מטלת מנחה (ממ"ן) 13

הקורס: 20441 - מבוא למדעי המחשב ושפת Java

חומר הלימוד למטלה: יחידות 5 – 6 **נושא המטלה:** לולאות ומערכים

מספר השאלות: 1 **משקל המטלה:** 5 נקודות

סמסטר: 2022ב **מועד אחרון להגשה:** 23.4.2022

(ת)

במטלה זו אנו משתמשים במחלקות Time1 ו- Flight שכתבנו בממ"ן 12.

אתם יכולים להשתמש במחלקות Time1 ו- Flight **שכתבתם** או בקבצים Time1.class ו- Flight.class שיהיו באתר ביחידה 6 בצמוד למטלה 13. נשים את הקבצים האלו באתר רק אחרי ההגשה של מטלה 12.

שימו לב שהקבצים שיש באתר הם קובצי class ולא הקוד ב- Java. אי אפשר לפתוח אותם אלא להשתמש בהם בלבד. לצידם יש קובצי html שהם ה- API של המחלקות Time1 ו- Flight.

אנא קראו את הכתוב במדריך **מדריך עזר קצר לשימוש בקובצי מחלקות (class) הקיימים בפרויקט שאתם יוצרים ב-Blue**. המדריך נמצא בלשונית "מדריכי עזר" ביחידה 3 באתר הקורס. כך תדעו איך להשתמש במחלקה שכבר כתובה, וניתנת לכם כקובץ class ללא הקוד.
שמנו באתר טסטר בסיסי לבדיקה ראשונית של המטלה. **חובה** להריץ את המטלה מול הטסטר ולבדוק שאין טעויות קומפילציה.

שאלה 1 - להרצה (100%)

המחלקה Airport מייצגת את לוח הטיסות בשדה התעופה בימה.

הייצוג נעשה על-ידי מערך ששומר את רשימת הטיסות. התכונות במחלקה הן:

- מערך של הטיסות Flight [] _flightsSchedule
- מספר הטיסות בלוח הטיסות (במערך) int _noOfFlights
- שם העיר בה נמצא שדה התעופה String _city

כמו כן קיים במחלקה קבוע שלם MAX_FLIGHTS המציין את המספר המקסימלי של טיסות בימה– 200.

הטיסות (כלומר האובייקטים מהמחלקה Flight) נמצאים במערך ברצף, **ללא "חורים" מתחילת המערך. המערך צריך להישאר כך (ללא חורים) לאחר כל פעולה.**

שימו לב, הטיסות במערך לא ממוינות ואין חשיבות לסדר הופעתן.

עליכם לכתוב את המימוש ב- Java של המחלקה Airport. מימוש המחלקה כולל את הסעיפים שלהלן:

1. הגדרת הקבוע/קבועים של המחלקה.
2. הגדרת התכונות של המחלקה.
3. בנאי שמקבל את שם העיר בה נמצא שדה התעופה ומאתחל את תכונות המחלקה כך שמערך הטיסות יהיה בגודל מקסימלי ומספר הטיסות הוא 0.
4. שיטה (addFlight) בוליאנית המוסיפה טיסה ללוח הטיסות. השיטה מקבלת את **הטיסה** כפרמטר. השיטה מחזירה ערך true אם ההוספה התבצעה כשורה, אם לא, השיטה תחזיר false. שימו לב שהמקור או היעד של הטיסה חייבים להיות העיר בה נמצא שדה התעופה. אתם יכולים להניח שהטיסה הזו לא קיימת כבר בלוח הטיסות. אין צורך לבדוק זאת.
5. שיטה (removeFlight) בוליאנית המוחקת טיסה מלוח הטיסות. השיטה מקבלת את **הטיסה** כפרמטר. השיטה מחזירה ערך true אם המחיקה התבצעה כשורה, אם לא, השיטה תחזיר false.
6. שיטה (firstFlightFromOrigin) המקבלת עיר כלשהי place, מחזירה את הזמן בו ממריאה הטיסה הראשונה באותו יום מהמקום place. אם אין אף טיסה באותו יום מהמקום place יוחזר null.
7. שיטה (howManyFullFlights) המחזירה מספר האומר כמה טיסות מלאות יש באותו יום.
8. שיטה (howManyFlightsBetween) המקבלת עיר place ומחזירה מספר האומר כמה טיסות יש באותו יום הממריאות משדה התעופה _city ונוחתות ב- place, או ממריאות מ- place ונוחתות ב- _city .
9. שיטה (mostPopularDestination) המחזירה את העיר הכי פופולרית באותו יום (כלומר העיר בה נוחתות הכי הרבה טיסות). אם אין טיסות בכלל בלוח הטיסות, יוחזר null. אם יש כמה ערים שהן פופולריות ביותר באותה מידה, תוחזר העיר הפופולרית הראשונה שנמצאה בלוח הטיסות.
10. שיטה (mostExpensiveTicket) המחזירה את **הטיסה** שהכרטיס שלה הוא היקר ביותר. אם אין טיסות בכלל בלוח הטיסות, יוחזר null. אם יש כמה טיסות שהן יקרות ביותר באותה מידה, תוחזר הטיסה היקרה ביותר הראשונה שנמצאה בלוח הטיסות (כלומר, מתחילת המערך).
11. שיטה (longestFlight) המחזירה את **הטיסה** הארוכה ביותר במערך הטיסות. אם אין טיסות בכלל בלוח הטיסות, יוחזר null. אם יש כמה טיסות שהן ארוכות ביותר באותה מידה, תוחזר הטיסה הארוכה ביותר הראשונה שנמצאה בלוח הטיסות (כלומר, מתחילת המערך).
12. שיטה (toString) המחזירה מחרוזת המתארת את כל הטיסות במערך הטיסות כסדרן לפי המערך (אין צורך למיין לפי זמנים או משהו אחר). כל טיסה תהיה בשורה נפרדת. ובתחילה תהיה כותרת. אם אין טיסות בכלל בלוח הטיסות, יוחזר null.

ראו את הדוגמא הבאה:

The flights for airport Tel-Aviv today are:

Flight from Tel-Aviv to London departs at 12:00. Flight is full.

Flight from New York to Tel-Aviv departs at 10:50. Flight is full.

Flight from Tel-Aviv to Paris departs at 11:35. Flight is not full.

בכל השיטות לעיל, אם מועבר אובייקט כפרמטר, אפשר להניח שהוא אינו null.

שימו לב לא לבצע aliasing במקומות המועדים.

תזכורת – השוואה בין אובייקטים (ומחרוזות) צריכה להיעשות בעזרת השיטה equals ולא על-ידי סימן השוויון ==.

מותר להוסיף שיטות נוספות (פרטיות בלבד), לפי ראות עיניכם, אבל אי אפשר להוסיף תכונות נוספות. כמו כן, כל התכונות חייבות להיות פרטיות!

לפניכם רשימת החתימות של הבנאי ושיטות המחלקה:

- `public Airport(String city)`
- `public boolean addFlight(Flight f)`
- `public boolean removeFlight(Flight f)`
- `public Time1 firstFlightFromOrigin (String place)`
- `public String toString()`
- `public int howManyFullFlights()`
- `public int howManyFlightsBetween (String city)`
- `public String mostPopularDestination()`
- `public Flight mostExpensiveTicket()`
- `public Flight longestFlight()`

אתם צריכים לכתוב בעצמכם API למחלקה, לבנאים ולשיטות לפי הנהוג בכתיבת API. כמו כן, עליכם לתעד בתיעוד פנימי כל מה שדורש הבהרה ואינו פשוט.

שימו לב,

באתר הקורס תמצאו גם טסטר לבדיקת האיות והפרמטרים של השמות של השיטות והמחלקה שאתם צריכים לכתוב. חובה עליכם לבדוק את המחלקה שכתבתם בטסטר זה, ולהגיש אותה רק אם הטסטר עובר קומפילציה. שימו לב שהטסטר לא מכסה את כל האפשרויות, ובפרט לא את מקרי הקצה. הוא רק בודק את השמות של השיטות במחלקות כלומר שגיאות קומפילציה. מאד מומלץ להוסיף לו בדיקות. **אם יש שיטה שלא כתבתם, כתבו חתימה והחזירו ערך סתמי כדי שהטסטר ירוצו עם המחלקה ללא שגיאות קומפילציה. אם הטסטר לא ירוץ ללא שגיאות קומפילציה הציון במטלה יהיה אפס.**

הגשה

1. הגשת הממ״ן נעשית בצורה אלקטרונית בלבד, דרך מערכת שליחת המטלות.
2. הקפידו לתעד בתיעוד פנימי וב- API את כל השיטות שיש במחלקות השונות.
3. הקפידו ששמות השיטות יהיו בדיוק כפי שכתוב במטלה. וכן שההדפסות יהיו בדיוק כפי שמופיע במטלה.
4. עליכם להגיש את הקובץ Airport.java, עטפו אותו בקובץ zip ושלחו. אין לשלוח קבצים נוספים.