# תכנות מונחה עצמים ופיתוח משחקים

# תרגיל 4 – Circle the Cat

## פרטי הסטודנטים

* רנאל בן סימן טוב
* עידן כהן

## הסבר כללי של התרגיל

בתרגיל זה מימשנו את המשחק "Circle the Cat", שהוא משחק מז'אנר משחקי הלוח שבו מוצג לשחקן לוח שהוא מעין חידה שעליו לפתור. במשחק זה ישנו לוח אריחים בגודל 11x11 ובמרכזו חתול. אריח יכול להיות חסום או פנוי. מטרת השחקן היא לכלוא את החתול כך שלא יהיה לו יותר לאן לזוז, כלומר להקיף אותו באריחים חסומים מכל צדדיו – במידה והצליח השחקן ניצח ומועבר לשלב הבא. במידה והחתול הצליח לברוח, כלומר להגיע לאריח שנמצא בקצות הלוח – השחקן הפסיד ועליו לנסות לפתור את השלב שוב.

## תיכון

מחלקת Controller שאחראית על ניהול המשחק בכללותו – בה נמצאת לולאת המשחק המרכזית וניהול זרימת השלבים השונים עד לסיום המשחק, וכן פונקציות ה-undo וה-reset. מכילה את האובייקטים המרכזיים במשחק כממברים ומנהלת אותם. אחראית על עיבוד הקלט מהמשתמש (לחיצות העכבר) והעברת מידע זה למחלקות שלהן זה רלוונטי.

מחלקת Board – אחראית לייצוג לוח האריחים במשחק, מכילה ממבר של מבנה הנתונים המרכזי במשחק: וקטור דו-ממדי של אריחים. מכילה פונקציה שמממשת את אלגוריתם BFS למציאת המסלול הקצר ביותר לקצות הלוח.

מחלקת Hexagon – אחראית לייצוג אריח בודד (משושה) במשחק, מהם מורכב הלוח.

מחלקת Cat – אחראית לייצוג החתול במשחק. מכילה בין היתר פונקציונליות של אנימציה, ומשתמשת באלגוריתם תזוזה מתוחכם.

מחלקת Level – יוצר השלבים האוטומטי שיוצר שלבים בהתאם לרמת הקושי – החלק משלב קל שבו הרבה אריחים חסומים מראש, ועד לשלב הקשה ביותר שבו מעט מאוד אריחים חסומים מראש.

מחלקת SideBar – מחלקה עם דגש גרפי שאחראית לייצוג תפריט הצד במשחק, הכולל את מספר השלב, מעקב אחר מספר הצעדים שנעשו עד כה, וציור כפתורי ה-undo וה-reset. מנטרת את הלחיצות על כפתורים אלה.

מחלקת ResourcesManager – מחלקת סינגלטון לשמירת משאבי המדיה הכבדים במשחק – טקסטורות, פונטים, צלילים ומוזיקה. מעצם היותה סינגלטון דואגת לכך שכל משאב יישמר פעם אחת בלבד ולא יישמר פעמים רבות שלא לצורך.

## רשימת הקבצים

* main.cpp
* macros.h – קובץ קבועים
* Controller.h+Controller.cpp – מחלקת ניהול המשחק
* Board.h+Board.cpp – מחלקה לייצוג הלוח
* Hexagon.h+Hexagon.cpp – מחלקה ייצוג אריח (משושה) בודד
* Cat.h+Cat.cpp – מחלקה לייצוג החתול
* Level.h+Level.cpp – יוצר שלבים אוטומטי
* SideBar.h+SideBar.cpp – מחלקה עבור תפריט הצד
* ResourcesManager.h+ResourcesManager.cpp – מחלקת סינגלטון לניהול משאבים

## מבני נתונים עיקריים ותפקידיהם

* במחלקת Board – וקטור דו ממדי של Hexagons לייצוג לוח המשחק.
* במחלקת Controller – מחסנית (stack) המתעדת את היסטוריית הצעדים של המשחק. בכל צעד היא שומרת את האריח שנחסם ואת מיקום החתול וכך ניתן לשחזר צעדים אחורה באמצעות כפתור ה-undo.
* במחלקת Level – set של מיקומים אליה אנו מכניסים את המיקומים של האריחים שברצוננו לחסום כבר בתחילת השלב הנוכחי. מעצם היותה set היא לא תתיר לנו להכניס אותו מיקום פעמיים וכך אנו נמנעים מהגרלות כפולות של אותם מיקומים ומוודאים כי מדובר במיקומים שונים.

## אלגוריתמים הראויים לציון

* אלגוריתם תזוזת החתול המבוסס על BFS – תזוזת החתול נעשית בצורה חכמה, החתול תמיד יודע מהו המסלול הקצר ביותר כדי להגיע לקצות הלוח. האלגוריתם מבוסס על חיפוש לרוחב בגרף (BFS). מריצים BFS מהמיקום של החתול (המקור), עד שמתגלה אריח ראשון שנמצא בקצות הלוח. זהו בהכרח האריח הקרוב ביותר לחתול. משחזרים את המסלול אחורה לכיוון החתול, וכך החתול יודע לפנות בכיוון הטוב לו ביותר (כך שיוכל לברוח הכי מהר).
* אלגוריתם שחזור הצעדים (undo) המבוסס על תיעוד הצעדים במחסנית. כאשר נלחץ כפתור ה-undo אנו מבצעים pop למחסנית וכך הצעד האחרון מבוטל.

## באגים ידועים

-

## הערות אחרות

* מימשנו את כפתור ה-undo באמצעות מחסנית ששומרת את היסטוריית הצעדים. כמו כן הוספנו כפתור reset לחזרה למצב ההתחלתי של כל שלב (מרוקן את מחסנית היסטוריית הצעדים).
* הוספנו אנימציה לתזוזת החתול וכן מוזיקת רקע, צלילים טקסטורות ומסכי מעבר.