

מכללת הדסה, החוג למדעי המחשב

מבוא לתכנות מונחה עצמים והנדסת תוכנה

סמסטר א', תשפ"א

פרויקט סוף סמסטר

תאריכי הגשה: ראו לוח זמנים בהמשך.

מטרת התרגיל:

בתרגיל זה נממש שוב את המשחק Lode Runner שמימשנו בתרגיל 2. ההבדלים המרכזיים הם שהפעם נשתמש בירושה ופולימורפיזם וכן נשתמש בספרייה הגרפית SFML.

מאז תרגיל 2 למדנו נושאים חדשים והתקדמנו באופן כללי בתכנות מונחה עצמים, כעת נוכל לעצב תוכן טוב יותר, יפה יותר ויעיל יותר של האפליקציה בהתאמה לעקרונות תכנות מונחה עצמים.

כמו כן, נוסיף תכונות נוספות לעומת מה שעשינו בתרגיל 2.

שינויים והוספות:

רוב פרטי המשחק נשארים בדיוק כפי שהיו בתרגיל 2, למעט מה שכבר הוזכר לעיל, שימוש בירושה, פולימורפיזם וממשק גרפי, וכן השינויים וההוספות המפורטים להלן.

משחק מונחה שעון:

המשחק כעת יהיה "בזמן אמת", מונחה שעון, ולא מונחה תורים. ראו פירוט בהמשך.

התנגשות בקירות או בין דמויות:

הגדרת התנגשות היא כאשר יש נקודה חופפת בין המלבנים החוסמים הרלוונטיים, כלומר המלבן החוסם את המכשול והמלבן החוסם את הדמות או המלבנים החוסמים של שתי הדמויות המתנגשות (שחקן ואויב).

הגבלות ו"בונסים" לשחקן:

בכל שלב יכולה להיות מוגדרת הגבלת זמן. אין חובה שבכל שלב תהיה ההגבלה הזו, אפשר ליצור שלבים שבכלל לא יהיו מוגבלים כך (למשל, אם רוצים שלבים ראשונים קלים ל"תרגול"), אבל יש חובה לתמוך באפשרות הזו.

אם עבר הזמן המוגדר, זה נחשב לפסילה של השחקן (יורדים "חיים") והשלב מתחיל מחדש (כלומר השלב כולו מתחיל מחדש, כולל המטבעות ש"נאכלו").

עוד תוספת היא "מתנות" המפוזרות בשלב. השחקן אוסף את המתנה על ידי "דריכה" עליה. סוגי המתנות שאתם נדרשים לממש הם:

1. הגדלת הזמן המוקצב (בהנחה שיש בשלב הזה מגבלה על הזמן).

2. הוספת ניקוד.

3. הוספת חיים.

4. הכנסת אויב נוסף למשחק ("מתנה" רעה).

לבחירתכם האם התוספות קבועות מראש או רנדומליות. אויבים יכולים לעבור במקום שיש בו מתנה, אבל הם כמובן לא יכולים לקחת אותה.

יכולת חפירה:

השחקן יכול לחפור בור ברצפה חשופה מימין או משמאל (באמצעות המקשים: x לבור ימני, z לשמאלי). הבור שנחפר הוא לכמה שניות בלבד. אחרי שהן חולפות הבור נסגר ומי שהיה בתוכו משנה את מיקומו לתא אחד מעל הבור. ההחלטה אם ניתן ללכת או לא על מי שנמצא בבור נתונה לשיקולכם.

מומלץ לממש את יכולת החפירה לאחר שהיכולות הבסיסיות יותר כבר עובדות היטב.

שלי המשחק:

עליכם לספק לפחות ארבעה שלבים (קל יותר עכשיו שיש לנו עורך גרפי:).

תזכורת ועדכונים לפורמט הקובץ:

בשורה הראשונה מופיעים שלושה מספרים: מספר השורות בשלב ומספר העמודות, כפי שהיה בתרגיל 4, וכעת גם הגבלת הזמן לשלב (בשניות). ניתן להשתמש במספר 1- כדי לסמן שהגבלת הזמן לא פעילה.

אחריה מופיעות שורות כמספר השורות שהוגדר עם הסימונים: # (קיר/רצפה), @ (שחקן), % (אויב), - (מוט), H (סולם), * (מטבע) ורווח למשבצות הריקות. בנוסף, יכול להופיע הסימון + ("מתנה").

נשאר לבחירתכם האם הפעם כל השלבים יופיעו בקובץ אחד (כמו בתרגיל 2) או שכל שלב מופיע בקובץ נפרד. בכל מקרה, נדרש שתהיה דרך להוסיף שלב חדש בלי צורך להוסיף את שם הקובץ החדש לקוד (למשל, שימוש בתבנית מסוימת לשמות הקבצים, כך שהתוכנית יודעת אוטומטית מה שם הקובץ הבא שהיא צריכה לנסות, או שימוש בקובץ שממנו קוראים את רשימת שמות קובצי השלבים).

באופן כללי, ניתן להניח שהקובץ תקין.

ניקוד:

הניקוד נותר כפי שהיה, על כל מטבע – 2 נקודות כפול מספר השלב, על סיום שלב – 50 נקודות כפול מספר השלב.

סוגי אויבים:

יהיו שלושה סוגי אויבים שיבדלו זה מזה באופן התנועה שלהם: תנועה רנדומלית, תנועה קבועה (למשל ימינה עד הסוף ואז שמאלה עד הסוף, הלך ושוב) או תנועה לעבר השחקן. הבחירה של סוג האויב נעשית בצורה אקראית ביצירת האויב בתחילת השלב (כזכור, את המידע על מיקום האויבים ומספרם אנחנו מקבלים בקריאה מהקובץ).

שימוש בתפריט:

עליכם לאפשר תפריט בסיסי של בחירת משחק חדש או יציאה. התפריט יוצג בתחילת ריצת התכנית או אחרי שהמשחק נגמר (כי השחקן סיים את כל השלבים בהצלחה או כי נגמרו לו החיים). לבחירתכם האם התפריט יופעל בעזרת העכבר או המקלדת (או שניהם). אם הוא מופעל בעזרת המקלדת ראוי לתמוך בבחירת אפשרות מהתפריט על ידי לחיצה על אות מתוך הכיתוב המופיע על הכפתור. במקרה כזה, נדגיש את האות על ידי קו תחתי (כמו בדוגמה כאן):

File Edit View Window Help

תצוגת מידע:

אתם נדרשים להציג עבור השחקן את מספר הנקודות שצבר, את מספר הפסילות שנשארו לו ואת הזמן שנותר לו, וכן להציג את מספר השלב הנוכחי.

תזוזה:

למרות שבשימוש במסך גרפי אנחנו יכולים להזיז את הדמויות גם באלכסון, לכאורה, בכל זאת נשאיר את התזוזות רק בקווים ישרים לאורך הצירים, ולכן גם המקשים של השחקן נשארים כפי שהם (חיצים לתזוזה לאחד מארבעת הצדדים בהתאמה).

הלחיצה על המקשים נשארת כשהייתה (רק שהפעם מזהים זאת בעזרת אירועים של SFML במקום בעזרת getch). ההבדל הוא שעכשיו אנחנו לא מתנהלים לפי תור אלא לפי השעון. ממילא, צריך לשים לב שקצב התזוזה לא יסתמך על קצב קבלת האירועים של key pressed, שאיננו אחיד, אלא על משך הלחיצה ועל השעון. הדרך לעשות זאת היא כך שהלחיצה רק קובעת האם השחקן יזוז ולא יזוז, ואילו כיוון הוא יזוז, כאשר נרצה להזיז אותו. התזוזה בפועל מתבצעת לפי הזמן שחולף. השחקן נעצר (גם בלי להיתקע במכשול) ברגע שהשחקן הפסיק ללחוץ על המקש (אירוע של key released).

מהירויות תזוזה:

עליכם להחליט כיצד לעדכן את מיקום הדמויות. מכיוון שהגדרנו שהמשחק מונחה שעון, הרי שתצטרכו לבצע זאת כל פרק זמן מסוים ותצטרכו לקבוע בכמה הדמות זזה בכל פרק זמן כזה, לפי הכיוון שאליה היא זזה כרגע. האם עדכון כל עשירית שנייה יהיה מוצלח? אולי פחות? אולי יותר? האם תזוזה של פיקסל או שניים סבירה? האם, כדי לתת די אתגר, האויבים צריכים להיות מהירים יותר (כלומר, יותר פיקסלים בכל איטרציה) מהשחקן? עד כמה מהירים? לא נקבע כללים בנושא, אלא נשאיר זאת עבורכם לנסות ערכים שונים ולהחליט איך ליצור משחקיות טובה. בכל מקרה, תעדו את החלטותיכם בקובץ ה-Readme.

עיצוב האובייקטים:

לגרפיקה מוצלחת כנראה שתמצאו את האובייקטים בתוכנה חיצונית או למצוא תמונות מתאימות ולטעון את התמונות בעזרת `sf::Texture`. במצב כזה, כדאי שהתמונה תהיה עם רקע שקוף (ניתן לעשות זאת בתמונות ששמורות כ-png, למשל), כדי למנוע רקע לבן שיתווסף לדמות. זה חשוב בעיקר אם מדובר בדמות שאין לכם צורה פשוטה שתכיל אותה ואתם צריכים להכיל אותה בתוך `sf::RectangleShape` או `sf::Sprite` שחוסם אותה. כמו כן, כדאי לדאוג שהדמות תתפוס ככל האפשר כמה שיותר מהמלבן החוסם, כדי שיהיה ברור יותר בשעת המשחק מתי נחשב שדמויות התנגשו זו בזו או במכשול.

סאונד:

אתם נדרשים להוסיף למשחק סאונד.

הערות:

- בניגוד לתרגיל 4, הפעם אין איסור להחזיק במחלקות משתנים מסוג הצורות של SFML. למעשה, אולי אפילו עדיף להחזיק את הצורות מוכנות כבר, וכמו כן לחשוב אלו מאפיינים (למשל, מיקום) אין צורך להחזיק בנפרד כי הם כבר נמצאים במאפייני הצורה.
- למעקב אחרי הזמן החולף – המחלקה הרלוונטית היא `sf::Clock`. אפשרות אחרת היא להשתמש בספריית `std::chrono` שמשולבת כחלק מהספרייה הסטנדרטית מאז C++11. שימו לב שאתם צריכים לבדוק בעצמכם כמה זמן עבר (בעזרת הפונקציה `getElapsedTime()` במקרה של `sf::Clock`, או באמצעים מתאימים במקרה של שימוש בספרייה הסטנדרטית). מן הסתם המקום המתאים הוא בלולאת האירועים או בסמוך לה. (ואגב לולאת האירועים, כדי להתנהל לפי השעון חייבים להשתמש הפעם ב-`pollEvent()` ולא ב-`waitEvent()`).
- הקפידו על תיכון נכון, מונחה עצמים, תוך שימוש ראוי בירושה ובפולימורפיזם. שימו לב, לא מספיק רק להשתמש בירושה. אתם נדרשים להשתמש בפולימורפיזם כראוי, ולהקפיד על חלוקת אחריות נכונה בין המחלקות.
- כרגיל, אתם מתבקשים לתעד בקובץ ה-`Readme` תוספות שעשיתם או החלטות לגבי דברים שלא מוגדרים בפירוש.
- כפי שהוזכר כבר בתרגיל 2, ונכון עוד יותר עבור הפרויקט, 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 :).
- אתם רשאים כמובן להוסיף על הרמה הבסיסית, הן מבחינה גרפית, הן תוספות לשיפור המשחקיות (סוגים נוספים של "מתנות", אויבים חכמים יותר וכדו') וכן בשימוש באלגוריתמים מתוחכמים או בתבניות עיצוב וכלי תכנות מתקדמים. בכל מקרה, אם הוספתם - כדאי לציין זאת בקובץ ה-`Readme`.

הגשת תיכון:

הגשת התיכון נעשית בהגשת דיאגרמת UML (מסוג class diagram) המציגה את המחלקות המתוכננות עם מאפיינים (שדות) עיקריים ופונקציות עיקריות שלהן ועם הקשרים בין המחלקות (לפחות ירושה, אבל עדיף גם הכלה). בנוסף, יוגש הסבר מילולי קצר שיסייע בהבנת הדיאגרמה וחלוקת האחריות בין המחלקות. ההסבר המילולי יכול להסבר קצר על כל מחלקה ואם יש צורך – גם על הקשר בין מחלקות שונות.

הקפידו על שמות משמעותיים גם למחלקות, גם לפונקציות המחלקה וגם למשתנים, כך שיהיה ברור מקריאת התיכון מה התפקיד של כל דבר.

חשוב להבין: בלוח הזמנים מוקדש זמן יחסית קצר עבור הגשת התיכון, מכיוון שמטרתו העיקרית היא לגרום לכם לתכנן לפני שתיגשו לממש את הפרויקט. ההערכה היא שלא תידרשנה לשם כך יותר ממספר שעות במקרה הגרוע ביותר. אתם לא צריכים לתכנן כאן את הקוד שלכם לפרטי פרטים, אלא רק לשבת ולחשוב איך נראה לכם כרגע שכדאי לממש את הדברים מבחינת חלוקת פונקציונליות בין המחלקות, מבחינת עצי ירושה וכדומה.

התיכון צריך להיות סביר והגיוני, אבל לא יקרה שום אסון אם במהלך כתיבת הפרויקט בפועל תגלו שאתם רוצים לשנות את חלוקת המחלקות שלכם וכדומה. זה לא אומר שזה בסדר אם התיכון יהיה מרושל מלכתחילה; זה כן אומר שלא צריך להילחץ אתו יותר מדי.

קיימות מספר שיטות מומלצות כדי לשרטט את הדיאגרמה. הסברים מפורטים נמצא באתר בקובץ ההנחיות ליצירת UML Class Diagram.

ההסבר המילולי יהיה בקובץ Word. זכרו, עדיף קובץ בעברית ברורה על פני קובץ באנגלית לא ברורה!

הגשת התיכון תתבצע כרגיל, כשהקבצים שאתם מגישים מכווצים לקובץ zip, ושם הקובץ, כרגיל, כולל את שמות המגישים אלא שהפעם במקום שם התרגיל תופיע המילה design, כלומר תבנית שם הקובץ היא Design_firstname_lastname.zip, עבור מגיש בודד, כאשר firstname_lastname הוא השם המלא (לדוגמה אלברט איינשטיין יגיש את התיכון כך: Design_albert_einstein.zip). במקרה של הגשה בזוג, שם הקובץ יהיה לפי התבנית Design_firstname1_lastname1_firstname2_lastname2.zip, עם שמות המגישים בהתאמה (ללא רווחים; כלומר, גם בשמות עצמם יש להחליף רווחים בקו תחתית, כפי המודגם לעיל). כמו כן, במקרה של הגשה בזוג, רק אחד מהמגישים יגיש את הקובץ ולא שניהם.

הצגת השלד:

כדי לגרום לכם להתקדם עם הפרויקט ולנצל את הזמן ולא לדחות הכול לסוף, כמו שעלול לקרות לרבים מאתנו, אתם נדרשים להציג התקדמות אחרי כשבועיים מהגשת התיכון. הצגת השלד תיעשה בזמן הסדנה. כל צוות ישב עם המתרגל/ת במשך 5-10 דקות ויציג את ההתקדמות הנוכחית והניקוד לשלב הזה ייקבע בהתאם.

אנחנו מודעים לכך שזה לא הרבה זמן אחרי תחילת הכתיבה, אבל כן נצפה לראות שלד. העדיפות היא אם ניתן להציג חלק מסוים שעובד כבר, אבל בכל מקרה נצפה לראות התקדמות מסוג כלשהו ולא רק את ההצהרות על המחלקות שבתכנון.

הצגת הפרויקט:

בזמן שייקבע, כל צוות בתורו יציג את הפרויקט בפני סגל הקורס (המרצה והמתרגל/ת הרלוונטיים). בשלב של הצגת הפרויקט, הוא אמור להיות גמור או לפחות כמעט גמור.

הגשת הפרויקט:

את הפרויקט מגישים באופן דומה לשאר התרגילים (רק עם השם Project במקום exN, כמפורט למטה). כמו תמיד, הנחיות הגשה מפורטות נמצאות בסוף הקובץ הזה.

לוח זמנים:

(בסוגריים מצוין פרק הזמן מאז נקודת הציון הקודמת.)

קמפוס הנביאים:

- תחילת הפרויקט – יום שני, 21/12/2020
- הגשת תיכון (תוך שלושה "ימי עבודה") – יום רביעי, 23/12/2020 (עד שעה 23:59)
- הצגת השלד (אחרי שבועיים) – יום חמישי, 7/1/2021, בזמן (התרגול ו)הסדנה
- הצגת הפרויקט (אחרי כשבועיים) – יום רביעי, 20/1/2021, ויום חמישי, 21/1/2021
- הגשת הפרויקט (באותו היום) – יום חמישי, 21/1/2021 (עד שעה 23:59)

ניקוד:

- הגשת התיכון – 5%
- הצגת השלד – 10%
- הצגת הפרויקט – 5%
- הגשת הפרויקט – 80%

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. **פורמט קובץ השלב, ופירוט אם יש צורך בצעדים נוספים להוספת שלב (למשל, הוספת שם הקובץ החדש לקובץ עם רשימת הקבצים).**
 6. רשימה של הקבצים שנוצרו על ידכם, עם הסבר קצר (לרוב לא יותר משורה או שתיים) לגבי תפקיד הקובץ.
 7. מבני נתונים עיקריים ותפקידיהם.
 8. אלגוריתמים הראויים לציון.
 9. באגים ידועים.
 10. הערות אחרות.
- יש לתמצת ככל שניתן אך לא לוותר על אף חלק. אם אין מה להגיד בנושא מסוים יש להשאיר את הכותרת ומתחתיה פסקה ריקה. תכתבו ב-README כל דבר שרצוי שהבודק ידע כשהוא בודק את התרגיל.

אופן ההגשה:

הקובץ להגשה: יש לדחוס כל קובץ הקשור לפרויקט, למעט מה שיצוין להלן, לקובץ ששמו `Project_firstname_lastname.zip`, כאשר `firstname_lastname` הוא השם המלא (לדוגמא אלברט איינשטיין יגיש את הפרויקט כך: `Project_albert_einstein.zip`). במקרה של הגשה בזוג, שם הקובץ יהיה לפי התבנית `Project_firstname1_lastname1_firstname2_lastname2.zip`, עם שמות המגישים בהתאמה (ללא רווחים; כלומר, גם בשמות עצמם יש להחליף רווחים בקו תחת, 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. שימו לב לשם הקובץ שאכן יכלול את שמות המגישים.

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

המלצה כללית: אחרי שהכנתם את הקובץ להגשה, העתיקו אותו לתיקייה חדשה, חלצו את הקבצים שבתוכו ובדקו אם אתם מצליחים לפתוח את התיקייה הזו ולקמפל את הקוד. הרבה טעויות של שכחת קבצים יכולות להימנע על ידי בדיקה כזו.

בהצלחה!