234124 - מבוא לתכנות מערכות
תרגיל בית מספר 4
סמסטר חורף 23/24 (אודיסיאה)

תאריך פרסום: 02/01/2024
תאריך הגשה: 01/02/2023 בשעה 23:59

1 הערות כלליות

- **תרגיל זה מהווה 8% מהציון הסופי**
- התרגיל להגשה **בזוגות** בלבד
- מענה לשאלות בנוגע לתרגיל יינתן אך ורק בפורום התרגיל בפיאצה או בסדנאות. לפני פרסום שאלה בפורום אנא בדקו אם כבר נענתה.
- קראו את התרגיל עד סופו לפני שאתם מתחילים לממש. חובה להתעדכן בעמוד הפיאצה של התרגיל, הכתוב שם מחייב.
- העתקות קוד בין סטודנטים ובפרט גם העתקות מסמסטרים קודמים תטופלנה. עם זאת – מומלץ ומבורך להתייעץ עם חברים על ארכיטקטורת המימוש.
- קבצי התרגיל נמצאים ב-**github** תחת הקישור הבא: https://github.com/cs234124-odyssey/ex4.git
- ניתן להשתמש במימוש שלכם מתרגיל בית 2 ו-3 אבל לא חובה.
- המסמך נכתב בלשון זכר מטעמי נוחות בלבד ומיועד לשני המינים.

2 MTMCHKIN – FULL edition

בתרגיל בית 4 נממש את המשחק כולו בגרסתו המלאה לקורס זה.

2.1 תיאור המשחק

כמו בתרגיל בית 2, מטרת המשחק היא לנצח בקרבות ולהגיע לרמה 10. הפעם המשחק יהיה מרובה שחקנים (מ-2 עד 6 שחקנים) ונציג סט חדש של קלפים.
בתחילת המשחק, כל שחקן בוחר את סוג הדמות אותה הוא רוצה לשחק. לאחר מכן, בכל סיבוב, כל שחקן שולף קלף אחד בתורו מחפיסת הקלפים ומשחק את הקלף. הקלף הנשלף, יכול להטיב עם השחקן ולהעלות את נקודות החיים, כמות המטבעות והרמות שלו, או להרע לו ולפגוע בנקודות החיים שלו ואף לאפס אותן כך שהשחקן יפסיד ויצא מהמשחק.
במהלך הסיבוב, כל שחקן בתורו שולף קלף מהחפיסה ומשחק את הקלף ששלף. לדוגמה – קלף קרב. כאשר כל השחקנים שלפו קלף ושיחקו אותו נגמר הסיבוב, ולאחריו מתחיל סיבוב חדש עד סוף המשחק.
כאשר שחקן מגיע לרמה 10 הוא מנצח, כאשר נקודות החיים שלו הגיעו ל-0 הוא מפסיד.
גם כשהוא מנצח וגם כשהוא מפסיד הוא מסיים את המשחק. המשחק ממשיך בלעדיו עד אשר כל אחד מהשחקנים מגיע לרמה 10 או שנקודות החיים שלו הגיעו לאפס.

2.2 שחקן/ית

2.2.1 מאפייני השחקן

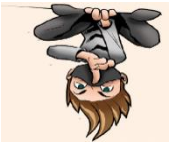
כמו בתרגיל בית 2, שחקן מאופיין ב:

- שם (name) - מורכב רק מאותיות (קטנות וגדולות) באנגלית וללא רווחים ואורכו לא יעלה על 15 אותיות.
- רמה (level) – מספר טבעי בטווח הערכים [1,10].
- כוח (force) – מספר שלם אי-שלילי ($\mathbb{N} \cup \{0\}$).
- נקודות חיים (HP – Health Points) – מספר שלם בטווח הערכים [0, maxHP].
- כמות מטבעות (coins) – מספר שלם אי-שלילי.

שחקן מתחיל ברמה (level) 1, עם 10 מטבעות (coins), כוח של 5, ונקודות חיים (HP) מלאות, כלומר שוות לנקודות חיים המקסימליות (maxHP) שהם 100 לכל השחקנים.

2.2.2 סוגי דמויות

יש 3 סוגי דמויות במשחק. לכל סוג דמות התנהגות ייחודית. כל שחקן במשחק יכול להיות אחת מהדמויות הבאות:



- **Ninja (נינג'ה)**

  שחקן מסוג זה הוא קל רגליים וכייס מעולה. כאשר Ninja מקבל מטבעות במשחק, הוא מכייס כמות שווה למה שהוא הרוויח, ובפועל מרוויח כמות כפולה של מטבעות מכמות המטבעות שהוא היה אמור לקבל.

- **Healer (מרפא)**

  שחקן מסוג זה הוא בעל יכולת התרפאות קסומה. כאשר קלף מעלה ל-Healer נקודות חיים הוא מקבל פי 2 מנקודות החיים שכל שחקן רגיל היה מקבל.

- **Warrior (לוחם)**

  Warrior היא דמות חזקה. כוח ההתקפה שלה מחושבת כך: force*2+level.
  (במקום force+level לכל השחקנים האחרים כמו בתרגיל בית 2).

2.3 קלפים

במשחק יש חפיסת קלפים, בגודל של 5 לפחות. חפיסת הקלפים יכולה להחזיק קלפים מכל הסוגים המתוארים בסעיף זה. לאחר כל שליפת קלף ושיחוקו, הקלף חוזר לסוף חפיסת הקלפים.

2.3.1 מאפייני הקלף

כל קלף מאופיין בשם (Name) עם אותם כללים כמו שם שחקן (מורכב רק מאותיות באנגלית, ללא רווחים, ואורכו לא יעלה על 15 אותיות) ובפעולה ייחודית לקלף.

2.3.2 סוגי קלפים

2.3.2.1 Battle Cards

קלפי קרב יכולים להוביל לאחת מ-2 תוצאות:
ניצחון – השחקן מנצח אם כוח ההתקפה שלו (force+level) גדול או שווה לכוחו (force) של קלף הקרב.
במקרה זה השחקן יעלה רמה אחת ויקבל מספר מטבעות (loot) המוגדר לקלף.
הפסד - במקרה זה נקודות החיים (HP – Health points) ירדו במספר נקודות שהקלף מגדיר (עד למינימום של 0).
יש 3 סוגי קלפי קרב:

- **Gremlin** – לגרמלין יש כוח של 5, Loot של 2 מטבעות, והוא עושה נזק של 10 נקודות חיים בעת הפסד.
- **Witch** – למכשפה יש כוח התקפה של 11, Loot של 2 מטבעות, והיא עושה נזק של 10 נקודות חיים בעת הפסד. בנוסף בעת הפסד למכשפה מאבדים יחידת כוח אחת.
- **Dragon** – לדרקון יש כוח התקפה של 25, Loot של 1000 מטבעות, ובעת הפסד הוא שורף את היריב כליל, הנקודות החיים מאופסות והשחקן מפסיד ומוצא מהמשחק.

2.3.2.2 Merchant Cards

קלפי סוחר. קלף זה מאפשר לקנות תוספת לנקודות החיים, או תוספת כוח, תמורת מטבעות (במידה ויש מספיק כסף.

```
You've ran into a Merchant!
Browse his wares:
Health Potion (heals 1 health point) : 5 coins per potion.
Force boost (adds 1 force): 10 coins per boost.
Enter 1 for Health Potion, 2 for force boost or 0 to leave.
Itay has 10 coins
1
Itay has paid 5 coins for 1 health potion!
Safe travels!
```

2.3.2.3 Treasure Card

קלף אוצר – מציאת תיבת אוצר
שמכילה 10 מטבעות. השחקן מקבל 10 מטבעות.

2.3.2.4 Well Card

באר – השחקן נופל לתוך הבאר ומאבד 10 נקודות חיים **אלא אם סוג השחקן הוא Ninja**, אז הוא מצליח להתחמק ולקפוץ החוצה.

2.3.2.5 Barfight Card

קטטת בר - הקלף מוריד את החיים ב-8 נקודות **אלא אם סוג השחקן הוא Warrior**.

2.3.2.6 Mana Card

מאנה - אנרגיית קסם. **רק ה-Healer** יודע להשתמש בה כראוי ולכן הוא יכול להעלות את נקודות החיים של עצמו ב-10 נקודות. היות והשחקן הוא Healer בפועל נקודות החיים של השחקן עולות ב-20.

2.4 Leaderboard – דירוג שחקנים

דירוג השחקנים (במהלך או בסוף המשחק) מוגדר באופן כזה שהשחקן הראשון שהגיע לרמה 10 (אם יש כזה) יהיה במקום הראשון, השחקן השני שהגיע לרמה 10 במקום השני, וכך הלאה. השחקן הראשון שאיבד את כל נקודות החיים שלו (אם יש כזה) ימצא במקום האחרון בדירוג השחקנים, השחקן השני שאיבד את כל נקודות החיים, ימצא במקום לפני האחרון וכך הלאה.
במידה ויש עדיין שחקנים שעוד לא הגיעו לרמה 10, וכמות נקודות החיים שלהם גדולה מאפס, נמקם אותם באמצע הדירוג, בין השחקנים ברמה 10, לשחקנים שהפסידו ויש להם אפס נקודות חיים. השחקנים באמצע יופיעו ב-leaderboard לפי סדר התורות שלהם, החל מהשחקן הבא בתור, וכן הלאה עד השחקן האחרון בסבב.

3 MTMCHKIN – FULL edition - INTERFACE

3.1 המחלקת Mtmchkin

בדומה לתרגיל בית 2, המחלקה Mtmchkin מחזיקה את תור השחקנים, חפיסת הקלפים ומנהלת את המשחק.
חתימת פונקציות הממשק שלה נתונות לכם בשלד של הקובץ Mtmchkin.h המסופק לכם. כמו כן מוגדרות לכם פונקציות עזר להדפסות במודול utilities.h.
שימו לב שאת כל ההדפסות של המשחק יש לבצע באמצעות הפונקציות שבמודול זה בלבד, על מנת לוודא התאמה לטסטים האוטומטיים. עם זאת ניתן (ומומלץ) להוסיף הדפסות משלכם במקרי טיפול בשגיאות במידה ולא ניתנה עבורן פונקצייה יעודית במודול זה.

3.1.1 בנאי

הבנאי מקבל שם של קובץ המייצג את חפיסת הקלפים.

```
Mtmchkin game("deck.txt");
```

3.1.1.1 אתחול חפיסת הקלפים

הבנאי מקבל שם קובץ המייצג את חפיסת קלפים. שורות הקובץ מגדירות את סוגי הקלפים שבחפיסה ואת הסדר שלהם, כאשר כל שורה בקובץ מכילה שם של קלף אחד, והיא מהפורמט הבא:

<cardName>

לדוגמה:

Mana
Gremlin
Witch
Mana
Barfight
Dragon
Treasure
Merchant
Gremlin

הבנאי מאתחל חפיסת קלפים לפי הרשום בקובץ. לדוגמה ברשימה מעל, הבנאי יתאחל חפיסה המכילה את הקלפים הבאים לפי הסדר: מנה, גרמלין, מכשפה, מנה, קטטת בר, דרקון, אוצר, סוחר וגרמלין.

3.1.1.2 אתחול אינטראקטיבי של תור השחקנים

השחקנים יאותחלו בצורה אינטראקטיבית דרך ערוץ הקלט הסטנדרטי. המשתמש תחילה יכניס את מספר השחקנים, ולאחר מכן את שמם ומקצועם (הדמות) של השחקנים כמתואר בדוגמה המוצגת למטה.
אם מכניסים גודל קבוצת שחקנים הגדול מ-6, או קטן מ-2, יש להציג למשתמשים את ההודעה המוצגת למטה ולבקש מהם להכניס גודל חדש. יש לטפל באופן דומה, בכל מקרה של קלט שגוי המתקבל מהמשתמש בצורה אינטראקטיבית. יש להדפיס הודעת שגיאה מתאימה, ולבקש מהמשתמש להקליט קלט מחדש. למשל אם הוקלדה מילה כאשר מצופה למספר, אם הוכנס מספר לא נכון של מילים בשורה וכן הלאה. במצב של שם קובץ לא תקין מסיבה כלשהי יש לזרוק חריגה מתאימה.
כאמור, בכל מצב שגיאה שעבורו לא ניתנה לכם פונקציית הדפסה ייעודית, יש להדפיס הודעת שגיאה (אינפורמטיבית) לבחירתכם.

בדוגמה להלן סדר התורות של השחקנים יהיה: דניאל, וסאלי.

```
Welcome to the world of MtmChkin!!!
Please enter team size: (2-6)
7
Invalid team size! Please enter a different size.
Please enter team size: (2-6)
2
Please insert the player name and class:
Daniel Ninka
You have entered an invalid class. Please try again.
Daniel Ninja
Please insert the player name and class:
Sally Witch
```

3.1.1.3 הדפסות

יש להיעזר בפונקציות הבאות בשביל ההדפסות בשלב אתחול המשחק:

- printStartGameMessage – מדפיסה את הודעת הפתיחה.
- printEnterTeamSizeMessage – מדפיסה את הבקשה מהמשתמש להכניס את מספר השחקנים.
- printInvalidTeamSize – הדפסת הודעה במצב בו המשתמש הכניס קלט לא תקין בתור מספר.
- printInsertPlayerMessage – מדפיסה את הבקשה להכנסת פרטי משתמש.
- printInvalidClass ו printInvalidName – מדפיסות הודעה אם הוקלד שם שחקן או שם דמות לא נכונים. (קודם נבדק השם, לאחר מכן סוג הדמות).

כל הפונקציות הללו מוגדרות ב-utilities.h.

3.1.2 playRound (סיבוב משחק)

בכל קריאה לפונקציית playRound, כל השחקנים שעדיין במשחק שולפים בזה אחר זה את הקלף הבא מחפיסת הקלפים לפי הסדר (לפי סדר התורות), משחקים את הקלף ומדפיסים את המידע התואם להשפעת הקלף על השחקן.
פעולת פונקציית הפעלת סיבוב משחק:

1. מדפיסים הודעה על תחילת סיבוב (printRoundStartMessage ב-utilities.h).
עבור כל שחקן בתור השחקנים שעדיין במשחק מבצעים את הפעולות הבאות:
1. מדפיסים הודעה על תחילת תור השחקן (printTurnStartMessage ב-utilities.h).
1. שולפים את הקלף הבא בחפיסת הקלפים.
2. משחקים את הקלף שנשלף.
3. מחזירים את הקלף שנשלף לסוף התור.
2. אם המשחק נגמר מדפיסים הודעה על כך (printGameEndMessage ב-utilities.h).

3.1.3 קבלת מספר הסיבוב הבא

getNumberOfRounds – תחזיר את מספר הסיבובים שהתקיימו עד כה (והסתיימו). אם המשחק רק החל ועוד לא התקיימו סיבובים – יש להחזיר 0.

3.1.4 בדיקת סיום המשחק

isGameOver – תחזיר false אם המשחק לא הסתיים, ו-true אם הסתיים. משחק מסתיים כאשר כל אחד מהשחקנים במשחק הגיע לרמה 10 או שנקודות החיים שלו הגיעו לאפס. הדפסת דירוג השחקנים (ה-leaderboard) תתבצע לערוץ הפלט הסטנדרטי בפורמט הבא:

```
The current ranking of the Team:
Ranking Player Name     Level   Force   HP      Coins   Job
1       Daniel          1       5       100     10      Warrior
2       Sally           1       5       100     10      Ninja
```

יש להיעזר בפונקציות printPlayerLeaderboard ו-printLeaderboardStartMessage הנמצאות ב-utilities.h לצורך ההדפסה. סדר ההדפסה מתבצע על פי המוסבר בסעיף 2.4.

3.1.5 דוגמת שימוש בממשק של Mtmchkin

בקוד הבא מאתחלים משחק עם חפיסת קלפים המיוצגת על ידי הקובץ deck.txt בה יש 5 דרקונים, ושחקנים המתקבלים על ידי המשתמש. מריצים סיבובים עד שהמשחק נגמר, או שהתקיימו כבר 100 סיבובים. לבסוף מדפיסים את דירוג השחקנים.
פלט הקוד לדוגמה נמצא לאחריו.
(שימו לב שההגבלה ל-100 סיבובים אינה דרישה מהמשחק אלא רק משמשת עבור דוגמת הקוד).

```
int main(){
   const int MAX_NUMBER_OF_ROUNDS = 100;
   Mtmchkin game("deck.txt");
   while(!game.isGameOver() && game.getNumberOfRounds() < MAX_NUMBER_OF_ROUNDS){
      game.playRound();
   }
   game.printLeaderBoard();
```

```
}
```

```
Welcome to the world of MtmChkin!!!
Please enter team size: (2-6)
2
Please insert the player name and class:
Daniel Ninja
Please insert the player name and class:
Sally Witch

-----------------------------------------
Start of round 1:

Start of Daniels's turn:
Player Daniel has been defeated by a Dragon.
Start of Sally's turn:
Player Sally has been defeated by a Dragon
The Game has ended!!!

The current ranking of the Team:
Ranking Player Name     Level   Force   HP      Coins   Job
1       Daniel          1       5       100     10      Warrior
2       Sally           1       5       100     10      Ninja
```

3.2 המחלקה Player

בדומה לתרגיל בית 2, המחלקה Player תייצג שחקן במשחק. ממשק מחלקת Player בתרגיל בית 4 נתון לבחירתכם. אתם רשאים להוסיף לו מתודות ואופרטורים. אנחנו ממליצים להתחיל מהממשק שניתן לכם בתרגיל בית 2 ולהוסיף/לשנות אותו.

3.3 המחלקות – Healer, Ninja, Warrior

המחלקות Healer, Ninja, Warrior מייצגות שחקנים מסוג מרפא, נינג'ה ולוחם בהתאמה. ממשק המחלקות הוא לבחירתכם ועל מחלקות אלה להתנהג בצורה הייחודית של דמויות אלה.
לדוגמה, עבור מרפא – כל פעם שהוא מקבל heal (מקלף Merchant או Mana), נקודות החיים שהוא מקבל מוכפלות ב-2.
התנהגות ייחודית נוספת שקיימת לכל מחלקה היא הדפסה שונה עבור אופרטור ההדפסה, לדוגמה בקוד הבא:

```
void printPlayerExample(const Player& somePlayer ){
  std::cout << somePlayer << std::endl;
}
```

אם השחקן הוא מסוג מרפא עם השם "someName" יודפס:

```
<name> <level> <Force> <Hp> <Coins> <Class>
someName      1       5     100   10    Healer
```

אם השחקן הוא מסוג נינג'ה יודפס:

someName 1 5 100 10 Ninja

אם השחקן הוא מסוג לוחם יודפס:

someName 1 5 100 10 Warrior

יש להשתמש בפונקציה printPlayerDetails בקובץ utilities.h.

3.4 המחלקה Card

בדומה לתרגיל בית 2, המחלקה מייצגת קלף במשחק. ממשק מחלקת Card בתרגיל בית 4 נתון לבחירתכם. אתם רשאים להוסיף לו מתודות ואופרטורים. אנחנו ממליצים להתחיל מהממשק שניתן לכם בתרגיל בית 2 ולהוסיף/לשנות אותו.

לצורך הדפסת פרטי הקלף יש להיעזר בפונקציות printCardDetails ו- printEndOfCardDetails הנמצאות בקובץ utilities.h. דוגמה להדפסה של קלף אוצר:

```
Card Details:
Name: Treasure
-------------------------------
```

קוד לדוגמה להדפסת פרטי קלפים:

```
void printCardExample(const Card& someCard){
   cout << someCard;
}
```

3.5 המחלקות – Witch, Gremlin, Dragon

המחלקות Witch, Gremlin, Dragon מייצגות קלפי קרב נגד המפלצות – מכשפה, גרמלין ודרקון בהתאמה. ממשק המחלקות נתון לבחירתכם ועל מחלקות אלה לייצג את המאפיינים הייחודים של כל מפלצת (לדוגמה - כוח), ולהתנהג בצורה הייחודית של קלפים אלו.

הדפסת פרטי קלף:
עבור קלפים אלו הדפסת פרטי הקלף תהיה בפורמט הבא:
עבור מכשפה:

```
Card Details:
Name: Witch
Force: 12
Damage upon loss: 10
Coins: 3
-------------------------------
```

עבור גרמלין:

```
Card Details:
Name: Gremlin
Force: 7
Damage upon loss: 10
Coins: 3
-------------------------------
```

עבור דרקון:

```
Card Details:
Name: Dragon
Force: 25
```

Damage upon loss: Infinite
Coins: 1000

יש להיעזר בפונקציה printMonsterDetails המופיעה ב-utilities.h.

הדפסה בעת ניצחון והפסד:
בקלפי קרב/מפלצות מתבצעת הדפסה בסוף פעולת הקלף של תוצאת הקרב.
ניצחון: אם השחקן ניצח את המפלצת – מודפסת ההדפסה הבאה:
Player <playerName> has defeated <monsterName> and rose 1 Level!

כאשר <playerName> זה שם השחקן, ו<monsterName> זה שם המפלצת.
יש להיעזר בפונקציה printWinBattle הנמצאת ב-utilities.h.

הפסד: אם השחקן הפסיד למפלצת – מודפסת ההדפסה הבאה:
Player <playerName> has been defeated by a <monsterName>.

יש להיעזר בפונקציה printLossBattle הנמצאת ב-utilities.h.

3.6 המחלקות – *Mana, Barfight, Well*

המחלקות Mana, Barfight, Well מייצגות קלפי מנה, קטטת בר ונפילה לבאר בהתאמה. אופן פעולתם מתואר למעלה. כל קלף עוזר/פוגע בדמות מסוימת, ומתעלם מדמויות אחרות.
לאחר סיום פעולת קלף זה יש להדפיס הודעת סיום בעזרת הפונקציות printWellMessage, printManaMessage ו-printBarfightMessage הנמצאות ב-utilities.h.

3.7 המחלקה – *Treasure*

המחלקה Treasure מייצת קלף אוצר. במפגש עם קלף זה השחקן מקבל 10 מטבעות. לאחר סיום פעולת הקלף יש להדפיס הודעה בעזרת הפונקציה printTreasureMessage.

3.8 המחלקה – *Merchant*

```
You've ran into a Merchant!
Browse his wares:
Health Potion (heals 1 health point) : 5 coins per potion.
Force boost (adds 1 force): 10 coins per boost.
Enter 1 for Health Potion, 2 for force boost or 0 to leave.
Jimmy has 10 coins
0
Jimmy has paid 0 coins
Safe travels!
```

המחלקה Merchant מייצגת קלף סוחר. מקלף סוחר ניתן לקנות שיפור אחד – נקודת חיים עבור 5 מטבעות או יחידת כוח בשביל 10 מטבעות. ניתן גם לוותר ולא לקנות דבר. השחקן בוחר את הפעולה בצורה אינטראקטיבית כאשר:
הכנסת קלט "0" – לא לקנות כלום, "1" – לקנות שיפור חיים, "2" – לקנות שיפור כוח.
ההדפסות יעשו בעזרת הפונקציות הבאו המוגדרות ב-utilities.h:
printMerchantInitialMessageForInteractiveEncounter – ההדפסה ההתחלתית.
printInvalidInput – אם מוכנס קלט לא תקין. במקרה הזה יש לבקש מהמשתמש להכניס קלט חדש.
printMerchantInsufficientCoins – הדפסת הודעה אם אין לשחקן מספיק כסף לקנות את השיפור שבחר. במקרה זה מסיימים את פעולת הקלף והתור עובר לשחקן הבא.
printMerchantSummary – הדפסת סיכום הקנייה בסוף פעולת הקלף. שימו לב להנחיות בסעיף 2.3.2.2.

הערה – מומלץ להשתמש בפונקציה getline כדי לקרוא את השורה, ו-stoi. שימו לב לתפוס חריגות במקרה הצורך.
ניתן להניח שהקלט מהמשתמש אינו מכיל רווחים מיותרים (לדוגמה " 0" לא תקין, "0" תקין).

3.9 חריגות

עליכם לממש חריגות **ככל הנדרש** על מנת לממש את התרגיל, ולזרוק אותן לפי הצורך.
את החריגות של התרגיל יש להגדיר בקובץ Exception.h.
עליכם להגדיר ולהשתמש ב-3 החריגות החדשות.
DeckFileNotFound
DeckFileFormatError
DeckFileInvalidSize
לכל מחלקה יש לוודא שהפונקציה exception::what() מחזירה מחרוזת המתארת את השגיאה, בהתאם לפירוט הבא:

Exception	Error Message
DeckFileNotFound	Deck File Error: File not found
DeckFileFormatError	Deck File Error: File format error in line <lineNumberInDeckfile>
DeckFileInvalidSize	Deck File Error: Deck size is invalid

עבור DeckFileFormatError יש להדפיס את מספר השורה בו הייתה שגיאה, לקרוא את סוג הקלף.
כמו כן, זכרו שחפיסת קלפים צריכה לכלול לפחות 5 קלפים.

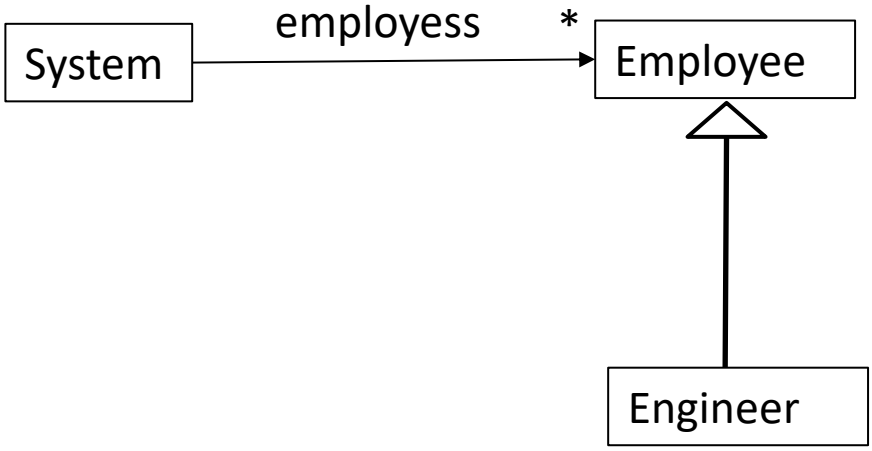
דגשים ודרישות מימוש

- קראו את התיעוד בקובץ Mtmchkin.h!
- הממשק של Mtmchkin מוגדר בקובץ Mtmchkin.h ואין להוסיף לממשק זה.
- אין לאפשר העתקה והשמה של מחלקה מסוג Mtmchkin.
- הממשק של כל מחלקה פרט ל-Mtmchkin נתון לבחירתכם פרט לשימוש באופרטור ההדפסה.
- מסופק לכם קובץ test.cpp המכיל טסטים למשחק, ותשתית לטסטים עתידיים. מומלץ להשתמש בתשתית זו לכתיבת טסטים נוספים.
- פונקציית ה-main שלכם תמצא בקובץ main.cpp. פונקציה זו תאתחל ותריץ את המחלקה Mtmchkin ותטפל בחריגות.
- לכל מחלקה יש לממש קובץ h ו-cpp משלה פרט לחריגות שצריכות להיות ממומשות כולן ב-Exception.h .
- המימוש חייב לציית לכללי כתיבת הקוד המופיעים תחת קובץ הקונבנציות. אי עמידה בכללים אלו תגרור הורדת נקודות.
- על המימוש שלכם להתבצע ללא שגיאות זיכרון (גישות לא חוקיות וכדומה) וללא דליפות זיכרון.
- המערכת צריכה לעבוד על שרת ALUF.
- ניתן להשתמש בקוד מתרגיל בית 2 ותרגיל בית 3, אך לא חובה. כתחליף לאוסף queue שמימשתם בתרגיל בית 3, ניתן להשתמש באוספים של STL שנלמדו בתרגולים ובהרצאות.
- בכל המימוש שלכם צריכות להתבצע הדפסות רק לערוץ הפלט הסטנדרטי.

<u>הערה</u>
שימו לב שבשום מצב אין לאפשר לתוכנית להכנס למצב לא תפקודי – למשל "להיתקע", או לקרוס בשל חריגה לא מטופלת. בכל מצב של שגיאה שלא תוארה בקובץ יש לטפל באופן המתאים ביותר, לרוב על ידי בקשה חוזרת מהמשתמש (אם הבעיה היא בגלל הקלדת קלט לא תקין), או על ידי יציאה מסודרת מהתוכנית (return מ-main) במקרה של תקלה שאינה מאפשרת את המשך ריצת התוכנית (למשל, בעיה בקובץ החפיסה).

4 חלק יבש - תכן

כפי שנלמד בהרצאות, בתהליך פיתוח מסודר יש להתחיל קודם כל בתכן (design). כתבו class diagram ב-UML של התכן שלכם הכולל את **כל** המחלקות העיקריות בהן תשתמשו. מחלקות המאפיינות collections אינן צריכות להיכלל בדיאגרמה. יש לכתוב את שמות המחלקות בלבד ללא השדות והמתודות של המחלקה כפי שמתואר בדוגמה הבאה.



5 הידור קישור ובדיקה

התרגיל ייבדק על שרת ALUF ועליו לעבור הידור בעזרת הפקודה הבאה:

```
g++ --std=c++11 -o mtmchkin -Wall -pedantic-errors -Werror -DNDEBUG *.cpp Cards/*.cpp
Players/*.cpp
```

לנוחיותכם, מסופקת לכם תוכנית "בדיקה עצמית" בשם finalCheck. התוכנית בודקת שקובץ ההגשה <ID1>_<ID2>.zip בנוי על פי מבנה הספריות ושמות הקבצים המתוארים להלן. כמו כן התוכנית מריצה את הטסטים שסופקו, כפי שירוצו על ידי הבודק האוטומטי.
הפעלת התוכנית ע"י הפקודה:

```
~mtm/public/odyssey/ex4/finalcheck <ID1>_<ID2>.zip
```

הקפידו להריץ את הבדיקה על קובץ ההגשה. אם אתה משנים אותו, הקפידו להריץ את הבדיקה שוב.

6 ולגרינד ודליפות זיכרון

המערכת חייבת לשחרר את כל הזיכרון שעמד לרשותה בעת ריצתה. על כן עליכם להשתמש ב valgrind שמתחקה אחר ריצת התכנית שלכם, ובודק האם ישנם משאבים שלא שוחררו. הדרך לבדוק האם יש לכם דליפות בתכנית היא באמצעות שתי הפעולות הבאות (שימו לב שחייב להיות main, כי מדובר בהרצה ספציפית):

1. קימפול של השורה לעיל עם הדגל g-.
2. הרצת השורה הבאה:

   valgrind --leak-check=full ./mtmchkin

   כאשר mtmchkin הוא שם קובץ ההרצה.

הפלט ש-valgrind מפיק אמור לתת לכם, במידה ויש לכם דליפות, את שרשרת הקריאות שהתבצעו וגרמו לדליפה. אתם אמורים באמצעות דיבוג להבין היכן היה צריך לשחרר את אותו משאב שהוקצה ולתקן את התכנית. בנוסף, valgrind מראה דברים נוספים כמו פנייה לא חוקית לזיכרון (שלא גררה segmentation fault) – גם שגיאות אלו עליכם להבין מהיכן הן מגיעות ולתקן.

7 בדיקת התרגיל

הבדיקה היבשה תכלול מעבר על הקוד והתכן (UML class diagram). למשל תכן לא נכון, שכפולי קוד, קוד מבולגן, קוד לא ברור, שימוש בטכניקות תכנות "רעות".
הבדיקה הרטובה תכלול את הידור התכנית המוגשת והרצתה במגוון בדיקות אוטומטיות. על מנת להצליח בבדיקה שכזו, על התוכנית לעבור הידור, לסיים את ריצתה ללא קריסה, ולתת את התוצאות הצפויות וללא דליפות זיכרון.

8 הגשה

את ההגשה יש לבצע דרך אתר המודל. הקפידו על הדברים הבאים:

- יש להגיש את קבצי הקוד מכווצים לקובץ zip (לא פורמט אחר).
- אין להגיש קבצים נוספים מלבד קובץ בשם **dry.pdf** המכיל את הפתרון לחלק היבש. וקבצי **h** וקבצי **cpp** אשר כתבתם או השלמתם בעצמכם המהווים פתרון לחלק הרטוב.
- הקבצים אשר מסופקים לכם יצורפו על ידנו במהלך הבדיקה, וניתן להניח כי הם יימצאו בתיקייה הראשית.
- ניתן להגיש את התרגיל מספר פעמים, רק ההגשה האחרונה נחשבת.
- על מנת לבטח את עצמכם נגד תקלות בהגשה האוטומטית, שימרו עותק של התרגיל על חשבון ה-ALUF/Github/Google Drive שלכם לפני ההגשה האלקטרונית ואל תשנו אותו לאחריה (שינוי הקובץ יגרור שינוי חתימת העדכון האחרון).
  - כל אמצעי אחר לא יחשב הוכחה לקיום הקוד לפני ההגשה.

בהצלחה !