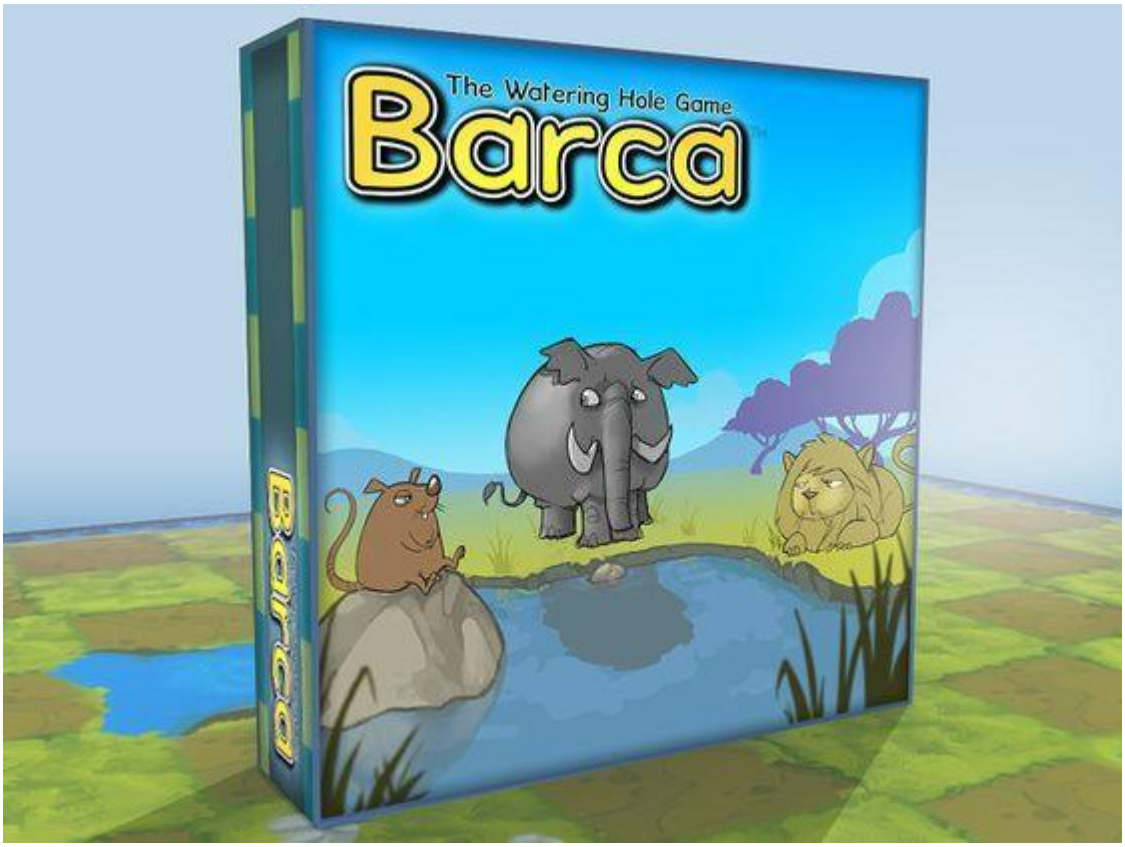
משחק – BARCA



שם המכללה: בסמח.

שם מגמת הלימוד: הנדסת תוכנה.

מסלול הכשרה: טכנאים מוסמכים.

שם המגיש: יהונתן טל

שם המנחה האישי: ניב שיראזי

תאריך הגשת ספר הפרויקט: 17/01/2024

תאריך :_____
_____

לכבוד
יחידת הפרויקטים
מה״ט

<u>הצעה לפרויקט גמר</u>

* יש להדפיס את כל הנתונים הנדרשים

1. <u>פרטי הסטודנטים</u>

שם הסטודנט	ת.ז .9 ספרות	כתובת	טלפון נייד	תאריך סיום הלימודים
יהונתן טל	324825470	הרברט סימון 10 ראשון לציון	0586304442	- קד״צ תוכניתן (נוב 20)

שם
המכללה __בסמ״ח_____ סמל המכללה :__71605__

מסלול ההכשרה : טכנאים מוסמכים.

מגמת לימוד : __הנדסת תוכנה__ מקום ביצוע הפרויקט : _בסמ״ח_

2. <u>פרטי המנחה האישי</u>

שם המנחה *	כתובת	טלפון נייד	תואר	מקום עבודה/תפקיד
ניב שיראזי	יוני נתניהו 8 פתח תקווה	0509595094	הנדסת תכנה	תכניתן בצוות פיתוח להב בשחר

* עבור מנחה אישי חדש יש לצרף קורות חיים, ניסיון מקצועי ותעודות השכלה לאישור מה״ט.

<u>יהונתן טל</u> <u>ניב שיראזי</u> _____
חתימת הסטודנט חתימת המנחה האישי חתימת הגורם המקצועי מטעם מה״ט

1. שם הפרויקט –**BARCA**
2. רקע

2.1. תיאור ורקע כללי

" בַּרְכָּה " (Barca) הוא משחק לוח שהומצא על ידי אנדרו קאלדוואל בשנת 2007. המשחק הינו משחק אסטרטגיה לשני שחקנים המשלב את היחסים הקיימים באבן, נייר ומספריים יחד עם תנועת כלי שחמט. הפרויקט הינו מימוש של המשחק במחשב. תהיינה האפשרות לשחק שחקן מול שחקן ואופציה נוספת של שחקן מול המחשב. המשחק מתרחש על גבי לוח משבצות x1010. לכל שחקן 6 כלי משחק: 2 עכברים, 2 אריות, 2 פילים. קרוב למרכז הלוח יש 4 משבצות מסומנות.

2.2. מטרת המערכת

בכדי לנצח, על השחקן להניח לפחות 3 חיות על המשבצות המסומנות (חיה 1 על כל משבצת).

3. מה הפרויקט אמור לחדש או לשפר

הפרויקט שכתבתי הוא לא היה לצורך חידוש/שיפור, אלא התנסות שלי במהלך קורס תכנות תחת מגמת פיתוח משחקי מחשב.

4. דרישות מערכת ופונקציונאליות

4.1. דרישות פונקציונאליות

רשימת דרישות המשתמש מהמערכת, מהן הפעולות בהן נדרשת המערכת לתמוך.

- המערכת תנהל את המשחק של השחקן ושל היריבים הממוחשבים.
- המערכת תטען את הלוח בתחילת המשחק.
- המערכת תבצע בדיקות כגון: האם כלי מסוים מפחיד כלי אחר, האם הכלים המפוחדים הם מאותו צד או לא, האם כלי מסוגל לבצע תזוזה מסוימת והאם צריך לחייב את השחקן לנקוט קודם פעולה אחרת, וכו'.
- המערכת תכריז על הודעות ניצחון במידה והשחקן ניצח את יריבו. המערכת תודיע על הפסד במידה והשחקן הפסיד.
- המערכת לא תאפשר לחדור או לעבור בין קירות הלוח.
- הנצחון יקבע לפי כמות הכלים הנמצאים על גבי 4 המשבצות המסומנות המייצגות את בורות המים.
- מבנה הנתונים עליו ייבדקו תנאי המשחק הוא מטריצה. בנוסף יהיה גרף בשביל אלגוריתם האויב.

5. בעיות צפויות במהלך הפיתוח ופתרונות:

5.1. אני תכניתן בצה"ל שעובד על פרויקטים טכנולוגים גדולים וחשובים. בנוסף, אני סטודנט לתואר ראשון במדעי המחשב. מפאת עומס בתחומים אלו, אני צריך לארגן את הזמן שלי בצורה הכי יעילה ואפקטיבית שיש.

5.2. פתרונות אפשריים:

5.2.1. אתכנן כמה שיותר מראש איך לחלק את העבודה לאורך הזמן, אקבע דד-ליינים וזמני עבודה קבועים, על מנת לפזר את זמן העבודה בצורה נוחה שתאפשר לי לשלב אותה בשגרת החיים העמוסה שלי.

5.3. שפות הפיתוח:

Java, awt– משום שאלו השפות שהתנסתי איתן כשרק התחלתי לכתוב את הפרויקט במהלך קורס תכנות. למדתי java במסגרת הקורס, ו-awt היא הספריית client הכי פשוטה ונוחה שנחשפתי אליה בעזרת הכוונה ממפקדי בקורס.

5.4. סביבת השרת (מקומי, וירטואלי, ענן, שירות אירוח)

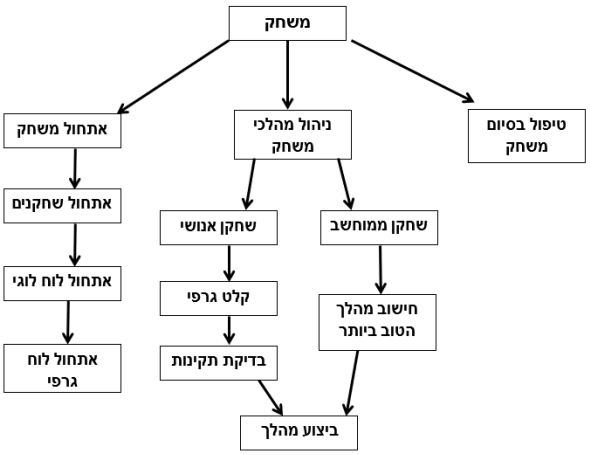
שרת מקומי

5.5. ממשק המשתמש/ לקוח- GUI

awt

6. תרשימי מערכת מרכזיים

6.1. Sequence diagram-



7. תיאור המרכיב האלגוריתמי-חישובי

אלגוריתם MiniMax- משחקים רבים בנויים על העיקרון של סט חוקים קבוע ומוגדר היטב ומשחק לסירוגין. לדוגמה: שחמט ודמקה. בתורת המשחקים, עץ מינימקס הוא (עץ סוג של מבנה נתונים) הפורס את האפשרויות למשחק של שחקן א', את התגובות של שחקן ב' לכל פעולה של שחקן א', את תגובותיו של א' לתגובותיו של ב' וכך הלאה. מעשית מוגבל עומקו של העץ על ידי הזמן וזיכרון המחשב העומדים לרשותנו (לדוגמה, מחשבי שחמט בונים עצים של כ-9 מהלכים קדימה). העלים בעץ שנוצר הם מצבים סטטיים שנגיע אליהם לאחר רצף של מהלכים (הנתיבים בעץ הם למעשה תרחישים אפשריים). ניתן ציון לכל מצב סטטי שכזה (מצב סטטי בלוח שחמט לדוגמה), שישקף כמה המצב טוב מבחינתנו. מובן שאם נבחר במהלך (ענף) המוביל לעלה עם הציון הגבוה ביותר, לא מובטח לנו שאכן נגיע לאותו עלה. העלה הטוב ביותר הוא התסריט האופטימלי מבחינתנו, וניתן להניח שיריב לא יוביל אותנו בהמשך הנתיב אלא יסיט אותו לנתיב הטוב ביותר מבחינתו. משפט המינימקס מאפשר לנו לדעת מה המהלך הטוב ביותר שנוכל לעשות, על בסיס המידע הנמצא בעץ.

סריקת העץ מתבצעת בצורה רקורסיבית, והסיבוכיות של אלגוריתם זה היא O(b^d) כאשר d מייצג את עומק העץ ו- b מייצג את הגורם המסועף (branching factor).

8. משאבים הנדרשים לפרויקט:

8.1. מספר שעות המוקדש לפרויקט, חלוקת עבודה בין חברי הצוות

היקף העבודה הוא בסדר גודל של כ-180 שעות

8.2. תוכנות נדרשות

intellij

8.3. ידע חדש שנדרש ללמוד לצורך ביצוע הפרויקט

התעמקתי יותר בשפה java, ולמדתי awt שכן לא יצא לי לפתח עם ספרייה זאת קודם לכן.

8.4. ספרות ומקורות מידע

8.4.1. http://stackoverflow.com/

8.4.2. http://www.java-forums.org

8.4.3. http://www.java-gaming.org/

9. תוכנית עבודה ושלבים למימוש הפרויקט

- הכרת הספריית **awt** ובניית שלד בסיסי למשחק בעזרת ה-**class**-ים שהיא מייחצנת
- פישוט תהליך התכנון- חשיבה על מחלקות עיקריות שבהן נשתמש
- בניית מחלקות הבסיס ויצירת אב טיפוס ראשוני הכולל את הלוח, הדמויות והאובייקטים על הלוח
- הרכבת ותכנות הבוט של היריב
- הכנסת לוגיקה הכוללת בתוכה מתי מנצחים ומתי מפסידים

10. תכנון הבדיקות שיבוצעו

10.1. נא פרט בטבלה, בדיקות תהליכיות ברמת המשתמש בהן נדרשת המערכת לעמוד (full flow).

10.1.1. בדיקה של האם אפשר לבצע את התזוזה הזו (מבחינת פחד של חיות).

10.1.2. בדיקה של האם ניתן לזוז לשם (מבחינת גבולות הלוח).

10.1.3. בדיקה של האם כלי שלי מפוחד והאם אני צריך להזיז אותו.

10.1.4. בדיקה של פסילה או נצחון כאשר שחקן תפס 3 מבורות המים.

11. בקרת גרסאות (**version control**)

נשתמש ב-**git** על מנת לעקוב בצורה ראויה אחר הגרסאות של הפרויקט.

יהונתן טל	ניב שיראזי
חתימת הסטודנט	חתימת המנחה האישי

3. הערות ראש המגמה במכללה

_____

_____

_____

4. אישור ראש המגמה

שם: _____ חתימה: _____ תאריך: _____

5. הערות הגורם המקצועי מטעם מה״ט

_____

_____

_____

6. אישור הגורם המקצועי מטעם מה״ט

שם : _____ חתימה : _____ תאריך :

_____

לכבוד

יחידת הפרויקטים

מה״ט

הצהרת סטודנט

עודכן (10/17)

דרך מנחם בגין 86 תל אביב ת.ד. 36049 מיקוד 67138

טלפון 03-7347521 : פקס : 03-7347644

שם הסטודנט : _____יהונתן טל_____ ת.ז. _____324825470___

שם המכללה בה לומד הסטודנט :_____בסמח_____

אני הח״מ, מצהיר בזאת כי פרויקט הגמר וספר הפרויקט המצ״ב נעשו על ידי בלבד.

פרויקט הגמר נעשה על סמך הנושאים שלמדתי במכללה ובאופן עצמאי.

פרויקט הגמר וספר הפרויקט נעשו על בסיס הנחייתו של המנחה האישי.

מקורות המידע בהם השתמשתי לביצוע פרויקט הגמר מצוינים ברשימת המקורות המצוינים בספר הפרויקט.

אני מודע לאחריות שהנני מקבל על עצמי על ידי חתימתי על הצהרה זו שכל הנאמר בה אמת ורק אמת.

חתימת הסטודנט : _____יהונתן טל_____ תאריך : _____16/12/2023_____

אישור המנחה האישי

הריני מאשר שהפרויקט בוצע בהנחייתי, קראתי את ספר הפרויקט ומצאתי כי הוא מוכן לצורך הגשת הסטודנט להגנה על פרויקט גמר.

שם המנחה: _____ניב שראזי_____ חתימה: _____ניב שיראזי___ תאריך: __16/12/2023_

אישור ראש המגמה

הריני מאשר שספר הפרויקט מוכן לצורך הגשת הסטודנט להגנה על פרויקט הגמר.

שם ראש המגמה: _____ חתימה _____ תאריך: _____

תוכן עניינים

תקציר

ספר זה מתאר את תהליך הפיתוח של המשחק Barca אותו בחרתי לבצע במסגרת פרויקט גמר של תכנית מה"ט. בספר תמצאו הסבר על המשחק, כיצד משחקים ותיאור הבעיות האלגוריתמיות של המשחק.

בנוסף, אפרט על תהליך החשיבה עברתי במהלך הפיתוח וכיצד פתרתי את הבעיות האלגוריתמיות.

תיאור הנושא

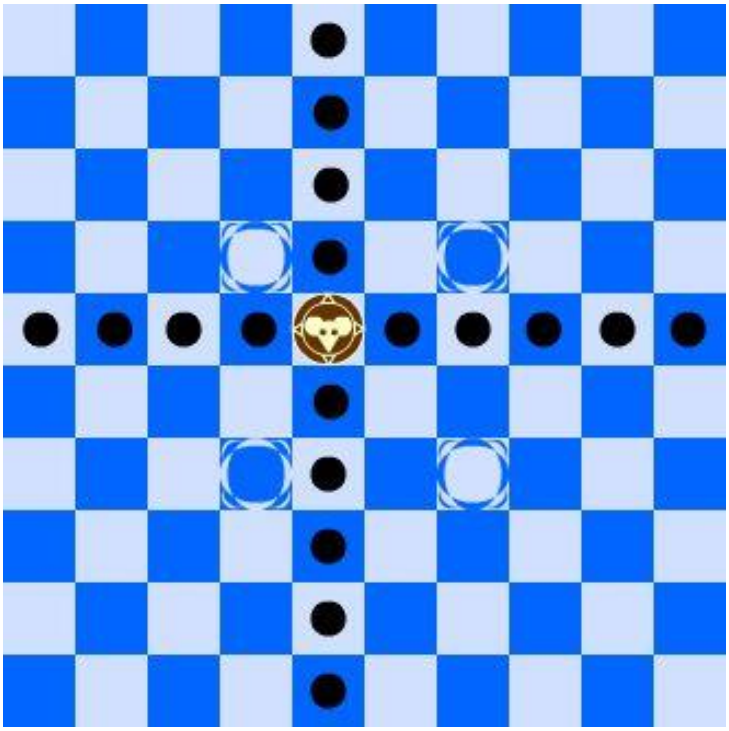
תכולת המשחק:

"בַּרְכַּה" (Barca) הוא משחק לוח שהומצא על ידי אנדרו קאלדוואל בשנת 2007. המשחק הינו משחק אסטרטגיה לשני שחקנים המשלב את היחסים הקיימים באבן, נייר ומספריים יחד עם תנועת כלי שחמט.

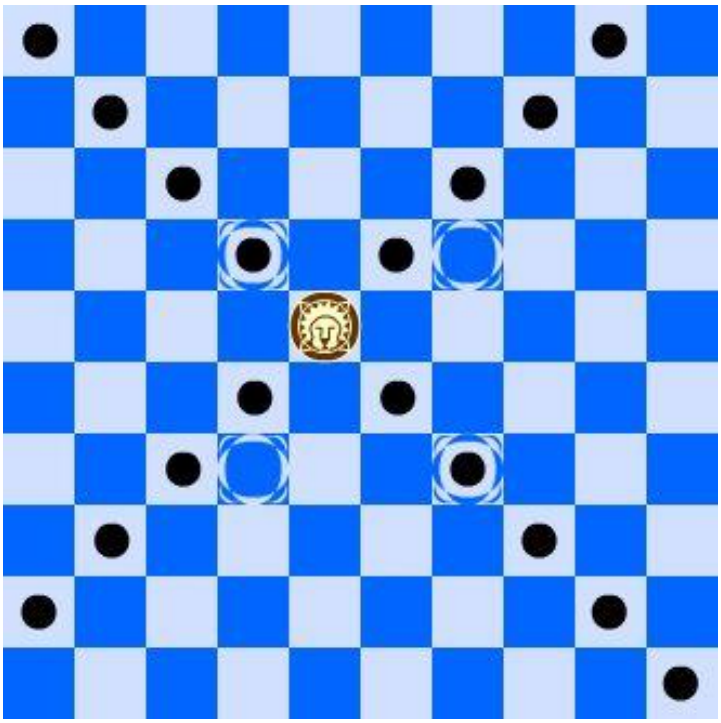
המשחק מתרחש על גבי לוח משבצות 10x10. לכל שחקן 6 כלי משחק: 2 עכברים, 2 אריות ו- 2 פילים. בדומה לשחמט, לכל חיה יש תנועה ייחודית שרק באמצעותה יכולה החיה להתקדם. קרוב למרכז הלוח ישנם 4 משבצות מיוחדות הנקראות בורות מים. משבצות אלו מסומנות ונראות קצת שונה משאר משבצות הלוח.

החיות והיחסים ביניהן:

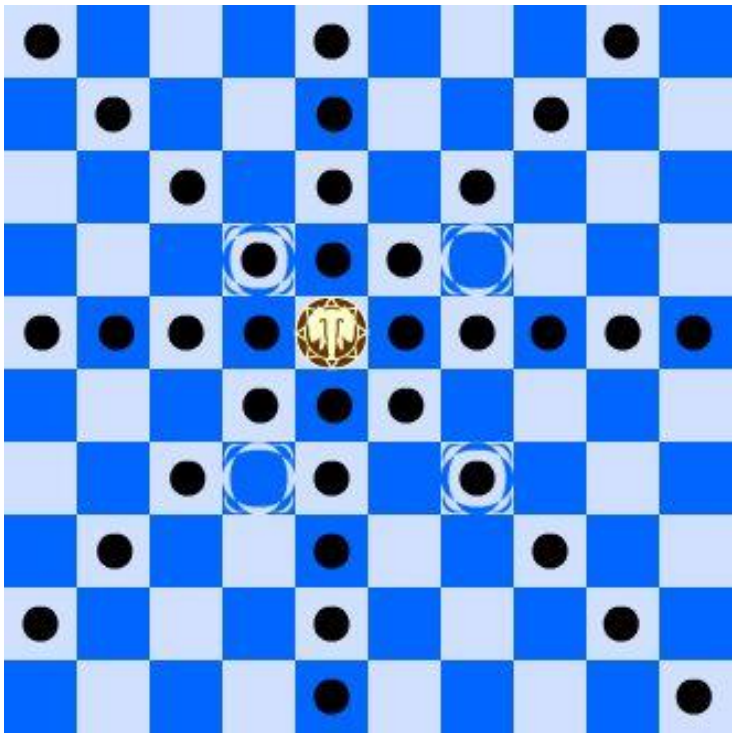
- **עכבר:** העכבר מסוגל לנוע בקו ישר בלבד. קדימה, אחורה, ימינה או שמאלה. העכבר מפחד מהאריה. לכן, אם נמצא אריה באחד מ- 8 המשבצות הסובבות אותו, חובה על העכבר לשנות מיקומו כך שלא ימצא אריה מסביבו.



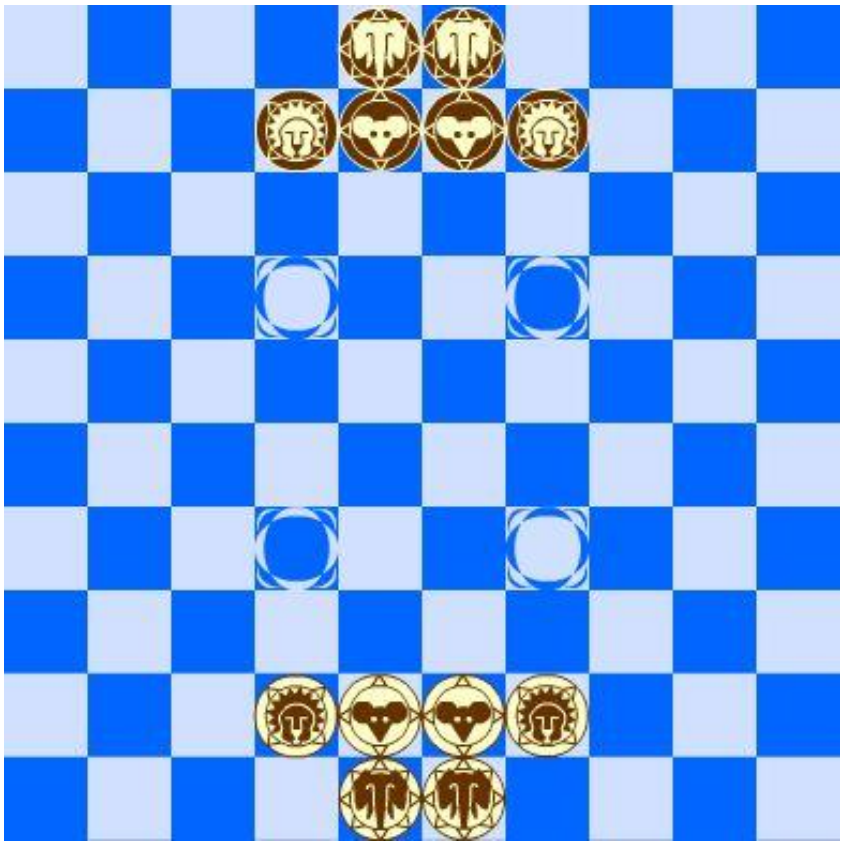
- **<u>אריה</u>:** האריה מסוגל לנוע אך ורק בקווים אלכסונים. האריה מפחד מהפיל. לכן, אם נמצא פיל באחד מ- 8 המשבצות הסובבות אותו, חובה על האריה לשנות מיקומו כך שלא ימצא פיל מסביבו.



- **<u>פיל</u>:** הפיל מסוגל לנוע בקווים אלכסונים וישרים. הפיל מפחד מהעכבר. לכן, אם נמצא עכבר באחד מ- 8 המשבצות הסובבות אותו, חובה על הפיל לשנות מיקומו כך שלא ימצא עכבר מסביבו.



המצב ההתחלתי של המשחק:



מהלך תור:

בדומה לשחמט, במהלך התור מותר לשחקן לבחור כלי על הלוח אותו יזיז. לאחר תנועת הכלי, התור עובר לשחקן היריב. כל עוד לא קיימת חיה "מפוחדת" השחקן רשאי לבחור להזיז איזה כלי שהוא רוצה. במידה ואחת החיות שלו "מפוחדת" חובה עליו להזיז את אותו הכלי המפוחד. במקרה ויש שני כלים "מפוחדים" או יותר, רשאי השחקן לבחור אחת מהחיות ה"מפוחדות" ולהזיז אותה. במידה והחיה ה"מפוחדת" תקועה באחת מפינות הלוח, רשאי השחקן להזיז כלים אחרים.

מטרת המשחק:

מטרת המשחק היא לתפוס כמה שיותר מבורות המים. זאת אומרת, לתפוס כמה שיותר משבצות מיוחדות. הרעיון הוא להשתמש בחכמה בעיקרון ה"הפחדות" על מנת לפנות כלים של היריב מן המשבצות המיוחדת ולתפוס אותן במקומו.

סיום המשחק:

המשחק מסתיים כאשר אחד השחקנים תפס 3 מתוך 4 בורות המים. זאת אומרת, באותו התור, מונחים 3 כלים של אותו השחקן על 3 מהמשבצות המיוחדות.

רקע תיאורטי

שם המשחק בַּרְכַּה (Barca) נקרא על שמו של המצביא והמדינאי בן קרתגו, חניבעל בן חמלקרת בַּרְכַּה. זאת משום שבמהלך מלחמה תקף חניבעל את רומא באמצעות צבא הכלל בתוכו גם פילים.

המשחק הומצא ב- 2007 על ידי אנדרו קאלדוואל (Andrew Caldwell).

המשחק אינו פופולרי במיוחד ולכן לא קיימות אסטרטגיות ספציפיות המוכרות עם המשחק. הדבר הבטוח הוא שהשגת הניצחון מצריכה תיאום של כל החיות בתמיכה זו בזו על מנת להחזיק בשניים מבורות המים, כנגד ניסיונות ההדחה של היריב, תוך יצירת פתח לכיבוש בור המים שלישי.

מושגים

- Model View Controller - MVC- תבנית עיצוב שמפרידה בין הגרפיקה למודל באמצעות בקר.
- Observer Design Pattern- תבנית עיצוב אשר מגדירה קשר אחד לרבים בין קבוצת אובייקטים כך כשאובייקט אחד משנה את מצבו, כל התלויים מקבלים הודעה ומעודכנים בהתאמה.
- עיקרון ה"הפחדות"- עיקרון של המשחק בַּרְבַּה לפיו כל כלי משחק מפחד מכלי משחק אחד אחר. אם כלי זה נמצא כאחד משכניו, הכלי נחשב כ"מפוחד" ומחויב לנוע מהמשבצת עליו נמצא בתור הבא (העכבר מפחד מהאריה, האריה מן הפיל, והפיל מן העכבר).

תיאור הבעיה האלגוריתמית

בעיות אלגוריתמיות בפיתוח המשחק:

במהלך בניית המשחק נתקלתי בבעיות אלגוריתמיות שונות שהייתי צריך לפתור על מנת לשמור על חוקי המשחק.

חוקיות תזוזה:

כל כלי משחק זז בצורה שונה. בכדי לבדוק את החוקיות של מהלך מסוים יש לבדוק שהכלי לא חורג מיכולות התזוזה שלו. בכדי לפתור בעיה זו השתמשתי בנוסחאות מתמטיות פשוטות אשר מחשבות האם המהלך חוקי על פי הפרש מיקומי המשבצות (משבצת מקור ומשבצת יעד). השתמשתי בעיקרון הפולימורפיזם כדי לתכלל את הפונקציה הבודקת חוקיות תזוזה של כלי כלשהו.

עיקרון ה"הפחדות":

אחת הבעיות העיקריות בפיתוח המשחק הייתה מימוש עיקרון זה. מדובר בעיקרון יחסית מורכב שעל פיו:

- לכל כלי יש כלי אחר שמסוגל "להפחיד" אותו ולכן יש לבדוק האם נמצא באחד מהמשבצות השכנות לו.
- אם קיים כלי "מפוחד" חייב השחקן להזיז אותו.
- אם קיימים יותר מכלי "מפוחד" אחד, יכול השחקן לבחור איזה מהם להזיז.
- אם כל הכלים "המפוחדים" תקועים, כלומר לא יכולים לברוח, יכול השחקן להזיז כל כלי פנוי אחר.

פתרון בעיה זו דורש סריקה של מצב הלוח הנוכחי ומכך להסיק אילו מהלכים יכול השחקן לבצע. בפתרוני סרקתי עבור כל תור אילו כלים מפוחדים. אם ישנם כלים מפוחדים שאינם תקועים, יוכל השחקן להזיז אך ורק כלים שסטטוס ה"פחד" שלהם (isScared) הוא אמת. במידה ואין כלים מפוחדים, או שהכלים המפוחדים תקועים, יוכל השחקן להזיז אך ורק כלים שסטטוס ה"פחד" שלהם הוא שקר.

הבעיה העיקרית – הכרעת מצבים:

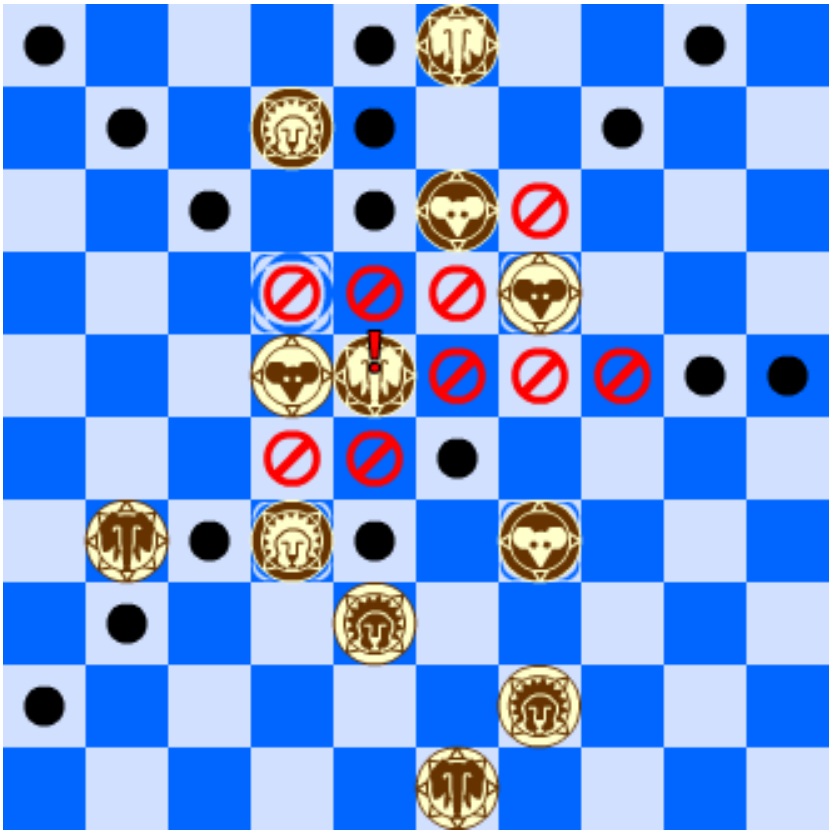
בַּרְבַּה הינו משחק אשר יכול להגיע לאינספור מצבים שונים ומגוונים. מצבים בהם כלי המשחק שלך "מפוחדים", כלים של היריב מוגנים על ידי כלים נוספים שלו, בור מים חסום על ידי שלושת החיות כך שלא ניתן לגשת ישירות אליו, ואפילו מצבים בהם אחד הכלים שלך "כלוא" באחת מקצוות הלוח.

על מנת לנצח במשחק יש לנתח את כלל המצבים בכל מהלך ולבחור באחד הטוב והיעיל ביותר בכל תור.

כדי לדעת באיזה מהלך לבחור בכל מצב, יש צורך באסטרטגיה ידועה מראש.

תפיסת בורות המים:

כדי לנצח עלינו לתפוס 3 מתוך 4 בורות המים שבמרכז הלוח. לשם כך עלינו לנצל את אלמנט ה"הפחדות" בצורה יעילה. אלמנט זה מאפשר לנו לפנות ולחסום בורות מים בפני השחקן היריב. העובדה שכלים בעלי אותו הצבע אינם פוחדים אחד מן השני מאפשרת בניית הגנה על חיה אחרת שנמצאת על אחד מבורות המים.



סקירת אלגוריתמים בתחום הבעיה

אלגוריתם MiniMax-

משחקים רבים בנויים על העיקרון של סט חוקים קבוע ומוגדר היטב ומשחק לסירוגין. לדוגמה: שחמט ודמקה. בתורת המשחקים, עץ מינימקס הוא סוג של מבנה נתונים) הפורס את האפשרויות למשחק של שחקן א', את התגובות של שחקן ב' לכל פעולה של שחקן א', את תגובותיו של א' לתגובותיו של ב' וכך הלאה. מעשית מוגבל עומקו של העץ על ידי הזמן וזיכרון המחשב העומדים לרשותנו (לדוגמה, מחשבי שחמט בונים עצים של כ-9 מהלכים קדימה). העלים בעץ שנוצר הם מצבים סטטיים שנגיע אליהם לאחר רצף של מהלכים (הנתיבים בעץ הם למעשה תרחישים אפשריים). ניתן ציון לכל מצב סטטי שכזה (מצב סטטי בלוח שחמט לדוגמה), שישקף כמה המצב טוב מבחינתנו. מובן שאם נבחר במהלך (ענף) המוביל לעלה עם הציון הגבוה ביותר, לא מובטח לנו שאכן נגיע לאותו עלה. העלה הטוב ביותר הוא התסריט האופטימלי מבחינתנו, וניתן להניח שיריב לא יוביל אותנו בהמשך הנתיב אלא יסיט אותו לנתיב הטוב ביותר מבחינתו. משפט המינימקס מאפשר לנו לדעת מה המהלך הטוב ביותר שנוכל לעשות, על בסיס המידע הנמצא בעץ.

סריקת העץ מתבצעת בצורה רקורסיבית, והסיבוכיות של אלגוריתם זה היא $O(b^d)$ כאשר d מייצג את עומק העץ ו- b מייצג את הגורם המסועף (branching factor).

שיטת Beta-Alpha-

האלגוריתם Beta-Alpha הוא הרחבה לאלגוריתם MiniMax. אלגוריתם זה מגיע לבחירה האופטימלית של MiniMax ללא מעבר על כל צמתי העץ, הוא מבצע גיזום במטרה לסלק חלקים גדולים בעץ מהתחשבות. במילים אחרות, האלגוריתם מזהה על איזה צמתים ניתן לדגל ללא חשש לאיבוד מידע.

סיבוכיות האלגוריתם היא $O(b^{\frac{d}{2}})$.

אסטרטגיה

במשחק כמו בַּרְבֵּה, בו לא קיימת אכילה של כלי היריב, נמצא קושי בדירוג המהלכים האפשריים. כמו כן, ניתן לגשת למשחק בהמון גישות שונות;

<u>גישה התקפית:</u>

עיקרון ה"הפחדות" במשחק פותח אפשרויות לסגנון משחק התקפי, בו ננסה להעסיק את היריב בבריחה מהכלים שלנו, ובכך להרחיק אותו מבורות המים. הרי, כל עוד ליריב קיימים כלים "מפוחדים" שאינם תקועים, **מחויב** הוא להזיז אותם. כך ניתן גם לעקב את היריב במידה ותכנן מהלך מסוים.

<u>גישה הגנתית:</u>

כמו שאפשר להשתמש בעיקרון ה"הפחדות" בכדי לתקוף, ניתן להשתמש בו כדי להגן. חשוב לזכור שהכלים בעלי אותו הצבע אינם מפחדים אחד מהשני. לפיכך, ניתן לצוות כלים מסוימים בכדי ליצור הגנה. למשל, ברגע שהצלחתי לתפוס בור מים בעזרת עכבר, אוכל לצוות לו פיל ובכך למנוע מן היריב לשים אריה צמוד לעכבר שלי. כמובן שניתן לפרק את ההגנה בעזרת מהלכים נכונים (בדוגמה הקודמת: לקרב עכבר שיפחיד את הפיל ויפתח את המקום לאריה) אך ההגנה תחזיק לפחות ל- 2 התורות הבאים מה שנותן זמן לתכנון המהלך העוקב.

<u>השחקן הממוחשב:</u>

לאחר שהתנסיתי בעצמי במשחק, הגעתי למסקנה שחייב להיות איזון בין 2 הגישות. לכן, בדירוג המהלכים עבור השחקן הממוחשב ניסיתי לאזן בין הערכים המספריים שמקבל כל מצב (הפחדת שחקן אחר, גישה לבור מים, תפיסת בור מים וכו') כך שיוכל השחקן לבחור במהלך הטוב ביותר בהתאם למצב הלוח הנוכחי.

מבנה נתונים

תיאור המבנה:

הלוח מיוצג באמצעות מטריצה בגודל 12X12 כאשר כל תא מכיל מצביע לטיפוס מסוג GameSquare.

הטיפוס GameSquare מייצג משבצת על גבי הלוח (המטריצה) ויכול להכיל בתוכו טיפוס נוסף מסוג GamePiece אשר מייצג כלי משחק כלשהו.

במבנה השתמשתי ב- 2 תבניות עיצוב בכדי לנהל את הצד הלוגי אל מול הצד הגרפי:

- MVC- Model View Controller: את הקוד חילקתי ל- 3 חלקים; מודל (לוגיקה), גרפיקה, ובקר. 2 חלקי הלוגיקה והגרפיקה מנוהלים על ידי הבקר וכך הקוד גמיש יותר לשינויים. שינוי בחלק הלוגיקה לא יחייב שינוי בחלק הגרפי או בבקר.
- Observer Design pattern: עשיתי שימוש בתבנית עיצוב זו כך שבעת שינוי לוגי במשחק, יתקיים באופן אוטומטי שינוי גרפי. תבנית זו שומרת על מצב המשחק מבלי להסתבך בשינוי המצב הגרפי עם כל פעולה לוגית.

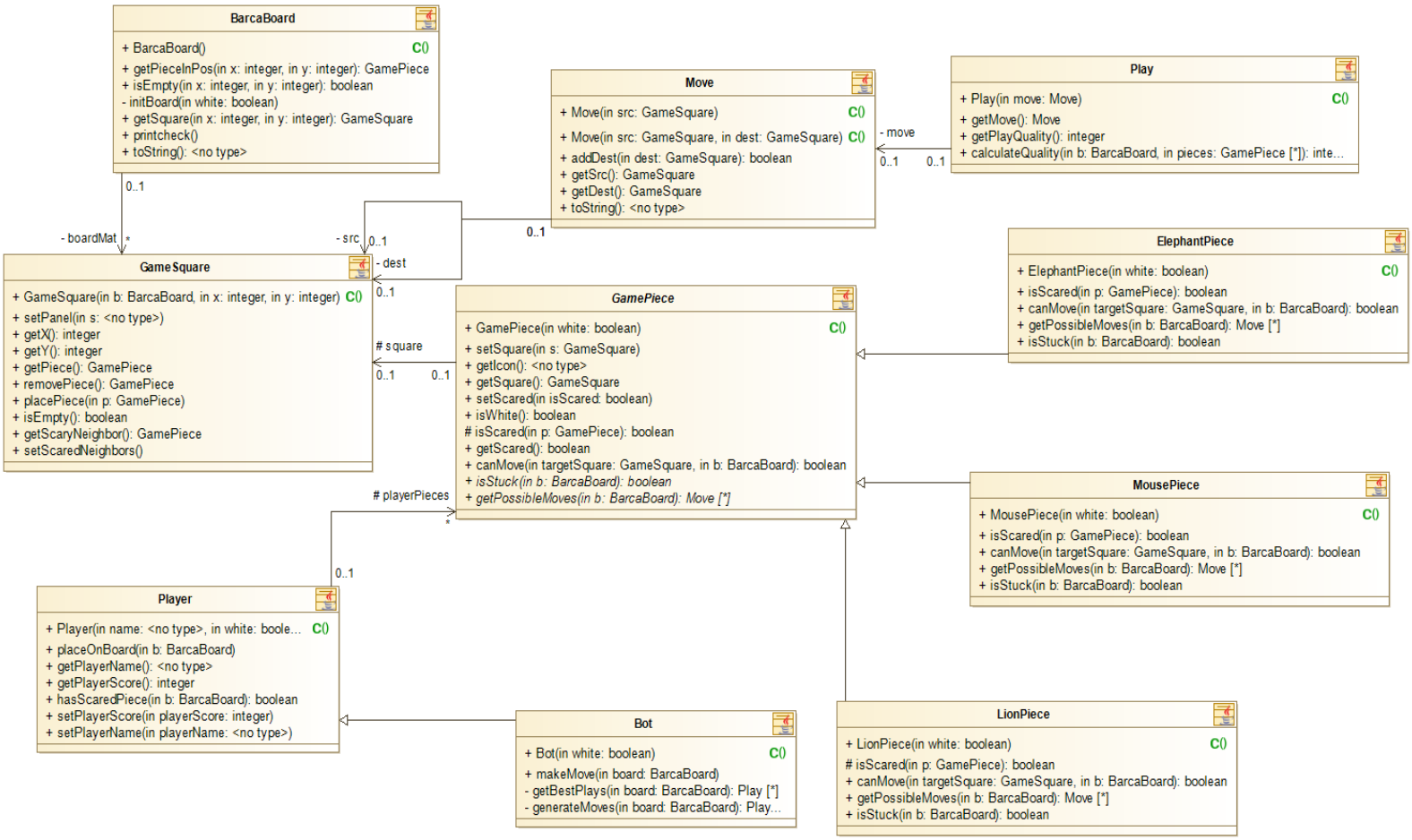
מדוע בחרתי במבנה זה:

כיוון החשיבה שלי בבחירת המבנה היה להישאר קרוב למציאות. הרי גם אם נסתכל על הלוח במציאות נוכל לראות סוג של מטריצה אשר כל תא בה יכול להכיל כלי משחק כלשהו. למטריצה אמנם יש חסרונות בתפיסת הזיכרון ובסריקת הלוח, אך לאחר חשיבה עם עצמי הבנתי שהחסרונות האלו לא עולים על היתרונות באלגוריתמיקה ובארכיטקטורה.

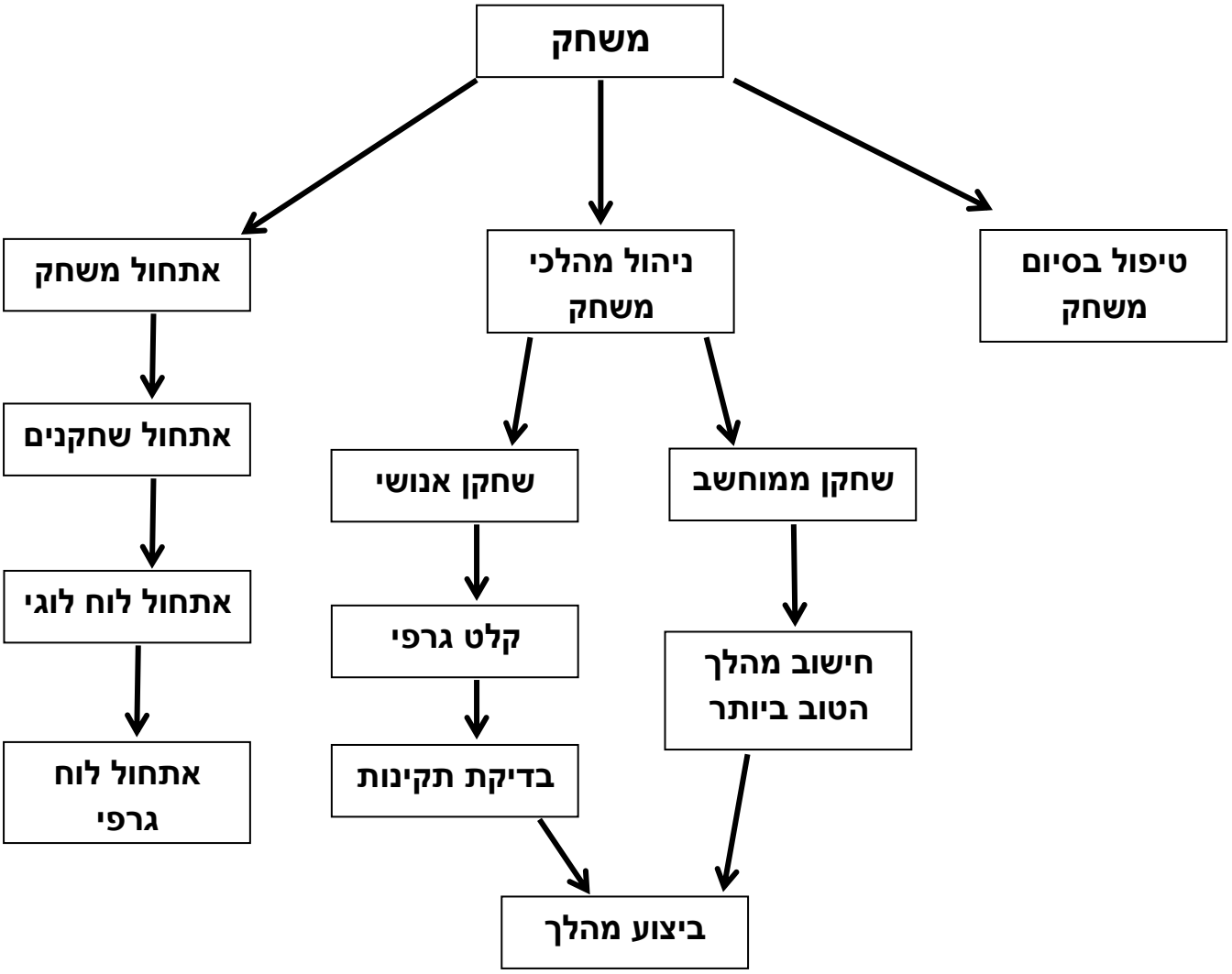
המטריצה מאפשרת לי לגשת לכל תא בגישה ישירה, מה שנותן יתרון רב בבדיקת השכנים באלמנט ה"הפחדות". על מנת שאלגוריתם הבדיקה יהיה קבוע בכל תא הוספתי שכבת מעטפת (לכן המטריצה היא בגודל 12X12 ולא 10X10 כמו לוח המשחק) כך שבדיקת השכנים תהיה זהה עבור כל תא (שכבת המעטפת תמיד תהיה ריקה מכלים).

מעבר לתיאור המציאות, סיבה נוספת לבחירה במבנה זה הינה הנוחות האישית בכתיבת המשחק עצמו. בהשוואת הזמן הניתן לעבודה על הפרויקט לעומת הקושי בניתוח מצבי המשחק, הבנתי כי איני יכול לבזבז זמן רב בבניית המשחק עצמו ומכיוון שמטריצה הינו מבנה בסיסי שהתנסיתי בו המון במהלך השנים, הרגשתי נוח לבחור בו.

תרשים מחלקות



ארכיטקטורה של הפתרון המוצע- Top-Down Level Design



תיאור סביבת העבודה ושפת התכנות

סביבת העבודה:

IntelliJ IDEA היא סביבת פיתוח משולבת שכתובה ב-Java, ב-Kotlin ובספריה הגרפית Swing של Java. היא פותחה על ידי תאגיד התוכנה הצ'כי JetBrains (ששמו הקודם היה IntelliJ) אשר ידוע בפיתוחן של סביבות פיתוח משולבות כגון Pycharm - לפיתוח בשפת פייתון, Rider - לפיתוח בסביבת מיקרוסופט (#J, סי שארפ, וכו'), PhpStorm - לפיתוח ב-PHP ו-Webstorm.

שפת התכנות:

Java – ג'אווה היא שפת תכנות מונחית עצמים אשר פותחה בחברת סאן מיקרוסיסטמס (ביום חברת-בת של אורקל) על ידי צוות בראשות ג'יימס גוסלינג בשנת 1991, והיא אחת משפות התכנות הנפוצות ביותר הנמצאות בשימוש כיום. השפה הוצגה לראשונה בשנת 1995, והיא מהווה את אחד ממרכיבי הליבה של פלטפורמת התוכנה ג'אווה.

התחביר של השפה מבוסס במידה רבה על התחביר של C++, אך כולל הרחבות רבות במטרה לאפשר תמיכה מובנית בתהליכונים, בינלאומיות, אבטחה ועבודה בסביבת האינטרנט ותכונות נוספות. לרוב עוברות תוכניות ג'אווה הידור ל־bytecode Java, שפת ביניים דמוית שפת מכונה, שאותה מריצה מכונה וירטואלית (Java Virtual Machine; JVM). הודות לכך התוכנית יכולה לרוץ על כל מחשב ועל כל מערכת הפעלה המריצים JVM, החל מטלפונים סלולריים ועד למחשבי-על.

תיאור ממשקים

- ArrayList.util.java – מיישם את כל פעולות הרשימה האופציונליות ומאפשר את כל האלמנטים, כולל null .בנוסף ליישום ממשק List ,מחלקה זו מספקת שיטות לתפעל את גודל המערך המשמש באופן פנימי לאחסון הרשימה.
- event.awt.java.* – מספק ממשקים ומחלקות להתמודדות עם סוגים שונים של אירועים המופעלים על ידי רכיבי AWT .
- awt.java.* – מכיל את כל המחלקות ליצירת UI ושימוש בגרפיקה ותמונות.

אלגוריתם ראשי

ניהול המשחק:

1. כל עוד אין מנצח:
   1.1. תור שחקן אנושי
   1.2. תור שחקן ממוחשב

בדיקת ניצחון:

1. 0 -> count
2. עבור כל בור מים:
   2.1. אם נמצא עליו כלי משחק של השחקן הנוכחי:
      2.1.1. count+1 -> count
3. אם count>2:
   3.1. עצור משחק והדפס מי ניצח

תור שחקן אנושי:

1. קלט: מהלך נבחר
2. אם מהלך תקין:
   2.1. בצע מהלך
   אחרת:
   2.2. אפס מהלך וחכה לקלט

תור שחקן ממוחשב:

1. סרוק את הלוח ומצא את כל המהלכים האפשריים
2. עבור כל מהלך:
   2.1. הפעל פונקציית דירוג המהלך
3. שים את כל המהלכים בעלי הדירוג המקסימלי במערך
4. בחר את אחד המהלכים בעלי הדירוג המקסימלי
5. בצע את המהלך הנבחר

פונקציות ראשיות בפרויקט

שם הפונקציה: initGame()

מקבלת:

מחזירה: void

תיאור: הפונקציה יוצרת את כל גורמי המשחק (לוח לוגי, לוח גרפי, שחקנים) ומאתחלת אותם.

יעילות אלגוריתמית: O(n) כאשר n מייצג את מספר המשבצות בלוח.

שם הפונקציה: win(int offset)

מקבלת: מספר שלם המייצג את אופסט השחקן שכעת תורו.

מחזירה: void

תיאור: הפונקציה בודקת האם השחקן שביצע את התור האחרון ניצח, משנה את סטטוס ריצת המשחק לשקר ומדפיסה את שם השחקן שניצח.

יעילות אלגוריתמית: O(1)

שם הפונקציה: getPossibleMoves(BarcaBoard b)

מקבלת: לוח משחק לוגי

מחזירה: רשימה – ArrayList<Move>.

תיאור: הפונקציה מחשבת את כל המהלכים האפשריים עבור הכלי הנוכחי.

יעילות אלגוריתמית: O(n) כאשר n מייצג את אורך הלוח (במשבצות).

שם הפונקציה: makeMove(BarcaBoard board)

מקבלת: לוח משחק לוגי

מחזירה: void

תיאור: הפונקציה מחשבת את הדירוג לכל המהלכים האפשריים עבור כל כלי, בוחרת את המהלך הטוב ביותר ומבצעת אותו.

יעילות אלגוריתמית: O(n+k) כאשר n מסמל את אורך הלוח (במשבצות) ו- k מסמל את מספר המהלכים החוקיים.

מדריך למשתמש

בכל תור עליך תחילה ללחוץ על הכלי הנך מעוניין להזיז ולאחר מכן את המשבצת אליה הנך מעוניין להזיז את הכלי שבחרת. בחירה בתנועה בלתי חוקית תאפס את התור ומצב הלוח לא ישתנה.
במידה וגילית חרטה בכלי שבחרת, תוכל לבחור תנועה לא חוקית במכוון, כך התור יתאפס ותוכל לבחור כלי אחר.

ברגע שבוצע תור, לא ניתן לבטלו.

זכור שהמטרה היא לתפוס 3 בורות מים! בחר את מהלכיך בחכמה, בהצלחה!

רפלקציה

פרויקט זה הינו פעם ראשונה שהתנסיתי בכתיבת משחק בשפת Java ומימוש של שחקן ממוחשב. במשך לימודי בבית הספר, בבסמ"ח ובמכללה למנהל, למדתי המון על אלגוריתמיקה, מבני נתונים, ניהול ותכנון של פרויקטים, תבניות עיצוב וכו'. פרויקט זה גרם לי להסתכל שוב על כל מה שלמדתי ולראות כיצד אני יכול להשתמש בידע שצברתי בכדי להגיע לתוצר הסופי.

בנוסף, נהניתי גם לחקור על נושא הבינה המלאכותית (נושא שלא למדתי עד כה). במהלך המחקר למדתי טכניקות בינה שונות ומגוונות וראיתי כיצד מבני נתונים באים לידי ביטוי במערכות בינה מתקדמות (לדוג' deep learning).

בפרויקט יצא לי להתנסות ב- 2 תבניות עיצוב: MVC ו- Observer. שתי תבניות העיצוב הללו הפכו את העבודה על הפרויקט ליותר פשוטה וממוקדת. ברגע שגיליתי בעיה באחד החלקים יכולתי להתמקד בתיקון החלק הספציפי מבלי שזה ישפיע על שאר הפרויקט. כך, בשונה מפרויקטים שעשיתי בעבר, לא הייתי צריך לבזבז את הזמן על עדכון קוד קודם שיתאים לשינויים.

בסופו של דבר, אני מרוצה שיצא לי להתנסות בפרויקט שכזה. איני לגמרי מרוצה מהתוצר הסופי. בהסתכלות לאחור, הייתי רוצה להיות החלטי יותר בבחירת הנושא לפרויקט וכך להרוויח לעצמי יותר זמן לעבודה.

לסיכום, אני מרגיש שתהליך העבודה על הפרויקט שיפר את החשיבה שלי בכל מני נושאים מעולם התכנות והראה לי שלמרות שטוב לדעת חומר בצורה תאורטית, חשוב לצבור ניסיון על מנת ליישם אותו בצורה הטובה ביותר.

ביבליוגרפיה

- BoardSpace, Barca: https://www.boardspace.net/barca/english/rules.html
- Play Barca: http://playbarca.com/
- UCI - Senior Projects Portal, Beating Barca: An Artificial Intelligence System Designed to Master the Barca Board Game: https://srproj.eecs.uci.edu/projects/beating-barca-artificial-intelligence-system-designed-master-barca-board-game
- Wikipedia, Barca (board game): https://en.wikipedia.org/wiki/Barca_(board_game)
- Wikipedia, IntelliJ IDEA: https://he.wikipedia.org/wiki/IntelliJ_IDEA
- Wikipedia, Java: https://en.wikipedia.org/wiki/Java_(programming_language)