**מחלקות, תכונות, מתודות ואובייקטים**

**שאלה 1**

כתוב מחלקה בשם Counter שמייצגת מונה.

תכונות המחלקה:

מספר שלם (ערך המונה)

המחלקה תכיל את המתודות הבאות:

Initialize(int value) – המתודה מאתחלת את המונה בערך הנתון כפרמטר.

Increment – המתודה מקדמת את ערך המונה ב-1.

Go – מדפיסה את המספרים מ- 0 ועד ערך המונה.

כתוב תוכנית ראשית שמגדירה אובייקט מסוג Counter, מאתחלת אותו בערך כלשהו, מקדמת אותו ומפעילה את המתודה Go.

export class Counter {

 num;

  constructor(num) {

   this.num=num;

  }

  SetNum(value) { this.num = value; }

  GetNum() { return this.num; }

  Initialize (value) {this.num=value;}

  Increment() {

    this.num++;

  }

  go() {

    for (let i = 0; i <= this.num; i++) {

      console.log(i);

    }}

  }

import {Counter} from './counter.js';

function Main() {

let c1 =new Counter();

c1.Initialize(5);

c1.Increment();

c1.go(console.log());

}

Main();

**שאלה 2**

* 1. כתוב מחלקה Frog המתארת צפרדע. נתוני המחלקה הם: שם (Name), צבע (Color), גיל (Age), משקל (Weight).
  2. מתודות המחלקה הן:

1. Show() – מדפיסה את כל הנתונים של הצפרדע.
2. Quack() – מדפיסה את המילה "quack!!!!", Age\*Weight)/2) פעמים ומשמיעה סאונד באותה כמות פעמים.
   1. כתוב תוכנית ראשית יוצרת אובייקט מטיפוס Frog ומפעילה את המתודות שכתבת.

import {Frog} from './frog.js';

function Main() {

const f1 =new Frog("kaka","green",5,2);

f1.Show();

f1.Quack();

}

Main();

export class Frog {

  name;

  color;

  age;

  weight;

   constructor(name,color,age,weight) {

    this.name=name;

    this.color=color;

    this.age=age;

    this.weight=weight;

   }

 Show()

 {

   console.log(`frog details`);

   console.log(`name: `+this.name);

   console.log(`color: `+this.color);

   console.log(`age: `+this.age);

   console.log(`weight: `+this.weight);

 }

 Quack()

 {

   let sum=age\*weight/2;

   for(let i=0;i<sum;i++)

   {

     console.log(`quack`);

   }

 }

 }

**שאלה 3**

1. הגדר מחלקה אשר מייצגת שעה. נתוני המחלקה הם מספרים שלמים אשר ייצגו את השעות, הדקות, ושניות ושם המדינה. לדוגמה: *השעה בישראל: 12:35:55*

במחלקה יוגדרו שתי מתודות:

* + - 1. ConverToSeconds – המתודה ממירה את השעה לשניות ומחזירה את כמות השניות. לדוגמא: עבור השעה 12:35:55, המתודה תחזיר 45355, שזה התוצאה של החישוב 12\*3600+35\*60+55
      2. Show – מציגה את השעה בפורמט hh:mm:dd.

כדי להדפיס אפסים מובילים (נניח עבור השעה 3, 4 דקות ו 5 שניות) ניתן לכתוב כך: 03:04:05

1. בתוכנית הראשית הגדירו מערך של "שעונים" בגודל 5.

התוכנית צריכה להציג ב- HTML כל מדינה ואת השעה שלה, וממירה אותה לשניות.

import {Clock} from './clock.js';

function Main() {

// דוגמה לשימוש במחלקה

const clocks = [

  new Clock(12, 35, 55, 'Israel'),

  new Clock(8, 15, 30, 'United States'),

  new Clock(17, 45, 0, 'United Kingdom'),

  new Clock(23, 10, 20, 'Japan'),

  new Clock(6, 0, 0, 'Australia')

];

for (const clock of clocks) {

  const totalSeconds = clock.convertToSeconds();

  console.log(`Country: ${clock.country}, Time: ${clock.show()}, Seconds: ${totalSeconds}`);

}

}

Main();

export class Clock {

  hours;

  minutes;

  seconds;

  country;

    constructor(hours, minutes, seconds, country) {

      this.hours = hours;

      this.minutes = minutes;

      this.seconds = seconds;

      this.country = country;

    }

    convertToSeconds() {

      return this.hours \* 3600 + this.minutes \* 60 + this.seconds;

    }

    show() {

      const formattedHours = this.hours < 10 ? `0${this.hours}` : this.hours;

      const formattedMinutes = this.minutes < 10 ? `0${this.minutes}` : this.minutes;

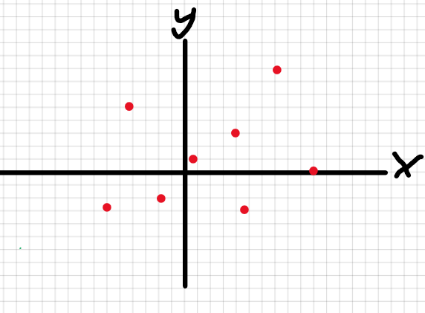
      const formattedSeconds = this.seconds < 10 ? `0${this.seconds}` : this.seconds;

      console.log(`${formattedHours}:${formattedMinutes}:${formattedSeconds}`);

    }

  }

**שאלה 4**

המחלקה Point מייצגת נקודה במערכת הצירים. ערכי המיקום יכולים להיות חיוביים או שליליים.

1. כתבו את הגדרת המחלקה על פי הנתונים הבאים:

|  |
| --- |
| class Point |
| **משתנים/תכונות:**  x – ממשי |
| y – ממשי |
| **מתודות נוספות:** |
| Show()  המתודה מציגה כפלט את שיעורי הנקודה בפורמט **(x,y)**  Equals(Point p)  המתודה מקבלת "נקודה" אחרת ומחזירה **אמת** אם הן זהות או **שקר** במידה ולא. |

export class Point {

  x;

  y;

    constructor(x, y) {

      this.x = x;

      this.y = y;

    }

    Show() {

      console.log(`(${this.x},${this.y})`);

    }

    Equals(Point p) {

      return this.x === p.x && this.y === p.y;

    }

  }

1. כתבו פעולה המקבלת 3 פרמטרים: מערך של נקודות (מובטח שהוא מלא), x וערך y.

הפעולה מחזירה **אמת** אם יש במערך נקודה שערכיה זהים לערכי x,y שהתקבלו כפרמטרים. אחרת הפעולה תחזיר **שקר**.

**רשמו דוגמה משלכם שעבורה הפעולה תחזיר אמת ודוגמה שעבורה הפעולה תחזיר שקר.**

import {Point} from './clock.js';

function Main() {

  const points = [

    new Point(1, 2),

    new Point(3, 4),

    new Point(5, 6)

  ];

  console.log(IfPointEquals(points, 3, 4)); // יחזיר true

  console.log(IfPointEquals(points, 7, 8)); // יחזיר false

}

function IfPointEquals(points, x, y) {

  for (let i = 0; i < points.length; i++) {

    if (points[i].x === x && points[i].y === y) {

      return true;

    }

  }

  return false;

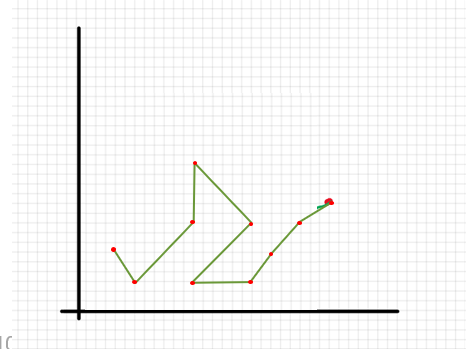
}

Main();

1. כתבו פעולה המקבלת 2 פרמטרים: מערך מלא של נקודות ואובייקט מטיפוס נקודה, ומחזירה **אמת** אם נמצאת במערך נקודה שערכיה זהים לנקודה שהתקבלה כפרמטר, אחרת תחזיר **שקר**.

יש להיעזר במתודה Equal שכתבת בסעיף א'.

**רשמו דוגמה משלכם שעבורה הפעולה תחזיר אמת ודוגמה שעבורה הפעולה תחזיר שקר.**

1. מסלול טיול מוגדר ע"י אוסף נקודות המייצגות מיקום.

אוסף הנקודות הוא מערך של Point.

ידוע כי כל הערכים בנקודות הם חיוביים.

כתבו פעולה חיצונית למחלקה, המקבלת את המערך של הנקודות ומחזירה את אורכו של המסלול.

**הנחיות עזר לחישוב אורכו של המסלול**:

* לסרוק את המערך ולבדוק זוגות של נקודות.
* ניתן לחשב מרחק בין שתי נקודות ע"י משפט פיתגורס
* לנקודה האחרונה אין בן זוג.

**רשמו דוגמה משלכם של מסלול בעל 4 נק' לפחות**