תרגול 9

חזרה רקורסיה

אוספים

קומפרטורים

שאלות לדוגמה ממבחנים

רקורסיה – שקף מתרגול 4!

- קידוד רקורסיבי הוא קטע שקוד שקורא לעצמו עם פרמטר שונה בכל קריאה.
- קוד רקורסיבי חייב להכיל תנאי עצירה (אחרת הקוד רץ לנצח).
- לדוגמה: פונקציה לחישוב חזקה: $a^b = a \cdot a \cdot \ldots_{b\ times} \cdot a$

```
Public static int power(a,b){
        if (b==0) return 1;
        return (a*power(a,b-1));
}
```

רקורסיה – דוגמא נוספת

```
// Sum of digits using recursion
static int sumOfDigits(int num) {
        // Base case: single-digit number
        if (num < 10) {
                return num;
        }
        else {
        // Recursive case: sum = last digit + sum of digits of remaining number
                return num % 10 + sumOfDigits(num / 10);
        }
}
```

רקורסיה

- פונקציה נקראת רקורסיבית אם היא קוראת לפונקציה רקורסיבית
- לדוגמה:

```
public static String reverseString(String s) { // main function has 1 input
            // calls the helper with 3 inputs: word, index and reversed word (starts with empty string)
    return helper(s, s.length() - 1, "");
  }

  private static String helper(String s, int index, String acc) {
    if (index < 0) {
      return acc;
    } else {
      return helper(s, index - 1, acc + s.charAt(index));
    }
  }
```

* פתרנו את התרגיל הזה בתרגול 3 בצורה איטרטיבית.
* אלא אם כן ביקשו ממכם במפורש לפתור בצורה רקורסיבית, אפשר לפתור איך שהכי מובן וברור לכם

אוספים - collections

- ראינו בתרגולים שאפשר "לאסוף" את האובייקטים שלנו
- לדוגמה
  - ✓מערך של תלמידים: Student[] classroom = new Student[25]
  - ✓מערך דינאמי של נקודות: ArrayList<Point2D> points = new ArrayList<Point2D>

- אפשר לבנות אוסף של מחלקות שעושות extend לאותו interface או יורשות מאותה המחלקה
- לדוגמה:
  - ✓ מערך דינאמי של צורות: ArrayList <GeoShapes> shapes = new ArrayList<GeoShapes>
- עכשיו אפשר להכניס למערך shapes כל צורה שיורשת מGeoShapes , ולהפעיל על המערך פעולות שקיימות ב GeoShapes (כי כל האיברים במערך מממשים את הפעולות האלה)

קומפרטור

- על מנת למיין דברים במהלך הקוד שלנו, צריך להשוות בין דברים מאותו הסוג
- יש דברים שברור לנו איך להשוות כמו מספרים (int, double וכו...)
- אפילו String אנחנו כבר מכירים שיטה להשוות (לקסיקוגרפי)
- אבל לאובייקטים שאנחנו יוצרים, אין דרך ברורה שהקוד שלנו יודע להשוות
- לכן כל מחלקה שנרצה להשוות בין 2 אובייקטים שלה נרצה שהמחלקה תממש את ה Interface Comparable

```
Interface Comparable<T>{
        public ststic int compare( T a, T b );
}
```

קומפרטור – דוגמא:

```
public class Student implements Comparator<Student> {
        ...
        //the rest of the class
        …
   @Override
   public int compare(Student a, Student b) {
      if (a.getID().compareToIgnoreCase(b.getID())>0)
         return 1;
      else if (a.getID().compareToIgnoreCase(b.getID())<0)
         return -1;
      else
         return 0;
   }
}
```

קומפרטור - המשך

- אם רוצים למיין לפי כמה אפשרויות אפשר לבנות מחלקה עבור כל סוג השוואה:

```
// Helper class implementing Comparator interface
class Sortbyage implements Comparator<Student> {
// Sorting in ascending order age
   public int compare(Student a, Student b) {
            if (a.getAge() > b.getAge()){
                        return 1;
            }else if(a.getAge() < b.getAge()){
                       return -1;
            }else{
                        return 0;}
   }
}
```

```
// Helper class implementing Comparator interface
class Sortbyname implements Comparator<Student>{
// Sorting in ascending order of name
   public int compare(Student a, Student b)
   {

      return a.name.compareTo(b.name);
   }
}
```

קומפרטור - המשך

```
public static void main(String[] args) {
    // Create an ArrayList of students
    ArrayList<Student> students = new ArrayList<>();
    students.add(new Student("Alice", 22));
    students.add(new Student("Bob", 20));
    students.add(new Student("Charlie", 25));
    students.add(new Student("David", 18));

    // Print unsorted ArrayList
    System.out.println("Unsorted Students:");
    printStudents(students);
```

```
    // Sort by age using Sortbyage comparator
    Collections.sort(students, new Sortbyage());
    System.out.println("Sorted Students by Age:");
    printStudents(students);

    // Sort by name using Sortbyname comparator
    Collections.sort(students, new Sortbyname());
    System.out.println("Sorted Students by Name:");
    printStudents(students);
}
```

תרגילים ברקורסיה:

- כתבו פונקציה רקורסיבית שבודקת האם מילה היא פלינדרום (כלומר מילה שהקריאה שלה מימין לשמאל ומשמאל לימין זהה. לדוגמה: ABBA, sos, kayak)
  public static Boolean isPalindrome(String str)
- כתבו פונקציה רקורסיבית שממירה מספר עשרוני ליצוג הבינארי שלו במחרוזת
  public static String ToBinary(int num)
- בהינתן מילה, כתבו פונקציה רקורסיבית שמוצאת את כל הפרמוטציות של האותיות במילה ומחזירה אותם במערך. (לדוג: עבור המילה "xyz" קיימים 6 פרמוטציות: "xyz", "xzy", "yxz", "yzx", "zxy", "zyx")

public static List<String> permutations(String s)

תרגילים ממבחנים:

שאלה 1:

נתון מערך של מספרים ממשיים (double), ניתן להניח שמערך תקין (לא null, ולא ריק).

1.1 (13 נקודות) כתבו פונקציה אשר מערבבת את המערך באופן אחיד, משמע שההסתברות של כל איבר במערך להיות בכל אינדקס במערך היא אחידה. **הדרכה**: בשאלה זו עליכם לכתוב את הפונקציה בעצמכם, ללא שימוש פונקציות ערבוב מובנות ב java. ניתן להניח שכל האיברים במערך שונים זה מזה.

```
public static void shuffle(double[] arr) //{…}
```

שאלה 2:

בשאלה זו נתייחס לממשק של צורות (GeoShape),

כזכור, הפעלת השיטה getClass().getSimpleName() מחזירה מחרוזת עם שם המחלקה ממנה נוצר האובייקט.

2.1 (8 נקודות) כתבו פונקציה סטטית שמקבלת שתי צורות s1,s2, ומחזירה אמת אם ורק אם אף אחת מהן אינה שווה ל null, וגם הן מאותה מחלקה.

```
public static boolean sameClass(GeoShape s1, GeoShape s2)  {…}
```

2.2 (17 נקודות) כתבו פונקציה סטטית שמקבלת מערך של צורות ומחזירה את **מספר** המחלקות **השונות** אליהן שייכות הצרות במערך.

```
public static int numOfClasses(GeoShape[] s) {…}
```

תרגילים ממבחנים:

בשאלה זו נניח שקיימת לכם המחלקה Q שמייצגת קוביית משחק **הוגנת** (בכל "זריקה" הקובייה יכולה לקבל ערך 1,2,3,4,5,6 בלבד, בהסתברות אחידה). למחלקה יש בנאי ריק והיא מממשת את הממשק Game בעל השיטות הבאות:

```
public interface Game() {
        void roll(); // roll the dice
        int getVal(); // return the value of the last roll.
}
```

1. שיטה roll() שלא מקבלת פרמטרים, ולא מחזירה ערכים, למעשה "זורקת" את הקובייה (או הקוביות).
2. שיטה getVal() מחזירה את ערך של הקובייה (או הקוביות) מהזריקה האחרונה (אם היא לא נזרקה עדיין מחזירה 0). שימו לב שכל עוד לא בוצע זריקה נוספת, הערך המוחזר לא ישתנה.

4.1 (7 נקודות)השלימו את המחלקה Q2 שמייצגת זוג קוביות משחק (הוגנות) בעלת בנאי ריק ושמממשת את הממשק Game

```
public class Q2 implements Game {  //...
}
```

המשך בעמוד הבא...

תרגילים ממבחנים:

4.2 (10 נקודות) השלימו את המחלקה Q2_Err שמייצגת זוג קוביות "מזויפות" – יש לה "עדיפות" עבור מספרים גבהים. משמע: הסיכוי לקבל סכום של 12 (6,6) יהיה גבוה יותר מהסיכוי לקבל 2 (1,1).

```
public class Q2_Err  //...
s{
}
```

4.3 (8 נקודות) הסבירו (במילים) כיצד ניתן לבדוק שהמחלקה Q2 מייצגת זוג קוביות הוגנות, והמחלקה Q2_Err אינה מייצגת זוג קוביות הוגנות.