מכללת הדסה, החוג למדעי המחשב
מבוא לתכנות מונחה עצמים והנדסת תוכנה
סמסטר א', תשפ"א

תרגיל 2

תאריך אחרון להגשה:

הנביאים – יום א', 22/11/2020, בשעה 23:59

שטראוס גברים – יום ג', 17/11/2020, בשעה 23:59

שטראוס נשים – יום ד', 18/11/2020, בשעה 23:59

מטרת התרגיל:

בתרגיל זה נתרגל תכנון ממשק (Interface designing), שימוש במחלקות מורכבות (מחלקות המכילות אובייקטים של מחלקות אחרות), העברת מידע בין אובייקטים, וכתלות בהתקדמות ההרצאות גם נושאים של שימוש נכון בהפניות (References) וב-const.

תיאור כללי:

בתרגיל זה נבנה תכנית המשתמשת במספר אובייקטים. לשם ביצוע המשימה האובייקטים יצטרכו להעביר מידע ביניהם.

התוכנית אותה עליכם לממש היא משחק המבוסס על Lode Runner. אפשר להתנסות במשחק כאן. אנחנו נממש גרסה פשוטה יותר של המשחק, בעלת שני מסכים (שלבים) לפחות, כפי שיפורט בהמשך. שימו לב שכרגע אנחנו עובדים עדיין בטרמינל שהכרנו עד היום והמשחק יתבצע באמצעות הדפסת תווים על המסך.

באופן כללי, המשחק בנוי ממבוך אשר בו מפוזר כסף שאותו על השחקן לאסוף. בזמן שהשחקן אוסף את הכסף, מסתובבים במבוך מספר אויבים שרוצים לאכול את השחקן. מטרתו של השחקן היא לגמור לאסוף את הכסף בלי שהאויבים יתפסו אותו. כאשר נאסף כל הכסף, המסך נגמר ומופיע במקומו מסך חדש. המשחק ממשיך כל עוד יש מסכים חדשים והשחקן לא נפסל. לשחקן יש "שלושה חיים", כך שהמשחק נגמר רק לאחר שהוא נאכל שלוש פעמים ע"י אויב. השחקן יכול לנוע על גבי רצפה רגילה, מוט או סולם, בו הוא יכול לעלות או לרדת.

אתם צריכים לחשוב על התיכון (Design) המתאים מבחינת אלו מחלקות צריכות להיות, חלוקת האחריות ביניהן, אלו פונקציות יהיו בכל אחת וכו'. שימו לב שאתם נדרשים לתאר את התיכון ולהסביר אותו בקובץ ה-Readme (ראו בסוף הקובץ, בחלק המתאר את קובץ ה-Readme).

פירוט הדרישות:

האובייקטים במשחק:

1. **שחקן** – זו הדמות שמייצגת את השחקן הראשי. מטרת המשחק היא שהשחקן יסיים לאסוף את כל מטבעות הכסף, בלי להיתפס על ידי האויבים.
2. **מטבעות כסף** – עצמים שאינם זזים שאותם השחקן רוצה לאסוף על מנת לסיים את השלב. לאחר שאסף, המטבע יעלם מהמסך.
3. **אויבים** – דמויות הנשלטות על ידי המחשב. מנסות להגיע אל השחקן. אם הן נוגעות בשחקן – השחקן נפסל ("יורד לו חיים").
4. **קירות** – הם עצמים שלא ניתן לעבור דרכם.
5. **רצפה** – השחקן יכול לנוע עליה רק לימין ולשמאל. אם הרצפה הסתיימה והשחקן ממשיך ללכת, הוא יפול עד שיגיע לרצפה אחרת מתחת.
6. **סולם** – השחקן יכול לטפס בו או לרדת בו. הכיוונים האפשריים הם רק למעלה ולמטה.
7. **מוט** – השחקן יכול לנוע מעליו לימין ולשמאל או להחליט ליפול. אם נפל, הוא נופל בבת אחת עד שמגיע לרצפה שמתחת. הכיוונים האפשריים הם ימין, שמאל ולמטה.

מהלך המשחק ותנועת הדמויות:

לשם פישוט, המשחק לא מתנהל "בזמן אמת" לפי שעון, כמו המשחק הרגיל, אלא לפי תור, אחד אחרי השני. ז"א, השחקן ינוע ואז כל אויב ינוע ושוב השחקן ינוע והאויבים אחריו.

בכל תור דמות יכולה לזוז לאחד מארבעת הכיוונים (אך לא באלכסון). דמות זזה בכל תור רק משבצת אחת.

נבחין בין המקרים הבאים לגבי התא המבוקש:

- אם התא המבוקש חסום (יש בו קיר או רצפה), התנועה לא תתבצע. מה כן יקרה? נבדיל בין השחקן לבין האויבים: אם אויב ניסה לזוז ונחסם, הוא מפסיד את התור, והתור עובר לדמות הבאה, לפי הסדר שתואר. לעומת זאת, אם השחקן ניסה לזוז בכיוון המכשול ונחסם, התנועה לא תתבצע אך גם התור נשאר שלו, עד שהוא יזוז לכיוון אחר.

- גבולות המסך גם הם משמשים כ"קירות". למשל, גם אם ניתן להגיע לקצה השמאלי של המסך בשורה מסוימת, לחיצה שמאלה נחשבת כמו ניסיון כניסה לתא חסום (עם הכללים דלעיל).

- אם השחקן נע לתא שיש בו אויב, המהלך חוקי (הוא מתבצע), אולם הוא "יאכל" מיד ויורד לו "חיים". אם השחקן "נאכל" ויש לו עוד "חיים", השלב יתחיל מחדש.

- כצפוי, גם אם אויב נע לתא שבו נמצא השחקן, השחקן "נאכל" כנ"ל.

- אם אויב נע לתא שיש בו אויב, המהלך חוקי. כלומר, שני אויבים יכולים להיות בתא אחד.
- אם השחקן נע לתא שיש בו כסף, הוא אוסף את הכסף, ז"א הכסף נעלם והניקוד שלו עולה. ראו להלן פירוט לגבי הניקוד.
- אם השחקן נכנס למשבצת שיש בה סולם, צורת ההצגה של השחקן תשתנה והשחקן יכול רק לעלות או לרדת.
- אם השחקן נעמד מעל מוט, הוא יכול לזוז עליו לימין ולשמאל או להחליט ליפול ממנו עד להתנגשות עם רצפה.

מקשי השחקן:

השחקן מזיז את הדמות שלו בעזרת מקשי החיצים (ראו הסבר בהמשך).

תזוזת האויבים:

מספר האויבים נתון לבחירתכם.

כמו כן, השיטה להזזת האויבים אף היא נתונה לבחירתכם. מספר נקודות מציון התרגיל מוקדשות לנושא זה וככל שהשיטה מוצלחת יותר, הניקוד בסעיף זה יעלה. תזוזה אקראית היא האפשרות הפשוטה כאן, אולם היא גם "מתומחרת" בהתאם. בכל מקרה, עליכם לציין בקובץ ה-readme מה השיטה שהחלטתם לממש ולהסביר את הבחירה בה, איך היא עובדת וכדומה.

סיום השלב:

אם השחקן אסף את כל הכסף במסך הנוכחי, הוא סיים בהצלחה את השלב, והתכנית צריכה לטעון את השלב הבא (אם יש). אם אין שלב נוסף, השחקן ניצח במשחק.

אם השחקן "נפגע" על ידי אויב, מספר ה"חיים" שנותרו לו קטן באחד. אם לא נשארו לו עוד חיים, המשחק מסתיים בכישלון של השחקן. אם נשארו חיים, השלב מתחיל מחדש.

מספר החיים ההתחלתי לשחקן יהיה קבוע ל־3 (תזכורת: אין להשתמש במספרים מפורשים, literals , ישירות בקוד hard-coded, אלא להשתמש בקבוע כלשהו לאתחול).

ניקוד:

כל מטבע שהשחקן אוכל מזכה אותו ב־2 נקודות כפול מס' השלב. כלומר בשלב הראשון, כל מטבע שווה 2 נקודות, בשלב השני – 4 נקודות וכן הלאה.

כל סיום שלב מזכה בבונוס של 50 נקודות כפול מס' השלב, כלומר סיום השלב הראשון יוסיף 50 נקודות, סיום המסך השני – 100 נקודות וכן הלאה.

מסכי המשחק:

מסכי המשחק יוגדרו בקובץ טקסט בשם Board.txt, והתכנית תקרא אותם מקובץ זה.

בקובץ הזה "נצייר" את השלבים באופן הבא:

השורה הראשונה לפני כל "ציור" שלב בקובץ תכיל מספר אחד המגדיר את גודל הלוח לשלב הנוכחי. כלומר אם המספר הינו N אזי השלב יהיה בגודל של N על N משבצות (כלומר, N שורות בנות N תווים כל אחת) וזה מספר השורות (והעמודות בכל שורה) שאנחנו מצפים לקרוא מהקובץ עבור השלב הזה.

השורות הבאות (N שורות, כאמור) מתארות את מבנה השלב ההתחלתי, באופן הבא:

רווח משמעותו משבצת ריקה.

@ – השחקן.

% – אויב.

- - מוט

H - סולם

S - שחקן על סולם

– רצפה או קיר. רצפה - אפשר לנוע עליה. בקיר - השחקן נתקע ולא יכול להמשיך

* – מטבע כסף.

כל המסכים נמצאים באותו הקובץ. הוסיפו שורה ריקה כדי להפריד בין מסך למסך.

ניתן להניח תקינות של הקובץ.

כפי שנאמר לעיל, אתם נדרשים ליצור לפחות שני מסכים, כאשר התוכנית שלכם נדרשת לתמוך במספר שלבים בלתי מוגבל. כל עוד יש בקובץ שלבים - התוכנית שלהם צריכה לקרוא אותם ולהמשיך לשלב הבא.

אם תבססו את הפרויקט על קובצי הפרויקט שקיבלתם מאיתנו, הקובץ יהיה בתיקיית resources וכבר מוגדרות שם ההגדרות כדי להעתיק אותו אוטומטית לתיקיית הפלט, כך שבהרצה הקובץ יהיה בתיקייה ביחד עם התוכנית.

הצגת לוח המשחק:

אחרי שכל הדמויות זזו, יש להציג למסך את לוח המשחק, מצב הניקוד ומספר השלב. כלומר, אם יש שחקן ואויב אחד אז סדר הדברים יהיה: תזוזת שחקן, תזוזת אויב, עידכון של התצוגה. בסיום המשחק, נציג הודעה לשחקן אם הוא ניצח או הפסיד, בהתאם לכללים שהוגדרו לעיל.

ציור הדמויות והאובייקטים השונים זהה לצורה שבה הם מופיעים בקובץ השלבים, זאת אומרת הדפסת תווים לטרמינל כמו שמופיעים בקובץ. כאשר קיימים מספר אובייקטים שונים באותו המקום, נעדיף להציג אויב מאשר להציג מטבע או סולם.

ניקוי לוח המשחק:

כדי לאפשר הצגה ברורה של העדכונים, נצטרך לנקות את המסך בין הדפסה להדפסה. זאת נשיג על ידי הפקודה system("cls") (ונוסיף בתחילת הקובץ: #include <cstdlib>). כדאי לציין ש-system משתמשת בפקודות המערכת, ולכן הפקודה המסוימת הזו תעבוד רק בסביבת Windows ולא במערכות הפעלה אחרות.

אפשרות נוספת היא להשתמש בקוד הבא (שגם הוא ספציפי ל־Windows):

```
SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), {0, 0});
```

בשביל הקוד הזה צריך את Windows.h ולכן מומלץ מאוד לעשות זאת בקובץ cpp. נפרד. הקריאה הזו מחזירה את הסמן לתחילת המסך, ולכן כל מה שנדפיס כעת ישכתב את מה שכבר מופיע. אם אתם בוחרים בדרך הזו, שימו לב לוודא שבכל השורות אתם מדפיסים את כל רוחב השורה (כולל רווחים) כדי למחוק את הטקסט שהיה שם קודם.

כדי לפשט, בקובץ io.h שאתם מקבלים מאיתנו, קיימות פונקציות Screen::resetLocation() ו־Screen::setLocation() שמאפשרות להזיז את הסמן למיקום הרצוי בשיטה הזו.

שימוש במקשי החיצים:

עד היום הכרנו איך לקבל מהמשתמש קלט שמכיל תווי ASCII, כאלה שיש להם ייצוג מודפס על המסך. כמו כן, שיטות הקלט הרגילות (בין אם זו cin, scanf או אפילו getc) דורשות מאתנו להמתין עד שהמשתמש ילחץ על מקש ה־Enter לפני שהן מתחילות לקבל את הקלט. המגבלה הראשונה מונעת מאתנו להשתמש במקשי החיצים בכלל. המגבלה השנייה מונעת מאתנו לקבל את הקלט מהמשתמש (לא משנה באלו מקשים נבחר לשם כך) בלי לחייב אותו ללחוץ Enter בכל צעד, דבר שאיננו סביר במשחק כמו זה.

כדי להשתמש במקשי החיצים, וכדי לקבל את המקש שנלחץ גם בלי המתנה ל־Enter, השתמשו בקובצי io.h ו־io.cpp שאנחנו נותנים לכם. כשתרצו לקבל תנועה מהמסך תקראו לפונקציה Keyboard::getch(). הפונקציה מחזירה int ובאמצעותו תוכלו להחליט מה לעשות עכשיו, לדוגמה

לאן להתקדם וכו'. (כמו שהזכרנו קודם: אין להשתמש במספרים מפורשים, literals, ישירות בקוד hard-coded).

הערה כללית:

הרבה פרטים הוגדרו לעיל במדויק, אולם תמיד, בכל פרויקט מספיק מורכב, יש התנהגויות שלא מוגדרות במדויק על ידי הדרישות. אפשר להתייעץ במקרה הצורך, אבל ההנחייה הכללית היא שעל דברים שלא נאמרו בהגדרות אתם מוזמנים להחליט בעצמכם מה הדרך הטובה יותר לביצוע הפעולה או לפתרון הבעיה (כלומר, טובה יותר מבחינת משחקיות, או סבירה מבחינת משחקיות וקלה מבחינה תכנותית :)).

קובץ ה-README:

יש לכלול קובץ README שיקרא README.doc, README.docx או README.txt (ולא בשם אחר). הקובץ יכול להיכתב בעברית ובלבד שיכיל את הסעיפים הנדרשים.

קובץ זה יכיל לכל הפחות:

1. כותרת.
2. פרטי הסטודנט: שם מלא כפי שהוא מופיע ברשימות המכללה, ת"ז.
3. הסבר כללי של התרגיל.
4. **תיכון (design): הסבר קצר מהם האובייקטים השונים בתוכנית, מה התפקיד של כל אחד מהם וחלוקת האחריות ביניהם ואיך מתבצעת האינטראקציה בין האובייקטים השונים.**
5. רשימה של הקבצים שנוצרו ע"י הסטודנט, עם הסבר קצר (לרוב לא יותר משורה או שתיים) לגבי תפקיד הקובץ.
6. מבני נתונים עיקריים ותפקידיהם.
7. אלגוריתמים הראויים לציון.
8. באגים ידועים.
9. הערות אחרות.

יש לתמצת ככל שניתן אך לא לוותר על אף חלק. אם אין מה להגיד בנושא מסוים יש להשאיר את הכותרת ומתחתיו פסקה ריקה. נכתוב ב־README כל דבר שרצוי שהבודק ידע כשהוא בודק את התרגיל.

אופן ההגשה:

הקובץ להגשה: יש לדחוס כל קובץ הקשור לתרגיל, למעט מה שיצוין להלן, לקובץ ששמו exN_firstname_lastname.zip, כאשר N הוא מספר התרגיל ו־firstname_lastname הוא השם המלא (לדוגמא אלברט איינשטיין יגיש כך את התרגיל הראשון: ex1_albert_einstein.zip). במקרה

של הגשה בזוג, שם הקובץ יהיה לפי התבנית exN_firstname1_lastname1_firstname2_lastname2.zip, עם שמות המגישים בהתאמה (ללא רווחים; כלומר, גם בשמות עצמם יש להחליף רווחים בקו תחתי, כפי המודגם לעיל). כמו כן, במקרה של הגשה בזוג, **רק אחד** מהמגישים יגיש את הקובץ ולא שניהם.

לפני דחיסת תיקיית הפרויקט שלכם יש למחוק את הפריטים הבאים:

- תיקייה בשם out, אם קיימת
- תיקייה בשם vs.

שתי התיקיות האלה נמצאות בתיקייה הראשית (זו שאנחנו פותחים בעזרת VS). התיקייה vs. לפעמים מוסתרת, אבל אם תפתחו את קובץ ה־zip שיצרתם, בוודאי תוכלו למצוא אותה ולמחוק אותה.

ככלל אצבע, אם קובץ ה־zip שוקל יותר ממ"ב אחד או שניים, כנראה שלא מחקתם חלק מהקבצים הבינאריים המוזכרים.

וודאו כי קובץ ה־zip מכיל תיקייה ראשית אחת, ורק בתוכה יהיו כל הקבצים ותתי התיקיות של הפרויקט.

את הקובץ יש להעלות ל־Moodle של הקורס למשימה המתאימה.

הגשה חוזרת: אם מסיבה כלשהי סטודנט מחליט להגיש הגשה חוזרת יש לוודא ששם הקובץ זהה לחלוטין לשם הקובץ המקורי. אחרת, אין הבודק אחראי לבדוק את הקובץ האחרון שיוגש.

כל שינוי ממה שמוגדר פה לגבי צורת ההגשה ומבנה ה־README עלול לגרור הורדת נקודות בציון.

מספר הערות:

1. נשים לב לשם הקובץ שאכן יכלול את שמות המגישים.

2. נשים לב לשלוח את תיקיית הפרוייקט כולה, לא רק את קובצי הקוד שהוספנו. תרגיל שלא יכלול את כל הקבצים הנדרשים, לא יתקבל וידרוש הגשה חוזרת (עם כללי האיחור הרגילים).

המלצה כללית: אחרי הכנת הקובץ להגשה, נעתיק אותו לתיקייה חדשה, נחלץ את הקבצים שבתוכו ונבדוק אם ניתן לפתוח את התיקייה הזו ולקמפל את הקוד. הרבה טעויות של שכחת קבצים יכולות להימנע על ידי בדיקה כזו.

<u>בהצלחה!</u>