בס"ד

README

**1. כותרת :**

תכנות מונחה עצמים ופיתוח משחקים – פרוייקט

**2. פרטי הסטודנטיות:**

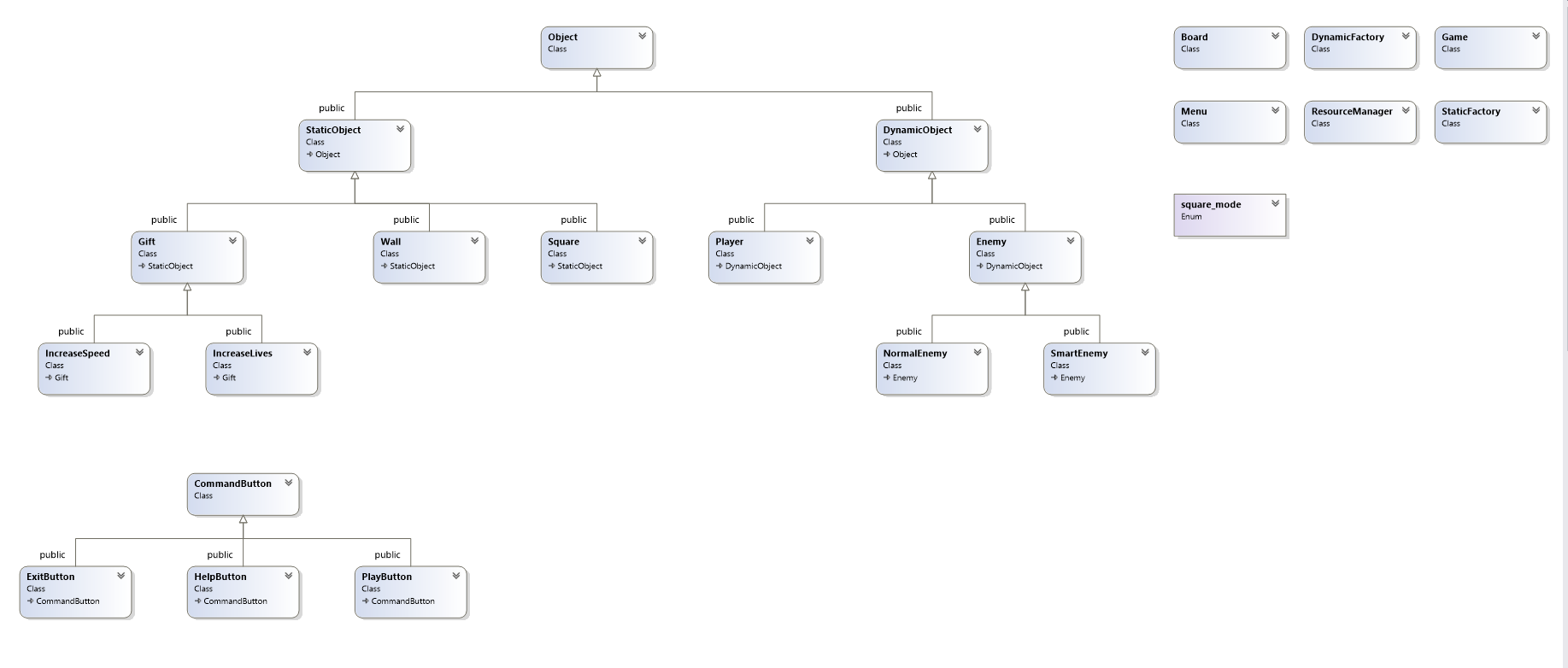
נאוה סוטו ת.ז. 208763516

חוי שמידט ת.ז. 200400364

**3. הסבר כללי של התרגיל:**

משחק "סוגר שטחים" שבו על השחקן לתחום שטחים, כאשר עליו להיזהר מהתנגשות באויבים. במהלך המשחק מופיעות מתנות אותן הוא יכול לאכול והן מוסיפות לו יכולות מגוונות. השלב מסתיים כאשר השחקן תחם את רוב הלוח.

**4. תיכון:**



**Object** - מחלקה המייצגת כל אובייקט במשחק, ותכונתה העיקרית היא החזקת .SPRITE

ממנה יורשות שתי מחלקות: StaticObject ו- DynamicObject .

**DynamicObject** - מייצגת את האובייקטים הפעילים שהם: השחקן והשדונים. זו מחלקה אבסטרקטית, כיוון שיש בה פונקצית move וירטואלית טהורה, אשר משותפת לכל הדמויות היורשות ממנה.

**StaticObject** - מייצגת את האובייקטים הפסיבים שהם: העוגיות והקירות. אלו אובייקטים שנקראים לתוך הלוח ונשמרים בווקטור.

מתוך מחלקת DynamicObject יורשים שתי מחלקות: Player ו-,Enemy אשר יש להן פונקציונאליות שונה בתזוזה וביכולות של כל אחד.

**Player**– אחראית על השחקן הראשי ותזוזתו.

**Enemy**– אחראית על השדונים ומתחלקת לשתי מחלקות שיורשות מימנה: SmartGhost ו- RandomGhost.

**SmartEnemy**– מייצגת את השדונים החכמים אשר רודפים אחרי השחקן ואחראית על תזוזתם.

**NormalEnemy**– מייצגת את השדונים הרנדומלים אשר תזוזתם נעשית באופן אקראי.

מתוך מחלקת StaticObject יורשות שתי מחלקות: Gift ו-. Wall

**Wall** – מייצגת את הקירות החוסמים מעבר של השחקנים.

**Gift**– מייצגת את העוגיות אותן אוכל השחקן.

ממחלקת Gift יורשות שתי מחלקות: IncreaseLives

ו-. IncreaseSpeed

**IncreaseLives**-אחראית להוספת חיים לשחקן.

**IncreaseSpeed-** אחראית להגברת מהירות לשחקן.

**Board** – מייצג את כל האובייקטים הסטטים על ידי ווקטור באופן פולימורפיזי.

Square-מייצגת משבצת בלוח אשר לכל משבצת קיים סטטוס שמתעדכן עם מהלך המשחק.

**ResourceManager** – זוהי מחלקה המתנהלת בעזרת .singletonכל הטעינות של התמונות, צלילים ,וטקסט הנצרכות במהלך התוכנית מתבצעות בה פעם אחת בלבד.

**StaticFactory-**מחזיקה map לכל האוביקטים היורשים מ- StaticObject . ב- map מוחזק תו המייצג את שמו של האוביקט ב-enum ופונקצית יצירה של אותו אוביקט.

כל אוביקט רושם את עצמו ובשעת הצורך קורא לפונקצית create שיוצרת בפועל.

**DynamicFactory-** מחזיקה map לכל האוביקטים היורשים מ- DynamicObject . ב- map מוחזק תו המייצג את שמו של האוביקט ב-enum ופונקצית יצירה של אותו אוביקט.

כל אוביקט רושם את עצמו ובשעת הצורך קורא לפונקצית create שיוצרת בפועל.

**Menu** - מנהלת את התפריט שבו המשתמש מחליט אם הוא רוצה לשחק או לצאת או לקבל מידע על המשחק.

**CommandButton**- אחראית על השדונים ומתחלקת לשתי מחלקות שיורשות מימנה:

ExitButton, HelpButton,Game

**ExitButton-** לאחר לחיצה על מקש ה-exit מתבצעת יציאה מהמשחק.

**HelpButton-** המסך אליו מגיעים לאחר לחיצה על מקש ה-help המכיל מידע על המשחק.

- **Game** המסך אליו מגיעים לאחר לחיצה על מקש ה-play

**5. קבצים:**

----------------------------------main.cpp-----------------------------------------------------------

מחלקה ראשית היוצרת מופע של Menu ומתחילה להריצו.

---------------------------------------- Menu.h + Menu.cpp--------------------------------------

ניהול תפריט המשתמש והכפתורים שנמצאים בו.

-----------------------------------------Board.h + Board.cpp-------------------------------------

מחלקת הלוח מייצגת את האובייקטים הסטטים, שומרת אותם בווקטור וקוראת להם באופן פולימורפיזי.

--------------------------------------- Object.h + Object.cpp-------------------------------------

המחלקה הבסיסית הראשית אשר מתארת את כל האוביקטים על הלוח ע"י זה שיש לה Sprite שלו יש תמונה ומיקום.

----------------------- DynamicObject.h + DynamicObject.cpp-------------------------------

מחלקה זו אחראית על כל האוביקטים הדינמיים במשחק. זוהי מחלקה אבסטרקטית שמכילה פונקציות שונות שהן וירטואליות טהורות.

-------------------------- StaticObject.h + StaticObject.cpp-------------------------------------

זוהי מחלקה אבסטרקטית המייצגת את האובייקטים הסטטיים.

---------------------------------- Player.h + Player.cpp---------------------------------------

מחלקה שאחראית על השחקן הראשי. מכילה שדות של ניקוד, חיים וכיוון.

------------------------------------- Enemy.h + Enemy.cpp-----------------------------------------

מחלקה זו אחראית על השדונים. מימנה יורשות שתי המחלקות: SmartEnemy ו- RandomGhost.

---------------------------- SmartEnemy.h + SmartEnemy.cpp------------------------------------

מחלקה שאחראית על השדונים החכמים, מיישמת את פונקציית moving באופן שהשדונים ירדפו אחרי השחקן.

------------------------- RandomGhost.h + RandomGhost.cpp---------------------------------

מחלקה שאחראית על השדונים הרנדומלים, מיישמת את פונקציית moving באופן שהשדונים יזוזו בצורה אקראית.

------------------------------- Gift.h + Gift.cpp------------------------------------------

מחלקה זו מייצגת את העוגיות אותן אוכל השחקן.. ממחלקה זו יורשות שתי מחלקות: IncreaseLives ו- IncreaseSpeed.

---------------------- IncreaseLives.h + IncreaseLives.cpp-------------------------------------

מחלקה זו אחראית להוספת חיים לשחקן.

---------------------- IncreaseSpeed.h + IncreaseSpeed.cpp-------------------------------------

מחלקה זו אחראית הגברת מהירות לשחקן.

--------------------------------- Wall.h + Wall.cpp------------------------------------------------

מחלקה זו מייצגת את הקירות ואחראית לציורם.

----------------- ResourceManager.h + ResourceManager.cpp--------------------------------

מחלקה זו אחראית על הטעינה של כל התמונות, צלילים ,וטקסט הנצרכות במהלך התוכנית .

מחלקה זו מיושמת בעזרת .singleton

----------------- StaticFactory.h + StaticFactory.cpp--------------------------------

מחלקה זו אחראית ליצירת האוביקטים הסטטייים.

----------------- DynamicFactory.h + DynamicFactory.cpp--------------------------------

מחלקה זו אחראית ליצירת האוביקטים הדינאמיים.

----------------- CommandButton.h + CommandButton.cpp--------------------------------

מחלקה זו אחראית על כפתורי התפריט וממומשת באמצעות פולימורפיזם.

----------------- Game.h + Game.cpp--------------------------------

מחלקה זו אחראית על הטעינה של כל התמונות הנצרכות במהלך התוכנית. מחלקה זו מיושמת בעזרת .singleton

----------------- ExitButton.h + ExitButton.cpp--------------------------------

מחלקה זו אחראית על הטעינה של כל התמונות הנצרכות במהלך התוכנית. מחלקה זו מיושמת בעזרת .singleton

----------------- HelpButton.h + HelpButton.cpp--------------------------------

מחלקה זו אחראית על הטעינה של כל התמונות הנצרכות במהלך התוכנית. מחלקה זו מיושמת בעזרת .singleton

ExitButton**-** אחראית לניהול התהליכים לאחר לחיצה על מקש ה- exit . כלומר, לבצע יציאה מהמשחק.

HelpButton**-** אחראית לניהול התהליכים לאחר לחיצה על מקש ה- help . כלומר, לספק מידע על המשחק.

-Gameהמסך אליו מגיעים לאחר לחיצה על מקש ה-play.כלומר, להתחיל שלב.

**6. מבני נתונים עיקריים ותפקידיהם:**

- **וקטור** מסוג StaticObject שמייצג את הלוח. הוא שומר את כל האובייקטים הסטטיים וקורא להם באופן פולימורפיזי.

- **תבנית סינגלטון**: תבנית זו נועדה למקרים בהם מעוניינים להגביל את יצירת המופעים של מחלקה  מסוימת למופע יחיד. היא שימושית כאשר נדרש רק מופע אחד על מנת לתאם פעולות במערכת.  בתוכנית זו מימשנו סינגלטון במחלקת ResourceManager אשר אחראית על טעינת התמונות, כוון שרצינו שטעינה של כל תמונה תתבצע פעם אחת בלבד.

- **map**: מילון הממומש על ידי עץ חיפוש מאוזן שכל צומת בו שומר מפתח וערך. השתמשנו במבנה נתונים זה במחלקת ResourceManager שבו שמרנו את התמונות שנטענו.

- **:double dispatch** מנגנון מיוחד לשליחת פונקציה אחת מספר פעמים עם אובייקטים שונים בכל פעם. הפונקציה collide, המטפלת במקרי התנגשויות במשחק, משתמשת במנגנון זה.

-תבנית **factory**-אחראית לניהול יצירת האוביקטים.

בנינו זאת כך שהאוביקטים נשמרים ב-map שמחזיק תו המייצג את שמו של האוביקט ב-enum ופונקצית יצירה של אותו אוביקט.

כל אוביקט רושם את עצמו ובשעת הצורך קורא לפונקצית create שיוצרת בפועל.

**7. אלוגריתמים הראויים לציון:**

- פונקציות שונות למימוש פולימורפיזם, כגון: פונקצית **moving** אשר נמצאת ב DynamicObjec כפונקציה וירטואלית טהורה, היא ממומשת במחלקות היורשות לכל אובייקט מחדש. נקראת כל פעם בהתאם לדמות שעליה מצביעים עכשיו.

- גם ה- constructor וה- distractor פועלים על ידי פולימורפיזם.

- פונקציות collide - מאחר שיש הבדלים בין ההתנגשות לאובייקטים שונים, יצרנו מספר פונקציות עם אותו אב טיפוס עבור כל אובייקט פעיל ואובייקט פסיבי.

פונקציות אלו פועלות בעזרת  .doubledispatch

- פונקצית סגירת השטח עובדת על פי עיקרון אלגוריתמי BFS/DFS

נתחיל מאחד הריבועים של הקו, מן הסתם הריבוע הראשון, היציאה מהשטח הסגור. יש לו שני שכנים "מעניינים", אחד מימינו ואחד משמאלו (או מעליו ומתחתיו, תלוי בכיוון הקו, אבל נתייחס לזה בתור ימין ושמאל, בלי הגבלת הכלליות). אלה השניים ששייכים לשטח שהיה פתוח עד כה ואחד מהם אולי הולך להיסגר. (תמיד ייתכן שהקו היה צמוד לשטח סגור קיים, אז ייתכן שיש לו רק שכן אחד או אולי אפילו אין בכלל, אם השטח הפתוח הזה היה רק ברוחב של הקו)נטפל בנפרד בשכן הימני ובשכן השמאלי (זה גם מה שיעזור לנו להחליט איזה מהצדדים, אם בכלל, לסגור).

נניח בה"כ שהתחלנו עם הימני. נעבור רקורסיבית על השכנים שלו (תוך שאנחנו מסמנים על מי כבר עברנו, כמו ב-BFS, כך שלא נעבור פעמיים על אותו ריבוע). הרקורסיה נמשכת על כל שכן ריק. היא נעצרת כשכבר אין שכנים ריקים או אם גילינו כדור בתוך השטח הזה. אם גילינו כדור - לא אמורים לסגור את השטח הזה בכלל (אבל עכשיו צריך לבדוק את הצד השמאלי, אולי אותו כן צריך לסגור). אם אין עוד שכנים ריקים - נמשיך ונעבור על ריבועים אחרים של הקו (וצריך לעבור על אותו הצד, כמובן). ברובם כבר לא יהיו שכנים לעבור עליהם (כי כבר הגענו אליהם קודם), אבל עדיין צריך לעשות את זה כדי למצוא מקרים שבהם באמצע הקו "נוגע" באלכסון בשטח סגור, ולכן הרקורסיה נעצרה שם. כשסיימנו לעבור ככה ולא גילינו שום אויב בשטח הזה - ממלאים את כל הריבועים האלה.

אם סגרנו צד אחד, כבר אין מה לעבור על הצד השני כי שם בוודאי שנמצא אויב, אבל אם לא סגרנו אחד - חייבים לבדוק את השני כדי לדעת אם יש שם אויב וכדי לדעת מה לסגור.

**8. באגים ידועים:**

-------

**9. הערות נוספות:**

ישנם דברים אשר עשינו (כמו תבנית strategy) ובגרסא הסופית של המשחק נאלצנו להוריד מבעיות כאלה ואחרות שנוצרו.

**תודה רבה ובדיקה נעימה!**☺