DAY-1/TASK-1

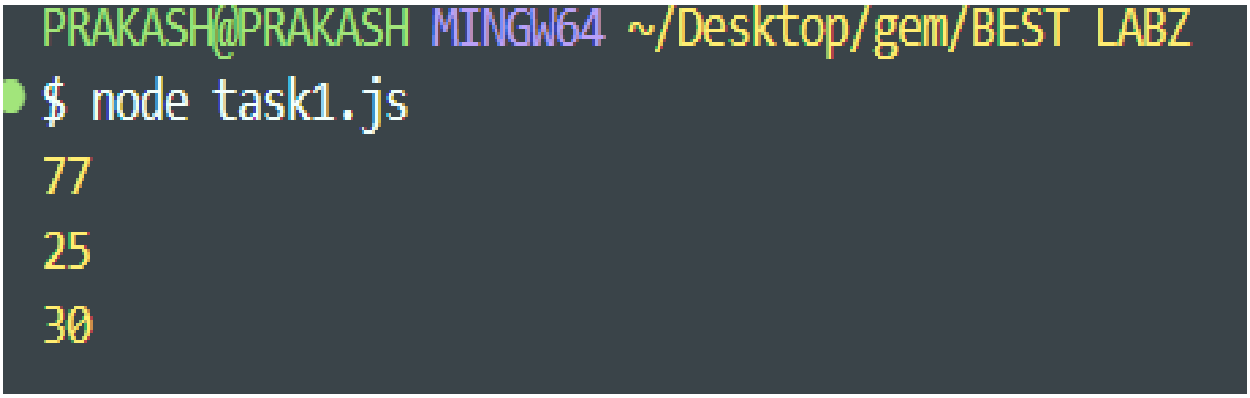
```
//ADD TWO NUMBERS                                                    Q-1
  let i = 5;
  let f = 1;
  let Summ = i + f;
  console.log(Summ);

//SUBTRACT TWO NUMBERS                                               Q-2
 let x = 10;
 let y = 5;
 let sum = 10-5;
 console.log(sum);

 //MULTIPLE TWO NUMBER                                               Q-3
let num1 = 5;
let num2 = 8;
let product = num1 * num2;
console.log(product);


  //Divide two numbers                                               Q-4
let numerator = 10;
let denominator = 2;
let result = numerator / denominator;
console.log(result);
```

OUTPUT :

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task1.js
6
5
40
5
```

```
  // CONVERT CEISIUS TO FUHRENHEIT                          Q-5
let celsiusTemperature = 25;
let fahrenheitTemperature = (celsiusTemperature * 9/5) + 32;
console.log(fahrenheitTemperature);




  // CONVERT FUHRENHEIT CEISIUS to CEISIUS                  Q-6
  let fahrenheiTemperature = 77;
let celsiuTemperature = (fahrenheiTemperature - 32) * 5 / 9;
console.log(celsiuTemperature);



//ADD THREE NUMBER                                          Q-7
function addThreeNumbers(a, b, c) {
  return a + b + c;
}
let result = addThreeNumbers(5, 10, 15);
console.log(result);
```

OUTPUT:



```
//SUBTRACT THREE NUMBER                                    Q-8
function subtractThreeNumbers(a, b, c) {
  return a - b - c;
}
let D = subtractThreeNumbers(20, 5, 3);
console.log(D);



// MULTIPLE THREE NUMBERS                                Q-9
function multiplyThreeNumbers(a, b, c) {
  return a * b * c;
}
let A = multiplyThreeNumbers(4, 5, 6);
console.log(A);



// DIVIDE THE THREE NUMBER                                  Q-10
function divideThreeNumbers(a, b, c) {
  return a / b / c;
}
let G = divideThreeNumbers(100, 5, 2);
console.log(G);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task1.js
12
120
10
```

DAY-2/TASK-2

```
// PRINT A SINGLE DICE                                          Q-1

function rollDice() {
    const result = Math.floor(Math.random() * 6) + 1;
    console.log(`You rolled a ${result}`);
}
rollDice();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
You rolled a 1

PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
You rolled a 6
```

```
// PRINT TWO DICE                                               Q-2
function rollDice() {

    function rollSingleDie() {
        return Math.floor(Math.random() * 6) + 1;
    }
    const die1 = rollSingleDie();
    const die2 = rollSingleDie();
    console.log(`You rolled a ${die1} and a ${die2}`);
}
rollDice();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
You rolled a 6 and a 3
```

```
//ADD TWO DICE                                          Q-3
function rollDice() {
    function rollSingleDie() {
        return Math.floor(Math.random() * 6) + 1;
    }
    const die1 = rollSingleDie();
    const die2 = rollSingleDie();
    const sum = die1 + die2;
    console.log(`You rolled a ${die1} and a ${die2}. The sum is ${sum}.`);
}
rollDice();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
You rolled a 3 and a 3. The sum is 6.
```

```
//PICK 5Random two digit number                          Q-4
function grg(count) {
    const numbers = [];
    for (let i = 0; i < count; i++) {

        const number = Math.floor(Math.random() * 90) + 10;
        numbers.push(number);
    }
    return numbers;
}
const randomNumbers = grg(5);
console.log('Random two-digit numbers:', randomNumbers);
```

Output:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
Random two-digit numbers: [ 87, 86, 64, 91, 57 ]
```

```
//PICK 5Random two digit number & GET AVERAGE NUM (TOTAL =5)   Q-5
const generateRandomTwoDigitNumbers = (count) =>
    Array.from({ length: count }, () => Math.floor(Math.random() * 90) + 10);
const calculateAverage = (numbers) =>
    numbers.reduce((sum, num) => sum + num, 0) / numbers.length;
const randomNumbers = generateRandomTwoDigitNumbers(5);
const average = calculateAverage(randomNumbers);
console.log('Random two-digit numbers:', randomNumbers);
console.log('Average:', average.toFixed(2));
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
Random two-digit numbers: [ 80, 72, 31, 84, 97 ]
Average: 72.80
```

```
//ADD 5Random THREE digit number & GET AVERAGE NUM (TOTAL =5)        Q-6
const generateRandomThreeDigitNumbers = (count) =>
    Array.from({ length: count }, () => Math.floor(Math.random() * 900) + 100);

const calculateAverage = (numbers) =>
    numbers.reduce((sum, num) => sum + num, 0) / numbers.length;

const randomNumbers = generateRandomThreeDigitNumbers(5);
const average = calculateAverage(randomNumbers);

console.log('Random three-digit numbers:', randomNumbers);
console.log('Average:', average.toFixed(2));
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
Random three-digit numbers: [ 242, 845, 640, 638, 481 ]
Average: 569.20
```

```
//ADD 5 TWO DIGIT RANDOM NUMBER USING FOR LOOP => INCREMENT TYPE          Q-6
```

```
function generateAndCalculateAverage() {
    const count = 5;
    const numbers = [];
    for (let i = 0; i < count; i++) {
       const number = Math.floor(Math.random() * 90) + 10;
        numbers.push(number);
    }
    const total = numbers.reduce((sum, num) => sum + num, 0);
    const average = total / numbers.length;
    console.log('Random two-digit numbers:', numbers);
    console.log('Average:', average.toFixed(2));
}
generateAndCalculateAverage();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
Random two-digit numbers: [ 34, 35, 12, 80, 41 ]
Average: 40.40
```

Q-7(II)

```
//ADD 5 TWO DIGIT RANDOM NUMBER USING FOR LOOP =>DECREMENT TYPE
function generateAndCalculateAverage() {
    const count = 5;
    const numbers = [];
    for (let i = count; i > 0; i--) {
        const number = Math.floor(Math.random() * 90) + 10;
        numbers.push(number);
    }
    const total = numbers.reduce((sum, num) => sum + num, 0);
    const average = total / numbers.length;
    console.log('Random two-digit numbers:', numbers);
    console.log('Average:', average.toFixed(2));
}
generateAndCalculateAverage();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
Random two-digit numbers: [ 42, 79, 90, 45, 57 ]
Average: 62.60
```

Q-8

```
//MULTIPLY 5 TWO DIGIT RANDOM NUMBER USING FOR LOOP
function multiplyRandomTwoDigitNumbers() {
    const count = 5;
    let product = 1;
    for (let i = 0; i < count; i++) {
        const number = Math.floor(Math.random() * 90) + 10;
        product *= number;
    }
    console.log('Product of the random two-digit numbers:',
product);
}
multiplyRandomTwoDigitNumbers();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task2.js
Product of the random two-digit numbers: 200204928
```

DAY-3/TASK-3

```
// SUM OF N DIGITS                                    Q-1
var n=10
var result = 0;
for (var i=1;i <=n;i++){
    result = result + i ;
    console.log(result);
}
```

OUTPUT:

```
$ node task3.js
1
3
6
10
15
21
28
36
45
```

```
//MULTIPLY OF N DIGITS                                Q-2
var n = 10;
var product = 1;
for (var i = 1; i <= n; i++) {
    product *= i;
}
console.log('Product of numbers from 1 to', n, 'is:', product);
```

OUTPUT:

```
$ node task3.js
1
3
6
10
15
21
28
36
45
55
Product of numbers from 1 to 10 is: 3628800
```

```
// SUM OF N TWO DIGITS EG: N = 123456=>12+34+56=102        Q-3
function sumOfTwoDigitGroups(n) {
    const str = n.toString();
    let sum = 0;
    for (let i = 0; i < str.length; i += 2) {
        const pair = str.substring(i, i + 2);
        sum += parseInt(pair, 10);
    }
    console.log('Sum of two-digit groups:', sum);
}
sumOfTwoDigitGroups(123456);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task3.js
Sum of two-digit groups: 102
```

```
//multiply of n two digits                                  Q-4
let n = 123456;
let nstr = n.toString();
let product = 1;
for (let i = 0; i < nstr.length; i += 2) {
    let pair = nstr.substring(i, i + 2);
    let num = parseInt(pair, 10);
    product *= num;
}
console.log(product);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task3.js
22848
```

```
//N FACTORIAL USING FOR LOOP =>ICREMENT TYPE=> N!=5!=>5*4*3*2*1=120  Q-5(i)
function factorial(n) {
    let result = 1;
    for (let i = 1; i <= n; i++) {
        result *= i;
    }

    return result;
}
const n = 5;
console.log(`Factorial of ${n} is: ${factorial(n)}`);
```

Output:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task3.js
Factorial of 5 is: 120
```

```
//N FACTORIAL USING FOR LOOP =>DECREMENT TYPE=> N!=5!=>5*4*3*2*1=120 Q-5(ii)
function factorial(n) {
    let result = 1;
    for (let i = n; i >= 1; i--) {
        result *= i;
    }
    return result;
}
const n = 5;
console.log(`Factorial of ${n} is: ${factorial(n)}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task3.js
Factorial of 5 is: 120
```

```
                                // DAY-4/TASK-4
//POWER OF 2'S USING INCREMENT TYPE : EG:2^4    Q-1(i)
function powerOfTwo(n) {
    let result = 1;
    for (let i = 0; i < n; i++) {
        result *= 2;
    }
    return result;
}
const n = 4;
console.log(`2^${n} is: ${powerOfTwo(n)}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task4.js
2^4 is: 16
```

```
//POWER OF 2'S USING DECREMENT TYPE    Q-1(ii)
function powerOfTwo(n) {
    if (n < 0) {
        return 'Exponent must be a non-negative integer';
    }
    let result = 1;
    for (let i = n; i > 0; i--) {
        result *= 2;
}
    return result;
```

```
}
const n = 4;
console.log(`2^${n} is: ${powerOfTwo(n)}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task4.js
2^4 is: 16
```

(i)POWER OF 2'S USING WHILE LOOP STATEMENT Q-2

```
function powerOfTwoWhileLoop(n) {
    if (n < 0) {
        return 'Exponent must be a non-negative integer';
    }
    let result = 1;
    let i = 0;
       while (i < n) {
        result *= 2;
        i++;
    }
    return result;
}
const n1 = 4;
console.log(`2^${n1} is: ${powerOfTwoWhileLoop(n1)}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task4.js
2^4 is: 16
```

(ii)ANOTHER METHOD IN LOOP STATEMENT Q-2

```
function powerOfTwoDoWhileLoop(n) {
    if (n < 0) {
        return 'Exponent must be a non-negative integer';
    }
    let result = 1;
    let i = 0;
```

```
    do {
        result *= 2;
        i++;
    } while (i < n);
    return result;
}
const n2 = 4;
console.log(`2^${n2} is: ${powerOfTwoDoWhileLoop(n2)}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task4.js
2^4 is: 16
```

```
//(iii)ONE ANOTHER METHOD IN LOOP    Q-2
function powerOfTwoArrayReduce(n) {
    if (n < 0) {
        return 'Exponent must be a non-negative integer';
    }
    const arr = Array(n).fill(1);
    const result = arr.reduce(accumulator => accumulator * 2, 1);

    return result;
}
const n3 = 4;
console.log(`2^${n3} is: ${powerOfTwoArrayReduce(n3)}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task4.js
2^4 is: 16
```

DAY-5/TASK-5

check whether number is prime or not : ***Q-1***

Eg: (i) i/p n=2

o/p => 2 is a prime number

```
function isPrime(n) {
    if (n < 2) {
        return `${n} is not a prime number`;
    }
    for (let i = 2; i < n; i++) {
        if (n % i === 0) {
            return `${n} is not a prime number`;
        }
    }
    return `${n} is a prime number`;
}
console.log(isPrime(2));
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task5.js
2 is a prime number
```

(ii) i/p=4

o/p =>4 is not a prime number ***Q-1***

```
function isPrime(n) {
    // Check if the number is less than 2
```

```
    if (n < 2) {
        return `${n} is not a prime number`;
    }
    for (let i = 2; i < n; i++) {
        if (n % i === 0) {
            return `${n} is not a prime number`;
        }
    }
    return `${n} is a prime number`;
}
console.log(isPrime(4));
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task5.js
2 is a prime number
4 is not a prime number
```

```
//I/P; N = 541
//O/P=> 541 IS NOT A PRIME NUMBER                    Q-2
function isPrime(n) {
    if (n < 2) {
        return `${n} is not a prime number`;
    }
    for (let i = 2; i < n; i++) {
        if (n % i === 0) {
            return `${n} is not a prime number`;
        }
    }
    return `${n} is a prime number`;
}
const number = 541;
console.log(isPrime(number));
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task5.js
541 is a prime number
```

```
THE NUMBER 72 BY USING THE PRIME FACTORIZE OF WRITTEN THE PRIME NUMBERS
EG :72                                                                Q-3
function primeFactorize(num, f = 2, result = '') {
    if (num < 2) {
        return result.trim();
```

```
    }
    if (num % f === 0) {
        result += f + ' ';
        return primeFactorize(num / f, f, result);
    } else {
        return primeFactorize(num, f === 2 ? 3 : f + 2, result);
    }
}
const number = 72;
const factors = primeFactorize(number);
console.log(factors);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task5.js
2 2 2 3 3
```

DAY-6/TASK-6

```
prime the number:- i/p :-n = 10; a = [1,2,3,5,6,7,8,9,10]                Q-1
function pn (n,cur=1){
    if (cur>n){
        return;
    }
    console.log(cur);
    pn(n,cur+1);
}
const n =10;
pn (n);
```

OUTPUT:-

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task6.js
1
2
3
4
5
6
7
8
9
10
```

```
//REVERSE AN ARRAY I/P: a =[1,2,3,4,5]                                   Q-2
function PR (arr,index){
    if (index<0){
        return;
```

```
    }
    console.log(arr[index]);
    PR (arr,index-1);
}
let a = [1,2,3,4,5];
PR (a,a.length-1);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task6.js
5
4
3
2
1
```

```
//PRINT THE ALL DAYS                                                    Q-3
function printDays() {
    const daysOfWeek = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday"];
    for (let day of daysOfWeek) {
        console.log(day);
    }
}
printDays();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task6.js
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
```

```
//diplay sunday
//print the monday                                                      Q-4
function displayDays() {
```

```
    const daysOfWeek = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday"];
    console.log(" Sunday");
    console.log(" Monday");
}
displayDays();
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task6.js
 Sunday
 Monday
```

DAY-7/TASK-7

```
                            day-7 /task-7
                            Q-1
*
* *
* * *
* * * *
function PSR(rows) {
    for (let row = 1; row <= rows; row++) {
        let pattern = "";
        for (let col = 1; col <= row; col++) {
            pattern += "*";
        }
        console.log(pattern);
    }
}
// Example usage
const numberOfRows = 5;
PSR(numberOfRows);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task7.js
*
**
***
****
*****
```

Q-2

```
* * * *
* * * *
* * * *
* * * *
```

```
function PSR(n, rows) {
    for (let row = 1; row <= rows; row++) {
        let pattern = "";
        for (let col = 1; col <= n; col++) {
            pattern += "*";
        }
        console.log(pattern);
    }
}
PSR(4, 5);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task7.js
****
****
****
****
****
```

Q-3

```
* * * *
* * *
* *
*
```

```
function PSR(n) {
    for (let row = n; row >= 1; row--) {
        let pattern = "";
        for (let col = 1; col <= row; col++) {
            pattern += "*";
        }
        console.log(pattern);
```

```
    }
}

PSR(5);
```

OUTPUT:

Q-4

```
  * * * *
    * * *
      * *
        *


function PSR(n) {
    for (let row = n; row >= 1; row--) {
        let pattern = '';

        for (let i = 1; i <= n - row; i++) {
            pattern += ' ';
        }
        for (let i = 1; i <= row; i++) {
            pattern += '*';
            if (i < row) {
                pattern += '';
```

```
            }
        }

        console.log(pattern);
    }
}
PSR(5);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task7.js
*****
 ****
  ***
   **
    *
```

Q-5

```
    *
   * *
  * * *
 * * * *
```

```
function PSR(n) {
    for (let row = 1; row <= n; row++) {
        let pattern = '';
        for (let col = 1; col <= n; col++) {

            pattern += (col <= n - row) ? ' ' : '*';
            pattern += (col > n - row && col < n) ? ' ' : ' ';
        }

        console.log(pattern.trimEnd());
    }
```

```
}
PSR(4);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task7.js
        *
      * *
    * * *
  * * * *
```

DAY-8/TASK-8

```
DESCENDING ORDER:-
// VAR a =[6,4,7,2,5]                                          Q-1

let arr = [6, 4, 7, 2, 5];

for (let i = 0; i < arr.length; i++) {
    for (let j = 0; j < arr.length - i - 1; j++) {
        if (arr[j] < arr[j + 1]) {
            let temp = arr[j];
            arr[j] = arr[j + 1];
            arr[j + 1] = temp;
        }
    }
}
console.log(arr);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task8.js
[ 7, 6, 5, 4, 2 ]
```

Q-2

a=[3,9,8,1,2]

```
var a = [3,9,8,1,2]
function fmm(arr){
    var min = arr[0]
    var max = arr[0]
    for(var i=1; i<arr.length;i++){
        min=arr[i]<min?arr[i]:min;
        max=arr[i]>max?arr[i]:max;
    }
    return {min:min,max:max};
}
var result= fmm(a)
console.log(result.min);
console.log(result.max);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task8.js
1
9
```

Q-3

ADD THESE NUMBER :-a = [3, 2, 5, 4]

```
var a = [3, 2, 5, 4];

function SumArray(arr) {
    var sum = 0;
```

```
    for (var i = 0; i < arr.length; i++) {
        sum += arr[i];
    }
    return sum;
}

var total = SumArray(a);
console.log(`Sum of the array: ${total}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task8.js
Sum of the array: 14
```

Q-4

```
    *
   * *
  * * *
 * * * *
* * * * *
```

```
function psg(rows) {
    for (let i = 0; i < rows; i++) {
        let line = '';
```

```
        for (let j = 0; j < 2 * rows - 1; j++) {

            if (j < rows - i - 1 || j > rows + i - 1) {
                line += ' ';
            } else {
                line += (j % 2 === (rows - i - 1) % 2) ? '*' : ' ';
            }
        }

        console.log(line);
    }
}

psg(5);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task8.js
    *
   * *
  * * *
 * * * *
* * * * *
```

DAY-9 / TASK-9

Q-1

```
****
****
####
####

let n=4;
for (let i=1;i<=n;i++){
    let char=(i%4 === 1||i%4===2)?'*':'#';
```

```
    let line="";

for (let j=0; j<n;j++){
    line += char;
}
console.log(line);
}
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task9.js
****

****

####

####
```

Q-2

```
**##
**##
**##
**##


function PSR(n){
    let pattern ="";
    for (let i=0;i<n;i++){
        if (i%4===0||i%4===1){
            pattern+='*'
```

```
        }else if (i%4===2||i%4===3){
            pattern+='#';
        }
    }
    for(let i=0;i<n;i++){
        console.log(pattern);
    }
}
PSR(4);
console.log("")
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task9.js
**##
**##
**##
**##
```

Q = 3

1
12
123
1234

```
var row =5
for (var i=1; i<=row;i++){
```

```
    var line='';
    for (var j=1;j<=i; j++){
        line+=(j)?j+'':'';

    }
    console.log(line.trimEnd());
}
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task9.js
1
12
123
1234
```

Q-4

```
   2
  24
 246
2468

var row = 4;

for (var i = 1; i <= row; i++) {
```

```
    var line = '';
    var currentnum = 2;
    for (var k = 1; k <= row - i; k++) {
        line += ' ';
    }
    for (var j = 1; j <= i; j++) {
        line += currentnum;
        currentnum += 2;
    }
    console.log(line);
}
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task9.js
   2
  24
 246
2468
```

DAY- 10 / TASK-10

```
//a = [1, 6, 7, 4, 3, 8, 4] ADD THESE NUMBER &FIND AVERAGE
                                                        Q-1
function CSA(arr) {
    let sum = 0;
    for (let i = 0; i < arr.length; i++) {
        sum += arr[i];
    }

    const avg = arr.length > 0 ? sum / arr.length : 0;

    console.log(`Sum: ${sum}`);
    console.log(`Average: ${avg}`);
}
const a = [1, 6, 7, 4, 3, 8, 4];
CSA(a);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task10.js
Sum: 33
Average: 4.714285714285714
```

```
                                Q-2
   fibonacci series write the code of this i/p: n= 10
function Gf(n) {
    if (n <= 0) {
```

```
        console.log('Please enter a number greater than 0');
        return [];
    }

    if (n === 1) return [0];

    let fib = [0, 1];
    for (let i = 2; i < n; i++) {
        fib[i] = fib[i - 1] + fib[i - 2];
    }

    return fib;
}

let result = Gf(10);
console.log(result);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task10.js
[
  0, 1,  1,  2,  3,
  5, 8, 13, 21, 34
]
```

I/P a = [3,4,3,6,4,4,6,8] IN THIS ARRAY HOW MANY SAME DIGITS ARE REPEATED WRITE THE CODE OS THIS:

```
function CEO(arr) {
    const elementCount = {};

    // Count occurrences of each element
    for (let i = 0; i < arr.length; i++) {
        const element = arr[i];
        elementCount[element] = elementCount[element] ? elementCount[element] +
1 : 1;
    }

    // Log occurrences of each element
    for (const element in elementCount) {
        console.log(`${element} occurs ${elementCount[element]} times`);
    }
}

const inputArray = [3, 4, 3, 6, 4, 4, 6, 8];
CEO(inputArray);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task10.js
3 occurs 2 times
4 occurs 3 times
6 occurs 2 times
8 occurs 1 times
```

DAY-11/TASK-11

Q1

FIND MAX VALUE WITHOUT SORTING

```
var a = [2,5,1,3,8,9];
function findmax(arr){
    if(arr.length === 0){
        return"array is empty"
    }
    var max = arr[0];
    for (var i=1; i<arr.length; i++){
      max= (arr[i]> max)?arr[i]:max;

    }
    return max;
    }
    console.log(findmax(a));
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task11.js
9
```

Q-2

I/P: a=["a","a","c","d","c","a","b"] IN THIS ARRAY HOW MANY SAME CHARACTERS ARE REPEATED: WRITE THE CODE OF THIS:

```
const array = ["a", "a", "c", "d", "c", "a", "b"];
const charCount = {};

for (let i = 0; i < array.length; i++) {
    const char = array[i];
    charCount[char] = charCount[char] ? charCount[char] +
1 : 1;
}

for (const char in charCount) {
    console.log(`${char} occurs ${charCount[char]} times`);
}
```

Output:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task11.js
a occurs 3 times
c occurs 2 times
d occurs 1 times
b occurs 1 times
```

Q-3

FIND MAX & MIN NUM WITHOUT SORTING

a=[5,1,2,6,8]

```
const a = [5, 1, 2, 6, 8];

function findMaxMin(arr) {
    if (arr.length === 0) {
        return { max: null, min: null };
    }

    let max = arr[0];
    let min = arr[0];

    for (let i = 1; i < arr.length; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
        if (arr[i] < min) {
            min = arr[i];
        }
    }

    return { max, min };
}

const { max, min } = findMaxMin(a);
console.log(`Max: ${max}`);
console.log(`Min: ${min}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task11.js
Max: 8
Min: 1
```

DAY-12/TASK-12

Q-1

```
* * * * *
 * * * *
  * * *
   * *
    *
```

```
function PSR(row){
    for (let i=0; i< row; i++){
        let line ='';
        const width =2 * row -1;
        for (let j=0; j<width; j++){
            line += (j<i||j>=width-i)?' ':(j% 2===(i%2)?'*':' ');

        }
        console.log(line);
    }
}
PSR(6)
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task12.js
* * * * * *
 * * * * *
  * * * *
   * * *
    * *
     *
```

Q-2

```
   *
  * *
 * * *
* * * *
 * * *
  * *
   *
```

```
function pd(n){
    if (n<1){
        console.log("please enter the positive number.");
        return
    }
    for (let i=0; i<2*n-1; i++){
        let line ='';
        const isupperPart = i<n;
        const spaces = isupperPart ? n-i-1 : i - n + 1;
        const stars = isupperPart? i+1 : 2*n - i-1;
        for (let s =0; s< spaces; s++){
            line+= ' ';
        }
        for (let st=0;st<stars;st++){
            line+= '* ';
        }
        console.log(line);
    }
}
pd(4)
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task12.js
    *
   * *
  * * *
 * * * *
  * * *
   * *
    *
```

(i)FIND 2ND MAX NUMBER WITHOUT SORTING

a = [5,1,2,6,8,7] ***Q-3***

```
const a = [5, 1, 2, 6, 8, 7];

function findSecondMax(arr) {
    if (arr.length < 2) {
        return null;
    }

    let max = arr[0] > arr[1] ? arr[0] : arr[1];
    let secondMax = arr[0] > arr[1] ? arr[1] : arr[0];

    for (let i = 2; i < arr.length; i++) {
        const num = arr[i];

        max = num > max ? num : max;
        secondMax = num < max && num > secondMax ? num : secondMax;
    }
    return secondMax === max ? null : secondMax;
}

const secondMax = findSecondMax(a);
console.log(`Second Max: ${secondMax}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task12.js
Second Max: 7
```

(ii) FIND 2ND Min NUMBER WITHOUT SORTING

a = [5,1,2,6,8,7] ***Q-3***

```
const a = [5, 1, 2, 6, 8, 7];

function findSecondMin(arr) {
    if (arr.length < 2) {
        return null;
    }

    let min = arr[0] < arr[1] ? arr[0] : arr[1];
    let secondMin = arr[0] > arr[1] ? arr[0] : arr[1];


    for (let i = 2; i < arr.length; i++) {
        const num = arr[i];

        min = num < min ? num : min;
        secondMin = num > min && num < secondMin ? num : secondMin;
    }

    return secondMin === min ? null : secondMin;
}

const secondMin = findSecondMin(a);
console.log(`Second Min: ${secondMin}`);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task12.js
Second Min: 2
```

DAY-13/TASK-13

```
 FIND 3rd Max & 3rd min NUMBER WITHOUT SORTING
 a = [5,1,2,6,8,7]                                              Q-1


function findKthMaxMin(arr, k) {
   function findKthMax(arr, k) {
      let maxValues = new Set();
      let currentMax;

      for (let i = 0; i < k; i++) {
         currentMax = null;
         for (let num of arr) {
            currentMax = (currentMax === null || (num > currentMax
&& !maxValues.has(num)))
                        ? num
                        : currentMax;
         }
         maxValues.add(currentMax);
      }
      return currentMax;
   }
   function findKthMin(arr, k) {
```

```
    let minValues = new Set();
    let currentMin;

    for (let i = 0; i < k; i++) {
      currentMin = null;
      for (let num of arr) {
        currentMin = (currentMin === null || (num < currentMin
&& !minValues.has(num)))
                    ? num
                    : currentMin;
      }
      minValues.add(currentMin);
    }
    return currentMin;
  }
  const thirdMax = findKthMax(arr, k);
  const thirdMin = findKthMin(arr, k);

  return { thirdMax, thirdMin };
}
const a = [5, 1, 2, 6, 8, 7];
const { thirdMax, thirdMin } = findKthMaxMin(a, 3);
console.log("Third Max:", thirdMax);
console.log("Third Min:", thirdMin);
```

OUTPUT:

```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task13.js
Third Max: 6
Third Min: 5
```

Q-2

SWAPPING WITHOUT USING TEMP & DON'T DECLARE ANY VARIABLE

var a = 8;

var b = 9;

```
var a = 8;
var b = 9;

a = a + b;
b = a - b;
a = a - b;

console.log("a:", a);
console.log("b:", b);
```

OUTPUT:

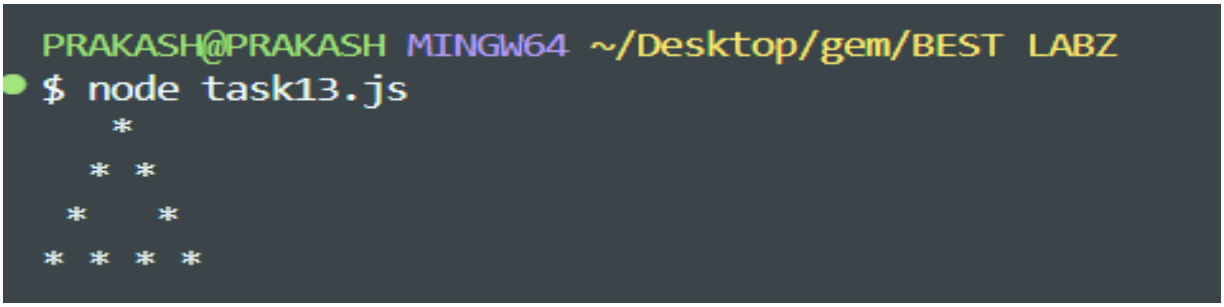
```
PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task13.js
a: 9
b: 8
```

Q-3

```
    *
  *   *
 *     *
* * * *
```

```
function PHP (n){
    for (let i=1;i<=n; i++){
        let row ='';
        for (j=1; j<=n;j++){
            row += (j<=n-i)?' ':'';
        }
        for (let j=1; j<=i;j++){
            row+=(j===1||j===i||i===n)?'* ' :'  ';
        }
        console.log(row);

    }

}
PHP(4)
```

OUTPUT:


PRAKASH@PRAKASH MINGW64 ~/Desktop/gem/BEST LABZ
$ node task13.js
*
* *
* *
* * * *