS> node demo.js
enter input1 : 4
enter input2 : 8
enter input3 : 1
1
PS E:\HTML_CSS_JS>
```

Question08:

Return addition of numbers that are divisible by 7 in the given range.

Input: start = 1, end = 20 Output: 21

Explanation: Here, the numbers divisible by 7 in the range are 7 and 14, and their sum is 21.

input: start = 10, end = 30 Output: 63

Here, the numbers divisible by 7 in the range are 14, 21, and 28, and their sum is 63.

```js
const prompt = require("prompt-sync")();
let ip1 = parseInt(prompt("Enter Input : "));
let ip2 = parseInt(prompt("Enter Input2 : "));
let count = 0;
for(let i = ip1; i <= ip2; i++){
    if(i % 7 == 0){
        count += i;
    }
}
console.log(count);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 1
Enter Input2 : 20
21
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 10
Enter Input2 : 30
63
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 100
Enter Input2 : 800
45150
PS E:\HTML_CSS_JS>
```

Question09:

Convert Decimal To Binary, Octal, Hexadecimal Number

```js
const prompt = require("prompt-sync")();
let num = parseInt(prompt("Enter Input : "));
console.log(`Binary : ${num.toString(2)}`);
console.log(`Octal  : ${num.toString(8)}`);
console.log(`Hexadecimal : ${num.toString(16).toUpperCase()}`);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 11
Binary : 1011
Octal  : 13
Hexadecimal : B
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 199
Binary : 11000111
Octal  : 307
Hexadecimal : C7
PS E:\HTML_CSS_JS>
```

Question10:

Convert Binary To Decimal, Octal, Hexadecimal Number

```js
const prompt = require("prompt-sync")();
let num = prompt("Enter Binary Number : ");
console.log(`Decimal : ${parseInt(num, 2)}`);
let decimal = parseInt(num, 2);
console.log(`Octal : ${decimal.toString(8)}`);
console.log(`Hexadecimal : ${decimal.toString(16).toUpperCase()}`);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Binary Number : 11010
Decimal : 26
Octal : 32
Hexadecimal : 1A
PS E:\HTML_CSS_JS> node demo.js
Enter Binary Number : 110011001
Decimal : 409
Octal : 631
Hexadecimal : 199
PS E:\HTML_CSS_JS>
```

Question11:

Convert Octal Number To Binary, Decimal, Hexadecimal Number

```js
const prompt = require("prompt-sync")();
let num = prompt("Enter Octal Number : ");
let decimal = parseInt(num, 8);
console.log(`Binary : ${decimal.toString(2)}`);
console.log(`Decimal : ${parseInt(num, 8)}`);
console.log(`Hexadecimal : ${decimal.toString(16).toUpperCase()}`);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Octal Number : 124
Binary : 1010100
Decimal : 84
Hexadecimal : 54
PS E:\HTML_CSS_JS> node demo.js
Enter Octal Number : 246
Binary : 10100110
Decimal : 166
Hexadecimal : A6
PS E:\HTML_CSS_JS>
```

Question12:

Hexadecimal To Binary, Octal, Decimal Number

```js
const prompt = require("prompt-sync")();
let num = prompt("Enter Hexadecimal Number : ");
let decimal = parseInt(num, 16);
console.log(`Binary : ${decimal.toString(2)}`);
console.log(`Decimal : ${parseInt(num, 16)}`);
console.log(`Octal : ${decimal.toString(8)}`);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Hexadecimal Number : A12
Binary : 101000010010
Decimal : 2578
Octal : 5022
PS E:\HTML_CSS_JS> node demo.js
Enter Hexadecimal Number : BB
Binary : 10111011
Decimal : 187
Octal : 273
PS E:\HTML_CSS_JS>
```

Question13:

Write a function that takes an unsigned integer and returns the number of '1' bits it has (also known as the Hamming weight).

Input: n = 00000000000000000000000000001011 decimal = 3 Output: 3

Explanation: The input binary string 00000000000000000000000000001011 has a total of three '1' bits.

Input: n = 00000000000000000000000010000000 decimal = 128 Output: 1

Explanation: The input binary string 00000000000000000000000010000000 has a total of one '1' bit.

```
JS demo.js ×
JS demo.js > ...
1  const prompt = require("prompt-sync")();
2  let input = parseInt(prompt("Enter Input : "));
3  let binary = input.toString(2);
4  count = 0;
5  for(let i = 0; i < binary.length; i++){
6      if(binary.charAt(i) == '1'){
7          count++;
8      }
9  }
10 console.log(`Output : ${count}`);
```

```
powershell ×
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 3
Output : 2
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 7
Output : 3
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 128
Output : 1
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 1314
Output : 4
PS E:\HTML_CSS_JS>
```

Question14:

Given a positive integer, check whether it has alternating bits: namely, if two adjacent bits will always have different values. Input: n = 5 Output: true Explain: The binary representation of 5 is: 101

```
JS demo.js ×
JS demo.js > ...
1  const prompt = require("prompt-sync")();
2  let input = parseInt(prompt("Enter Input: "));
3  let prev = input % 2;
4  input = Math.floor(input / 2);
5  let consecutiveEqualBits = false;
6  while (input > 0) {
7      let cur = input % 2;
8      if (cur == prev) {
9          consecutiveEqualBits = true;
10         break;
11     }
12     prev = cur;
13     input = Math.floor(input / 2);
14 }
15 if (consecutiveEqualBits) {
16     console.log(false);
17 } else {
18     console.log(true);
19 }
```

```
powershell ×
Enter Input: 7
false
PS E:\HTML_CSS_JS> node demo.js
Enter Input: 5
true
PS E:\HTML_CSS_JS> node demo.js
Enter Input: 1365
true
PS E:\HTML_CSS_JS>
```

Question15:

47. Given two integers dividend and divisor, divide two integers without using multiplication, division, and mod operator. The integer division should truncate toward zero, which means losing its fractional part. For example, 8.345 would be truncated to 8, and -2.7335 would be truncated to -2.

Input: dividend = 10, divisor = 3 Output: 3

Input: dividend = 7, divisor = -3 Output: -2

```
demo.js
const prompt = require("prompt-sync")();
let ip1 = parseInt(prompt("Enter Input 1: "));
let ip2 = parseInt(prompt("Enter Input 2: "));
let ans = ip1 / ip2;
if (ans > Math.pow(2, 31) - 1) {
    ans = Math.pow(2, 31) - 1;
}
if (ans < -Math.pow(2, 31)) {
    ans = -Math.pow(2, 31);
}
ans = Math.floor(ans);
console.log(ans);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 10
Enter Input : 3
3
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 7
Enter Input : -3
-3
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 5
Enter Input : 3
1
PS E:\HTML_CSS_JS> node demo.js
Enter Input 1: 7
Enter Input 2: -3
-3
PS E:\HTML_CSS_JS> node demo.js
Enter Input 1: -2147483648
Enter Input 2: -1
2147483647
PS E:\HTML_CSS_JS>
```

Question16:

You have numCoconuts coconut shells that are initially full of water. You can exchange numExchange empty coconut shells from the market with one full coconut shell of water. The operation of drinking a full of water turns it into an empty coconut shell. Given the two integers numCoconuts and numExchange return the maximum number of coconut shells of water you can drink. Input: numCoconuts = 9 numExchange = 3 output: 13

```
demo.js
const prompt = require("prompt-sync")();
let ip1 = parseInt(prompt("Enter Input 1: "));
let ip2 = parseInt(prompt("Enter Input 2: "));
let total = ip1, empty = ip1;
while(empty >= ip2){
    let temp = Math.floor(empty / ip2);
    total += temp;
    empty = temp + empty % ip2;
}
console.log(total);
```

```
powershell
PS E:\HTML_CSS_JS> node demo.js
Enter Input 1: 9
Enter Input 2: 3
13
PS E:\HTML_CSS_JS>
```

Question17:

there are many needy people who need some money are standing in a row. the person who stands at ith position need i * i rs money. you have X rs. and you want to fulfil the needs of people. when you donate the money to any people it is deducted from x. you have to find how many peoples can fulfil their needs by your money.

input: 14 output: 3 input: 35 output: 4

```js
const prompt = require("prompt-sync")();
let ip = parseInt(prompt("Enter Input : "));
if(ip <= 0){
    console.log(0);
}
let count = 0; i = 1;
while(ip >= i * i){
    ip -= i * i;
    count++;
    i++;
}
console.log(count);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 14
3
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 35
4
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 50
4
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 100
6
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 70
5
PS E:\HTML_CSS_JS>
```

Question18:

Compute the natural logarithm of 2 by adding up to n terms in the series 1 – ½ + 1/3 – ¼ + 1/5 - ..... 1/n where n is a positive integer and input by user.

```js
const prompt = require("prompt-sync")();
let sum = 0;
let ip = parseInt(prompt("Enter Input : "));
for(let i = 1; i <= ip; i++){
    if(i % 2 == 1)
        sum += 1.0 / i;
    else
        sum -= 1.0 / i;
}
console.log("Output : "+sum);
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 5
Output : 0.7833333333333332
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 2
Output : 0.5
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 10
Output : 0.6456349206349207
PS E:\HTML_CSS_JS> node demo.js
Enter Input : 7
Output : 0.7595238095238095
PS E:\HTML_CSS_JS>
```

Question19:

Raman needs to work on a project for the next N hours. He has a work plan represented as a binary string S of length N, where 1 indicates work time and 0 indicates free time. Raman wants to take naps during his free time, and he needs at least K consecutive hours of free time to take a nap. The goal is to find the maximum number of naps Raman can take during the given N hours. Your Task -You don't need to read input or print anything. Complete the function sleptwithRaman() which takes two space-separated integers N and K and a binary string S of length N. Return a single integer, which represents the maximum number of naps Raman can take.

Input: 3<->4 1<->1010<->4 2<->0100<->11 3<->00100000001 Output: 2 1 2

Test Case 1: Raman can take two naps starting from the 2nd and 4th hours respectively.

Test Case 2: Raman can take a nap starting from the 3rd hour.

Test Case 3: Raman can take one nap starting from the 5th hour and another starting from the 8th hour. solve the problem in java

demo.js

```
const prompt = require("prompt-sync")();
function func(N, K, S){
    let max = 0, time = 0;
    for(let i = 0; i < N; i++){
        if(S.charAt(i) == '0')
            time++;
        else
            time = 0;
        if(time >= K){
            max++;
            time = 0;
        }
    }
    return max;
}
let T = parseInt(prompt("Enter T : "));
while(T-- > 0){
    let N = parseInt(prompt("Enter N : "));
    let K = parseInt(prompt("Enter K : "));
    let S = prompt("Enter S : ");
    let op = func(N, K, S);
    console.log(op);
}
```

```
PS E:\HTML_CSS_JS> node demo.js
Enter T : 3
Enter N : 4
Enter K : 1
Enter S : 1010
2
Enter N : 4
Enter K : 2
Enter S : 0100
1
Enter N : 11
Enter K : 3
Enter S : 00100000001
2
PS E:\HTML_CSS_JS>
```