**תרגיל 4, חריגות – Lights:**

**שמות מגישים:** איתן שלו 315511741, יונה הראל 315427047. **שם מרצה:** ד"ר אלחנתי מיכל, אמיתי לוי, יחזקאל ברנט.

**מטרת התרגיל**: שימוש במבני נתונים, איטרטורים ואלגוריתמים בסגנון STL וריענון השימוש ב־SFML.

**הסבר כללי:**

מימשנו משחק מחשב. משחק זה הוא מז'אנר משחקי הפאזל, בהם מוצג לשחקן לוח עם "חידה" שהוא צריך לפתור לפי כללי המשחק. לשחקן מוצג לוח משושה שעליו מופיעות נקודות (”נורות") המפוזרות במרווחים שווים. מכל נקודה יוצאת זרוע אחת או יותר. מספר הזרועות המקסימלי הוא 6 זרועות עבור נקודות שלא נמצאות על גבולות המשושה, ועבור נקודות שעל הגבול הוא נמוך יותר, בהתאם למספר השכנים. כאשר זרועות של נקודות סמוכות מכוונות זו כלפי זו הן מחוברות ונוצר קשר בין הנורות. הנורה המרכזית דולקת תמיד, וכל נורה שמחוברת לנורה דולקת (כלומר, מחוברת לנורה המרכזית ישירות או דרך נורות אחרות) נדלקת אף היא. המטרה של השחקן היא להדליק את כל הנורות בלוח. בכל לחיצה על נקודה כלשהי, היא מסתובבת ב־60 מעלות, יחד עם זרועותיה. לחיצה של כפתור העכבר השמאלי מסובבת עם כיוון השעון, ולחיצה של הכפתור הימני – נגד כיוון השעון.

התכנית מתנהלת בלולאה שמקבלת מחוות מהמשתמש בעשרת לחיצות על גבי מסך המשחק. עבור כל מחווה היא פועלת בהתאם ומשנה את מבנה הנתונים בהאם למה שנלחץ. נסביר בקצרה אודות הצורה בה בנוי הפרויקט שלנו על מנת להסביר כיצד הוא נבנה וכיצד צורת הבניה הזו תורמת לחלק בין המרכיבים השונים של המשחק- הפרדנו בין מחלקות שמנהלות את כל הinterface – כל הsfml, לעומת המחלקה שמנהלת את הdata. – הסבר-יש לנו מבנה נתונים שבמחלקה שלו יש את כל האלגוריתמים והפונקציות הדרושות על מנת לשמור את הנתונים- ללא קשר לsfml. כעת יש מחלקה נוספת שהיא מנהלת את כל החלק הsfml-י והיא מקבלת את המידע הדרוש לה להצגת המשחק ובעצם לנהל את כל חווית המשתמש מול המסך. היא לא קובעת כלום בניגוד למחלקה של התנונים אשר היא מכתיבה למחלקה של הsfml מה לעשות בעצם.

גם בתרגיל זה לא השתמשנו במסוף (terminal ) אלא בחלון של sfml . את היררכיית הירושה שלנו בנינו באופן הזה שתהיה לנו מחלקת גרף(שמנהל את ה data בלבד , ומחלקת GraphInterface )שמנהלת את כל הקשור ב-sfml( . הן בעצם די מקבילות לכאורה ועושות דברים דומים ולכן לכל אחת גם יש מחלקה בשם point לגרף וlight לגרף האספמלי- שהיא בעצם מנהלת את כל הקשור בענייני הנקודה הפרטית עצמה . שכן אנו מחזיקים מבנה נתונים המכיל נקודות כאלו - מלא נוקדות.

**תיכון (desing):**

**המחלקות+אובייקטים:**

**מחלקת Globals: אחראית על כל המשתנים הקבועים בתכנית.**

**מחלקת Board:** זוהי המחלקה הבסיסית -הסבר: המחלקה שאחראית על מהנה התנונים מבחינה אלגוריתמית- שם מייצרים את כל השלבים (יצרנו פונקצייה ליצירת שלבים – היא מקבלת מספר שמייצגת את מספר הקודקודים שיהיו בשורה המרכזית , וממנו בונה שורות – חשוב לציין שהיא עובדת עבור כל אינדקס שלם חיובי ממש. אחראי על כל עניין הטיפול בשלבים -למחוק שלב שהסתיים להמשיך לשלב הבא, וכולי. **עשינו לו פונקציה שהורסת את כל מהנה הנתונים בסיום שלב מסיבה כלשהי.**

**כעת יש לנו עוד 2 מחלקות עבור** הלוגיקה של המשחק **שהן: Graph,Point**

**ועוד 2 מחלקות עבור** המימוש הגרפי **שהן: GraphInterface, Light.**

**מחלקת Graph: מכילה מבנה נתונים שמכיל זוגות של קורדינטות (x,y) אשר כל קורדינטה היא בעצם אובייקט מסוג Point. הלוח משתמש בחישובים שלה על מנת להעביר למחלקת גרףאינטרפייס**

**מחלקת Point: המבנה הבסיסי של כל נקודה גאומטרית בגרף , מחלקת גרף משתמשת בה על מנת לערוך את כל החישובים שצריך מבחינה חישובית.**

**מחלקת Graph Interface: שומרת את כל הנתונים הרלוונטים להצגת הגרף. מכילה וקטור של light שכך שומרים את כל הנתונים. שומרת וקטור של lights.**

**מחלקת Light: שומרת את כל הדאטא עבור נקודה שנשתמש בזה במהלך ריצת התכנית להציג למשתמש את הלוח בחווית משתמש. ולשם כך כל קודקוד מכיל וקטור של** RectangleShape **ששומרת את כל הצלעות שלו. ובאמצעןת** Vector2f שומר את המיקום של כל קודקוד **.**

**זוגות המחלקות האלו (בורוד ובירוק שתיארתי לעיל) פועלות בהרמוניה יחד כך שעל תפקיד האחת – עבור חווית המשתמש להתיג את המשחק ואילו תפקיד האחת היוא לחשב, לשמור, לספק נתונים רלוונטים אודות המשחק. דרך מחלקת Board הן לכאורה 'מתקשרות, יחד. חשוב היה לנו להפריד באופן זה שכן יש תפקיד ברור לכל אחת וזהו תפקיד שונה ואין סיבה מבחינה תכנותית לערבבן לעניות דעתינו.**

**השתמשנו באיטרטורים כדי לעבור יעיל על מבני הנתונים, כל פונקציה שעוברת על צלעות או כל הקודקודים – כל פונקציה עם המימוש שלה. למדנו להתשמש בזה וזה עזר לנו המון.**

**מבני נתונים עיקריים ותפקודם:**התכנית מכילה וקטור של נקודות עבור הגרף המתמטי ווקטור של נקודות עבור הגרף להצגה באמצעות sfml. חשוב לציין שבתוך כל וקטור מהם יש כמו שתיארתי למעלה ורטור של צלעות המתאים לכל קודקוד בהתאם לשיוכו לתפקיד – אם הוא point אזי המיקום שלו יהיה זוג של איקס וואי לעומת light ששומר אחרת את המיקום ואת הצלעות שלו.

השתמשנו בפונקציית למדה כדי לעבור על מבני נתונים בצורה יעילה יותר.כמו כן השתמשנו בפונקציות של השפה – בalgorithem כדי לעבור על מבני נתונים בהתאם לכל פונקציה למשל השתמשנו בפונקציית for\_if לעבור על כל הקודקודים שמבצעים פעולה כלשהיא.

(\*)האינטרקציה בין האובייקטים השונים מתבצעת בצורה נקייה דרך מחלקת Board.

וקטור של נקודות, וקטור של נקודה לsfml-. וקטור של צלעות עבור הנקודה הרגילה ועבור הנקודה לsfml.

הערות:

1. תמכנו באופן יצירת השלב נקבע ע"י אלגוריתם שביצענו. יש לציין שהמחשב יכול לתכנן שלב בגודל אינסופי וזה לא מוגבל עד 3 שלבים אלא המשחק עצמו מפסיק בשלב בגודל 9 קודקודים במרכז, בהתאם לדרישות התרגיל אבל עקרונית ניתן לשנות את הקבועENDGAME לאיזה אינדקס שנבחר ושם נסיים. מצורף בנספח שלושת השלבים הנדרשים לצורך הגשה ספציפית תרגיל זה אבל הפרויקט עובד לכל סוג של שלב עקרונית.
2. יצרנו את הקוד בצורה גנרית - המשתנים הגלובליים קובעים את אופי המשחק: כך למשל המשתנה DIFICULTY שלנו קובע לאחר יצירת עץ פורש מינימלי – עוד כמה צלעות נוסיף לכל קודקוד ואז לאחר יצירת השלב נוסיף x צלעות מקסימום לכל קודקוד , שכן זה לא בהכרח שאם רמה הקושי תהיה 3 אזי לכל קודקוד תתווספנה עוד 3 צלעות אלא מקסימום עוד 3 – יכול להתווסף אחד שתיים או אפילו לא להתווסף וזה נבחר אקראית גם כן .

***נספחים:***  
  
להלן סיום שלבי המשחק: (לכל שלב יש מבחר שלבים שהמחשב קובע בזמן ריצה – הם אינם שלבים קבועים).

שלב א' שלב ב' שלב ג'

**ב ב ר כ ה**