# **תרגיל 1 – משחק צוללות לקונסול**

## מגישים:

1. אור מנצור – 204311997 (or.mantzur@gmail.com)
2. דודי יחקאל – 200441749 (dudi.2705@gmail.com)

## הערות לבדיקת התרגיל:

קבצי מקור:  
כל קבצי קוד המקור ממוקמים תחת תיקיית Source Code.

מיקום קובץ Bat:  
קובץ הBat ממוקם בתיקיה יחד עם הקובץ jar של הלוגיקה וזאת בכדי שקבצי שמירת המשחק ימוקמו   
באותה התיקייה.

אופן מימוש התפריט:  
בכל שלב יוצגו בתפריט רק הפריטים הרלוונטיים למצב המשחק באותו רגע.  
לדוגמא: כל עוד לא נטען משחק לא ניתן יהיה לבחור "start game" ואם כבר התחיל משחק לא ניתן יהיה לטעון משחק נוסף עד שהשחקן לא יוצא מהמשחק הנוכחי.

הצגת סטטיסטיקות:

1. ספירת כמה תורות בוצעו עד כה:

כל התקפה חוקית (תקיפת תא על הלוח שלא נתקף בעבר) נספרת כתור, אם לדוגמא שחקן פוגע בספינה ולאחר מכן מפספס (שני מהלכים) יספרו לשחקן זה שני תורות.

1. משך זמן מתחילת המשחק:

משך הזמן נספר החל מרגע טעינת המשחק, אם המשחק נטען מקובץ xml אז הזמן יתחיל להיספר החל מלחיצה על פקודה 2. אם המשחק נטען מקובץ שמור אז הזמן ייספר החלר מרגע טעינת המשחק.

בדיקות המבוצעות על קובץ הxml :

1. בהרצת פקודת טעינת המשחק (פקודה 1):
   1. טעינת קובץ
      1. הקובץ שניתן קיים
      2. הקובץ שניתן אינו תיקייה
      3. הקובץ הוא מסוג xml
   2. בדיקת תקלות JAXB
   3. בדיקה שגודל הלוח הוא בין 5-20
   4. בדיקת **סוגי** הספינות שיש למקם על הלוח
      1. בדיקה שרשימת סוגי הספינות אינה ריקה
      2. אורך הספינה הוא מספר אי שלילי
      3. הscore של הספינה הוא מספר אי שלילי
      4. אין שתי ספינות עם אותו הID
2. בדיקות לוגיות המבוצעות על קובץ הxml בהרצת פקודת תחילת משחק (פקודה 2):
   1. בדיקה שאין שחקן שהוא null (ביצירת משחק חדש שולחים את שני השחקנים בctor)
   2. בדיקה שאכן קיים קובץ שנטען מxml (חוסם את האפשרות להתחיל מחדש משחק שנטען משמירה)
   3. בדיקה לוגית שאין התנגשויות בהוספת הספינות ללוח (אין התנגשות של ספינות, אין ספינות סמוכות, אין ספינות שממוקמות מחוץ ללוח)
   4. כמות הספינות המוגדרת לכל לוח תואמת את כמות הספינות שמוקמו בכל לוח.

## הסבר כללי על אופן מימוש התרגיל:

### ישנם שני מודולים בתרגיל זה

1. BattleshipsGameLogic –מכיל את לוגיקת המשחק. ממומש באופן כזה שניתן יהיה לעשות בו שימוש מכל UI שנבחר.
2. BattleshipsConsoleUI – מכיל את ה-UI של התרגיל עבור משחק על גבי הConsole. מודול זה מכיל את נקודת הכניסה (Main) של המשחק והוא עושה שימוש במודול של הלוגיקה המתואר לעיל.

### תיאור המחלקות העיקריות בכל מודול:

**BattleshipsGameLogic**

1. GamesManager

מחלקה זו מנהלת את המשחק (בתרגילים הבאים תנהל גם יותר ממשחק אחד).   
מחלקה זו מממשת את הממשק IGamesLogic שתתואר בהמשך.

* 1. Data Members:
     1. כל השחקנים הרשומים למשחק (בתרגיל זה יש רק שניים).
     2. כל המשחקים הקיימים במשחק (בתרגיל זה יש רק אחד).
  2. Methods:
     1. המתודות שיתוארו בממשק IGamesLogic.

1. IGamesLogic:

ממשק אשר מחצין את המתודות הרלוונטיות בלוגיה לUI.

הסיבה העיקרית לקיום מחלקה זו היא בכדי לאפשר עבודה זהה עם הלוגיקה בין כל UI עתידי.

בנוסף, מחלקה זו מפשטת את השימוש במודול זה ע"י חשיפת המתודות הבאות:

* 1. טעינת קובץ XML עבור הגדרות המשחק
  2. התחלת משחק
  3. תקיפת משבצת בלוח השחקן
  4. קבלת זמן המשחק עד כה
  5. הצבת מוקש
  6. שמירת המשחק הנוכחי
  7. טעינת משחק שנשמר בעבר
  8. יצאה מסודרת מהמשחק

1. Game:
   1. Data Members:
      1. הגדרות המשחק
      2. שני השחקנים
   2. Methods:
      1. initGame – אתחול המשחק הכולל בתוכו בדיקה לוגית של כל הגדרות המשחק.

לדוגמא, בדיקה שאין חפיפה בין הספינות, שכל הספינות שמוגדרות בקובץ אכן נמצאות על כל אחד מלוחות השחקנים וכו'.

* + 1. resetGame – אתחול מחדש עבור משחק קיים, איפוס לוחות השחקנים למצב הראשוני. במידה והשחקן רוצה לסיים את המשחק הנוכחי, אך רוצה להשתמש באותו משחק ששיחק.
    2. saveToFile – מאפשר לשחק לשמור את המשחק הנוכחי בכל עת מרגע שהתחיל את המשחק (בונוס – יפורט בנפרד).
    3. loadFromFile – מאפשר לשחקן לטעון משחק ששמר בעבר. אפשרות זו ניתנת כל עוד אין משחק פעיל (בונוס – יפורט בנפרד).

1. GameSettings:
   1. Data Members:
      1. גודל הלוח
      2. מספר מוקשים לשחקן
      3. סוג המשחק
      4. סוג הספינות שיש לטעון ומספרם
   2. Methods:
      1. loadGameFile – טעינת קובץ XML עבור הגדרות המשחק מהמשתמש ובדיקה ראשונית של תקינות ההגדרות. לדוגמא, בדיקה שגודל הלוח הוא מספר תקין בטווח, בדיקה שהלוח לא ריק מספינות, בדיקה שאין שני סוגי ספינות שונות בעלות או מזהה יחודי וכו'.
2. Player:
   1. Data Members:
      1. ID
      2. שם
      3. לוח המשחק
      4. לוח המשחק של היריב
      5. נקודות במשחק
      6. זמן משחק
      7. מספר התורות ששוחקו
   2. Methods:
      1. attack – תקיפה של משבצת בלוח. מתודה זו מחזירה את תוצאת התקיפה. לדוגמא: השחקן פגע בצוללת, מוקש, משבצת שכבר תקף בעבר וכו'.
      2. plantMine – הטמנת מוקש בלוח ספינות השחקן (בונוס – יפורט בהמשך).
3. ComputerPlayer (בונוס):

מחלקה זאת יורשת מהמחלקה Player שתיארנו לעיל.

* 1. Data Members (הסבר מעמיק יותר ניתן בחלק של ההסברים על הבונוסים):
     1. רשימת כל המהלכים האופציונליים שיבצע המחשב
     2. רשימת כל התאים ה"חשודים" כספינות
     3. לוח המשקף את מצב לוח הספינות של השחקן (אילו תאים עדיין רלוונטיים לתקיפה)
     4. גודל הספינה המינימלי
  2. Methods:
     1. getNextPositionToAttack – מחזיר את מיקום המשבצת של המהלך המושכל הבא הניתן ע"י אלגוריתם שעליו נפרט בחלק של ההסברים על הבונוסים.
     2. attack – תקיפה של משבצת בלוח. מתודה זו מחזירה את התוצאה עקב התקיפה. לדוגמא, השחקן פגע בצוללת, מוקש, משבצת שכבר תקף בעבר וכו'.

1. Board:
   1. Data Members:
      1. לוח המשחק
      2. רשימת ספינות
   2. Methods:
      1. addShipToBoard – הוספת ספינות ללוח המשחק בעת אתחול הלוח.
      2. allShipsWereSunk – בכל שלב במשחק ניתן לבדוק האם כל הספינות שעל הלוח הוטבעו.
2. BoardCell:

תא בלוח המשחק. Board הוא למעשה מערך דו מימדי של BoardCells

* 1. Data Members:
     1. מיקום התא
     2. האובייקט שנמצא בתא (ספינה, מוקש, מים)

1. GameObject:

מחלקה אבסטרקטית אשר ממנה יורשים כל האובייקטים על הלוח (ספינה, מוקש, מים).

* 1. Data Members:
     1. מיקום על הלוח
  2. Methods:
     1. getAttackResult – מתודה אבסטרקטית המבצעת את הפעולות הנדרשות בפגיעה באובייקט (בדיקה האם הספינה הוטבעה לדוגמא) ומחזירה את תוצאת הפגיעה

**BattleshipsConsoleUI**

1. ConsoleUIManager:

מחלקה אשר מנהלת את ה UI של המשחק באמצעות ה Console.

* 1. DataMembers:
     1. מנהל המשחק של הלוגיקה
     2. המשחק הפעיל כעת
     3. תפריט
  2. Methods:
     1. run – הרצת המשחק אשר מנוהל באופן הבא:  
        כל עוד לא נבחרה האפשרות לצאת מהמשחק:
        1. הצגת התפריט
        2. בחירת פעולה ע"י המשתמש
        3. ביצוע הפעולה שנבחרה
        4. המתנה להקשת אנטר ע"י השחקן

1. Menu:
   1. DataMembers:
      1. מצב המשחק הפעיל
   2. Methods:
      1. display – הצגת התפריט עם האפשרויות הרלוונטיות למצב המשחק.
2. BoardPrinter:
   1. Methods:
      1. printBoards – הדפסת הלוחות של השחקן הרלוונטי

## **פירוט בונוסים שמומשו:**

1. מימוש המוקש – באפשרות השחקן בכל תור לבחור את אפשרות 7 על מנת להניח מוקש על הלוח כפי שמוגדר בתרגיל.

כאשר שחקן תוקף מוקש הוא מקבל נקודה עבור הפגיעה. אם המוקש גורם לפגיעה בספינה של השחקן התוקף גם השחקן השני יקבל נקודה.

במידה והמוקש גורם לפיצוץ של מוקש אחר שני השחקנים יזוכו בנקודה אך במקרה זה בשני הלוחות יסומן "החטאה"

1. הרצת משחק לאחר סיום משחק עקב פרישה או ניצחון.

במידה ושחקן מנצח או פורט מהמשחק יוצג לו שוב התפריט עם האפשרויות הבאות:

* 1. טעינת קובץ משחק חדש (1) – המשתמש יתבקש להזין שוב Path עבור קובץ xml
  2. התחלת משחק (2) – אפשרות זו תוצג רק במידה והמשחק האחרון נטען מקובץ xml ולא ממשחק שמור. בבחירת אפשרות זו יתחיל המשחק מההתחלה.
  3. טעינת משחק שמור (9) – יוצגו למשתמש כל המשחקים השמורים אותם הוא יכול לטעון
  4. הגדרת משחק נגד מחשב (10) – יוכל המשתמש לבחור האם המשחק הבא ישוחק נגד המחשב או נגד שחקן "רגיל"
  5. יציאה (0) – יציאה מוחלטת מהמשחק. אפשרות זו מוצגת בכל שלב במשחק.  
     \*\*אמנם רשום בתרגיל שאת אפשרות זו יש לממש כפקודה "(נגיד 8...)" אך החלטנו לממש אותה כפקודה 0 כך שהיא תהיה האחרונה, ולמקם בפקודות 8-10 את הבונוסים הנוספים.

1. טעינה ושמירה של המשחק
   1. שמירה לקובץ (8):  
      המשחק מאפשר לשמור את המשחק הנוכחי בכל נקודה שהמשחק רץ.

השחקן מתבקש לתת שם לקובץ בו ישמר המשחק. קובץ זה יישמר עם הסיומת .dat  
הקובץ ישמר בתיקייה בשם "Saved Games" , שתיווצר באופן אוטומטי במיקום רלטיבי למקום ממנו מורץ המשחק (אם הפרויקט יורץ מהBat שנמצא בתיקיית BattleShip התיקייה תיווצר באותו המקום)

* 1. טעינה מקובץ (9):

אופציה זו ניתנת כל עוד אין משחק פעיל.

בעת אפשרות זו, המערכת תציג למשתמש את כל המשחקים שנשמרו בעבר.

מהם יוכל השחקן לבחור.

במידה ואין משחקים שמורים, תוצג הודעה מתאימה למשתמש.

1. שחקן ממוחשב

לאחר טעינת קובץ משחק ולפני תחילת משחק (בין פקודות 1 ל2) באפשרות השחקן לבחור אם ברצונו לשחק נגד שחקן ממוחשב וכן לבחור האם השחקן הממוחשב יהיה שחקן 1 או 2. (\*בכל מקרה השחקן הראשון שישחק הוא השחקן האנושי ולא הממוחשב)

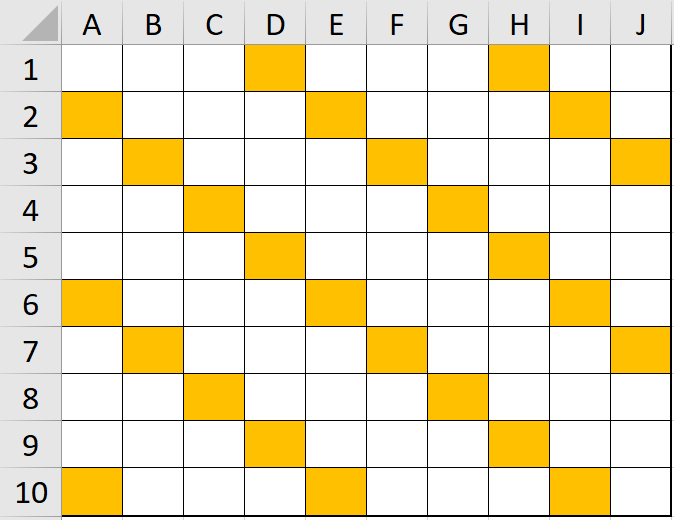
אופן מימוש השחקן הממוחשב:

השחק הממוחשב הוא למעשה נורש של שחקן רגיל המממש מתודה שמחזירה את התא הבא שברצונו לתקוף וכן מממשת מחדש את המתודה attack.

המתודה תחילה קוראת לפונקציה attack של השחקן הרגיל ולאחר מכן מבצעת עדכונים ברשימת המהלכים האפשריים של השחקן הממוחשב כמתואר מטה.

האלגוריתם לפיו מחליט השחקן הממוחשב איזה תא לתקוף:

ראשית מייצר השחקן הממוחשב רשימה של כל המהלכים שאותם עליו לבצע כך שבוודאות יפגע בכל הספינות בלוח במינימום תורות. רשימה זו היא אלכסונים על הלוח המתחילים מהאלכסון השמאלי תחתון כאשר המרווח בין כל אלכסון הוא אורך הספינה המינימלית על הלוח פחות 1.

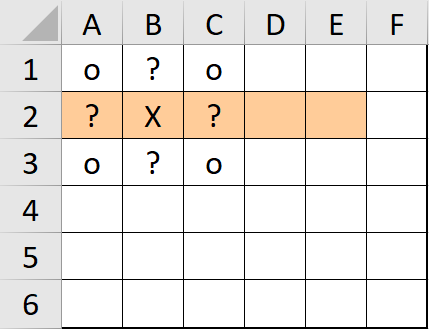
לדוגמא אם הלוח בגודל 10 וגודל הספינה המינימלי הוא 4, כל התאים אותם ינסה המחשב לתקוף הם התאים הצבועים בכתום באיור הבא:

נשים לב שלא ניתן למקם על הלוח הנ"ל ספינה באורך גדול או שווה ל4 לכן בוודאות נפגע בכל הספינות שעל הלוח.

בנוסף ללוח הצוללות ולוח המעקב של המחשב, מחזיק המחשב גם לוח של תאים "רלוונטיים" לתקיפה (מערך דו-מימדי זהה לגודל הלוח של ערכים בוליאניים, בו true מייצג תא שעליו יש לנו את המידע הנדרש ואין צורך לנסות לתקוף אותו)

לאחר תקיפה של המחשב, במידה והייתה פגיעה בצוללת יסמן המחשב את כל התאים שנמצאים באלכסון לפגיעה כתאים שאינם רלוונטיים לבדיקה, גם אם לפני כן אותם תאים היו "חשודים" (לא ייתכן שתהיה בהם צוללת). לאחר מכן יוסיף לרשימת התאים החשודים את כל התאים שנמצאים מעל או מצידי הפגיעה, וזאת בתנאי שהם לא מסומנים כ"לא רלוונטיים" (מקרה כזה יכול לקרות כאשר זוהי הפגיעה השניה בספינה שכבר נפגעה).

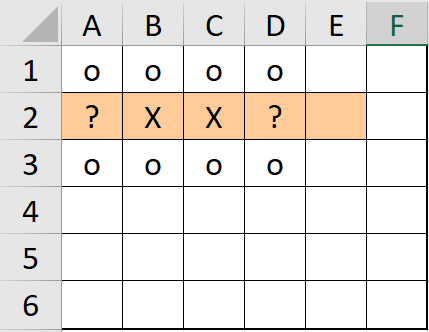
דוגמא לצורך המחשת האלגוריתם

אם המחשב פגע בספינה (מסומנת בכתום) בתא B2 ,

התאים המסומנים o יוכנסו לרשימת התאים הלא רלוונטיים

והתאים המסומנים ? יוכנסו לרשימת התאים החשודים

בתור הבא של השחקן הממוחשב תינתן עדיפות לרשימת התאים ה"חשודים" והמחשב ינסה לתקוף את התא הראשון ברשימה זו,

אם המחשב יתקוף את תא C2 לדוגמא מצב הרשימות יעודכן לפי

הסימונים הבאים:

נשים לב שהתאים B3 וB1 סומנו כלא רלוונטיים למרות שלא נתקפו,

התא D2 נוסף לרשימת החשודים ואילו B2 לא נוסף כיוון שמצבו כבר ידוע בלוח הבוליאני

לאחר שתקיפה של המחשב תחזיר "ספינה טבעה" יהפוך המחשב את כל התאים החשודים ללא רלוונטיים וימשיך לתקוף לפי הרשימה הראשית את התאים שמצבם אינם ידוע. (כלומר, הערך שלהם בטבלה הבוליאנית הוא false)