**פרויקט גמר – תכנות מונחה עצמים**

**בנימין רוסין 211426598, יוחאי בניטה 322636036**

בפרויקט הגמר שלנו, ממשנו את המשחק jetpack joyride.

במהלך המשחק, בעזרת מקש הרווח, השחקן רץ וקופץ על גבי לוח אינסופי, כאשר מטרתו היא לאסוף כמה שיותר מטבעות תוך כדי התחמקות מטילים ולייזרים שעלולים להרוג את השחקן.

על גבי הלוח יופיעו אובייקטים נוספים כמו מדענים וצ'קלקות – כתפאורה, מתנות, ואלמנטים נוספים שישנו את אופן הפעולה של השחקן.

כאשר השחקן יפסל, הוא יתבקש להקליד את שמו ובמידה והניקוד שלו (מספר המטבעות) יהיה די גבוה, הוא יתווסף לטבלת השיאים באחד משלושת המקומות הראשונים בהתאמה.

**קבצים ותיכון:**

הקבצים מסודרים בסדר הירושה

* **MenuCommand:** מחלקה זו משמשת כמחלקת בסיס (בעזרת שימוש בתבנית עיצוב command) עבור הכפתורים שיופיעו בתפריט הראשי. היא מכילה פונקציות למימוש הגרפיקה של הכפתורים כאשר הכפתורים עצמם ימומשו מהמחלקות היורשות הבאות:
  + **StartGameCommand:** מחלקה זו מממשת את כפתור התחלת המשחק, ממנה ניכנס אל הפונקציה שמריצה את המשחק.
  + **HighScoreCommand:** מחלקה זו מממשת את הכפתור שלחיצה עליו תביא לצפייה בטבלת השיאים.
  + **RulesGameCommand:** מחלקה זו מממשת את הכפתור שלחיצה עליו תציג לנו את חוקי המשחק.
  + **BackCommandMenu:** מחלקה זו מממשת כפתור חזרה לתפריט מהמסכים של טבלת השיאים וחוקי המשחק.
  + **NextCommandMenu:** מחלקה זו מממשת כפתור לדפדוף קדימה בחוקי המשחק.
  + **PrevCommandMenu:** מחלקה זו מממשת כפתור לדפדוף אחורה בחוקי המשחק.
* **BoardCommand:** מחלקה זו משמשת כמחלקת בסיס (בעזרת שימוש בתבנית עיצוב command) עבור הכפתורים שיופיעו במהלך המשחק. היא מכילה פונקציות למימוש הגרפיקה של הכפתורים כאשר הכפתורים עצמם ימומשו מהמחלקות היורשות הבאות:
  + **BackCommand:** מחלקה זו מממשת כפתור יציאה מהמשחק וחזרה לתפריט הראשי.
  + **MuteCommand:** מחלקה זו מממשת כפתור השתקה לסאונד במהלך המשחק.
  + **PauseCommand:** מחלקה זו מממשת כפתור השהייה למשחק.
* **ObjectFactory:** מחלקה זו היא מחלקה תבניתית מייצרת את האובייקטים השונים במשחק בעזרת תבנית עיצוב Factory.
* **PowerUpFactory:** מחלקה זו מייצרת סטייט מיוחד עבור השחקן במשחק בעת לקיחת מתנה בעזרת תבנית עיצוב Factory.
* **Object:** מחלקה זו משמשת כמחלקת בסיס עבור כל האובייקטים במשחק. היא אחראית על ציור האובייקטים, האנימציות שלהם, והתנועה. ממחלקה זו ירשו המחלקות הבאות:
  + **StaticGameObjects:** מחלקה זו משמשת כמחלקת בסיס עבור כל האובייקטים הסטטיים במשחק. ממנה ירשו המחלקות הבאות:
    - **Light:** מחלקה זו מממשת את הצ'קלקה בלוח המשחק שמתפקדת כתפאורה.
    - **Coin:** מחלקה זו מממשת את יצירת המטבעות ע"י תבניות שהוגדרו בפונקציות שיוגרלו.
    - **Gift:** מחלקה זו משמשת כמחלקת בסיס עבור יצירת המתנות השונות במשחק.ממנה ירשו המחלקות הבאות:
      * **Gmoney:** מחלקה זו תממש את piggy במשחק. היא תגריל את מספר המטבעות שיהיו piggy.
      * **Gpower:** מחלקה זו תממש את הכניסה למתנת ה-powerup במשחק.
      * **Gspeed:** מחלקה זו תממש את הכניסה למתנת ה-speedy במשחק.
  + **MovingGameObjects:** מחלקה זו משמשת כמחלקת בסיס עבור כל האובייקטים הזזים במשחק. ממנה ירשו המחלקות הבאות:
    - **Player:** מחלקה זו מממשת את השחקן עצמו, התנועות, המשחקיות, מעבר בין מצבים ועדכון המהירות כתלות בזמן.
    - **Scientist:** מחלקה זו מממשת את המדענים שברקע המשחק (תפאורה) ואת תנועתם.
    - **Laser:** מחלקה זו מממשת את הלייזרים שבמשחק ומגרילה בין לייזרים סטטיים ללייזרים מסתובבים.
    - **Missile:** מחלקה זו מממשת את הטילים במשחק, קובעת את מיקומם לפי מיקום השחקן ומממשת את תנועתם.
* **PlayerAnimationState:** מחלקה זו מממשת תבנית עיצוב מסוג state עבור מימוש התנועות והמצבים השונים של השחקן. ממנה ירדו המחלקות הבאות:
  + **DeadPlayerState:** מחלקה זו מממשת את המוות של השחקן.
  + **JumpAnimationPlayerState:** מחלקה זו מממשת את הקפיצות של השחקן.ממנה ירשו המחלקות הבאות שיממשו מצבים שונים:
    - **RegularJumpState:** מחלקה זו מממשת את הקפיצה של השחקן הרגיל.
    - **TankJumpState:** מחלקה זו מממשת את הקפיצות של השחקן כאשר הוא מסוג Tank.
    - **SpeedPlayerState:** מחלקה זו מממשת בפועל את המתנה speedy שנותנת לשחקן מהירות והגנה.
  + **WalkAnimationPlayerState:** מחלקה זו מממשת את ההליכה של השחקן.ממנה ירשו המחלקות הבאות שיממשו מצבים שונים:
    - **RegularWalkState:** מחלקה זו מממשת את ההליכה של השחקן הרגיל.
    - **TankWalkState:** מחלקה זו מממשת את ההליכה של השחקן כאשר הוא מסוג Tank.
    - **GravityPlayerState:** מחלקה זו מממשת בפועל את מצב ה-gravity המתקבל לאחר קבלת מתנה מסוג powerup.
* **MissileAnimationState:** מחלקה זו מממשת תבנית עיצוב מסוג state עבור מימוש כל המצבים השונים של הטיל.ממנה ירשו המחלקות הבאות שיממשו מצבים שונים:
  + **AlertMissileState:** מחלקה זו מממשת את ההתראה לפני ירי הטיל, כאשר ההתראה תכוון את הטיל לפי מיקום השחקן.
  + **FireMissileState:** מחלקה זו מממשת את ירי הטיל ותנועתו.
* **LaserState:** מחלקה זו מממשת תבנית עיצוב מסוג state עבור מימוש כל המצבים השונים של הלייזר.ממנה ירשו המחלקות הבאות שיממשו מצבים שונים.
  + **LaserRotatingState:** מחלקה זו מממשת את הלייזר המסתובב.
  + **LaserStaticState:** מחלקה זו מממשת את הלייזר הסטטי.
* **Board:** מחלקה זו מממשת את לוח המשחק, האלגוריתם ליצירת הלוח, ההתנגשויות וניהול המשחק עצמו.
* **CollisionHandling:** מחלקה זו היא מחלקה סטטית המממשת טבלת התנגשויות (HitMap) עם פונקציות למימוש ההתנגשויות ובנוסף מממשת את הפונקציות עצמן עבור כל אובייקט.
* **Controller:** מחלקה זו מממשת את המעטפת של התכנית עצמה, מממשת את האלגוריתם להסטת הלוח, הכפתורים שבמשחק, ניהול טבלת השיאים ועוד.
* **Menu:** מחלקה זו מממשת את התפריט הראשי של המשחק, המסכים של החוקים וטבלת השיאים, הכפתורים של התפריט הראשי וכן של מסך החוקים וטבלת השיאים ועוד.
* **Utilities:** קובץ זה מממש פונקציות יצירת צורות למטבעות, פונקציות נוספות הנצרכות עבור ה- factory, ומחלקת func-tor עבור מימוש מחיקת אובייקטים ישנים מהלוח.

**תיכון:**

מחלקת הmenu אחראית על הצגת התפריט, טיפול כולל במצב התפריט, טעינת ושמירת טבלת השיאים לקובץ והצגת חוקי המשחק. המחלקה מחזיקה מצביע לcontroller המנהל את המשחק.

מחלקת הcontroller מממשת את ממשק המשתמש, מנהלת את לולאת המשחק הראשית הכוללת תנועה, הדפסות וסאונד. מחזיקה אובייקט מסוג board.

במחלקת הלוח מנוהל המשחק בפועל, מחזיקה מצביע לשחקן ולשתי רשימות אובייקטים, מנהלת את תנועת השחקן ומממשת את תנועת ואנימציית יתר האובייקטים. בנוסף במחלקה זו ממומשת פונקציית הגרלת שלב, עבור כל סוג אובייקט אנו מגרילים את פרק הזמן עד להיווצרות האובייקט הבא מסוג זה, בעת יצירת האובייקט יוגרל פרק הזמן עד האובייקט הבא. כלל פרקי הזמן מנורמלים ביחס למהירות השחקן שהולכת וגוברת. סוג המתנה, מצב הלייזר, צורת וגודל המתנות גם הם יוגרלו ע"י התוכנית. כלל ההתנגשויות ממומשות באמצעות טבלת התנגשויות, התנגשויות עם תפאורה (מדענים ומנורות) אינו משפיע על השחקן, בעת התנגשות במתנת powerup הסוג יוגרל בעת זימון הfactory.

מחלקת objectFactory הינה מחלקה תבניתית אשר משומשת שלוש פעמים בתוכנית שלנו, פעם עבור יצירת אובייקטים סטטיים, פעם עבור אובייקטים נעים ופעם נוספת עבור יצירת צורות של מטבעות, מכיוון שבחלק מהמקרים חוזר יותר מאובייקט יחיד, פונקציית היצירה מחזירה רשימת אובייקטים. בנוסף, מפני שחלק מהאובייקטים צריכים לקבל מצביע לשחקן לצורך קביעת מיקומם וחלק מהאובייקטים אינם צריכים, יצרנו overloading לפונקציית הרישום, כך שתוכל לקבל פונקציות יצירה משני הסוגים ובמידת הצורך יוצרת פונקציית מעטפת אשר מקבלת את כל הפרמטרים אך עוצרת אותם מלהמשיך הלאה במידה ואין צורך לחשוף מידע זה.

בנוסף, קיים מימוש לfunc-tor תבניתי אשר מקבל בבנאי שלו ערך X אשר נמצא מחוץ למסך, הפונקציה עובדת באופן כזה שהיא מחזירה 'אמת' עבור אובייקטים שכבר עברו מן המסך ואנו משתמשים בה על מנת למחוק אובייקטים מיותרים ממבני הנתונים.

**מבני נתונים:**

* 2 רשימות המחזיקות מצביעים מסוג unique\_ptr עבור האובייקטים הזזים והסטטיים בהתאמה. הבחירה בlist- היא מכיוון שבתכנית מתקיימות הרבה מחיקות והוספות כאשר המחיקות לא בהכרח מקצה הרשימה.
* וקטורי כפתורים שמחזיק מצביעים מסוג unique\_ptrלכפתורים במשחק.הבחירה ב-vector היא מכיוון שבתכנית יש צורך במעבר סדרתי וגישה אקראית.
* מפה במחלקת הfactory שמחזיקה מפתח לפי סוג האובייקט ומצביע לפונקציה היוצרת את האובייקט.
* מפה של וקטורים של IntRect שמחזיקהאת הגדלים של התמונות של האובייקטים בsprite sheet, מרווחים ביניהם ומספר הטקסטורות.הבחירה ב-mapהיא כדי להחזיק את המפתחות והווקטור המתאים בזוגות.
* מפה עבור טבלת ההתנגשויות המחזיקה type\_index של אובייקט ומצביע לפונקציה המנהלת את ההתנגשות עבור האובייקט.הבחירה ב-mapהיא כדי להחזיק את ה type\_index והמצביעים לפונקציה בזוגות.
* מפה מסוג multimap המחזיקה ניקוד ושמות של השחקנים. הבחירה ב multimapהיא ע"מ לאפשר תיקו בין שחקנים וכדי להחזיק את הניקוד והשמות בזוגות. טבלת השיאים נשמרת בקובץ כך שבין משחקים טבלת השיאים נשמרת.

**אלגוריתמים:**

* אלגוריתם להזזת הלוח כתלות בזמן שחלף.
* אלגוריתם ליצירת האובייקטים על הלוח – מייצר אובייקטים בעזרת הגרלות לפי קבועי זמן שונים לאורך כל המשחק.
* אלגוריתם להעלאת מהירות השחקן – ככל שהזמן עובר, מהירות השחקן גדלה והמשחק נעשה מאתגר יותר.
* אלגוריתם לניהול ההתנגשויות – ממומש ע"י חישוב המטריצה ההיפוך של הטרנספורמציה של האובייקט והפעלתה על השחקן, לאחר מכן מבוצעת בדיקה אם יש חפיפה בין האובייקטים ופניה לפונקציה לניהול ההתנגשות.
* אלגוריתם לכיוון הטיל על השחקן – במהלך ההתראה על הגעת טיל, ההתראה עוקבת אחרי מיקום השחקן ולאחר מכן הטיל משוגר לפי מיקום השחקן.

**באגים ידועים:**

* בעת משחק בpowerup מסוג gravity, בעת לקיחת מתנת מהירות כאשר נמצאים בהיפוך של השחקן, אזי בסיום מתנת המהירות קיימת בעיה עם החזרה למצב הקודם.