תוכנה 1

תרגיל מספר 9

הורשה, חידות Java

<u>הנחיות כלליות:</u>

קראו בעיון את קובץ נהלי הגשת התרגילים אשר נמצא באתר הקורס.

את התרגיל הבא ניתן להגיש באחת משתי הדרכים הבאות:

1. הגשה במערכת ה-moodle (http://moodle.tau.ac.il/) עפ"י ההנחיות הבאות:

יש להגיש קובץ zip יחיד הנושא את שם המשתמש ומספר התרגיל (לדוגמא, עבור המשתמש **yael3** יקרא הקובץ **yael3**_hw9.zip). קובץ ה-zip יכיל:

א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז.

ב. תיקיית **src** ובתוכה היררכיית התיקיות כפי שקיבלתם בקובץ הזיפ, כולל קבצי הג'אווה שסופקו לכם (אשר נוסף להם הקוד שלכם). (שימו לב שתיקיית ה-src עצמה נמצאית ברמה העליונה בקובץ הזיפ, כלומר היא לא מוכלת באף תיקיה אחרת).

ג. קובץ PDF בשם answers.pdf המכיל את התשובות לשאלות.

2. הגשה בmoodle וגם הגשה בgit.
   צרו את הrepository שלכם מתוך הקישור הבא:
   https://classroom.github.com/a/s2n3egl0
   להנחיות להגשת הgit ניתן לעקוב אחר ההנחיות ממטלה 3.

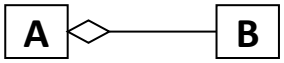
נא לא להשתמש בפקודה System.exit()! היא מחבלת בבדיקות אוטומטיות. אין כל צורך לעשות בה שימוש, כאשר תוכניות יכולות להסתיים ע"י הגעה לסוף מתודת ה-main.

יצירת פרוייקט והגשה: חזרו על ההוראות ממטלה 3 לגבי יצירת פרויקט וייבוא הקבצים. התהליך הוא זהה במטלה זו: יש ליצור פרויקט ג'אווה חדש באקליפס (ולשים לב, למיקום של הworkspace במחשבכם). כעת יש להיכנס לתיקיית הפרויקט במחשב, ולהעתיק לשם את תיקיית ה-src מתוך קובץ הזיפ, כך שתיקיית ה-src הקיימת תידרס. תיקיית ה-src הזו היא התיקיה שתצרפו לזיפ בתום כתיבת הקוד. חזרו כעת לאקליפס, ובלחיצה ימנית על הפרוייקט בחרו refresh.

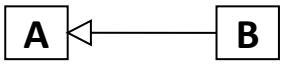
חלק א': הורשה (80%)

<u>הערה כללית</u>: בתרגיל זה אתם מתבקשים, בין היתר, לשרטט דיאגרמות של מחלקות. השתמשו בסימונים הבאים בלבד:

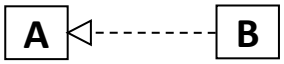
- (aggregation) יחס של **הכלה** (למשל, ל-A יש שדה מטיפוס B)



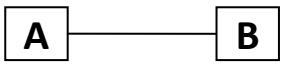
- **ירושה** (B מחלקה היורשת את A):



- **מימוש** (B מחלקה מממשת\ מנשק היורש את המנשק A):



- (association) קשר כללי שאינו נופל בקטגוריות הקודמות. למשל, A משתמש במשתנה מטיפוס B באחת המתודות.



יש לציין: בתוך כל מלבן <<interface>>, <<abstract>> או <<class>>, ואת שם המנשק או המחלקה.

אין צורך לציין: מספרים ושמות שדות על יחסי אגרגציה ואסוציאציה; שמות מתודות ושדות בתוך מלבני המחלקות; יחסים "עקיפים" בין מחלקות (כלומר, אם C יורש מ-B שיורש מ-A, אין צורך לציין קשר בין C ל-A אלא אם יש ביניהם קשר ישיר בנוסף, למשל של הכלה)

יצירת הדיאגרמות: ניתן לעשות זאת דרך PowerPoint, Word, תוכנת הציור המועדפת עליכם או לסרוק שרטוט (בכתב ברור!).

Starfleet Command

בתרגיל זה נבנה מערכת תוכנה לניהול צי חלליות עתידני.

בתחילה נבנה מחלקות שייצגו את אנשי הצוות ואת סוגי החלליות השונים תוך שימוש במנשקים, מחלקות אבסטרקטיות והורשה. לאחר מכן, ניצור אובייקטים של מחלקות אלו ולבסוף נדפיס מספר דוחות המציגים חיתוכי מידע שונים על צי החלליות שלנו כגון עלות אחזקה כוללת, כוח-אש כולל של חלליות הצי ועוד.

הנחיות כלליות:

א. אין לשנות הגדרות של מנשקים, ובפרט לא חתימות של מתודות.
ב. מותר להוסיף לכותרות של מחלקים ומנשקים extends ו-implements.
ג. ניתן להוסיף מחלקות עזר כרצונכם, כלל מחלקות אבסטרקטיות.
ד. ניתן להוסיף פונקציות עזר כרצונכם למחלקות שאתם מממשים.
ה. השמות של כל המחלקות שתוסיפו על דעת עצמכם (לא כלל שמות של מחלקות ומנשקים שהוזכרו בקובץ ההנחיות והשלד) מוכרחים להתחיל במילה my.
ו. למנשקים לא ניתן להוסיף מתודות אבסטרקטיות, אך ניתן (אם כי לא חובה) להוסיף מתודות עם מימוש (default or static).
ז. הקוד של הטסטר יתקמפל רק לאחר השלמת המימוש.

Starfleet Personnel

צי החלל של פדרציית הכוכבים המאוחדת כולל 3 סוגי אנשי צוות (Crew-Members):

1. CrewWomen – חברת צוות אנושית רגילה.
2. Officer – חברת צוות אנושית שהינה קצינה (בעלת דרגת קצונה).
3. Cylon – חברת צוות סיילונית (רובוטית דמויית אדם). מכיוון שסיילונים מתחזים לבני אדם, ניתן לראותם בצוותים של כל כלי הטייס בצי. סיילונית אינה יכולה להיות גם קצינה. לקריאה נוספת על סיילונים: (https://en.wikipedia.org/wiki/Cylon_(Battlestar_Galactica

להלן תיאור השירותים בהם תתמוך כל אחת מהמחלקות המייצגות את סוגי אנשי הצוות הנ"ל:

CrewWomen

שם השירות	טיפוס החזרה	הסבר
getName()	String	שם חברת הצוות (מחרוזת המציינת שם ייחודי לכל חברת צוות).
getAge()	int	גילה של חברת הצוות (בשנות כדור-הארץ, למשל 28).
getYearsInService()	int	מספר שנות השירות של חברת הצוות (בשנות כדור-הארץ, למשל 1).

Officer

חברת צוות שהינה קצינה תכלול את כל התכונות של חברת צוות רגילה, ובנוסף תהיה לה גם דרגת קצונה:

שם השירות	טיפוס החזרה	הסבר
getRank()	OfficerRank	הדרגה של הקצין (מיוצג ע"י Enum בשם OfficerRank).

Cylon

שם השירות	טיפוס החזרה	הסבר
getName()	String	שם חברת הצוות (מחרוזת המציינת שם ייחודי לכל חברת צוות).

getAge()	int	גילה של חברת הצוות (בשנות כדור-הארץ, למשל 28).
getYearsInService()	int	מספר שנות השירות של חברת הצוות (בשנות כדור-הארץ, למשל 10).
getModelNumber()	int	מספר המודל על פיו נוצרה הסיילונית (בין 1 ל 12 - הניחו כי האתחול יהיה תקין).

שימו לב:

- בהמשך נגדיר מנשק בשם **CrewMember** אשר ייצג חברת צוות מסוג כלשהו.
- הטיפוס **OfficerRank** הוא Enum המגדיר קבועים המציינים את דרגות הקצונה. טיפוס זה נתון לכם.
- ההתייחסות לאנשי הצוות היא בלשון נקבה אך מתייחסת לחברי צוות משני המינים.
- בהמשך אנו נגדיר מחלקות עבור חלליות. נציין כבר עכשיו כי ניתן להניח שאף איש צוות לא מאייש יותר מחללית אחת.

Starfleet Ships

צי החלל של פדרציית הכוכבים המאוחדת כולל 6 סוגי חלליות:

1. Transport Ship – חללית תובלה המאפשרת שינוע נוסעים ומטען בין בסיסי חלל.
2. Fighter – חללית קרב (Battleship) קטנה ומהירה.
3. Bomber – חללית קרב (Battleship) גדולה בעלת עוצמת אש אדירה.
4. Stealth Cruiser –חללית קרב (Battleship) מהירה בעלת יכולת חמקנות (Stealth).
5. Cylon Rider – חללית קרב מהירה המאושיית רק ע"י סיילוניות (Cylons).
6. Colonial Viper – חללית קרב מהירה המאושיית רק ע"י חברות צוות אנושיות (CrewWomen).

להלן פירוט המאפיינים של סוגי החלליות השונות:

Spaceship

נתחיל בתיאור השירותים המשותפים לכל סוגי החלליות. עבור כל חללית (להלן Spaceship) נגדיר את השירותים הבאים:

שם השירות	טיפוס החזרה	הסבר
getName()	String	שם החללית (מחרוזת המציינת שם ייחודי לכל חללית).
getCommissionYear()	int	שנת ייצור (בשנות כדור הארץ, למשל 2241).
getMaximalSpeed()	float	מהירות מקסימלית (שבר בין 0 ל-10).
getFirePower()	int	סכום כוח-אש של כל כלי הנשק המותקנים בחללית (מספרים שלמים, ביחידות של כוח-אש). לכל חללית יש כוח-אש בסיסי מובנה של 10 יחידות כוח-אש. בחלליות קרב מתווסף כוח-אש נוסף מכלי הנשק המותקנים על החללית.
getCrewMembers()	Set<? Extends CrewMember>	חברי הצוות המאיישים את החללית (CrewMember הינו מנשק המייצג איש-צוות מסוג כלשהו).
getAnnualMaintenanceCost()	int	עלות אחזקה שנתית כוללת (מספרים שלמים) ביחידות של דולר-פדרציה. נתון זה יחושב באופן שונה לכל סוג חללית על פי המפורט בהמשך.

Transport Ship

עבור חללית תובלה נגדיר את כל השירותים של חללית המובאים לעיל, בתוספת ההגדרות הבאות:

שם השירות	טיפוס החזרה	הסבר
getCargoCapacity()	int	יכולת נשיאת מטען, ביחידות של מגה-טון (מספר שלם)
getPassengerCapacity()	int	יכולת נשיאת נוסעים, ביחידות של מספר נוסעים (מספר שלם)
getAnnualMaintenanceCost()	int	עלות האחזקה השנתית הכוללת של ספינת תובלה מורכבת מסכום הרכיבים הבאים: • עלות אחזקה שנתית בסיסית לספינת תובלה (3000 דולר). • עלות של 5 דולר לכל מגה-טון של יכולת נשיאת מטען (כלומר CargoCapacity * 5 דולר). • עלות של 3 דולר פר יכולת נשיאת נוסע (כלומר PassengerCapacity * 3 דולר).

Fighter

חללית קרב מהירה. נגדיר עבורה את כל השירותים של חללית בנוסף להגדרות הבאות:

שם השירות	טיפוס החזרה	הסבר
getWeapon()	List<Weapon>	רשימת כלי הנשק המותקנים על חללית הקרב. עבור כל נשק (Weapon) נשמור את הנתונים הבאים: • שם כלי הנשק. • כוח-אש (ביחידות כוח-אש). • עלות אחזקה שנתית (בדולרים).
getFirePower()	int	כוח האש המצטבר של חללית קרב הינו סכום כוח-האש של כל הנשקים המותקנים, בנוסף לכוח האש המובנה של כל חללית.
getAnnualMaintenanceCost()	int	עלות האחזקה השנתית של חללית קרב מסוג Fighter מורכבת מסכום הרכיבים הבאים: • עלות אחזקה שנתית בסיסית לספינת קרב Fighter (2500 דולר). • עלות אחזקה השנתית של כלי הנשק (סכום עלות האחזקה של כל כלי הנשק המותקנים על חללית הקרב). • עלות אחזקת מנועי החללית ששוה ל 1000 דולר כתלות במהירות החללית המקסימלית (MaximalSpeed * 1000, מעוגל לשלמים).

Bomber

חללית קרב כבדה בעל יכולת הפצצה מרשימה. נגדיר עבורה את כל השירותים של חללית בנוסף להגדרות הבאות:

שם השירות	טיפוס החזרה	הסבר
getWeapon()	List<Weapon>	רשימת כלי הנשק המותקנים על חללית הקרב. עבור כל נשק נשמור את הנתונים הבאים: • שם כלי הנשק. • כוח-אש (ביחידות כוח-אש). • עלות תחזוקה שנתית (בדולרים).
getFirePower()	int	כוח האש המצטבר של חללית קרב הינו סכום כוח-האש של כל הנשקים המותקנים, בנוסף לכוח האש המובנה של כל חללית.
getNumberOfTechnicians()	int	מספר הטכנאים המוצבים על החללית (מספר שלם בטווח 0- 5 – אין צורך לבדוק את הקלט). הטכנאים אינם משפיעים על חישובי גודל הצוותים המובאים בהמשך.
getAnnualMaintenanceCost()	int	עלות האחזקה השנתית של חללית קרב מסוג Bomber מורכבת מסכום הרכיבים הבאים: • עלות אחזקה שנתית בסיסית לספינת קרב מסוג Bomber (5000 דולר). • עלות האחזקה השנתית של כלי הנשק (עלות האחזקה של כל כלי הנשק המותקנים על חללית הקרב). • כל טכנאי המוצב על החללית מוזיל את עלויות האחזקה השנתיות על כלי הנשק ב-10%. כלומר, עלות תחזוקת כלי הנשק מופחתת בשיעור של 0- 50% כתלות במספר הטכנאים (מס' הטכנאים נע בין 0 ל 5). יש לעגל את המחיר לשלמים אחרי חישוב ההוזלה ביחס לסכום עלויות כלי הנשק.

StealthCruiser

חללית קרב מהירה הכוללת גם יכולת חמקנות מתקדמת, משמשת למשימות סיור בעומק שטח האויב. נגדיר עבורה את כל השירותים של חללית קרב מהירה (Fighter), בנוסף להגדרות הבאות:

שם השירות	טיפוס החזרה	הסבר

getAnnualMaintenanceCost()	int	עלות האחזקה השנתית של חללית קרב מסוג StealthCruiser מורכבת מסכום הרכיבים הבאים:
		• עלות אחזקה שנתית חללית קרב (Fighter).
		• תוספת שנתית בגין תחזוקה של מנוע החמקנות (Cloaking device).
		  ○ על כל חללית מסוג StealthCruiser מותקן מנוע חמקנות יחיד.
		  ○ עלות האחזקה השנתית של מנוע החמקנות תלויה במספר מנועי החמקנות הקיימים בצי. עלות האחזקה של כל מנוע

	* חמקנות מחושבת כמספר המנועים בצי 50 דולר פדרציה (למשל אם קיימים בצי 4 מנועי חמקנות, עלות האחזקה של כל אחד מהם היא 200 דולר-פדרציה). ○ ניתן להניח שמספר מנועי החמקנות בצי שווה למספר המופעים שנוצרו עבור סוג החללית StealthCruiser.

ColonialViper

חללית קרב מהירה המאיישת ע"י חברות צוות אנושיות. נגדיר עבורה את כל השירותים של חללית קרב מהירה , בנוסף להגדרות הבאות:

שם השירות	טיפוס החזרה	הסבר
getAnnualMaintenanceCost()	int	עלות האחזקה השנתית של חללית קרב מסוג ColonialViper מורכבת מסכום הרכיבים הבאים: • עלות אחזקה שנתית בסיסית לספינת קרב מסוג ColonialViper (4000 דולר). • עלות אחזקה השנתית של כלי הנשק (סכום עלות האחזקה של כל כלי הנשק המותקנים על חללית הקרב). • עלות אחזקה שנתית על טיפול בכל חברת צוות המוצבת על הספינה. תחזוקת כל חברת צוות עולה 500 דולר פדרציה. • עלות אחזקת מנועי החללית שהוא 500 דולר כתלות במהירות החללית המקסימלית.

CylonRider

חללית קרב מהירה המאוישת ע"י סיילוניות. נגדיר עבורה את כל השירותים של חללית קרב מהירה , בנוסף להגדרות הבאות:

שם השירות	טיפוס החזרה	הסבר
getAnnualMaintenanceCost()	int	עלות האחזקה השנתית של חללית קרב מסוג CylonRider מורכבת מסכום הרכיבים הבאים: • עלות אחזקה שנתית בסיסית לספינת קרב מסוג CylonRider (3500 דולר). • עלות אחזקה השנתית של כלי הנשק (סכום עלות האחזקה של כל כלי הנשק המותקנים על חללית הקרב). • עלות אחזקה שנתית על טיפול בכל חברת צוות סיילונית המוצבת על הספינה. תחזוקת כל חברת צוות עולה 500 דולר פדרציה.

• עלות אחזקת מנועי החללית שהוא 1200 דולר כתלות במהירות החללית המקסימלית.

מה עליכם לעשות ?

1. <u>הגדרת מנשקים (Interfaces) (5%)</u>

<u>הגדרת המנשק CrewMember</u>

- הגדירו מנשק בשם **CrewMember** אשר ייצג חברת צוות בצי החלל. על המנשק לכלול את המתודות getName(), getAge(), getYearsInService().
- על כל מחלקה המייצגת חברת צוות לממש מנשק זה.

<u>הגדרת המנשק Spaceship</u>

- הגדירו מנשק בשם **Spaceship** אשר ייצג חללית בצי החלל. על המנשק לכלול את המתודות המאפיינות חללית כלשהי (היעזרו בטבלה המתארת Spaceship כללי לעיל).
- על כל מחלקה המייצגת חללית לממש מנשק זה.

✓ הגדרת מנשק מאפשרת לנו לעבוד בצורה אחידה עם מחלקות שונות המממשות אותו. למשל - נוכל ליצור אוסף פולימורפי המכיל אובייקטים של חלליות מסוגים שונים ולגשת אליהם בצורה אחידה דרך המתודות המוגדרות במנשק *Spaceship*.

✓ יש לממש את כל המחלקות והמנשקים בחבילה: `il.ac.tau.cs.sw1.ex9.starfleet`. שלדי המחלקות והמנשקים נתונים לכם בתיקיית קבצי התרגיל.

2. <u>הגדרת עץ ההורשה (10%)</u>

- נתחו את הדמיון בין המחלקות השונות שהוגדרו לעיל עבור חברות צוות ועבור חלליות, ובנו עצי הורשה מתאימים אשר יכללו מנשקים, מחלקות אבסטרקטיות, מחלקות קונקרטיות ומחלקות עזר אם קיימות.
- יחסי ההורשה בין המחלקות אמורים למנוע שכפול קוד בין מחלקות.
- שרטטו את היחסים בין המחלקות השונות על פי המוגדר בראש התרגיל <u>והגישו את דיאגרמת המחלקות בקובץ התשובות</u>.
- וודאו שהשרטוט שלכם מכיל את כל המחלקות הקונקרטיות המפורטות בתחילת הסעיף הבא.

3. <u>מימוש המחלקות (25%)</u>

בהתבסס על עצי ההורשה אותם הגדרתם ולפי פירוט המתודות שהובא בטבלאות לעיל, ממשו את המחלקות הבאות:

1) CrewWoman
2) Officer
3) Cylon
4) TransportShip
5) Fighter
6) Bomber
7) StealthCruiser
8) ColonialViper
9) CylonRider

- מימוש המחלקה Weapon נתון לכם ומופיע בתיקיית קבצי הפרוייקט.

- במידה ובחרתם להגדיר מחלקות אבסטרקטיות, ממשו גם אותן. (זכרו להתחיל שם כל מחלקה שהוספתם מעבר לקובץ ההנחיות והשלד עם המילה my).
- בנאים - לכל מחלקה ניצור בנאי המקבל את כל הפרמטרים הנדרשים לאתחול שדות המחלקה. חתימות הבנאי של כל מחלקה מופיעות בשלד המחלקות.
- בבנאי של Fighter מותר, אך לא חובה, לשנות את הפרמטר
  Set<CreMember> crewmembers
  להפוך להיות
  Set<? Extends CrewMember> crewmembers

- עבור המחלקה `StealthCruiser`, נממש שני בנאים, כאשר אחד מהם לא יקבל רשימת נשקים :

```
public StealthCruiser(String name, int commissionYear, float maximalSpeed,
Set<CrewMember> crewMembers)
```

היות ומרבית החלליות מסוג `StealthCruiser` יכללו רק תותחי לייזר סטנדרטיים בתור חימוש, בנאי זה (אשר אינו מקבל את הנשקים כפרמטר) יצור אובייקט המייצג חללית מסוג `StealthCruiser` עם רשימת נשקים הכולל את האובייקט הבא בלבד:

```
new Weapon ("Laser Cannons",10,100)
```

על בנאי זה לעשות שימוש בבנאי המלא שהוגדר קודם לכן (שימו לב שקריאה לבנאי אחר של המחלקה יכולה להיעשות רק מהשורה הראשונה של הבנאי).

- מתודת toString() – בכל אחת מהמחלקות עליכם לממש מתודת toString() המחזירה מחרוזת המתארת את נתוני המחלקה.
  - המחרוזת תתחיל בשם המחלקה, ואח"כ מוסטים לימין ע"י טאב בודד יופיעו נתוני המחלקה לפי הסדר והפורמט המודגמים בהמשך (סדר הופעת השדות יהיה: שדות המחלקה המשותפים לכל סוגי החלליות, אח"כ תופיע עלות האחזקה השנתית, ולאחר מכן השדות הספציפיים לאותה המחלקה).
  - מתודת ה-toString() עשויה לקרוא למתודה באותו שם במחלקת האם.
  - הקפידו שהמחרוזות שנוצרות יהיו **זהות** לאלו המוצגות בקובץ הפלט הנלווה. **מומלץ להשתמש ב copy-paste ממסמך זה ולא להקליד את המחרוזות ידנית.**

להלן דוגמא למחרוזת המיוצרת ע" מתודת ה-toString של מחלקת `TransportShip` (הדגש הוא על הפורמט ולא על נכונות הערכים!):

```
TransportShip
	Name=USS Lantree
	CommissionYear=23571
	MaximalSpeed=5.1
	FirePower=9
	CrewMembers=8
	AnnualMaintenanceCost=40000
	CargoCapacity=3000
	PassengerCapacity=10000
```

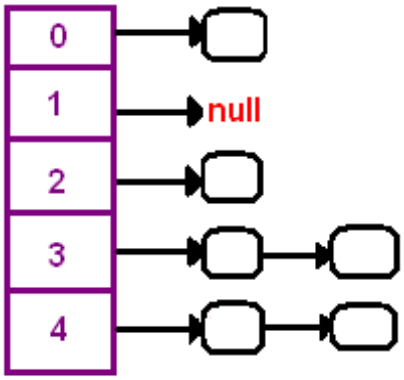
- דריסת המתודות equals() ו-hashCode()

- **public boolean** equals(Object obj)
- **public int** hashCode()

על מנת שנוכל לאחסן אובייקטים של מחלקות שיצרנו במבני נתונים המבוססים על HashTable יש לדרוס את המתודות ()equals (הבודקת זהות מול אובייקט אחר) ואת המתודה ()hashCode (המחזירה ערך גיבוב). וודאו שכל מחלקה המייצגת איש צוות או חללית מכילה דריסה של 2 מתודות אלו (אך הימנעו משכפול קוד מיותר תוך שימוש בהורשה).

- ✓ שימו לב ששדה השם מהווה ערך מזהה ייחודי עבור אנשי צוות ועבור חלליות.
- ✓ היעזרו באקליפס ליצירה אוטומטית של מתודות אלו (**Source>Generate hashCode() and equals()...**), אך וודאו שאתם מבינים את הקוד שנוצר.

- מבנים מסוג Hash עובדים באופן הבא:
  -
  - מבני נתונים אלה ממומשים באופן הבא:



  - לכל אובייקט יש hashCode שמשמש כאינדקס במערך. לכל תא במערך יש רשימה מקושרת שבה נשמור את כל האיברים להם אותו ה hashCode (מה שמכונה collision – התנגשות ב hashCodes). בד"כ הפיזור של האיברים יהיה אחיד ולכן על מנת לשלוף/להכניס איבר מסויים לא נצטרך לעבור על כל האיברים באוסף, אלא רק על אלה החולקים איתו את אותו ה hashCode, ומספרם יהיה קטן.
  - במידה וקיימים מספר אובייקטים להם אותו HashCode, נשתמש ב equals בשביל להבדיל ביניהם.
    - בהכנסה: אם נכניס אובייקט ל HashSet שבו כבר קיים אובייקט (או אובייקטים) עם אותו ה HashCode, נבדוק את הערך של equals על שניהם. אם מדובר באובייקטים שונים, שניהם יכנסו ל HashSet, אחרת זו תהיה הכנסה כפולה ורק אחד מהם יופיע ב HashSet.
    - בשליפה: למשל, ב contains של Set או get של Map: נחשב את ערך ה hashCode של האובייקט. מבין כל האובייקטים להם אותו ה hashCode נחפש את האובייקט שלנו, ע"י שימוש ב equals.
  - קריאה נוספת: http://coding-geek.com/how-does-a-hashmap-work-in-java/

- תמיכה במיון של אובייקטים מסוג איש צוות או חללית

ייתכן ותרצו שהמחלקות שיצרתם יממשו את המנשק Comparable כדי שניתן יהיה להשתמש בהן עם מתודות או מבני נתונים הדורשים הגדרת יחס סדר (כגון Collection.sort או כמפתחות של TreeMap). לחילופין, תוכלו בהמשך להגדיר מחלקת עזר חיצונית המממשת את המנשק Comparator ולספק אותה כמגדירת יחס סדר למתודה או לבנאי של מבנה הנתונים הרלבנטי.

הערות כלליות לסעיף זה:

- אתם רשאים להוסיף שדות, מתודות ומחלקות עזר נוספות בכל אחת מהמחלקות שלכם כל זמן שאתם לא פוגעים בחתימות ובמנשק המוגדרים לעיל.
- שימו לב לנראות השדות בכל אחד משלבי היררכיית הירושה. לא ניתן לגשת לשדות המוגדרים כפרטיים במחלקת האם.
- הקפידו להשתמש בקבועים כשאלו נדרשים.

4. StarfleetManager (40%)

מחלקה זו (עבורה נתון לכם השלד) תכיל מספר מתודות סטטיות המקבלות אוסף חלליות ומחזירות חיתוכים שונים על פי הפירוט הבא:

```
1. public static List<String>
   getShipDescriptionsSortedByFirePowerAndCommissionYear
   (Collection<Spaceship> fleet)
```

(5%) המתודה תחזיר רשימה של מחרוזות המתארות את חלליות הצי, כאשר החלליות ממוינות קודם לפי עוצמת אש (**בסדר יורד**), אחר כך לפי שנת ייצור (**בסדר יורד**), ואחר מכן על פי שם החללית (**בסדר עולה**). זה אומר שאם לשתי חלליות עוצמת אש זהה, נשווה את שנות הייצור, ואם הן זהות, נשווה את שמות החלליות.
כל איבר ברשימה המוחזרת יהיה מחרוזת שהינה תוצר של מתודת ה-()toString של אובייקט החללית המתאים.

```
2. public static Map<String, Integer> getInstanceNumberPerClass
   (Collection<Spaceship> fleet)
```

(5%) המתודה תחזיר מפה המכילה עבור כל שם מחלקה של חללית את מספר האובייקטים שנוצרו מהמחלקה (רק אם נוצרו, אין לכלול מחלקות שלא נוצרו מהן אובייקטים).

✓ ניתן להשתמש במתודה ()getClass על כל אובייקט כדי לדעת מאיזו מחלקה הוא (מקבלים חזרה אובייקט מסוג Class ואז ניתן לקבל את שם המחלקה באמצעות המתודה ()getSimpleName).

```
3. public static int getTotalMaintenanceCost (Collection<Spaceship>
   fleet)
```

(5%) המתודה תחזיר את סך כל עלויות האחזקה של כל חלליות הצי ע"י סכימת עלויות האחזקה של כל חללית בצי.

```
4. public static Set<String> getFleetWeaponNames
   (Collection<Spaceship> fleet)
```

(5%) המתודה תחזיר אוסף מסוג קבוצה המכיל מחרוזות המייצגות את שמות כלי הנשק השונים (ללא חזרות) המותקנים על חלליות הצי.

```
5. public static int
   getTotalNumberOfFleetCrewMembers(Collection<Spaceship> fleet)
```

(5%) המתודה תחזיר את מספר אנשי הצוות הכולל בצי (סכום אנשי הצוות המוצבים בכל חללית)

```
6. public static float getAverageAgeOfFleetOfficers(Collection
   <Spaceship> fleet)
```

(5%) המתודה תחזיר את הגיל הממוצע של קציני הצי (ניתן להניח שקיים לפחות קצין אחד בצי, גם בסעיפים הבאים).

```
7. public static Map<Officer, Spaceship>
   getHighestRankingOfficerPerShip(Collection<Spaceship> fleet)
```

(5%) המתודה תמצא את הקצין בעל הדרגה הבכירה ביותר המוצב על כל חללית בצי, ותחזיר מפה הממפה לכל קצין כזה את החללית בה הוא מוצב. ניתן להתעלם מחלליות עליהם לא מוצבים קצינים.

שימו לב שהטיפוס OfficerRank שמסופק לכם, הוא Enum אשר מונה את דרגות הקצונה על פי סדר הבכירות שלהן. Enum בג'אווה מממש את המנשק Comparable ועל כן ניתן למיין לפיו.

```
8.  public static List<Map.Entry<OfficerRank, Integer>>
    getOfficerRanksSortedByPopularity (Collection<Spaceship> fleet)
```

(5%) המתודה תחזיר רשימה של אובייקטים מסוג Map.Entry המכילים זוגות של דרגה, ומספר המופעים שלה בקרב קציני הצי. הרשימה המוחזרת תהיה ממוינת בסדר עולה לפי מספר מופעי הדרגה בצי, והמיון המשני יהיה בסדר עולה של הדרגות (כלומר, שתי דרגות שמופיעות אותו מספר פעמים יסודרו על פי הדרגות בסדר עולה). שימו לב ש enum מממש comparable ולכן אובייקטים מטיפוס OfficerRank הם ברי השוואה. לא צריך לכלול דרגות עם 0 מופעים.

הנחיה: בנו מפה אשר מאחסנת עבור כל דרגה את מספר המופעים שלה, אח"כ השתמשו ב-()Map.getEntries לקבלת אוסף זוגות המייצג את המפה, העבירו את האוסף לרשימה, מיינו את הרשימה והחזירו אותה.

בנוסף, עבור כל הפונקציות ניתן להניח שfleet לא מכילה כפילויות.

5. StarfleetManagerTester

המחלקה StarfleetManagerTester מייצרת צי של חלליות על צוותיהן ומשתמשת בכל המחלקות והמתודות שכתבתם כדי להדפיס דוח מסכם למסך. לאחר שסיימתם את מימוש כל המחלקות, הריצו את המחלקה StarfleetManagerTester שמסופקת לכם בשלמותה ובדקו שהפלט המודפס על ידכם זהה לפלט המצורף לקבצי התרגיל (בכל מקום שבו מופיעה הזחה, ניתן להניח שמדובר בטאב בודד).

- ✓ המחלקה StarfleetManagerTester מייצרת צי חלליות וצוותים בצורה אוטומטית (אך לא רנדומאלית, על מנת שתוכלו לקבל פלט זהה לשלנו בכל הרצה). בשירותים אשר מחזירים מבנה נתונים לא מסודר (Set, Map) מבוצע סידור לפני ההדפסה, על מנת לוודא פלט אחיד.
- ✓ שמות אנשי הצוות והחלליות אמורים להיות ייחודיים ועל כן יצרנו שמות מהצורה "James #121".

הערות כלליות:

- במהלך התרגיל יש מקרים בהם עליכם לעגל מספר. ניתן להסיק את כיוון העיגול לפי קובץ הפלט המצורף.
- חוץ מאשר במקרים שצוינו במפורש בהנחיות, ניתן להניח את תקינות הקלט .
-

חלק ב': חידות java (20%)

בשאלה זו נשלים קוד ג'אווה שמשמעותו אינה ידועה, תוך תרגול של הבנת הקוד וחזרה על עקרונות שנלמדו בכיתה בנושאים שונים. באתר הקורס נתונה לכם החבילה `il.ac.tau.cs.sw1.ex9.riddles`, ותחתיה ארבע חבילות: `first, second, third, forth`. כל חבילה מכילה שתי מחלקות\מנשקים, A ו- C, והמימוש של \מנשק B הוא ריק(המחלקות ממוספרות מ 1-4 בהתאם לחבילות). עליכם להשלים את B מבלי לשנות את הקוד של A ו-C ומבלי להוסיף קבצים אחרים, כך ש:

1. כל הקוד יעבור קומפילציה ללא שגיאות וללא אזהרות.
2. בכל המחלקות C נתונה פונקציית `main`. אם נריץ את התכנית C עם ארגומנט אחד לפחות, היא תמיד תסיים את הריצה ללא שגיאות ותדפיס `!success`

הערות:

- יש להגיש רק את ארבעת קבצי B (`B1, B2, B3, B4`) שכתבתם בתוך מבנה החבילות המתאים.
- אין מגבלה על המימוש של B, כל עוד הוא מימוש תקין ב Java.
- מומלץ, בהינתן שגיאת קומפילציה, לחשוב תחילה בעצמכם כיצד לפתור אותה, לפני שתיעזרו בהצעות של `eclipse`. אם אתם משתמשים בהצעות אלה, היזהרו לא לשנות בטעות את קבצי A או C, משום ששינוי זה עשוי לפגוע בנכונות הפתרון שלכם.
- בדקו את עצמכם ע"י הרצה של התכנית עם ארגומנטים שונים. את התכנית בחבילה `second` כדאי להריץ מספר פעמים עבור כל קלט ליתר ביטחון, מכיוון שהיא משתמשת בערך רנדומי.
- הקוד לא תמיד מקיים את קונבנציות הקידוד של `java` ונועד להיות קשה לקריאה. בפרט, לא מופיעות הערות בגוף הקוד. מכיוון שכך, אין להסיק ממנו על צורת כתיבה נכונה ב `java`, אלא להיפך.
- אסור להדפיס דבר נוסף מלבד `!success`.

בהצלחה!