**Kadane’s Algorithm (Maximum Product Subarray):**

class Solution {

    maxProduct(arr,n){ //O(n)

        let start = 0

        let product = 1

        let maxProd = Math.abs(arr[0])

        for(let i = start; i < arr.length; i++){

            product\*=arr[i]

            console.log("product ===>",product)

            console.log("maxProd ===>",maxProd)

            if( Math.abs(product) > maxProd){

                maxProd = Math.abs(product)

            }

            if( Math.abs(product) === 0){

                product = 1

            }

        }

        return maxProd

    }

}

const prod = new Solution()

console.log("<======>",prod.maxProduct([6, 0, -10, 1, 2,4,0,15,-6]))

**HashMaps:**

//JS ES6 Map

//enhancement over using an object

/\*

-Map has a delete method

-Map has a clear method to clear all data at once in a Map

-Map keeps insertion order in case you would like to iterate in order

-Map had forEach method

-Look up an item in O(1) time by key

\*/

let m = new Map();

m.set(10,100)

m.set(9, true)

m.set(10, false)

m.set("Apple","Fruit")

console.log("m=======>",m)

console.log(m.get(10));

console.log(m.get(100));

console.log(m.size);

console.log(`\*\*\*\*\*\*\*mmmmmmmmmmmmmmmmmmmm\*\*\*\*\*\*\*\*`);

for(t of m) {

  console.log(t)

}

for(t of m.keys()) {

  console.log(t)

}

for(t of m) {

  console.log("t[0]======>",t[0])

}

for(t of m.values()) {

  console.log(t)

}

for(t of m) {

  console.log("t[1]======>",t[1])

}

m.delete(10)

console.log("m after delete=======>",m)

m.clear()

console.log("m after clear=======>",m)

console.log(`\*\*\*\*\*\*\*Above For Loop For m\*\*\*\*\*\*\*\*`);

class Contact {

  constructor(name, age, phoneNumber) {

    this.name = name;

    this.age = age;

    this.phoneNumber = phoneNumber;

  }

}

const contracts = new Map();

contracts.set("Shane", new Contact("Shane Crouch", 12, "111-111-1111"));

contracts.set("Rosy", new Contact("Rosy Stark", 13, "222-222-2221"));

contracts.set("Gabriel", new Contact("Gabriel Taco", 13, "333-333-3331"));

console.log(`Each Contract in The Map`);

contracts.forEach((contact) => console.log(contact));

console.log(`\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*`);

contracts.forEach((contact) => console.log(contact.name));

console.log(`\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*`);

console.log(contracts);

contracts.clear();

console.log("After Clearing");

console.log(contracts);

console.log(`\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*`);

class Contact1 {

  constructor(name, age, phoneNumber, parent) {

    this.name = name;

    this.age = age;

    this.phoneNumber = phoneNumber;

    this.parent = parent;

  }

}

const contracts1 = new Map();

contracts1.set(

  "Shane",

  new Contact1("Shane Crouch", 12, "111-111-1111", { fn: "ABC", mn: "DEF" })

);

contracts1.set(

  "Rosy",

  new Contact1("Rosy Stark", 13, "222-222-2221", { fn: [1, 2, 3], mn: "JKL" })

);

console.log(`Each Contract in The Map contracts1`);

contracts1.forEach((contact) => console.log(contact.parent));

contracts1.forEach((contact) => console.log(contact.parent.fn));

console.log(`\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*`);

console.log("Getting a specific contact's phone number in O(1) constant");

console.log(contracts1.get("Rosy").phoneNumber);

//Check whether the Map has specific property

if (!contracts1.has("Sinha")) {

  console.log("Adding contact Sinha");

  contracts1.set(

    "Sinha",

    new Contact1("Ashish Sinha", 14, "444-444-4441", { fn: "ABCF", mn: "DEFG" })

  );

}

console.log(contracts1.get("Sinha"));

//Delete item from Map

console.log(contracts1);

console.log(contracts1.delete("Sinha"));

console.log("After Delete");

console.log(contracts1);

console.log(`\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*`);

//The constructor takes in array of arrays. Each child array is a key value pair

const PLanguage = new Map([

  [1, "JavaScript"],

  [2, "Python"],

  [3, "Java"],

]);

console.log(PLanguage.size);

console.log(PLanguage.keys());

const Keys = PLanguage.keys();

for (let key of Keys) {

  key += 1;

  console.log(key);

}

console.log("PLanguage.values()==========>",PLanguage.values());

console.log("PLanguage.entries()==========>",PLanguage.entries());

for (let entry of PLanguage.entries()) {

  console.log(`Key: ${entry[0]} Value: ${entry[1]}`);

}