Exceptions Homework

מטרה:

בתרגיל זה נכתוב אפליקציה קטנה לתרגול נושא Exceptions.

המחלקה תכיל שתי מחלקות ראשיות, Employee ו - SalesPerson, מהן נרצה ליצור אובייקטים בצורה חוקית - אחרת לזרוק חריגה מתאימה בדבר הבעיה שהתרחשה.

דרישות:

על התרגיל להיות מסודר וקריא. יש להקפיד על תיעוד במקומות הנדרשים - על פי שיקול דעתכם ובהתחשב בעובדה שמתכנת אחר ייקרא את הקוד שלכם בעתיד.

הקפידו על מוסכמות השפה כפי שנלמדו בשיעורים עד כה.

הנחיות:

לפניכם הנחיות כלליות לכתיבת האפליקציה. אמנם, על פי שיקול דעתכם תוכלו להשתמש בפונקציות עזר לבחירתכם ובכלים מתקדמים - גם במידה ועדיין לא נלמדו.

<u>האפליקציה תכיל את המחלקות הבאות:</u>

1. Employee - עובד בחברה.
2. SalesPerson - איש מכירות בחברה, <u>יורש מ Employee.</u>
3. InvalidEmployeeCertException - מחלקה המגדירה חריגה עבור מספר רשיון בלתי תקין של SalesPerson.
4. InvalidEmployeeSalaryException - מחלקה המגדירה חרגיה עבור משכורת בלתי חוקית של SalesPerson.

<u>כתיבת המחלקה **InvalidEmployeeSalaryException**:</u>

המחלקה יורשת מ RuntimeException ומכילה את הבנאי הבא בלבד:

```
public InvalidEmployeeSalaryException(String message) {
    super(message);
}
```

<u>כתיבת המחלקה **InvalidEmployeeCertException**:</u>

המחלקה יורשת מ Exception ומכילה את הבנאי הבא בלבד:

```
public InvalidEmployeeCertException(String message) {
    super(message);
}
```

כתיבת המחלקה **Employee**:

המחלקה תכיל את השדות הבאים:

1. שדה סטטי קבוע בשם MAXIMUM_SALARY המאותחל לערך 40,000 המגדיר את ערך המשכורת הגבוה ביותר עבור עובד בחברה.
2. שדה קבוע מטיפוס String השומר את שם העובד.
3. שדה קבוע מטיפוס int השומר את גיל העובד.
4. שדה קבוע מטיפוס double השומר את משכורת העובד.

המחלקה תכיל את הפונקציות הבאות:

1. בנאי שחתימתו:

```
public Employee(String name, int age, double salary)
```

המאתחל את שדות המחלקה המתאימים, אלא אם כן שדה המשכורת אינו תקין - במצב כזה יש לזרוק חריגה מסוג InvalidEmployeeSalaryException עם תיאור מתאים.

כתיבת המחלקה **SalesPerson**:

מחלקה זו יורשת מהמחלקה **Employee**.

המחלקה תכיל את השדות הבאים:

1. שדה מטיפוס String בשם certificationNumber השומר את מספר אישור העסקה של איש המכירות בחברה. שדה זה חייב להכיל ערך מתאים לחוקים הבאים:
   a. פורמט מספר אישור העסקה הינו ###-???. כלומר, שלוש ספרות כלשהן, מכף הפרדה ולאחר מכן שלושה תווים כלשהם.

לצורך ניתוח מספר אישור העסקה של איש המכירות, השתמשו בפונקציה הסטטית indexOf של המחלקה String, יחד עם הפונקציה הסטטית substring של המחלקה String.

המחלקה תכיל את הפונקציות הבאות:

1. המאתחל את שדות המחלקה בהתאם, אלא אם כן מספר אישור העסקה של איש המכירות הינו בלתי חוקי. במקרה של ערך שגוי כזה, זרקו את החריגה InvalidEmployeeCertException. לצורך כך, השתמשו בפונקציה הבאה:
2. פונקצית עזר לבנאי הנ"ל שחתימתה:

```
private static String requireValidCertificationNumber(String certNum)
```

ותפקידה לדרוש לפורמט תקין של הארגומנט certNum המתקבל לפרמטר הראשון.

במידה ו certNum אינו חוקי - תזרוק הפונקציה את החריגה InvalidEmployeeCertException, אחרת תחזיר את certNum כפי שהתקבל. רעיון טוב הוא להעזר אף בפונקציה נוספת שתבדוק את התקינות עצמה, ובפונקציה הזו, requireValidCertificationNumber להשתמש בה בכדי להחליט על זריקת חריגה.

צרו את הפונקציה main במחלקה חדשה, צרו מספר מופעים של המחלקה SalesPerson, תפסו את החריגות, הכניסו קלטים שונים ובדקו את תקינות הקוד שלכם.

בונוס:

ממשו את ה interface הבא במחלקה SalesPerson:

```
public interface Comparable<T> {
    int compareTo(T other);
}
```

כלומר:

```
public class SalesPerson extends Employee implements Comparable<SalesPerson>
```

הפונקציה compareTo תשווה בין שני אובייקטים מטיפוס SalesPerson ותחזיר את הערך 1 במידה ולאובייקט this ערך משכורת גבוה מזה של other, את הערך 1- במידה ולאובייקט other ערך משכורת גבוה מזה של this, או 0 אחרת, כלומר במידה וערך המשכורת של this ו other שווים.

צרו מספר מופעים של המחלקה SalesPerson, השוו ביניהם והדפיסו למסך מי הוא ה SalesPerson שמשכורתו היא הנמוכה ביותר.

בהצלחה!