מאגר מעבדות לשפות התכנות החדשות

ניתן להשתמש בחומרים לצורך הוראה בלבד.

לא ניתן לפרסם את החומרים או לעשות בהם כל שימוש מסחרי

ללא קבלת אישור מד"ר תמר פז.

המעבדה בקובץ זה מיועדת לתלמידים הלומדים מדעי המחשב בשפת התכנות ג'אווה והיא מותאמת לסביבת אקליפס.

המעבדה מיועדת לשיעורי המעבדה והיא מבוססת על שיטת ההוראה לפיה הלימוד של כל נושא חדש ייפתח בהתנסות אישית במעבדה. לאחריה, יבוא דיון כיתתי, שבעקבותיו ייפתרו משימות שונות.

המעבדה מתרכזת בהעמקה בנושא של מחלקות ועצמים.

עצם כפרמטר ..... עמוד 2

עצם כערך מוחזר ..... עמוד 5

פעולה בונה מעתיקה ..... עמוד 10

עצם מורכב ..... עמוד 12

מערך כתכונה ..... עמוד 17

תכונות מופע ותכונות מחלקה ..... עמוד 23

פעולות מופע ופעולות מחלקה ..... עמוד 28

מערך: התכונה Length ..... עמוד 34

המעבדה מיועדת להעמקה עם הנושא של מחלקות ועצמים לאחר היכרות ראשונית עם הנושא (מעבדות 8,9).

במעבדה נעשה שימוש במחלקות Bucket ו- Date כפי שנכתבו במעבדות 8,9. בעמוד הבא מובאים ממשקי המחלקות.

ממשק המחלקה Bucket:

תיאור הפעולה	החתימה של הפעולה
פעולה בונה: יוצרת עצם מטיפוס Bucket על פי פרמטרים נתונים. הנחה: הפרמטרים אינם שליליים.	Bucket (double capacity, double currentAmount)
מחזירה את הקיבולת של הדלי.	double getCapacity ()
מחזירה את הכמות הנוכחית של הדלי.	double getCurrentAmount ()
מקבלת כמות להוספה ומוסיפה אותה לכמות הנוכחית בדלי. במידה והתוספת תגרום ל"גלישה", הפעולה תוסיף לדלי רק עד לקיבולת שלו ותפלוט הודעה מהי הכמות העודפת. הנחה: הקלט איננו שלילי.	void addAmount (double amount)
מקבלת כמות לריקון ומרוקנת אותה מהכמות הנוכחית בדלי עליו היא פועלת. אם הכמות לריקון גדולה מהכמות הנוכחית, הפעולה תרוקן את כל הדלי. הנחה: הקלט איננו שלילי.	void removeAmount (double amount)
מרוקנת את הדלי עליו היא פועלת.	void emptyAll()
מחזירה true אם הדלי ריק, false אחרת.	boolean isEmpty()
ממלאת את הדלי בכל הקיבולת שלו.	void fillAll()
מחזירה true אם הדלי מלא בכמות השווה לקיבולת שלו, false אחרת.	boolead isFull()
מחזירה מחרוזת הכוללת את השמות של התכונות של עצם מטיפוס דלי ואת הערכים שלהם עבור הדלי הנוכחי.	String toString()

ממשק המחלקה Date:

תיאור הפעולה	החתימה של הפעולה
פעולה בונה: יוצרת עצם מטיפוס Date על פי פרמטרים נתונים. הנחה: day מספר שלם 1-31 , month מספר שלם 1-12, year שלם וחיובי.	Date (int day , int month , int year)
פעולה בונה: יוצרת עצם מטיפוס Date בהתאם לנתונים שייקלטו מהמשתמש.	Date()
מחזירה את השנה.	int getYear ()
מחזירה את החודש.	int getMonth ()
מחזירה את היום.	int getDay ()
מחזירה את מספר הימים בשנה (366 אם השנה מתחלקת ב- 4 ללא שארית. 365 אחרת).	int daysInYear()
מחזירה true אם התאריך "מעניין", ו- false אחרת. תאריך "מעניין" הוא תאריך בו היום שווה לחודש ובנוסף, אם היום הוא בן ספרה אחת אז היום שווה גם לספרה האחרונה של השנה, ואם היום הוא בן שתי ספרות אז היום שווה גם לשתי הספרות האחרונות של השנה (תאריכים "מעניינים": 6.6.2006 , 12.12.1912).	boolean isInteresting()
מחזירה את מספר הימים שחלפו מתחילת השנה עד לתאריך הנוכחי. למשל אם התאריך הנוכחי הוא 6.3.2006 אז מספר הימים שחלפו הוא: 31+28+6=65 (ינואר 31, פברואר 28).	int daysPass()
מעדכנת את היום בהתאם לפרמטר הנתון. הנחה: הערך של הפרמטר הוא מספר שלם 1-31	void setDay (int newDay)
מעדכנת את החודש בהתאם לפרמטר הנתון. הנחה: הערך של הפרמטר הוא מספר שלם 1-12	void setMonth (int newMonth)
מחזירה מחרוזת המתארת את התאריך בצורה הבאה: שנה / חודש / יום	String toString()

© כל הזכויות שמורות, תמר פז

עצם כפרמטר

משימה 1 – חלק א'

כל הפעולות שהגדרנו עד כה במחלקה Bucket פעלו על עצם מטיפוס Bucket וקיבלו פרמטרים מטיפוסים המוגדרים בשפה (למשל, double). אפשר גם להגדיר פעולה שתקבל כפרמטר עצם ששייך למחלקה אחרת או עצם ששייך למחלקה הנוכחית.

נוסיף כעת למחלקה **Bucket** פעולה pourInto ("שפוך לתוך") שתקבל כפרמטר עצם מטיפוס Bucket. כלומר, הפעולה pourInto תתייחס **לשני עצמים** מטיפוס Bucket. הראשון, הדלי הנוכחי עליו פועלת הפעולה, והשני, הדלי שיתקבל כפרמטר.

הפעולה תעביר לתוך הדלי שיתקבל כפרמטר את הכמות המקסימלית שאפשר להעביר **מתוך** הדלי הנוכחי (הדלי עליו פועלת הפעולה). למשל, אם הקיבולת של הדלי שיתקבל כפרמטר היא 10 והכמות הנוכחית בו היא 7, והכמות הנוכחית של הדלי הנוכחי היא 8, אזי הפעולה תדאג להעביר 3 ליטר מהדלי הנוכחי לדלי שיתקבל כפרמטר. כלומר, לאחר ביצוע הפעולה: הדלי שיתקבל כפרמטר יהיה מלא (יהיו בו 10 ליטרים) ובדלי הנוכחי יהיו 5 ליטרים. אבל עבור אותו דלי שיתקבל כפרמטר, אם הכמות הנוכחית של הדלי הנוכחי היא 2, אזי הפעולה תדאג להעביר את 2 הליטרים מהדלי הנוכחי לדלי שיתקבל כפרמטר ולאחר ביצוע הפעולה, בדלי שיתקבל כפרמטר יהיו 9 ליטרים והדלי הנוכחי יהיה ריק.

לפניכם שלד של הפעולה.

```
public void pourInto (Bucket otherBucket) // Bucket הפרמטר שהפעולה תקבל יהיה מטיפוס otherBucket
{
  // otherBucket הוא עצם מטיפוס Bucket ולכן ניתן להפעיל עליו את הפעולות המוגדרות במחלקה Bucket
   double freeSpace = otherBucket.getCapacity()-otherBucket.getCurrentAmount();
   // otherBucket הוא המקום הפנוי בדלי freeSpace
   if (this.currentAmount<freeSpace) // otherBucket אם הכמות הנוכחית בדלי הנוכחי קטנה מהמקום הפנוי ב-
   { // otherBucket ניקח את כל הכמות הנוכחית מהדלי הנוכחי ונוסיף אותה ל-
     otherBucket.addAmount (this.currentAmount);
     this.currentAmount =_____ ;
   }
  else  // otherBucket-הכמות הנוכחית בדלי הנוכחי גדולה מהמקום הפנוי ב
    { // otherBucket-ניקח מהדלי הנוכחי רק כמות השווה למקום הפנוי ב- otherBucket ונוסיף אותה ל
      otherBucket.addAmount (freeSpace);
      this.currentAmount = this.currentAmount - _____;
    }
}
```

תזכורת: this = התייחסות לדלי הנוכחי

- השלימו את הפעולה והוסיפו אותה למחלקה Bucket

כאשר פעולה מקבלת כפרמטר עצם ששייך למחלקה הנוכחית היא מתייחסת לשני עצמים מטיפוס המחלקה. ההתיחסויות לעצם הנוכחי הן באמצעות הקידומת this. למשל this.currentAmount וההתיחסויות לעצם שהתקבל כפרמטר היא באמצעות שם הפרמטר. למשל, _____

© כל הזכויות שמורות, תמר פז

תזכורת: ממשק המחלקה Bucket לאחר השינויים שבצעתם בו עד כה:

תיאור הפעולה	החתימה של הפעולה
פעולה בונה: יוצרת עצם מטיפוס Bucket על פי פרמטרים נתונים. הנחה: הפרמטרים אינם שליליים.	Bucket (double capacity, double currentAmount)
מחזירה את הקיבולת של הדלי.	double getCapacity ()
מחזירה את הכמות הנוכחית של הדלי.	double getCurrentAmount ()
מקבלת כמות להוספה ומוסיפה אותה לכמות הנוכחית בדלי. במידה והתוספת תגרום ל"גלישה", הפעולה תוסיף לדלי רק עד לקיבולת שלו ותציג כפלט הודעה מהי הכמות העודפת. הנחה: הקלט איננו שלילי.	void addAmount (amount)
מקבלת כמות לריקון ומרוקנת אותה מהכמות הנוכחית בדלי עליו היא פועלת. אם הכמות לריקון גדולה מהכמות הנוכחית, הפעולה תרוקן את כל הדלי. הנחה: הקלט איננו שלילי.	void removeAmount (amount)
מרוקנת את הדלי עליו היא פועלת.	void emptyAll()
מחזירה true אם הדלי ריק, false אחרת.	boolean isEmpty()
ממלאת את הדלי בכל הקיבולת שלו.	void fillAll()
מחזירה true אם הדלי מלא בכמות השווה לקיבולת שלו, false אחרת.	boolean isFull()
מעבירה לתוך הדלי שהתקבל כפרמטר את הכמות המקסימלית שאפשר להעביר מתוך הדלי הנוכחי.	void pourInto (bucket otherB)
מחזירה מחרוזת הכוללת את השמות של התכונות של עצם מטיפוס דלי ואת הערכים שלהם עבור הדלי הנוכחי.	String toString()

משימה 1 – חלק ב'

- כיתבו מחלקה עם פעולה ראשית (main) שתבדוק את הפעולה pourInto. (הקפידו לכתוב את המחלקה החדשה בתוך הפרויקט BucketProject). הפעולה הראשית תבצע:
  1. תיצור שני דליים בהתאם לנתונים שיקלטו מהמשתמש.
  2. תבדוק מיהו הדלי בעל הכמות הנוכחית הקטנה יותר ותעביר אליו את הכמות המקסימלית האפשרית מהדלי האחר. רמז: דוגמה להפעלה (זימון) של הפעולה pourInto: **b1.pourInto(b2)** הפעלה זו תגרום להעברת הכמות המקסימלית שניתן מ b1 (הנוכחי) אל b2 (הפרמטר).
  3. תציג כפלט את התכונות של שני הדליים ואת ערכיהן.
- הריצו את התכנית שלוש פעמים עבור הנתונים הבאים ובדקו שהתקבלו הפלטים המבוקשים.

מספר הרצה	דלי ראשון: קיבולת	דלי ראשון: כמות נוכחית	דלי שני: קיבולת	דלי שני: כמות נוכחית
1	20	12	20	10
2	20	8	20	5
3	20	8	20	12

© כל הזכויות שמורות, תמר פז

גם עצם יכול להיות פרמטר של פעולה!

כאשר עצם משמש כפרמטר של פעולה, בתוך קובץ המחלקה **ההתיחסות אליו איננה באמצעות this משום שהוא הפרמטר של הפעולה ולא העצם עליו הפעולה פועלת.**

משימה 2

בסרט "מת לחיות 2", מופיעה המשימה הבאה: נתונים שני דליים ובריכה מלאה. הקיבולת של הדלי הראשון היא 5 ליטרים, והקיבולת של הדלי השני היא 7 ליטרים. המטרה: למלא את אחד מהדליים (כרצונכם) ב- 4 ליטרים בדיוק. מותר לבצע רק את הפעולות הבאות:

1. למלא דלי בכמות של כל הקיבולת שלו.
2. להעביר מים מדלי אחד לדלי אחר עד שהדלי ממנו מעבירים מתרוקן או עד שדלי אליו מעבירים מלא בכל הקיבולת שלו.

כתבו פעולה ראשית שתבצע את המשימות הבאות:

- תייצר שלושה דליים: b5 בעל קיבולת של 5 ליטרים וכמות נוכחית 0, b7 בעל קיבולת של 7 ליטרים וכמות נוכחית 0, ו- pool שייצג את הבריכה שהקיבולת שלה 1000 והכמות נוכחית 1000.
- תכתוב את ההוראות הדרושות לביצוע המשימה, כולל הדפסה של הכמות הנוכחית של הדליים בסיום ביצוע המשימה. הקפידו להשתמש רק בפעולות המותרות.
  ההדפסות תהיינה מלוות בהודעות מתאימות.

הקלידו את הפעולה במחלקה חדשה, שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

משימה 3 – חלק א'

חיזרו כעת למחלקה **Date** והוסיפה לה את הפעולה before שתקבל כפרמטר עצם מטיפוס Date . הפעולה תחזיר true אם התאריך הנוכחי קודם לתאריך שהועבר כפרמטר, ותחזיר false אחרת. למשל אם התאריך הנוכחי הוא 12.7.2006 והתאריך שהתקבל כפרמטר הוא 17.8.2006 הפעולה תחזיר true. כלומר, הפעולה before תתייחס **לשני עצמים** מטיפוס Date: התאריך הנוכחי עליו הפעולה פועלת, והתאריך שיתקבל כפרמטר.

החתימה של הפעולה: **public boolean before (Date otherDate)**

כיתבו את הפעולה. זיכרו להתייחס לתאריך הנוכחי בלבד באמצעות this .

משימה 3 – חלק ב'

- כתבו מחלקה עם פעולה ראשית שתבדוק את הפעולה החדשה before. הפעולה תיצור שני עצמים מטיפוס Date (בהתאם לנתונים שיוקלדו על-ידי המשתמש), ותפעיל עליהם את הפעולה before.
- הקלידו את הפעולה במחלקה חדשה (הקפידו לפתוח מחלקה חדשה כחלק מהפרויקט **DateProject**),
- הריצו את הפעולה מספר פעמים כך שתבדוק מקרים שונים: כאשר שני התאריכים באותו חודש ובאותה שנה אבל היום שונה, כאשר החודש שונה וכאשר השנה שונה. כמו כן דאגו להריץ גם מקרים בהם פעולה תחזיר true וגם מקרים בהם בפעולה תחזיר false .

© כל הזכויות שמורות, תמר פז

עצם כערך מוחזר

משימה 4 – חלק א'

כל הפעולות שכתבנו עד כה במחלקות שמגדירות עצם, לא החזירו ערך או שהחזירו ערך מטיפוס שהוגדר בשפה. פעולה במחלקה יכולה להחזיר גם ערך מטיפוס המחלקה עצמה. נוסיף כעת למחלקה Date את הפעולה **nextDate** שתחזיר את התאריך העוקב לתאריך הנוכחי. למשל, אם התאריך הנוכחי הוא 1.8.2006 אזי הפעולה תבנה ותחזיר עצם מטיפוס Date ותיתן לתכונותיו את הערכים: 2.8.2006. ואם התאריך הנוכחי הוא 31.12.2006 אזי הפעולה תבנה ותחזיר עצם מטיפוס Date ותתן לתכונתיו את הערכים: 1.1.2007

לפניכם שלד של הפעולה. השלימו אותו.

```
public Date nextDate()
{
//      נאתחל את d,m,y בתכונות של התאריך הנוכחי. לאחר מכן, אם אנחנו לא בסוף חודש, נוסיף 1 ל d.
//      אם אנחנו בסוף חודש, נעבור לתחילת החודש הבא (נציב 1 ב d, ונוסיף 1 ל m).
//      אם כתוצאה מהשינוי הגענו לחודש 13 (סימן שהתאריך המקורי היה היום ה _____),
//      נעבור לתחילת השנה הבאה: נציב 1 ב m, ונוסיף 1 לשנה
    int d,m,y;
    d=this.day;
    y=this.year;
    m=this.month;
    if (m ==2 && (y%4 != 0)&& d<28)   d++;  // בשנה הנוכחית, פברואר הוא 28 יום.
    else if (m ==2 && y%4 ==0 && _____)  d++;  // בשנה הנוכחית, פברואר הוא 29 יום

        else if ((m==4 || m==6 || m==9 || m==11)&& d<30)  d++;
            else if ((m==1 || m==3 || m==5 || m==7 || m==8 || m==10 || m==12)&& _____)
                d++;
              else { \\ התאריך הנוכחי הוא יום האחרון של חודש
                    d=1;
                    _____;
                    if (m==13)  \\ עוברים לשנה חדשה
                     {
                        m=1;
                        y++;
                     }
                }
    Date d2= new Date (d,m,y);
    return d2;
}
```

הוסיפו את הפעולה החדשה למחלקה Date

משימה 4 – חלק ב'

- פתחו מחלקה חדשה וכיתבו בה פעולה ראשית שתשתמש בפעולה nextDate. הפעולה תקלוט 20 תאריכים ותודיע כמה מהתאריכים הם "כמעט מעניינים". תאריך "כמעט מעניין" הוא תאריך שהיום העוקב לו הוא תאריך "מעניין" (על פי ההגדרה בממשק המחלקה Date. למשל 5.6.06).
- שימרו, הריצו ובדקו שהתקבל הפלט המבוקש.

© כל הזכויות שמורות, תמר פז

משימה 5 – חלק א'

לפניכם ממשק של המחלקה **Time**

כותרת הפעולה	תיאור הפעולה
Time (int hour, int minute , int second)	פעולה בונה: יוצרת עצם מטיפוס Time על פי פרמטרים נתונים. הנחה: hour שלם 0-23 , minute , second שלמים 0-59
Time ()	פעולה בונה: דואגת לקליטת נתונים תקינים מהמשתמש (שעה: מספר שלם 0-23, דקות ושניות: מספרים שלמים 0-59), ויוצרת עצם מטיפוס Time בהתאם לערכים שנקלטו.
int getHour()	מחזירה את השעה.
int getMinute()	מחזירה את הדקות.
int getSecond()	מחזירה את השניות.
void setHour (int newHour)	מעדכנת את השעה בהתאם לפרמטר הנתון. הנחה: הערך של הפרמטר שלם 0-23
void setMinute (int newMinute)	מעדכנת את הדקות בהתאם לפרמטר הנתון. הנחה: הערך של הפרמטר שלם 0-59
void setSecond (int newSecond)	מעדכנת את השניות בהתאם לפרמטר הנתון. הנחה: הערך של הפרמטר שלם 0-59
int firstTime (Time otherTime)	מקבלת כפרמטר עצם מטיפוס Time. מחזירה 1 אם העצם הנוכחי מוקדם יותר, 0 אם הזמנים שווים, 2 אם הפרמטר מוקדם יותר.
Time plus (int second)	מחזירה עצם מטיפוס Time שגדול ב- second שניות מהעצם הנוכחי. second יכול להיות גדול מ-60
String toString()	מחזירה מחרוזת הכוללת את הזמן המדויק בצורה הבאה: ***שניות:דקות:שעה***

ממשו את המחלקה Time. כלומר, כתבו את ההוראות למימוש המחלקה.

יש לשלב במחלקה פעולת עזר getValidNum שתקבל מספר טבעי ותדאג לקלוט מהמשתמש מספר בתחום שבין 0 לבין המספר שקיבלה. הפעולה תחזיר את המספר הנקלט. למשל, אם הפעולה תקבל את הערך 59, היא תדאג לקלוט מהמשתמש מספר בתחום שבין 0 לבין 59 ותחזיר את הערך הנקלט.

תזכורות:

1. כדי לכתוב מחלקה חדשה צריך להגדיר תחילה פרויקט חדש (File -- New -- Project.. -- Next> -- הקלדת שם לפרויקט -- Next > -- Finish). ובפרויקט החדש יש להגדיר מחלקה חדשה בשם Time.
2. התכונות של המחלקה אינן חלק מהממשק משום שהתכונות הן פרטיות למחלקה והן לא מעניינו של מי שמשתמש במחלקה. אך קובץ המחלקה צריך להתחיל בתיאור התכונות.

© כל הזכויות שמורות, תמר פז

משימה 5 – חלק ב'

אורית החליטה לפתוח בצעידה יומית. אורית החליטה לצעוד כל יום 50 שניות יותר מאשר ביום הקודם לו. פיתחו מחלקה חדשה וכתבו בה פעולה ראשית שתקלוט את הזמן שאורית צעדה ביום הראשון (שעות, דקות ושניות). הפעולה תודיע כמה ימים (כולל היום הראשון) על אורית לצעוד עד שתצעד שעתיים או יותר. למשל, אם ביום הראשון אורית צעדה 1:55:50 (שעה, 55 דקות ו- 50 שניות), אזי הפעולה תודיע כי על אורית לצעוד 6 ימים (כיוון שביום השני היא תצעד 1:56:40, ביום השלישי 1:57:30 , ביום הרביעי 1:58:20, ביום החמישי 1:59:10, וביום השישי 2:00:00).

הנחיות:

1. יש להגדיר עצם מטיפוס Time ולהשתמש בפעולות הממשק.
2. בניית העצם תעשה באמצעות אחת מהפעולות הבונות כרצונכם.
3. הקפידו שלא לבנות עצם חדש עבור כל יום אלא לעדכן את העצם שבניתם עבור היום הראשון.

- שימרו, הריצו ובדקו שהתקבל הפלט המבוקש.

משימה 6

שני ילדים החליטו לקיים ביניהם תחרות ריצה. בכל סיבוב של התחרות יימדד הזמן בו כל אחד מהם עבר את המסלול ויקבע המנצח באותו סיבוב (זה שעבר את המסלול בפחות זמן). התחרות תסתיים כאשר מספר הניצחונות של שני הילדים יהיה שווה.

פיתחו מחלקה חדשה וכתבו בה פעולה ראשית שתקלוט את הזמן בו עבר כל אחד מהתלמידים את המסלול בסיבוב הראשון, תדפיס את הזמנים בצורה יפה (הערך המוחזר על-ידי הפעולה toString) כולל הודעות מתאימות, ותבדוק מיהו המנצח. לאחר מכן הפעולה תקלוט, תדפיס את הזמנים ותבדוק מיהו המנצח בסיבוב הבא. התהליך יפסק כאשר מספר הניצחונות של שני הילדים יהיה שווה. לסיום, הפעולה תציג כפלט את מספר הסיבובים שנערכו.

<u>רמז</u>: יש להגדיר עצם מטיפוס Time עבור כל ילד ולהשתמש בפעולות הממשק.

הקפידו שלא לבנות עצמים חדשים עבור כל סיבוב אלא לעדכן את העצמים שבניתם עבור הסיבוב הראשון.

- שימרו, הריצו ובדקו שהתקבל הפלט המבוקש.

© כל הזכויות שמורות, תמר פז

משימה 7 – חלק א'

לפניכם ממשק של המחלקה **Student**

תיאור הפעולה	כותרת הפעולה
פעולה בונה: יוצרת עצם מטיפוס Student הכולל שם של תלמיד וציון (מספר שלם 0-100). הפעולה קולטת את הערכים המתאימים מהמשתמש.	Student ()
פעולה בונה: יוצרת עצם מטיפוס Student בהתאם לפרמטרים נתונים. הנחה: garde מספר שלם 0-100.	Student (String name , int grade)
מחזירה את שם התלמיד.	String getName()
מחזירה את הציון.	int getGrade()
מעדכנת את השם של התלמיד ואת הציון שלו בהתאם לפרמטרים הנתונים.	void setAll (String newName, int newGrade)
הפעולה קולטת מהמשתמש שם של תלמיד וציון (מספר שלם 0-100), ומעדכנת את השם ואת הציון של התלמיד הנוכחי.	void setAll ()
מקבלת כפרמטר עצם מטיפוס Student. מחזירה עצם חדש מטיפוס Student שתכונותיו זהות לתכונות של התלמיד בעל הציון הגבוה יותר מבין הפרמטר והעצם הנוכחי.	Student greater (Student otherStudent)
מקבלת כפרמטר עצם מטיפוס Student. מחזירה עצם חדש מטיפוס Student שתכונותיו זהות לתכונות של התלמיד בעל הציון הנמוך יותר מבין הפרמטר והעצם הנוכחי.	Student smaller (Student otherStudent)
מחזירה מחרוזת הכוללת את התכונות של העצם בצורה הבאה: name = *שם התלמיד* , grade = *הציון*	String toString()

ממשו את המחלקה Student (זיכרו לפתוח תחילה פרויקט חדש).

יש לשלב במחלקה פעולת עזר getValidGrade שתהווה מסננת קלט הדואגת לקבל ציון חוקי (0-100).

הפעולה תדאג לקלוט מהמשתמש מספר בתחום שבין 0 לבין 100 ותחזיר את הערך הנקלט.

- איזו הרשאת גישה נתתם לפעולה getValidGrade ? _____ מדוע? _____

_____

© כל הזכויות שמורות, תמר פז

ראינו כבר כי ג'אווה מאפשרת להגדיר מספר פעולות בעלות אותו שם בתנאי שהן שונות זו מזו ברשימת הפרמטרים. השינוי יכול להתבטא במספר הפרמטרים ו/ או בטיפוסים שלהם. במקרה שלנו, הגדרנו במחלקה Student שתי פעולות בונות ששונות זו מזו במספר הפרמטרים, ושתי פעולות setAll שאף הן שונות זו מזו במספר הפרמטרים. בעת הזימון של הפעולה המהדר (הקומפיילר) יבצע את הפעולה המתאימה לרשימת הפרמטרים שתסופק לו.

משימה 7 – חלק ב'

כתבו פעולה ראשית שקולטת את השמות של 20 תלמידי הכיתה ואת הציונים שלהם בהסטוריה. הפעולה תציג כפלט את השם ואת הציון של התלמיד בעל הציון הגבוה ביותר. עליכם להשתמש בפעולות המחלקה Student. רמז: יש להגדיר שני עצמים מטיפוס Student, אחד עבור התלמיד הנוכחי בו מטפלים. ואחד עבור התלמיד בעל הציון הגבוה ביותר.

- שימרו, הריצו ובדקו שהתקבל הפלט המבוקש.

משימה 7 – חלק ג'

שנו את הפעולה הראשית כך שיודפסו גם הנתונים של התלמיד שהשיג את הציון הגבוה ביותר וגם הנתונים של התלמיד שהשיג את הציון השני.

רמז: חישבו כמה עצמים צריכים להגדיר כעת.

- שימרו, הריצו ובדקו שהתקבל הפלט המבוקש.

משימה 7 – חלק ד'

הוסיפו לפעולה הראשית את ההוראות הדרושות כך שיודפס גם מספר התלמידים שהציון שלהם נמוך מ-56.

הוסיפו תחילה פעולה למחלקה Student. הפעולה החדשה תקבל מספר ותבדוק האם הציון הנוכחי קטן ממנו. הפעולה תחזיר true אם הציון הנוכחי קטן מהמספר שיתקבל כפרמטר, ותחזיר false אחרת.

- שימרו, הריצו ובדקו שהתקבל הפלט המבוקש.

שימרו את המחלקה Student . נזדקק לה במשימה הבאה!

© כל הזכויות שמורות, תמר פז

פעולה בונה מעתיקה

משימה 8 – חלק א'

המחלקה הבאה היא הרחבה של משימה 7ב'.

הפעולה הראשית קולטת את השמות של 20 תלמידי הכיתה ואת הציונים שלהם בהסטוריה. הפעולה פולטת את השם ואת הציון של התלמיד בעל הציון הגבוה ביותר, ואת השם והציון של התלמיד בעל הציון הנמוך ביותר. פיתחו מחלקה חדשה (בפרויקט בו מוגדרת המחלקה Student), הקלידו בה את המחלקה הנוכחית, שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

```
public class FirstLastStudent
{
  public static void main(String[] args)
  {
      Student st = new Student ("",0);
      Student first = new Student ();
      Student last = new Student ("",0);
      last.setAll (first.getName(), first.getGrade());
      for (int i=  2 ; i<21; i++)
      {
        st.setAll();
        first = st.greater(first);
        last = st.smaller(last);
      }
      System.out.println("the first student is “+ first.toString());
      System.out.println("the last student is “+ last.toString());
   }
}
```

משימה 8 – חלק ב'

- תפקיד ההוראה: Student last = new Student ("",0);
  הוא ליצור עצם חדש מטיפוס Student ולהציב הפניה אליו במשתנה _____.
- מטרת ההוראה: last.setAll (first.getName(), first.getGrade());
  היא להעתיק לעצם שההפניה אליו נמצאת במשתנה last, את התכונות של העצם שההפניה אליו נמצאת במשתנה _____.

למעשה, רצינו ליצור **עותק** של העצם שההפניה אליו נמצאת במשתנה first, ולהציב במשתנה last הפניה לעצם החדש. עשינו זאת באמצעות העתקת הערכים של התכונות.

לעצם מטיפוס Student יש רק שתי תכונות אבל אילו היו לו יותר תכונות, העבודה היתה מייגעת.

ג'אווה מאפשרת ליצור עצם שהוא עותק של עצם קיים.

לשם כך, היכנסו לקובץ המחלקה Student **והוסיפו** בה את הפעולה הבונה הבאה.

```
public Student (Student otherSt)
{
   this.name = otherSt.name;
   this.grade = otherSt.grade;
}
```

- הפעולה הבונה החדשה מקבלת כפרמטר עצם מטיפוס _____.
- המילה this מתייחסת לעצם הנוכחי. כלומר, לעצם הנבנה על-ידי הפעולה.
- משמעות הביטוי: this.name = otherSt.name; היא: העתקת הערך של התכונה name של העצם שהתקבל כ_____, אל התכונה name של העצם הנוכחי.

© כל הזכויות שמורות, תמר פז

פעולה בונה שיוצרת עצם שהוא עותק של עצם קיים נקראת **פעולה בונה מעתיקה**.

מבנה של פעולה בונה מעתיקה:

```
public שם המחלקה (שם המחלקה משתנה)
{
  this.תכונה1 = משתנה.תכונה1;
   …
  this.תכונהn = משתנה.תכונהn;
}
```

משימה 8 – חלק ג'

שנו כעת את הפעולה הראשית כך שתשתמש בפעולה הבונה החדשה. (מחקו את שתי ההוראות המודגשות בפעולה הראשית שבסעיף א, ורשמו במקומן את ההוראה המודגשת).
שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

```
public class FirstLastStudent
{
  public static void main(String[] args)
  {
      Student st = new Student ("",0);
      Student first = new Student ();
      Student last = new Student (first);
      for (int i=  2 ; i<21; i++)
      {
        st.setAll();
        first = st.greater(first);
        last = st.smaller(last);
      }
      System.out.println("the first student is “+ first.toString());
      System.out.println("the last student is “+ last.toString());
   }
}
```

זימון של פעולה בונה מעתיקה:

```
שם המחלקה שם העצם החדש = new שם המחלקה(משתנה–הפניה לעצם קיים)
```

משימה 9

- היכנסו לקובץ המחלקה Date והוסיפו גם בה פעולה בונה מעתיקה. הפעולה הבונה החדשה תקבל עצם מטיפוס _____ ותבנה עצם חדש מטיפוס Date.
- צרו מחלקה חדשה בפרויקט בו נמצאת המחלקה Date וכתבו בה פעולה ראשית שתבדוק את הפעולה הבונה החדשה שהוספתם למחלקה Date.
- שימרו, הריצו ובדקו שהתקבל הפלט לו ציפיתם.

© כל הזכויות שמורות, תמר פז

עצם מורכב (עצם כתכונה)

משימה 10 – חלק א'

כל התכונות שפגשנו עד כה היו מטיפוסים שהוגדרו בשפה. אבל תכונה יכולה גם להיות מטיפוס מחלקה. למשל, למחלקה StudentAge יש שתי תכונות: שם התלמיד מטיפוס String ותאריך לידה מטיפוס Date.

- למחלקה StudentAge שתי פעולות בונות

איך מזהים פעולה בונה?

פעולה בונה היא פעולה ששמה הוא _____.

- הפעולה הבונה הראשונה מקבלת שני פרמטרים: name מטיפוס _____ ו- birthday מטיפוס _____.
- הפעולה הבונה השניה מקבלת _____ פרמטרים: שם התלמיד, יום, חודש ושנה.

```
public class StudentAge
{
  private String name;
  private Date birthday;

  public StudentAge (String name, Date birthday)
 {
     this.name = name;
     this.birthday = new Date (birthday);
 }
 public StudentAge (String name, int day , int month , int year)
 {
     this.name = name;
     this.birthday = new Date (day , month , year);
 }
 public String getName()
 {
    return this.name;
 }
 public Date getBirthday()
 {
    return this.birthday;
 }
 public String older (StudentAge otherStudent)
 {
     if (this.birthday.before(otherStudent.birthday)
          return this.name;
     else return otherStudent.name;
 }
 public String toString()
 {
    return (this.name+ " was born at "+ this.birthday.toString());
 }
}
```

פתחו מחלקה חדשה בשם StudentAge והקלידו בה את המחלקה הנוכחית.

מכיוון שהמחלקה משתמשת בעצם מטיפוס Date, היא צריכה להכיר את המחלקה Date, לכן, פיתחו את המחלקה החדשה בתוך הפרויקט המכיל את המחלקה Date

או: פיתחו פרויקט חדש, הקלידו בו את המחלקה החדשה ודאגו שהמחלקה Date תוכר בפרויקט החדש (עימדו עם העכבר על הפרויקט החדש, ולחצו על המקש הימני. בחלון שנפתח, בחרו ב- Properties, בצד השמאלי של החלון שנפתח כעת לחצו על Java Build Path. באותו חלון, בלשוניות למעלה לחצו על Libraries, ובצד הימני של אותו חלון לחצו על Add Class Folder. כעת נפתח חלון שמופיעים בו השמות של כל הפרויקטים שכתבתם עד כה, לחצו על הריבוע שמשמאל ל- DateProject (שימו לב שמופיע הסימן V). אשרו (OK) ואשרו גם את המסך הבא (שוב OK)).

© כל הזכויות שמורות, תמר פז

משימה 10 – חלק ב'

פתחו מחלקה חדשה (בתוך הפרויקט בו הקלדתם את המחלקה StudentAge) והקלידו בה פעולה ראשית שתיצור שני עצמים מטיפוס StudentAge, ותציג כפלט את תכונותיהם (באמצעות הפעלת הפעולה toString). כל עצם יווצר על-ידי הפעלה של פעולה בונה אחרת.

- שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

- **עצם יכול להיות תכונה של מחלקה.**
- כאשר למחלקה יש תכונה מטיפוס עצם, ניתן להגדיר בה **פעולה בונה שמקבלת פרמטר מטיפוס העצם**, למשל, public StudentAge (String name, Date birthday)
- וניתן להגדיר **פעולה בונה שמקבלת פרמטרים מטיפוס התכונות של העצם**. למשל,

---

- כאשר פעולה בונה מקבלת פרמטרים מטיפוס התכונות של העצם, צריך **ליצור את העצם** בתוך הפעולה הבונה. למשל,

```
public StudentAge (String name, int day , int month , int year)
 {
    this.name = name;
    this.birthday = new Date (day , month , year);
 }
```

עצם, שבו עצם אחר משמש כתכונה, נקרא: **עצם מורכב**

משימה 10 – חלק ג'

הפעולה older שמוגדרת במחלקה StudentAge מקבלת כפרמטר עצם מטיפוס _____. הפעולה מחזירה את השם של התלמיד הגדול ביותר מבין התלמיד הנוכחי והתלמיד שהתקבל כ_____.

```
public String older (StudentAge otherStudent)
 {
    if (this.birthday.before(otherStudent.birthday))
        return this.name;
    else return otherStudent.name;
 }
```

- הוסיפו לפעולה הראשית האחרונה שכתבתם בדיקה כי הפעולה older תקינה.
- שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

בפעולה older המוגדרת במחלקה StudentAge, נעשה שימוש בפעולה before המוגדרת במחלקה Date.

התכונה birthday היא מטיפוס Date, לכן ניתן להפעיל עליה את כל הפעולות המוגדרות במחלקה Date.

© כל הזכויות שמורות, תמר פז

משימה 11 – חלק א'

פתחו מחלקה חדשה בפרויקט בו הקלדתם את המחלקה StudentAge, והקלידו בה את המחלקה Check הבאה.

- שימרו והריצו.

```
public class Check
{
        static Scanner input = new Scanner(System.in);
        public static void main(String[] args)
        {
                Date d1= new Date (9,12,2007);
                StudentAge st = new StudentAge ("gil",d1);
                d1.setDay(20);
                System.out.println (st.toString());
        }
}
```

שינוי של העצם Date שההפניה אליו היא במשתנה d1, **כן** / **לא** (מחקו את המיותר) בא לידי ביטוי בעצם StudentAge שההפניה אליו היא במשתנה st .

משימה 11 – חלק ב'

שנו כעת את הפעולה הבונה הראשונה במחלקה StudentAge כך שתתקבל הפעולה הבאה:

```
public StudentAge (String name, Date birthday)
{
     this.name = name;
     this.birthday = birthday;
}
```

לאחר השינוי, הקפידו לשמור את המחלקה StudentAge.

הריצו בשנית את המחלקה Check.

האם הפעם, השינוי של העצם Date שההפניה אליו היא במשתנה d1, בא לידי בעצם StudentAge שההפניה אליו היא במשתנה st ? _____.

מדוע? _____

_____

שנו חזרה את הפעולה הבונה במחלקה StudentAge כך שתתקבל הפעולה המקורית שמופיעה במשימה 10 חלק א'.

© כל הזכויות שמורות, תמר פז

משימה 12 – חלק א'

נוסיף כעת למחלקה StudentAge שתי פעולות בונות חדשות:

פעולה בונה ראשונה:

פעולה בונה שלא מקבלת אף פרמטר. הפעולה תקלוט נתונים מהמשתמש (עם מסננת קלט כפעולה פנימית) ותציב אותם בעצם שתבנה.

פעולה בונה שניה:

פעולה בונה מעתיקה. פעולה בונה שמקבלת עצם מטיפוס studentAge, יוצרת עצם חדש מטיפוס studentAge ומעתיקה אליו את התכונות של העצם שהתקבל כפרמטר.

לפניכם שלד של הפעולה הבונה המעתיקה. השלימו אותו והוסיפו למחלקה StudentAge.

```
public StudentAge (StudentAge st)
{
  _____ = st.name;
  this.birthday = new Date (st.birthday);
}
```

גם במחלקה שאחת מהתכונות שלה היא מטיפוס עצם, ניתן להגדיר פעולה בונה מעתיקה

שימו לב כי בעת השמת הערך לתכונה birthday יצרנו גם עצם מטיפוס Date.

מה היה קורה אילו לא יצרנו את העצם מטיפוס Date ? כלומר אילו השמת הערך היתה:

```
this.birthday = st.birthday;
```

כעת, במחלקה StudentAge יש ארבע פעולות בונות והמחלקה תקינה! משום שהפעולות שונות זו מזו ב

משימה 12 – חלק ב'

הוסיפו למחלקה StudentAge את הפעולות הבאות:

1. פעולה setAll שלא תקבל אף פרמטר. הפעולה תקלוט מהמשתמש ערכים חדשים לכל התכונות ותעדכן בהתאם את העצם הנוכחי.
2. פעולה olderSt שמקבלת כפרמטר עצם מטיפוס StudentAge, בונה ומחזירה עצם מטיפוס StudentAge הזהה בתכונותיו לתכונות של התלמיד הגדול מבין העצם הנוכחי והעצם שהתקבל כפרמטר.
3. פעולה youngerSt שמקבלת כפרמטר עצם מטיפוס StudentAge, בונה ומחזירה עצם מטיפוס StudentAge הזהה בתכונותיו לתכונות של התלמיד הקטן מבין העצם הנוכחי והעצם שהתקבל כפרמטר.

© כל הזכויות שמורות, תמר פז

משימה 13 – חלק א'

כתבו פעולה שתקלוט את השמות ואת תאריכי הלידה של 15 ילדי השכונה. הפעולה תציג כפלט את השם ואת תאריך הלידה של הילד הגדול ביותר.

אל תבנו עצם חדש עבור כל ילד. בכל "סיבוב" של הלולאה, עדכנו את אותו העצם.

- שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

משימה 13 – חלק ב'

חיזרו על המשימה מהסעיף הקודם. אבל הפעם: מספר הילדים אינו ידוע. קליטת הנתונים תפסק עם קליטת שנה מספר 1 (אפשר לקלוט עבור שנה זו גם שם ותאריך לידה מלא).

רמז: משמעות הביטוי **st.getBirthday().getYear()** היא: הפעלת הפעולה getYear על העצם המוחזר מהפעולה getBirthday .

- שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

כאשר עצם משמש כתכונה, ניתן להרכיב פעולות משתי המחלקות: המחלקה אליה שייך העצם, והמחלקה בה הוא משמש כתכונה. למשל, בביטוי **st.getBirthday().getYear()** מופעלת הפעולה getYear (מהמחלקה Date) על העצם המוחזר מהפעולה getBirthday (מהמחלקה studentAge). הדבר אפשרי כיוון שקודם מתבצעת הפעולה getBirthday המחזירה עצם מטיפוס Date, ולאחר מכן מופעלת הפעולה _____ שמוגדרת במחלקה Date

משימה 13 – חלק ג'

הוסיפו לפעולה שכתבתם בסעיף הקודם את ההוראות הדרושות כך שתציג כפלט **גם** את השם ואת תאריך הלידה של הילד הצעיר ביותר בכיתה.

- שימרו, הריצו ובדקו שהתקבל הפלט הרצוי.

© כל הזכויות שמורות, תמר פז

מערך כתכונה

משימה 14

במחלקה Family נשמרת אינפורמציה על משפחה.
האינפורמציה הנשמרת היא:

_____

_____

privateNames היא תכונה מטיפוס מערך שאיבריו הם מטיפוס _____

- השלימו את המחלקה Family.
- הקלידו את המחלקה.
- פתחו מחלקה חדשה והקלידו בה פעולה ראשית שתבנה עצם מטיפוס Family ותפעיל עליו את הפעולה printFamily.

- שימרו, הריצו ובדקו שהתקבל הפלט המבוקש.

```
import java.util.Scanner;
public class Family
{
   Scanner input = new Scanner(System.in);
   private String familyName;
   private int numPeople;
   private String [ ] privateNames;
   public Family ()
   {
        System.out.print ("enter family name");
        this.familyName =input.next();
        System.out.print ("enter num people in family");
        this.numPeople = input.nextInt ();
        privateNames = new String[this.numPeople];
        for (int i=0;i<this.numPeople; i++)
        {
            System.out.println("enter people name");
            this.privateNames[i] = input.next();
        }
   }
   public void printFamily()
   {
        System.out.println ("family name is: "+ this.familyName);
        System.out.println ("private names: ");
        for (int i=0;i<_____; i++)
             System.out.println (privateNames[i]);
   }
}
```

מערך הוא עצם! לכן גם מערך יכול לשמש כתכונה של מחלקה!
בעת הגדרת התכונות של המחלקה, מצהירים על משתנה שתהיה בו הפניה למערך. בשלב זה עדיין לא נבנה העצם מטיפוס מערך! בפעולה הבונה, שבונה עצם מטיפוס המחלקה, צריך גם לבנות את העצם מטיפוס מערך! הבניה הזו נעשית באמצעות ההוראה:

שם המשתנה = new ***טיפוס נתונים*** [***אורך המערך***]

© כל הזכויות שמורות, תמר פז

משימה 15

בעבר כתבנו תכנית שקולטת עבור כל אחת מהקבוצות הרשומות באיגוד הכדורסל את שמה, את מספר הניצחונות שצברה בעונה האחרונה ואת מספר ההפסדים שספגה. קליטת הנתונים נפסקה עם קליטת השם "aaa" (ונדרשנו שלא לקלוט נתונים נוספים עבור שם זה).

התכנית הציגה כפלט:

א. את השם של הקבוצה שהיחס בין מספר הניצחונות שלה לבין סה"כ המשחקים ששיחקה הוא הגדול ביותר (אם למספר קבוצות יש את היחס הגדול ביותר, התכנית הציגה כפלט את השמות של כולן).

ב. את השמות של הקבוצות שהיחס בין מספר הניצחונות שלהן לבין סה"כ המשחקים ששיחקו הוא מתחת לממוצע.

השמות של הקבוצות שימשו כמזהים (אין שתי קבוצות או יותר בעלות אותו שם. ואי אפשר לשנות את השמות של הקבוצות).

הנחה: באיגוד רשומות לכל היותר 30 קבוצות.

כדי לפתור את המשימה, כתבנו שתי מחלקות. מחלקה Team (שמייצגת קבוצה אחת) ומחלקה ראשית.

המחלקה Team :

```
import java.util.Scanner;
public class Team
{
  Scanner input = new Scanner(System.in);
  private String teamName;
  private double ratio;

  public Team(String name)
  {
        this.teamName = name;
        System.out.println ("enter num of wins");
        double win= input.nextDouble();
        System.out.println ("enter num of losses");
        double loss= input.nextDouble();
        this.ratio = win/(win+loss);
  }

  public String getName ()
  {
        return this.teamName;
  }

  public double getRatio ()
  {
        return this.ratio;
  }
}
```

למחלקה Team שתי תכונות:
teamName - השם של הקבוצה
ratio - היחס בין מספר הניצחונות לבין סה"כ המשחקים שהקבוצה שיחקה.

נשנה כעת את הפתרון.

נוסיף מחלקה חדשה Union שיהיו בה שתי תכונות: מערך שכל איבר בו יהיה עצם מטיפוס Team, ומספר שלם שיציין את מספר הקבוצות באיגוד.

במחלקה הזו יהיו כל הפעולות שאנו מגדירים על האיגוד.

© כל הזכויות שמורות, תמר פז

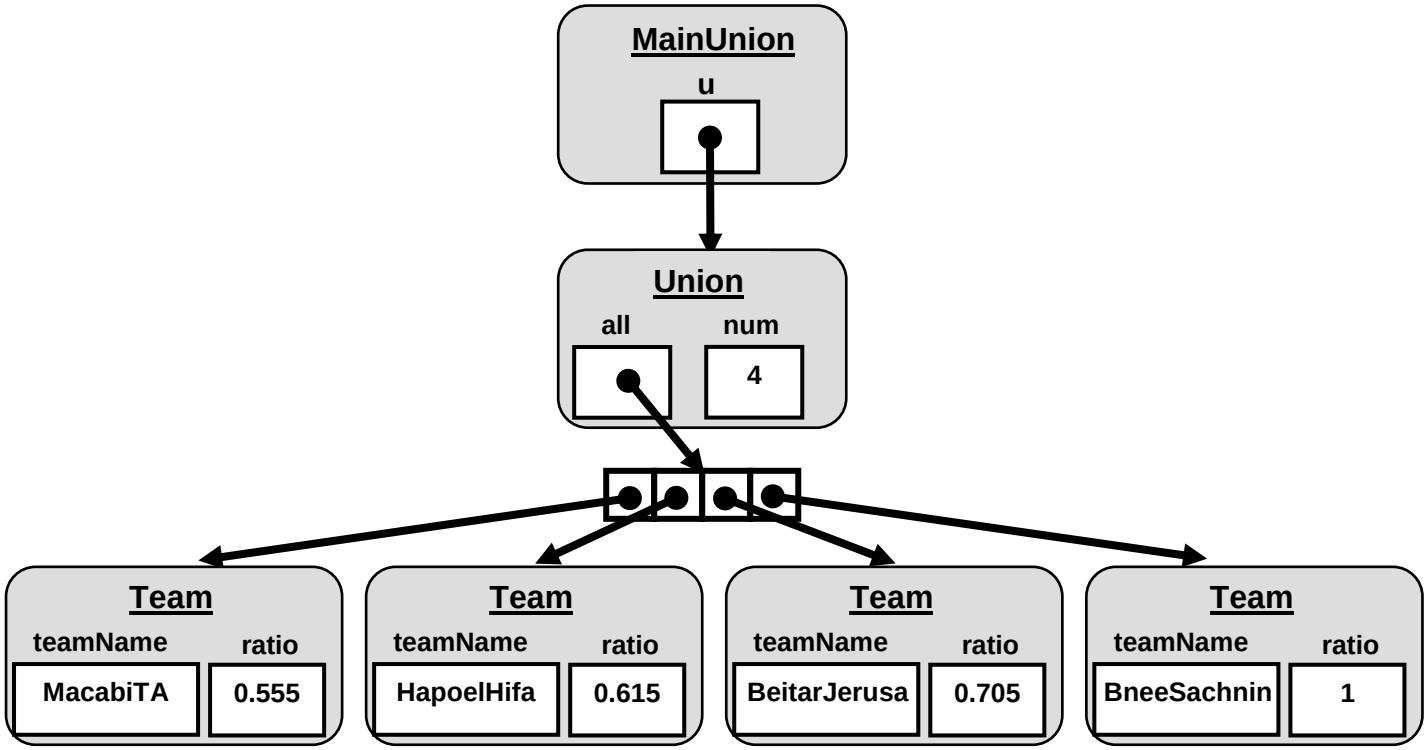
```
import java.util.Scanner;
public class Union
{
    private Scanner input = new Scanner(System.in);
    private Team [ ] all; //  תכונה 1: מערך שכל איבר בו הוא עצם מטיפוס Team
    private int num;    // תכונה 2: מספר שלם המציין את מספר הקבוצות שהכנסנו למערך

    public Union () // הפעולה הבונה
    {
        this.all = new Team[30]; // באיגוד רשומות לכל היותר 30 קבוצות
        this.num = 0;
        System.out.println ("enter first team name");
        String name = input.next();
        while (!name.equals("aaa"))
        {
                this.all [this.num]= new Team(name);
                // יצירת עצם חדש מטיפוס Team והשמת הפניה אליו בתא מספר num של המערך (התכונה) all
                this.num++;     // הוספת 1 למונה מספר הקבוצות
                System.out.println ("enter team name");
                name = input.next();
        }
    }

    public void printBiggest ()
    // פולטת את השמות של הקבוצות שהשיגו את היחס הגדול ביותר בין מספר הניצחונות לבין מספר המשחקים
    {
        double big= this.all[0].getRatio(); // היחס הגדול ביותר
        for (int i =1; i<this.num; i++)
            if (this.all[i].getRatio()>big)
                big= this.all[i].getRatio();
        System.out.println ("biggest ratio:");
        for (int i =0; i<this.num; i++)
           if (this.all[i].getRatio()==big)
                System.out.println (all[i].getName());
    }

    public void underAverage ()
    // פולטת את השמות של הקבוצות שהיחס בין מספר הניצחונות שלהן לבין מספר המשחקים הוא מתחת ליחס הממוצע
    {
        double sum=0;
        for (int i =0; i<this.num; i++)
            sum = sum+ all[i].getRatio();
        double ave = sum/this.num;  // היחס הממוצע
        System.out.println ("ander avarage:");
        for (int i =0; i<this.num; i++)
           if (this.all[i].getRatio()<ave)
                System.out.println (all[i].getName());
    }
}
```


© כל הזכויות שמורות, תמר פז

- פתחו מחלקה חדשה בפרויקט בו הקלדתם את המחלקה Team, והקלידו בה את המחלקה Union.
- פתחו מחלקה נוספת באותו פרויקט, והקלידו בה את המחלקה הראשית הבאה.

```
public class MainUnion
{
	public static void main(String[] args)
	{
		Union u=new Union();
		u.printBiggest();
		u.underAverage();
	}
}
```

הריצו את המחלקה ובדקו כי הפעולה אכן מבצעת את המשימה המבוקשת.



- המחלקה MainUnion היא מחלקה ראשית שמוגדרת בה רק פעולה אחת main. בפעולה זו בונים עצם אחד מטיפוס Union ומפעילים עליו את הפעולות printBiggest ו- _____ .
- הפעולה הבונה של המחלקה Union בונה עצם אחד מטיפוס מערך (התכונה all) ומספר עצמים מטיפוס _____
- למופע היחיד של העצם מטיפוס Union יש שתי תכונת: num ו- all.
- במשתנה all יש הפניה למערך שבכל איבר בו הפניה לעצם מטיפוס _____

שימרו את שלוש המחלקות הקשורות למשימה זו. נזדקק להן בהמשך!

© כל הזכויות שמורות, תמר פז

משימה 16 (מעובד מתוך קבוצת מדעי המחשב, אוניברסיטת תל אביב)

חברת ההסעות BusExpress מפעילה N קווי אוטובוס. כתבו תכנה שתקלוט את מספר הקווים N, ואחריו N קווים. עבור כל קו ייקלט מספר הקו, מספר התחנות שהקו עובר בהן, ושמות של תחנות בהן הוא עובר. השמות של התחנות ייקלטו בהתאם לסדר הנסיעה של האוטובוס. למשל, hadar אחריה grand אחריה ziv וכן הלאה. לאחר קליטת הנתונים, התכנה תציג כפלט:

1. רשימה מסודרת של כל הקווים. למשל,

```
The lines are:
Line number 18 , stations: hadar  grand  ziv
Line number 20 , stations: hadar  renez
Line number 22 , stations: ziv  grand  horev  carmel
Line number 15 , stations: carmel  rambam  hadar  hamifraz
```

2. הודעה האם קיים או לא קיים קו שתחנתו הראשונה היא התחנה האחרונה של קו אחר.

הנחיה:

יש לכתוב 3 מחלקות:

1. מחלקה Line שתכיל אינפורמציה עבור קו יחיד. למחלקה זו יהיו 3 תכונות: מספר הקו, מספר התחנות שהקו עובר בהן, ומערך שיכיל את השמות של התחנות.
2. מחלקה BusExpress שתכיל אינפורמציה עבור על הקווים. למחלקה זו יהיו שתי תכונות: מערך שכל איבר בו יהיה עצם מטיפוס _____, ומספר שיציין את מספר הקווים.
3. מחלקה ראשית.

משימה 17

כדי לא להשתעמם בהפסקות, החליטו 25 תלמידי הכיתה להקים מאגר משחקים. בתחילת השנה כל תלמיד התחייב להביא מספר מסוים של משחקים. בהמשך השנה, הוא יכול להביא משחקים כרצונו. בסוף השנה הילד שהביא הכי הרבה משחקים יקבל פרס.

כתבו תכנה שתקלוט את השמות של כל תלמידי הכיתה (בשלב זה, כל תלמיד הביא את מספר המשחקים ההתחלתי שנקבע). לאחר מכן, ייקלט שם של תלמיד ומספר משחקים שהביא (תלמיד יכול להביא משחקים מספר פעמים). קליטת הנתונים תפסק עם קליטת השם zozo. לבסוף, התכנה תציג כפלט את מספר המשחקים הכללי שנאסף, ואת השם של התלמיד שיזכה בפרס (ייתכן כי מספר תלמידים יזכו בפרס).

הערה: מספר המשחקים שכל ילד הביא בתחילת השנה ייוצג כקבוע.

השמות של התלמידים משמשים כמזהים (אין שני תלמידים או יותר בעלי אותו שם. ואי אפשר לשנות את השמות של התלמידים).

דוגמה להרצה של התכנה (נניח כי בכיתה 5 תלמידים ובתחילת השנה, כל ילד הביא 2 משחקים):

```
Enter 5 names
Adi
Lital
Efrat
```

© כל הזכויות שמורות, תמר פז

```
Dorit
Orit
Enter student name: lital
Enter amount of games 3
Enter student name: efrat
Enter amount of games 2
Enter student name: dorit
Enter amount of games 1
Enter student name: efrat
Enter amount of games 2
Enter student name: orit
Enter amount of games 4
Enter student name: zozo
The class collect 22 games
The winner: efrat  orit
```

הסבר הדוגמה: כיוון שיש 5 תלמידים ובתחילת השנה כל תלמיד הביא 2 משחקים, נאספו בתחילת השנה 10 משחקים, לאחר מכן מאספו עוד 12 משחקים.

הנחיה:

לצורך פתרון המשימה עליכם לכתוב 3 מחלקות: מחלקה Pupil שתכיל אינפורמציה עבור תלמיד יחיד (שמו ומספר המשחקים שהביא), מחלקה Class17 שתכיל אינפורמציה עבור הכיתה, ומחלקה ראשית.

משימה 18 – חלק א'

כתבו תכנה שקולטת את מספר תלמידי הכיתה. עבור כל תלמיד, התכנה תקלוט את השם של התלמיד ואת מספר המקצועות שלמד השנה. עבור כל מקצוע ייקלט שם המקצוע, מספר יחידות הלימוד והציון הסופי שהתלמיד השיג במקצוע.

התכנה תציג כפלט דוח הכולל אינפורמציה עבור כל תלמיד. האינפורמציה תכלול גם את הממוצע המשוקלל שלו. למשל,

```
Student name is: Ortal
Subject =cs  points = 5  grade = 97
Subject =history  points = 3  grade = 82
Subject =gremer  points = 1  grade = 86
Average = 90
Student name is: Ragi
Subject =cs  points = 3  grade = 100
Subject =math  points = 1  grade = 80
Average = 95
Student name is: Tomer
Subject =math  points = 5  grade = 90
Subject =history  points = 2  grade = 80
Subject =english  points = 2  grade = 88
Average = 87
```

משימה 18 – חלק ב'

הוסיפו את ההוראות הדרושות כך שהדוח שהתכנה תציג כפלט יהיה ממוין לפי ציון משוכלל יורד. למשל, בדוגמה האחרונה, תוצג כפלט האינפורמציה עבור Ragi, לאחריה האינפורמציה עבור Ortal, ולאחריה האינפורמציה עבור Tomer.

© כל הזכויות שמורות, תמר פז

תכונות מופע ותכונות מחלקה

משימה 19 – חלק א'

המחלקה Basketball שומרת עבור כל שחקן את שמו ואת מספרו הסידורי בקבוצה. השחקן הראשון שיצטרף יקבל את המספר 1, השחקן השני יקבל את המספר 2 וכן הלאה.

פתחו פרויקט חדש והקלידו בו את המחלקה Basketball.

```
public class Basketball
{
  // The attributes of the class
   private String name;
   private int number ;
   private static int counter=0;

   // A constructor for the class
   public Basketball (String name)
   {
         this.name = name;
         counter++;
         this.number =counter;
    }
   public String getName()
   {
         return this.name;
   }
   public int getNumber()
   {
         return this.number;
   }
   public void setName (String newName)
   {
         this.name = newName;
   }
  public String toString ()
  {
         return ("number  "+this.number+"  is "+this.name);
  }
}
```

פתחו מחלקה נוספת באותו פרויקט בו הקלדתם את המחלקה Basketball, והקלידו בה את הפעולה הראשית הבאה:

```
public static void main(String[] args)
{
    Scanner input = new Scanner(System.in);
    System.out.println ("enter name");
     String name =  input.next();
     Basketball b1;
     b1= new Basketball (name);
     System.out.println(b1.toString());
     System.out.println ("enter other name");
     name =  input.next();
     Basketball b2;
     b2= new Basketball (name);
     System.out.println(b1.toString());
     System.out.println(b2.toString());
}
```

שימרו, הריצו וכתבו את הפלט:

_____

_____

_____

השורה הזו חוזרת פעמיים וזו לא טעות

© כל הזכויות שמורות, תמר פז

לכל עצם במחלקה Basketball יש שתי תכונות: name ו- number. התכונות הללו קיבלו את ערכיהן עבור כל עצם שיצרנו (שם השחקן נקבע על-ידי המשתמש, ומספר השחקן נקבע על-ידי הפעולה הבונה). במילים אחרות, לכל **מופע** של המחלקה יש עותק מקומי של שתי התכונות הללו והן מאפיינות אותו. תכונות כאלה, שנוצרות עבור כל עצם (או: מאפיינות מופע מסוים), נקראות **תכונות מופע.**
למחלקה Basketball יש גם תכונה נוספת! התכונה counter היא **תכונה של המחלקה**. בזיכרון המחשב קיים רק עותק אחד של התכונה ללא קשר למספר העצמים שנוצרו מהמחלקה. המילה **static** בהגדרת התכונה מציינת כי מדובר **בתכונה שאינה קשורה לעצם מסוים!** תכונה כזו, שמאפיינת מחלקה ולא מופע מסוים שלה נקראת **תכונת מחלקה**.

משימה 19 – חלק ב'

השלימו: משמעות השורה **private static int counter=0;** היא:

private- תכונה פרטית למחלקה, אינה ניתנת לשימוש מחוץ לקובץ המחלקה.

static – תכונה שאינה קשורה לעצם מסוים, אינה קשורה למופע מסוים של המחלקה.

int - תכונה מטיפוס _____

countrer - _____

0 - ערך התחלתי לתכונה.

משימה 19 – חלק ג'

בכל הפעלה של הפעולה הבונה (כאשר יוצרים עצם מטיפוס המחלקה = יוצרים מופע של המחלקה), הערך של התכונה counter גדל ב- 1. והערך החדש מוצב בתכונה _____ של העצם הנוכחי.

```
public Basketball (String name)
{
      this.name = name;
      counter++;
      this.number =counter;
}
```

שנו כעת את הפעולה toString כך שתתקבל הפעולה הבאה:

```
public String toString ()
{
  return ("number  "+this.number+"  is "+this.name + "   there are "+this.counter+" players");
}
```

שימרו, הריצו את הפעולה הראשית והשלימו:

כאשר הדפסנו את התכונות של העצם הראשון (b1) לפני יצירת העצם השני (b2) הערך של counter היה _____ וכאשר הדפסנו את התכונות של העצם הראשון לאחר יצירת העצם השני, הערך של counter היה _____.

counter היא תכונת מחלקה (שקיים רק עותק אחד שלה) ולא תכונת מופע (שקיימת עבור כל עצם), לכן, שינוי שלה בעת יצירת העצם השני בא לידי ביטוי גם כשמדפיסים את התכונות של העצם הראשון.

© כל הזכויות שמורות, תמר פז

משימה 19 – חלק ד'

שנו כעת את הפעולה toString כך שתתקבל הפעולה הבאה:

```
public String toString ()
{
  return ("number  "+this.number+"  is "+this.name + "   there are "+ Basketball.counter+" players");
}
```

שימרו, הריצו את הפעולה הראשית ובדקו שהפלט של הפעולה לא השתנה.

ניתן לפנות אל תכונת מחלקה באמצעות עצם מטיפוס המחלקה (באמצעות מופע של המחלקה).

כלומר, ***שם התכונה*.this** למשל, _____

ואפשר לפנות אל תכונת מחלקה **באמצעות שם המחלקה**. כלומר, ***שם התכונה*.שם המחלקה.**

למשל, _____

הפניה באמצעות שם המחלקה (Basketball) עדיפה כיוון שהיא מדגישה זו תכונה של מחלקה.

משימה 20 – חלק א'

- הוסיפו כעת לפעולה הראשית את ההוראה: System.out.println (Basketball.counter); .
- שימרו – מופיע הסימן המעיד על **שגיאה!** נסו להסביר מדוע: _____

_____

רמז: חישבו על הרשאת הגישה של התכונה counter

משימה 20 – חלק ב'

- עברו לקובץ המחלקה Basketball ושנו את הגדרת התכונה counter כך שהיא תוגדר כבעלת הרשאת גישה פומבית public. כלומר הגדרת התכונה תהיה: **public static int counter=0;**
- חזרו לפעולה הראשית והוסיפו בה במקומות שונים את ההוראה: System.out.println (Basketball.counter); הוסיפו את ההוראה גם לפני יצירת העצם הראשון!
- שימרו, הריצו והשלימו:

  ההוראה Basketball.counter מחזירה את הערך של התכונה _____.

משימה 20 – חלק ג'

- הוסיפו כעת בתחילת הפעולה הראשית את ההוראה: Basketball.counter=100; .
- שימרו, הריצו והשלימו:

  ההוראה Basketball.counter=100; משנה את הערך של התכונה counter של המחלקה

  _____.

© כל הזכויות שמורות, תמר פז

- counter היא **תכונת מחלקה** (מוגדרת כתכונה static) שקיים רק עותק אחד שלה בזיכרון המחשב ללא קשר למספר העצמים שנוצרו מהמחלקה.
- אם מגדירים אותה כבעלת הרשאת גישה פרטית **(private)** למשל, private static int counter=0; אפשר לגשת אליה ולשנות אותה רק מתוך המחלקה בה היא מוגדרת.
- אם מגדירים אותה כבעלת הרשאת גישה פומבית **(public)** למשל, public static int counter=0; אפשר לגשת אליה גם ממחלקות שאינן המחלקה בה היא מוגדרת. אפשר לראות את ערכה ממחלקות אחרות ואפשר גם לשנות את ערכה ממחלקות אחרות.
- **גישה לתכונת מחלקה תעשה רק באמצעות שם המחלקה**. כלומר, ***שם התכונה.שם המחלקה***
- תכונת מחלקה קיימת **לפני יצירת העצם הראשון** מטיפוס המחלקה. ולכן ניתן לגשת אליה גם לפני יצירת העצם הראשון.

משימה 20 – חלק ד'

- לאחר יצירת העצם b1 בפעולה הראשית, הוסיפו את ההוראה

System.out.println ("counter=" + b1.counter);

- שימרו, הריצו והשלימו:

כאשר תכונת מחלקה מוגדרת כבעלת הרשאה public אפשר לגשת אליה גם דרך מופע של המחלקה. אולם הדבר <u>אינו מומלץ</u> כיון שאנו רוצים להדגיש כי מדובר בתכונה של המחלקה ולא בתכונה של עצם מסוים.

הערה: גם אם נגדיר תכונת מופע (למשל את name) כבעלת הרשאה public נוכל לגשת אליה ממחלקות אחרות. אולם לא נעשה זאת כי נרצה לשמור על התכונות כפרטיות למחלקה.

משימה 21 – חלק א'

בגלל בעיות תקציב החליטו להגביל את מספר השחקנים בנבחרת. במקרה כזה, נרצה להוסיף למחלקה Basketball קבוע שישמור את מספר השחקנים המקסימלי המותר. כאשר נרצה ליצור עצם חדש (עבור שחקן נוסף) נבדוק האם הגענו למספר המקסימלי המותר, ורק אם לא הגענו למספר המקסימלי המותר נגדיר את העצם החדש. לשם כך, נוסיף למחלקה Basketball תכונת מחלקה maxPlayers שתכיל את מספר השחקנים המקסימלי המותר: **public static final int MAXpLYERS = 20;**

משמעות ההגדרה: MAXpLYERS היא תכונת מחלקה (static), קבועה (final), פומבית (public), מטיפוס int שערכה 20. תכונת מחלקה כזו שערכה לא יכולה להשתנות נקראת: **תכונת מחלקה קבועה**.

- הוסיפו את התכונה לרשימת התכונות של המחלקה Basketball.

בעמוד הבא, מופיעה המחלקה Basketball בשלמותה.

© כל הזכויות שמורות, תמר פז

כך צריכה להראות כעת המחלקה Basketball

```
public class Basketball
{
   // The attributes of the class
    private String name;
    private int number ;
    public static int counter=0;
    public static final int MAXpLAYERS = 20;
    // A constructor for the class
    public Basketball (String name)
    {
            this.name = name;
            counter++;
            this.number =counter;
     }
    public String getName()
    {
           return this.name;
    }
    public int getNumber()
    {
           return this.number;
    }
    public void setName (String newName)
    {
           this.name = newName;
    }
   public String toString ()
   {
      return ("number  "+this.number+"  is "+this.name + "   there are "+ Basketball.counter+" players");
   }
}
```

משימה 21 – חלק ב'

עברו כעת לפעולה הראשית והדפיסו בה את הערך של MAXpLYERS .

- MAXpLYERS היא תכונת מחלקה ולכן הפניה אליה היא: _____
- כיוון ש MAXpLYERS היא תכונת מחלקה אפשר לפנות אליה בפעולה הראשית גם לפני שמגדירים עצמים מטיפוס המחלקה Basketball. בדקו זאת!
- שימרו, הריצו ובדקו שהתקבל הפלט המצופה.

משימה 21 – חלק ג'

- כיוון שגם counter וגם MAXpLYERS הן בעלות הרשאת גישה פומבית (public), אפשר לבדוק את ערכם גם בפעולה הראשית. שנו כעת את הפעולה הראשית כך שלא תאפשר יצירה של יותר שחקנים מהמקסימום המותר. כלומר, לפני כל יצירה של עצם מטיפוס Basketball, הפעולה תבדוק אם מספר השחקנים הוא כבר MAXpLYERS, ותיצור את העצם רק במידה ו-counter קטן מ- MAXpLYERS. במקרה שלא תיצור עצם, הפעולה תציג כפלט הודעה כי מספר השחקנים הוא כבר המקסימום המותר.
- שנו כעת את הפעולה הראשית כך שתנסה ליצור מעל 20 עצמים מהמחלקה Basketball (אפשר להשתמש בלולאה).
- שימרו, הריצו ובדקו כי כאשר מנסים ליצור יותר עצמים מהמספר המקסימלי המותר, הפעולה לא יוצרת עצם נוסף, ומודיעה על כך למשתמש.

שימרו את המחלקה Basketball. נזדקק לה במשימה הבאה.

© כל הזכויות שמורות, תמר פז

פעולות מופע ופעולות מחלקה

משימה 22 – חלק א'

הוסיפו למחלקה Basketball (מהמשימה הקודמת), את הפעולה freePlace הבאה, והשלימו:

```
public static int freePlace ()
{
   return Basketball.MAXpLAYERS -Basketball.counter;
}
```

הפעולה freePlace לא מקבלת אף פרמטר ומחזירה את _____

- את כל הפעולות שפגשנו עד עכשיו, ניתן היה לבצע על עצם מסוים. פעולות כאלה שמתבצעות על מופע מסוים של המחלקה נקראות **פעולות מופע**. הפעולה freePlace שונה מהן, זו פעולה שהמחלקה מבצעת ללא קשר למופע מסוים שלה. פעולה כזו, המתבצעת על-ידי המחלקה ולא על עצם מסוים שלה, נקראת **פעולת מחלקה**.
- המילה static בחתימה של הפעולה freePlace מציינת כי זו **פעולת מחלקה** המתבצעת על-ידי המחלקה ולא על עצם מסוים שלה.

משימה 22 – חלק ב'

- פתחו מחלקה חדשה וכתבו בה את הפעולה הראשית הבאה.
- שימרו, הריצו וכתבו כאן את הפלט:

_____

_____

```
public static void main(String[] args)
{
   Scanner input = new Scanner(System.in);
   System.out.println ("free place = " + Basketball.freePlace());
   System.out.println ("enter name");
   String name =  input.next();
   Basketball b1 = new Basketball (name);
   System.out.println(b1.toString());
   System.out.println ("free place = " + Basketball.freePlace());
}
```

מכיוון שפעולת מחלקה מתבצעת על-ידי מחלקה ולא על עצם מסוים, ניתן להפעיל אותה גם **לפני** שהוגדרו עצמים מהמחלקה (כשם שניתן לגשת לתכונת מחלקה לפני שהוגדרו עצמים מהמחלקה).

משימה 22 – חלק ג'

- הוסיפו למחלקה Basketball פעולת מחלקה המחזירה את אחוז התפוסה בנבחרת (היחס בין מספר השחקנים בפועל לבין מספר השחקנים המותר).
- הוסיפו לפעולה הראשית הוראה שתבדוק את פעולת המחלקה החדשה שכתבתם.
- שימרו, הריצו ובדקו שהודפס הפלט לו ציפיתם.

© כל הזכויות שמורות, תמר פז

משימה 22 – חלק ד'

- הוסיפו למחלקה Basketball את הפעולה הבאה.
- שימרו – מופיע הסימן המעיד כי **ההוראה אינה תקינה**! נסו להסביר מדוע

```
public static int problem ()
{
   return this.number +Basketball.counter;
}
```

_____

_____

כיוון ש**פעולות מחלקה** אינן קשורות לעצם מסוים וניתן אפילו להפעיל אותן לפני שהוגדרו עצמים מהמחלקה, פעולות מחלקה לא יכולות להשתמש בתכונות של עצם מסוים! הן **יכולות להשתמש רק בתכונות של המחלקה** (תכונות מחלקה בעלות הרשאת גישה static).

משימה 23

שנו כעת את הגדרת תכונת המחלקה counter כך שתהיה בעלת הרשאה פרטית. כלומר:

private static int counter=0;

כעת counter היא תכונה פרטית של המחלקה Basketball ולכן לא נוכל לגשת אליה ממחלקות אחרות. יש לכך יתרון כי כך נוכל להגן על התכונה ולהיות בטוחים שבשום מחלקה אחרת לא ישנו את ערכה. אבל יש גם חסרון: במחלקות אחרות (למשל בפעולה הראשית) לא נוכל לדעת מהו ערכה של התכונה.

הציעו פתרון לבעיה _____

_____

- נוכל לכתוב פעולת מחלקה שתחזיר את הערך של תכונת המחלקה counter.
  כתבו במחלקה Basketball פעולת מחלקה getCounter שמחזירה את ערכה של תכונת המחלקה counter. זיכרו להגדיר את הפעולה החדשה כפעולה static.
- חזרו לפעולה הראשית וכתבו בה הוראה לבדיקת הערך המוחזר על-ידי פעולת המחלקה החדשה getCounter שכתבתם. כתבו את ההוראה גם לפני שמוגדרים עצמים מהמחלקה וגם לאחר שמוגדרים עצמים.
- שימרו, הריצו ובדקו שהפלט מתאים לציפיות שלכם.

כאשר **תכונת מחלקה** היא בעלת הרשאת גישה פומבית (**public**) **ניתן לגשת אליה** גם ממחלקות אחרות.

כאשר **תכונת מחלקה** היא בעלת הרשאת גישה פרטית (**private**) **לא ניתן לגשת** אליה ממחלקות אחרות. כדי שגם במחלקות אחרות אפשר יהיה לדעת מהו ערכה, יש לכתוב **פעולת מחלקה** (פעולה **static**) שמחזירה את ערכה של התכונה.

© כל הזכויות שמורות, תמר פז

משימה 24

בספריה העירונית רוצים לשמור עבור כל ספר את האינפורמציה הבאה: קוד הספר, שם הספר, מספר עותקים ברשות הספריה, מספר עותקים מושאלים.

קוד הספר מורכב משתי האותיות הראשונות של שמו ולאחריהן מספרו הסידורי (המספר הסידורי של הספר הראשון שהספריה תרכוש יהיה 1,של הספר השני 2 וכן הלאה).

כתבו תכנה לניהול הספריה. התכנה תאפשר למשתמש לבצע את הפעולות הבאות:

1. להוסיף לספריה מספר עותקים מספר כלשהו (המשתמש יתבקש לספק את השם של הספר ואת מספר העותקים שנרכשו).
   התכנה תבדוק אם הספר כבר קיים בספריה ותדפיס הודעה מתאימה. במידה והספר כבר קיים בספריה, התכנה תוסיף את מספר העותקים שנרכשו למספר העותקים שיש כבר מהספר.
   במידה והספר עוד לא קיים בספריה, התכנה תיצור עבורו עצם חדש עם מספר העותקים שנרכשו.
2. לשאול ספר מהספריה (ואז המשתמש יתבקש לספק את השם של הספר הרצוי).
3. להחזיר לספריה ספר ששאל (ואז המשתמש יתבקש לספק את השם של הספר המוחזר).
4. לקבל רשימה של כל הספרים שבספריה. לכל ספר את הקוד שלו, שמו, מספר עותקים ברשות הספריה, מספר עותקים מושאלים.
5. לקבל את מספר הספרים השונים הפנויים להשאלה (כאן מסכמים עותק אחד מכל ספר שיש ממנו עותקים פנויים).
6. לקבל את המספר הכללי של העותקים המושאלים מהספריה (כאן מסכמים את מספר העותקים המושאלים מכל הספרים).

לאחר ביצוע הפעולה הרצויה, התכנה תבקש מהמשתמש פעולה נוספת וכן הלאה עד שהמשתמש יבקש לסיים.

הערות:

- בפעולות 2,3 יש לבדוק כי הספר אכן רשום בספריה. במידה והספר אינו נמצא, יש להודיע על כך למשתמש.
- בפעולה 2 יש לבדוק כי נותר עותק להשאלה. במידה ולא נותר, יש להודיע על כך למשתמש.
- בפעולה 3 צריך לבדוק שהספר המבוקש הושאל מהספריה. במידה ולא הושאל, יש להודיע על כך למשתמש.

הנחיות:

- לצורך פתרון המשימה עליכם לכתוב 3 מחלקות: מחלקה Book שתכיל אינפורמציה עבור ספר יחיד, מחלקה Library שתכיל אינפורמציה עבור הספריה, ומחלקה ראשית.
- **במחלקה Book** יכללו הפעולות הבאות:

1. פעולה בונה: יוצרת ספר חדש (המשתמש יספק את השם של הספר החדש ואת מספר העותקים שנרכשו).
2. פעולה שמחזירה את השם של הספר הנוכחי.
3. פעולה שמחזירה את מספר העותקים המושאלים מהספר הנוכחי.
4. פעולה שמחזירה את מספר העותקים הפנויים (שאינם מושאלים) מהספר הנוכחי.
5. פעולה שמוסיפה מספר עותקים לספר הנוכחי.
6. פעולה ששואלת מהספריה את הספר הנוכחי.

© כל הזכויות שמורות, תמר פז

7. פעולה שמחזירה לספריה את הספר הנוכחי.
8. פעולת toString שיוצרת מחרוזת מהפרטים (ערכי התכונות) של הספר הנוכחי.

דוגמה להרצה של התכנה:

```
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
1
Enter book name
LoveStory
Enter num copies
3
The library does not has this book. we add it
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
2
Enter book name
Jerusalem
Sorry, the library does not has this book
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
1
Enter book name
Help
Enter num copies
2
The library does not has this book. we add it
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
1
Enter book name
LoveStory
Enter num copies
4
The library has this book. we add copies
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
3
```

© כל הזכויות שמורות, תמר פז

```
Enter book name
Help
You did not borrow this book
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
2
Enter book name
Help
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
2
Enter book name
Help
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
2
Enter book name
Help
No copies to borrow
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
4
Book code= Lo1      Book name= LoveStory       NumCopies= 7  Borrow= 0
Book code= He2      Book name= Help            NumCopies= 2  Borrow= 2
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
5
The Library has 1 free different books
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
6
The Library has 2 borrow books
2 -- borrow book
3 -- return book
```


© כל הזכויות שמורות, תמר פז

```
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
2
Enter book name
LoveStory
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
6
The Library has 3 borrow books
Enter:
1 -- add copies to new or existing book
2 -- borrow book
3 -- return book
4 -- show all books
5 -- show number of free different books in library
6 -- show number of borrow books in library
7 -- exit the program
7
Good bye
```

מספר הערות:

- שם הספר אינו ניתן לשינוי ולכן אל תכתבו פעולה המאפשרת לעדכן את שם הספר.
- למחלקה **Library** תהיה תכונה מערך שבכל איבר בו תהיה הפניה לעצם מטיפוס Book.

<u>הנחה</u>: יש בספריה לכל היותר 50 ספרים.

- לפניכם הצעה **לפעולה הראשית**.

בהצעה זו, getSign היא פעולה שיש להגדיר. חישבו מה היא מקבלת ומה היא אמורה להחזיר.

השלימו את המחלקה,
שימרו הריצו ובדקו שהיא מבצעת את הנדרש.

```
public static void main(String[] args)
{
    Library lib = new Library();
    int answer = getSign();
    while (answer != 7)
    {
      switch (answer)
      {
         case 1 : lib.addBook(); break;
         case 2 : lib.borrowBook(); break;
         case 3 : lib.returnBook(); break;
         case 4 : lib.showAll(); break;
         case 5 : lib.showNumFreeBooks(); break;
         case 6 : lib.showNumBorrow(); break;
      }
      answer = getSign();
    }
    System.out.println ("Good bye");
}
```


© כל הזכויות שמורות, תמר פז

מערך: התכונה length

משימה 25 – חלק א'

לפניכם שלד של מחלקה שקולטת ערכים לשני מערכים a ו- b. לפני קליטת הערכים לכל מערך, המחלקה שואלת את המשתמש מהו מספר הערכים שהמערך יכיל, ומגדירה את המערך באורך הרצוי.

המחלקה בונה מערך חדש c שהוא מערך האיחוד של a ושל b. כלומר, מכיל את כל הערכים שמופיעים גם ב- a וגם ב- b.

הנחה: בכל אחד מהמערכים, לא יופיע אותו ערך יותר מפעם אחת.

שימו לב, במחלקה מופיעה הוראה חדשה

- טענו את שלד המחלקה, השלימו והריצו.

- השלימו:

ההוראה **a.length** מחזירה את מספר ה _____

במערך _____

```
import java.util.Scanner;
public class Union
{
  static Scanner input = new Scanner(System.in);
  public static void main(String[] args)
  {
    System.out.println ("enter number of items for first array");
    int lengthA = input.nextInt();
    int  [ ] a = new int [lengthA];
    kelet (a, "first");
    System.out.println ("enter number of items for second array");
    int lengthB = input.nextInt();
    int  [ ] b = new int [lengthB];
    kelet (b, "second");
    int  [ ] c = new int [_____];
    int count = unionArray (a,b,c);
    pelet (c, count);
  }
  public static void kelet (int [ ] ar, String name)
  {
    System.out.println ("enter "+ ar.length +" items for "+name+" array");
    for (int i=0; i< ar.length; i++ )
        ar[i] = input.nextInt();
  }
  public static void pelet (int [ ] ar, int count )
  {
    System.out.print ("the numbers in array: ");
    for (int i=0; i<count; i++ )
       System.out.print ("  "+ ar[ i ] );
    System.out.println ("");
  }
  public static int unionArray  (int[ ] a, int[ ] b, int[ ] c)
  {
    for (int i= 0; i< a.length ; i++)
        c[i] = a[i];
    int count = a.length;
    for (int i= 0; i< b.length; i++)
      if (! isIn (b[i],c))  {
                    c[count] = b[i];
                    _____;
                  }
    return count;
  }
  public static boolean isIn (int x , int[ ] ar)
  {
    boolean flag=false;
    for (int i= 0; i< ar.length ; i++)
        if (x== ar[i])
            flag = true;
    return flag;
  }
}
```


© כל הזכויות שמורות, תמר פז

- **length** היא **תכונה** של המחלקה Array. התכונה שומרת את _____
- length היא **תכונת מופע** (כיוון שהיא קשורה לעצם מסוים) וגם הפניה אליה היא באמצעות עצם, למשל כך: ar.length

  איך יודעים ש length היא תכונת מופע? _____

  _____
- כל תכונות המופע (הקשורות לעצם מסוים) שפגשנו עד כה היו בעלות הרשאת גישה פרטית (לכן ניתן היה לגשת אליהן רק מהמחלקה בה הן הוגדרו). length היא תכונת מופע בעלת הרשאת גישה **פומבית** ולכן ניתן לגשת אליה ממחלקות שונות.

length היא **תכונת מופע פומבית** של המחלקה Array.
הפניה אליה היא באמצעות עצם. כלומר, length. *שם משתנה*

לא להתבלבל: במחלקה String, length היא פעולה ולכן זימנו אותה כך: sen.length()

ובמחלקה Array , length היא תכונה! ולכן מזמנים אותה כך: ar.length

משימה 25 – חלק ב'

הוסיפו למחלקה את ההוראות הדרושות כך שיתכן שאותו ערך יופיע יותר מפעם אחת באותו מערך. ועדיין נרצה שמערך האיחוד יכיל כל ערך רק פעם אחת. כלומר, יתכן שהמספר 5 יופיע פעמיים או יותר במערך a ועדיין נרצה שהוא יופיע רק פעם אחת במערך האיחוד.

- שימרו והריצו.

© כל הזכויות שמורות, תמר פז

משימה 26 – חלק א'

בקניון הזהב יש מספר חנויות. בכל חנות מספר שונה של מוצרים. לכל מוצר יש שם ומחיר.

כתבו תכנה שתקלוט עבור כל חנות את שמה ואת מספר המוצרים הנמכרים בה. עבור כל מוצר ייקלט שמו ומחירו. קליטת הנתונים תפסק כאשר ייקלט שם של חנות "end" (אין לקלוט נתונים נוספים עבור חנות זו).

התכנה תציג כפלט עבור כל חנות את מספרה (החנות הראשונה שתקלט תקבל את המספר 1, השניה 2, וכן הלאה), את שמה, את המחיר הממוצע למוצר, ואת השמות של המוצרים שנמכרים בה ומחירם.

המוצרים יהיו ממויינים בסדר עולה של מחירם.

התכנה גם תציג כפלט את המחיר הממוצע למוצר בקניון.

הנחות: 1. הקלט תקין. 2. יש 20 חנויות לכל היותר.

יש לפתח ארבע מחלקות (כולל המחלקה המכילה את הפעולה הראשית). דוגמה להרצה:

```
*** enter shop name ***
crazyShop
enter number of products
3
enter product name
pants
enter product price
200
enter product name
shirt
enter product price
100
enter product name
sweater
enter product price
160
*** enter shop name ***
eko
enter number of products
2
enter product name
shoes
enter product price
240
enter product name
jeans
enter product price
120
*** enter shop name ***
end

shop number 1 is crazyShop
average price = 153.33333333333334
shirt cost 100.0
sweater cost 160.0
pants cost 200.0
shop number 2 is eko
average price = 180.0
jeans cost 120.0
shoes cost 240.0
average price in the mall is 164.0
```


© כל הזכויות שמורות, תמר פז

משימה 26 – חלק ב'

במשימה האחרונה הנחנו כי בקניון יש 20 חנויות לכל היותר.

מדוע היינו זקוקים להנחה הזו? _____

_____

נפתור כעת את אותה המשימה בלי ההנחה לגבי מספר החנויות בקניון.

מכיוון שאנו לא יודעים מהו מספר החנויות המקסימלי, נחליט על מספר כלשהו סביר של חנויות, למשל 15 ובעת בנית המערך, נבנה אותו בהתאם לגודל עליו החלטנו למשל כך: allShops = new Shop [15];.

כעת, לפני כל הוספה של חנות חדשה למערך החנויות, נצטרך לבדוק אם יש עדיין מקום במערך. במידה והמערך כבר מלא, נרצה להגדיל אותו כדי שנוכל להוסיף לו את החנות החדשה. אבל, כיוון שאי אפשר להגדיל מערך, נבנה מערך חדש גדול יותר מהמערך הישן, נעתיק לתחילתו את המערך הישן, ונשנה את המשתנה בו מוחזקת ההפניה למערך הישן כך שתוחזק בו ההפניה למערך החדש. למשל,

```
Shop[ ] temp = new Shop [allShops.length + 10];
for (int i = 0; i < allShops.length; i++)
        temp[i] = allShops [i];
allShops = temp;
```

temp - מערך חדש בעל 10 איברים יותר מאשר ב allShops . בכל איבר תהיה הפניה ל Shop.

שימו לב, **השינוי הוא רק במחלקה "קניון"** שאחת מתכונותיה היא מערך של חנויות.

- תפקיד הלולאה הוא להעתיק את כל האיברים של המערך allShops אל האיברים הראשונים של המערך _____.
- תפקיד ההוראה allShops = temp הוא לשים במשתנה allShops הפניה למערך החדש.

הוראה זו חיונית משום ש_____

_____

שנו כעת את הפתרון למשימה האחרונה כך שתיפתר ללא ההנחה כי יש 20 חנויות לכל היותר.

מערך דינמי הוא מערך שניתן לשנות את גודלו בזמן הרצת התכנית.

בג'אווה אין אפשרות להגדיר מערך דינמי. כדי להגדיל מערך, אפשר **לבנות מערך חדש גדול יותר מהמערך הישן, להעתיק לתחילתו את המערך הישן, ולשנות את המשתנה בו מוחזקת ההפניה למערך הישן כך שתוחזק בו ההפניה למערך החדש**.

© כל הזכויות שמורות, תמר פז