תרגול מס' 12 - חריגות

הוסיפו לפרויקט package בשם T12.

תרגיל 1

נמחיש את מנגנון החריגות באמצעות מחלקות המנהלות רשימת ניגון של שירים.
הקוד ההתחלתי לתרגול נמצא בדף הקורס ב- MAMA

חלק א' - הורידו את שלושת הקבצים הבאים וצרפו אותם אל החבילה T12

שיר (**Song**) מאופיין על ידי שם (name, מחרוזת) ומשכו בשניות (seconds, שלם גדול מ-0)
במחלקה Song מוגדרות התכונות, בנאי מלא, ומתודות מאחזרות וקובעות. בנוסף מוגדרות המתודות הבאות:

- play - המדמה את "ניגון השיר" על ידי הדפסת שמו ואת משכו בשניות.
- העמסה של play - המקבלת פרמטר נוסף המציין מהיכן לנגן את השיר.

רשימת נגינה (**PlayList**) מאופיינת על ידי songs, רשימה של שירים **בעלי שמות שונים**.
במחלקה PlayList מוגדרת התכונה היחידה ובנוסף מוגדרות המתודות הבאות:

- ***addSong*** - מקבלת כפרמטר שיר ומוסיפה אותו בסוף הרשימה
- ***playSong*** - מקבלת שם של שיר וזמן התחלה "ומנגנת" את השיר המתאים החל מהזמן המבוקש
- העמסה של ***playSong*** - המקבלת במקום שם של שיר את מספרו ברשימה

במחלקה הראשית **T1Main** הוראות עבור :

- יצירת רשימת נגינה
- הוספה של שני שירים לרשימת הנגינה
- ניגון השיר הראשון ברשימה מתחילתו ועד סופו
- ניגון השיר S2 החל מהשנייה ה-100
- בהמשך ישנן ארבע הוראות "בעייתיות" המסומנות בהערה

אילו מבין ההוראות תגרומנה לשגיאה לוגית? אילו מבין ההוראות תגרומנה לשגיאת ריצה?
הריצו התכנית לאחר שהסרתם/ן ההערות מההוראות הללו. ראו מה מודפס.
סמנו את ההוראות האלו שוב בהערה.
נסו לזהות הוראות קוד רגישות נוספות קיימות במחלקות ועשויות לגרום לשגיאות.
בשל מגבלת זמן בתרגול נטפל רק בחלק מהן.

חלק ב'- דוגמאות לאופן טיפול בחריגות

א. ההוראה הבאה מהווה שגיאה לוגית - כי משך השיר צריך להיות מספר גדול מ-0

```
Song s3 = new Song("S3", -240);
```

1. בבנאי המחלקה **Song** כשערך הפרמטר השני קטן שווה ל-0 נזרוק עצם חדש מסוג **Exception** עם המלל:
"Song duration must be greater than 0"
2. **בעיה ראשונה**: הוראת הזריקה מסומנת בשגיאה
פתרון: יש להוסיף לכותרת הבנאי את ההכרזה throws Exception
3. **בעיה שנייה**: הוראות יצירת השירים ב- main מסומנות בשגיאה
פתרון א': יש להוסיף לכותרת של ה- main את ההכרזה throws Exception
פתרון ב': נעטוף את הוראות יצירת השירים בבלוקים של try/catch
בחרו בפתרון א'
4. ב - T1Main הסירו את ההערה מהוראה זו והריצו התכנית. מה מופיע? לאחר ההרצה שימו ההוראה בהערה

ב. גם בהוראה הבאה ישנה שגיאה לוגית - כי שניית התחלת הנגינה המבוקשת (190) גבוהה ממשך השיר (180)

```
pl.playSong(1, 190);
```

1. במחלקה Song במתודה ***play*** כשהפרמטר startAt חורג ממשך השיר נזרוק עצם חדש מסוג **Exception** עם ההודעה:
"Song start should be between 0 and <seconds>"
כאשר במקום <seconds> יופיע ערך התכונה seconds של השיר.
2. **בעיה ראשונה**: הוראת הזריקה מסומנת בשגיאה
פתרון: השתמשו במנגנון של eclipse כדי להוסיף לכותרת המתודה את ההכרזה throws Exception
3. **בעיה שניה**: ההוראה במתודה ***play*** מסומנת בשגיאה
פתרון: השתמשו במנגנון של eclipse כדי להוסיף בלוק try/catch. מדוע נבחר באופציה הזו?
4. **בעיה שלישית**: במחלקה **PlayList** במתודות ***playSong*** ישנן הוראות המסומנות בשגיאה
פתרון: השתמשו במנגנון של eclipse כדי להוסיף לכותרת המתודה את ההכרזה throws Exception
5. ב - T1Main הסירו את ההערה מהוראה זו והריצו התכנית. מה מופיע? לאחר ההרצה שימו ההוראה בהערה

ג. ההוראה הבאה תגרום לשגיאת ריצה - כי ברשימה לא קיים שיר שזהו שמו. מהי החריגה המופיעה?

```
pl.playSong("S3", 0);
```

1. נתרגל הגדרת חריגה **מותאמת אישית**
2. הוסיפו לחבילה T12 את קובץ המחלקה **MissingSongException** היורשת מהמחלקה **Exception**
3. במחלקה **PlayList** במתודה ***playSong*** הוסיפו הוראה של זריקת עצם חדש מהמחלקה **MissingSongException** המתרחשת כאשר שם השיר שהועבר בפרמטר אינו מופיע ברשימת השירים. הודעת השגיאה תהיה:

"There is no song in this playlist whose name is <songName>"

כאשר במקום <songName> יופיע ערך הפרמטר songName.

4. מדוע לא נדרש לשנות את כותרת המתודה ***playSong***?

תשובה: מכיוון שעצם השגיאה שנזרק הוא מסוג MissingSongException שהוא סוג של Exception

5. ב - T1Main הסירו את ההערה מהוראה זו. מחקו מכותרת ה- main את ההכרזה throws Excption
6. **בעיה**: כמעט כל ההוראות מסומנות בשגיאה

פתרון: השתמשו במנגנון של eclipse כדי להוסיף בלוק try/catch אחד.

העבירו אל בלוק ה- try את כל ההוראות <u>למעט</u> הוראת יצירת רשימת הנגינה שתופיע לפני בלוק ה- try

7. הריצו התכנית.
8. הוסיפו בלוק catch שני לשגיאה מסוג MissingSongException. מופיע סימון שגיאה. מדוע?

תשובה: בלוק ה- Catch הראשון "ימסך" ויקלוט אליו גם שגיאות מהסוג MissingSongException.

9. העבירו את בלוק ה- catch החדש כך שיופיע מיד לאחר בלוק ה- catch. בתוכו הדפיסו את e.getMessage()
10. הוסיפו ב- main בלוק של finally ובו הוראה מתאימה לניגון כלל השירים ברשימת השירים

<u>חלק ג' - השלימו עצמאית דוגמאות נוספות לאופן טיפול בחריגות</u>

סעיף זה דומה לסעיף ג של חלק ב' - ותת הסעיפים שלו יושלמו בהמשך

ד. גם בהוראה הבאה ישנה שגיאה לוגית - כי מספר השיר המבוקש (5) גבוה ממספר השירים ברשימה (2)

```
pl.playSong(5, 100);
```

עוד תרגילים בסיסיים בנושא חריגות ב- java תוכלו למצוא בקישור הבא:

https://personales.unican.es/corcuerp/java/Labs/LAB_15.htm

תרגיל 2

א. הוסיפו לפרויקט את המחלקה האבסטרקטית **צורה** (Shape) וכתבו בה את:

1. התכונה שם הצורה (name, מחרוזת).
2. בנאי מלא המקבל פרמטר עבור התכונה name
3. מתודה אבסטרקטית לחישוב שטח הצורה

```
public abstract double area();
```

4. מתודה אבסטרקטית להדפסת ערכי התכונות הייחודיות לכל מחלקה יורשת

```
public abstract void show();
```

5. המתודה print המדפיסה את שם הצורה, שטחה (על ידי זימון של area) וערכי התכונות (על ידי זימון של show)

```
public abstract class Shape{
        private String name;

        public Shape(String name) {
                this.name = name;
        }

        public abstract double area();
        public abstract void show();

        public void print() {
          System.out.println(name+":");
          show();
          System.out.println("Area = "+ area());
        }
      }
```

ב. הוסיפו לפרויקט את המחלקה IllegalNegativeValueException היורשת מ- Exeption ומוסיפה להודעה מחרוזת "value must be positive".

ג. הוסיפו לפרויקט את המחלקה **מלבן** (Rectangle) היורשת מ- Shape וכתבו בה את:

1. התכונות רוחב (width, שלם) וגובה (height, שלם)
2. דריסת ומימוש המתודות האבסטרקטיות מהמחלקה Shape

```
public class Rectangle extends Shape{

        private double a, b;

        @Override
        public double area() {
         return a*b;
        }

        @Override
        public  void show() {
          System.out.println("a = " + a + ", b = " +  b);
        }
}
```

3. מתודות set לתכונות רוחב וגובה.

המתודות תזרוקנה חריגה IllegalNegativeValueException כאשר רוחב וגובה המלבן קטן או שווה לאפס.

4. בנאי מלא המזמן את הבנאי של מחלקת הבסיס.

הבנאי יזרוק חריגה IllegalNegativeValueException כשערך אחד הפרמטרים עבור רוחב וגובה המלבן קטן או שווה לאפס.

5. לרכז את הלוגיקה של זריקת IllegalNegativeValueException במקום אחד - פעולת עזר סטטית במחלקת Shape.

ד. הוסיפו לפרויקט את המחלקה **עיגול** (Circle) היורשת מ- Shape וכתבו בה את:

1. התכונה רדיוס (radius, שלם), מיקום מרכז המעגל על הציר האופקי (x, שלם) ועל הציר האנכי (y, שלם)

2. דריסת ומימוש המתודות האבסטרקטיות מהמחלקה Shape

```
public class Circle extends Shape{

        private double x, y, radius;
        @Override
        public double area() {
         return Math.PI * radius * radius;
        }

        @Override
        public void show() {
         System.out.println("center = (" + x + ", "+ y+ "), radius = " + radius);
        }
}
```

3. מתודת set לתכונת רדיוס.

המתודה תזרוק חריגה IllegalNegativeValueException כאשר רדיוס קטן או שווה לאפס.

4 . בנאי מלא המזמן את הבנאי של מחלקת הבסיס.

הבנאי יזרוק חריגה IllegalNegativeValueException כאשר רדיוס קטן או שווה לאפס.

ה. הוסיפו לחבילה T12 קובץ מחלקה T2Main עם פונקציה ראשית (main) וכתבו בה הוראות מתאימות:

1. צרו עצם של מלבן או עיגול.
2. בעת יצירת העצם גרמו לשגיאה כלשהי על ידי זימון הבנאי עם ערך שלילי באחד הפרמטרים
3. השתמשו ב- try/catch כדי ללכוד את השגיאות ולטפל בהן על ידי העברת ערך תקין כפרמטר לבנאי

תרגיל 3

מטריצה היא מבנה נתונים מלבני בו תאים מסודרים בשורות.

נגדיר את **הממשק Matrix** עם כותרות המתודות הבאות:

מה מבצעת	כותרת המתודה
מחזירה את מספר השורות במטריצה	int **getRowsNum**()
מחזירה את מספר העמודות במטריצה	int **getColumnsNum**()
מחזירה את הערך במטריצה בתא ה-j של השורה ה- i. זורקת IllegalArgumentException אם i או j חורגים ממימדי המטריצה	int **get**(int i, int j)
מחזירה מטריצה חדשה של סכומי ערכים במיקומים זהים במטריצה m ובמטריצה הנוכחית. זורקת IllegalArgumentException אם מימדי המטריצה m והמטריצה הנוכחית שונים.	Matrix **add**(Matrix m)
יוצרת ומחזירה מטריצה חדשה שערכיה הן המכפלה של ערכי המטריצה הנוכחית ב- k.	Matrix **mult**(int k)

א. הוסיפו לפרויקט את הממשק **Matrix** וכתבו בו את כותרות המתודות המתוארות בטבלה שלעיל

ב. הוסיפו לפרויקט את המחלקה **NaiveMatrix** הממשת את הממשק **Matrix** וכתבו בה:

1. בנאי המקבל כפרמטרים את מספר השורות (n, שלם) ואת מספר העמודות (m, שלם)
   בהתאם לערכי הפרמטרים n ו - m הבנאי ייצור מטריצה ויאתחל את כל הערכים שלה ל-1.
   הבנאי יזרוק IllegalArgumentException אם המימדים אינם חיוביים
   חשבו אילו תכונות נדרשות להגדיר במחלקה NaiveMatrix.
2. דרסו את המתודה toString כך שתחזיר מחרוזת המכילה את כל ערכי המטריצה.
   בין כל זוג ערכים באותה שורה יופיע התו פסיק ובין כל שתי שורות יופיע התו '\n'
3. ממשו את המתודות של הממשק

ג. הוסיפו לחבילה T12 קובץ מחלקה T3Main עם פונקציה ראשית (main) וכתבו בה הוראות מתאימות:

1. צרו עצם מהמחלקה **NaiveMatrix** כך שישקף מטריצה ריבועית בגודל 5*5
2. הדפיסו את נתוני המטריצה
3. זמנו המתודות get ו - add עם ערכים חריגים