**חלק רטוב**

**תיעוד חיצוני:**

**מטרת הקוד:**

התוכנית מסמלצת משחק אינטרקטיבי של Lunar Lockout עם המשתמש. התכנית בונה לוח לפי נתונים המתקבלים מהמשתמש אותו המשתמש צריך לפתור במס' מהלכים וזמן מוגבלים (גם הם ניתנים ע"י המשתמש), כאשר התוכנית מקבלת מהמשתמש את המהלך (בפורמט של קאורדינטה בלוח+כיוון) ומדפיסה בכל תור את הלוח המעודכן, את המהלכים שבוצעו עד כה ואת מס' המהלכים שנותרו. במקרה של ניצחון, הפסד או שגיאה (כפורמט בחוקי המשחק להלן) תודפס הודעה מתאימה.

**חוקי המשחק:**

לוח המשחק בנוי ממשבצות כאשר כל משבצת יכולה או להיות ריקה, או להכיל מחסום, רובוט, או אסטרונאוט. מטרת המשחק היא להביא את האסטרונאוט אל נקודת הסיום שהוגדרה. הכלים שניתנים להזזה הם הרובוטים והאסטרונאוט. ניתן להזיז כלי בכיוון מסוים רק אם יש כלי באותו הכיוון שאפשר "להיתקע" בו, כלומר חוסם את דרכו אך לא אם תנועה זו משאירה את הכלי במקומו. המשחק מסתיים בניצחון כאשר האסטרונאוט נעצר בנקודת הסיום. המשחק מסתיים בכישלון אם הלוח לא נפתר בתום מס' המהלכים או הזמן שהוקצבו.

**שיטת עבודת התוכנית:**

1. התוכנית תדפיס בקשות לקלט הנ"ל מהמשתמש אחת-אחרי-השנייה שיתקבל ע"י לחיצה על המקש 'Enter', אותן היא תסרוק.
2. התוכנית תדפיס מסך הכנה עם הגבלות המשחק כפי שניתנו ע"י השתמש ותחכה ללחיצה על המקש 'Enter' כדי להתחיל את המשחק ולהריץ את השעון.
3. כל תור התוכנית תדפיס את מצב הלוח הנוכחי ותבדוק האם הלוח נפתר. אם כן- היא תדפיס את המהלכים ששוחקו עד כה ואת ההודעה "Well done, you won!" ותסתיים. אם לא, היא תדפיס את המהלכים ששוחקו עד כה ואת מס' המהלכים שנותרו ותבקש מהמשתמש להקליד את המהלך הבא שיתקבל ע"י לחיצה על המקש 'Enter', אותו היא תסרוק. אם במהלך ההמתנה לקלט תם הזמן המוקצב- התוכנית תדפיס "Time's up. Game Over!" ותסתיים.
4. במידה והמהלך היה לא חוקי לפי החוקים הנ"ל- תודפס השגיאה: Illegal move. Try again"". במקרה זה המהלך ייספר והמשתמש יתבקש להכניס מהלך נוסף.
5. במידה והמהלך היה חוקי- התוכנית תבצע את המהלך ותעבור לתור הבא- כלומר, תבצע שוב את השלבים החל משלב 2.

**משתנים ותפק;ידם :**

להלן רשימת המשתנים שהוגדרו בקוד ותפקידם:

duration: .blkw 2 –Time limit as given by the user, in int or string representation

num\_moves: .blkw 1 –Limit on number of moves as given by the user, in int or string representation

print\_pointer: .blkw 1 – Pointer for printing a string in subroutines printf and printc

print\_busy: .word 0 – Flag marking the printer is currently busy used in printf and printc

scan\_pointer: .blkw 1 - Pointer for scanning input to a string in subroutines scanf and inp

done\_scanning: .word 0– Flag marking the program has finished scanning input used in printf and printc

times\_up: .byte 0 – Flag marking the given time has elapsed

nRows: .byte 0 – Number of rows in Board, as given by the user

nCols: .byte 0 – Number of columns in Board, as given by the user

Board: .blkw 50 – The game board, as given by the user

Finish: .blkw 1 – The address of the finish square in Board

square: .blkw 1 – Content of 1 byte in Board, used in subroutine printBoard

Moves: .blkw 100 – Array of bytes holding all moves that have been used so far in the game

counter: .blkw 1 – Count number of clock interrupts, used in subroutine clock

row: .byte 1 – holds the row number of a , used in subroutine movePiece

col: .byte 1 – holds the column number of a piece, used in subroutine movePiece

direction: .byte 1 - holds the direction of the current move, used in subroutine movePiece

Move: .blkw 3 – Holds current move as given by the user

rate: .word 1000 – Number of clock interrupts per second

temp: .blkw 1 – Used to store temp data in various subroutines

drop\_line: .byte 10., 13., 0 – String representation of 'Enter', used in printf

string1:.ascii <Please enter number of rows:> – Used in printf

string2:.ascii <Please enter number of columns:> – Used in printf

string3:.ascii <Please enter game duration in seconds and max number of moves:> – Used in printf

string4:.ascii <Please enter the board:> – Used in printf

string5:.ascii <We are all set. Total time: > – Used in printf

string6:.ascii < sec> – Used in printf

string7:.ascii <Max number of allowed moves: > – Used in printf

string8:.ascii < moves> – Used in printf

string9:.ascii <Press Enter key when you are ready…> – Used in printf

str10:.ascii <Current Board:> – Used in printf

str11:.ascii <Moves:> – Used in printf

str12:.ascii < moves left, please enter your move:> – Used in printf

str13:.ascii <Illegal move. Try again> – Used in printf

str14:.ascii <No more moves allowed. Game Over!> – Used in printf

str15:.ascii <Final Moves:> – Used in printf

str16:.ascii <Well done, you won!> – Used in printf

str17:.ascii <Time's up. Game Over!> – Used in printf

שגרות התוכנית:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| שם השגרה | תפקיד השגרה | אוגר קישור | פרמטרים | אופן הפעולה |
| setupGame | ביצוע כל ההכנות למשחק (עד סעיף 6) | pc | * str1-9- בשטח הגלובלי * buffer- בשטח הגלובלי * nRows- בשטח הגלובלי * nCols- בשטח הגלובלי * Board- בשטח הגלובלי * duration- בשטח הגלובלי * num\_moves- בשטח הגלובלי | 1. הדפסת ההוראות ע"י printf וה-str המתאימה 2. קריאת הקלט ע"י scanf כמחרוזות דרך buffer לnRows, nCols- וBoard- 3. קריאת הקלט ע"י scanf כמחרוזת דרך buffer ל-duration ו-num\_moves בעזרת initDurAndNumMoves 4. הדפסת מסך ה-ready (סעיף 5) ע"י printf ובעזרת המשתנים המאותחלים 5. עיבוד המשתנים להמשך עבודה ע"י handleInput |
| initDurAndNumMoves | אתחול duration ו-num\_moves | pc | * buffer- בשטח הגלובלי * duration- בשטח הגלובלי * num\_moves- בשטח הגלובלי | לולאה עם איטרטור על buffer:   1. מעתיקים ל-duration עד שמגיעים לרווח 2. כשמגיעים לרווח- מדלגים עליו 3. מעתיקים ל-num\_moves עד שמגיעים לסוף buffer |
| handleInput | המרת משתנים מ-string ל-int ועיבוד Board | pc | * nRows- בשטח הגלובלי * nCols- בשטח הגלובלי * duration- בשטח הגלובלי * num\_moves- בשטח הגלובלי * Board- בשטח הגלובלי * Finish- בשטח הגלובלי | 1. המרת nRows מ-string ל-int 2. המרת nCols מ-string ל-int 3. המרת duration מ-string ל-int 4. המרת num\_moves מ-string ל-int 5. החלפת 'O' ב-Board ב'\*' 6. שמירת הכתובת של הבית ב-Board שבו נמצא 'F' ב-Finish |
| play | ניהול המשחק האינטרקטיבי | pc | * Finish- בשטח הגלובלי * num\_moves- בשטח הגלובלי * str13-17- בשטח הגלובלי * Moves- בשטח הגלובלי | לולאה על num\_moves שקטן ב-1 בכל איטרציה:   1. הדפסת הלוח הנוכחי ע"י printBoard 2. בדיקה האם האסטרונאוט נמצא במשבצת הסיום ע"י השוואת תוכן Finish ל-A. אם כן- הדפסת המהלכים עד כה והודעת ניצחון ויציאה מהשגרה. אם לא:    1. הדפסת המהלכים עד כה ובקשה למהלך הבא ע"י printMoves    2. קליטת הקלט ע"י scanMove    3. בדיקה אם הזמן תם תוך כדי scanMove. אם כן- הדפסת הודעת הפסד מתאימה ע"י printf ויציאה מהשגרה. אם לא:       1. בדיקת חוקיות המהלך ע"י movePiece.          1. מהלך לא-חוקי:             1. הדפסת הודעת שגיאה ע"י printf             2. חיסור 1 מ-num\_moves             3. בדיקה אם תמו המהלכים ע"י השוואת num\_moves ל-0. אם כן- הדפסת הודעת הפסד מתאימה ע"י printf ויציאה מהשגרה. אם לא: חזור ל-2.2          2. מהלך חוקי:             1. ביצוע המהלך ע"י executeMove             2. חיסור 1 מ-num\_moves             3. בדיקה אם תמו המהלכים ע"י השוואת num\_moves ל-0: אם כן- הדפסת הודעת הפסד מתאימה ע"י printf ויציאה מהשגרה. אם לא: חזור ל-1. |
| printBoard | הדפסת Board | pc | * str10- בשטח הגלובלי * Board- בשטח הגלובלי * nRows- בשטח הגלובלי * nCols- בשטח הגלובלי * square- בשטח הגלובלי | 1. הדפסת כותרת ע"י printf 2. לולאה על Board בעזרת איטרטורים על nRows ו-nCols:  * העתקת הבית ל-square והדפסתו ע"י printf והגדלת האיטרטור על מס' העמ' * כשהאיטרטור על מס' העמ' שווה ל-nCols- רד שורה והגדל את האיטרטור על מס' השורות * כשהאיטרטור על מס' השורות שווה ל-nCols- צא מהשגרה |
| printMoves | הדפסת המהלכים עד כה והבקשה למהלך הבא | pc | * str11-12- בשטח הגלובלי * Moves- בשטח הגלובלי * num\_moves- בשטח הגלובלי | 1. הדפסת המהלכים עד כה ע"י printf ו-Moves 2. המרת num\_moves מ-int ל-string בעזרת int2string 3. הדפסת num\_moves ע"י printf 4. המרת num\_moves בחזרה מ-string ל-int בעזרת string2int 5. הדפסת הבקשה למהלך הבא ע"י printf |
| scanMove | סריקת קלט המהלך מהמשתמש | pc | * times\_up- בשטח הגלובלי * Move- בשטח הגלובלי * coord בשטח הגלובלי | 1. סריקת הקלט כ-string ל-Move ע"י scanf 2. אתחול coord בערכי Move, כשערכי row ו-col מומרים מ-string ל-int ע"י string2int |
| movePiece | מקבלת את הלוח, מימדי הלוח, מיקום על הלוח וכיוון ומחזירה את המיקום של הרובוט/אסטרונאוט אחרי הזזתו בכיוון הנבחר. | r5 | קלט: המיקום על הלוח והכיוון (אחד מתווי האסקי 'L, 'R, 'U, 'D) מועברים inline בסדר הנתון (המיקום מועבר ראשון). הלוח מועבר בשטח המשותף Board וממדי הלוח מועברים בשטח המשותף nCols ו nRows.  פלט: מיקום האסטרונאוט/רובוט אחרי הזזתו יוחזר במחסנית. במידה והצעד לא חוקי יוחזר המיקום (-1,-1). | 1. קריאת את המשתנים שהועברו inline לתוך התוויות row, col, direction. 2. בדיקת תקינות הנתונים שהוכנסו (שורה ועמודה) והחזרת שגיאה אם אחד מהם מחוץ לתחומי הלוח. 3. אתחול את r1 כך שיצביע על המשבצת שנבחרה במהלך ונבדוק באיזה כיוון המהלך. כעת השגרה פועלת לפי הכיוון, אך כיוון שבכל כיוון קורה כמעט בדיוק אותו הדבר, נסביר רק את אחד הכיוונים, בהנחה ימין:   הרג'יסטר r4 ישמש כמצביע שמחפש את המשבצת שאליה יזוז הכלי אם המהלך חוקי, ולכן נעביר אליו את הכתובת של המשבצת הנוכחית מ r1. ע"י חישוב, נגרום ל r3 להצביע על המשבצת שבתחילת השורה הבאה. כעת מתחילה לולאה בה כל פעם מזיזים את r4 משבצת אחת בכיוון הנבחר, בודקים אם יצאנו מהשורה (השוואה ל r3), ובודקים את תוכן המשבצת. אם הגענו לסוף השורה ולא מצאנו כלי או מחסום בדרך, אז המהלך אינו חוקי. במקרה כזה מחזירים שגיאה (מידע לגבי השגיאה בפלט). אם מצאנו כלי במהלך סריקת השורה, נחזיר את r4 משבצת אחת אחורה, ונקבל את המשבצת אליה יגיע הכלי בסוף המהלך. |
| executeMove | ביצוע המהלך וכל מה שכרוך בכך. | pc | * הכתובת החדשה שהתקבלה מ-movePiece- במחסנית * coord- בשטח הגלובאלי * nCols- בשטח הגלובאלי * Board- בשטח הגלובאלי * Move- בשטח הגלובאלי * Moves- בשטח הגלובאלי | 1. ביצוע ההחלפה ב-Board ע"י שימוש ב-coord ובתוצאה של movePiece במחסנית. 2. הוספת Move למערך ה-Moves והוספת ',' בין הערכים ו'<-' בין המהלכים. |
| string2int | המרת מערך של תווי ascii לערכו המספרי בבסיס אוקטלי | pc | * המערך המומר- במחסנית | כפי שנלמד בתרגול |
| Int2string | המרת מס' מערכו המספרי בבסיס אוקטלי למערך של תווי ascii | pc | * המס' המומר- במחסנית | 1. חלוקת המס' ב-108. 2. המרת המנה והשארית לערכן המספרי 3. בדיקת השארית:    1. השארית שווה לאפס- דריסת הערך במחסנית במנה המומרת    2. השארית שונה מאפס- דריסת הערך במחסנית במנה ובשארית |
| dropLine | הדפסת ירידת שורה | pc | אין | הדפסת הערכים המספריים של 'Enter' ע"י printf |
| Scanf | מקבלת קלט מהמקלדת עד הקשת Enter ומכניסה את המחרוזת שהתקבלה למערך היעד. | pc | קלט: כתובת מערך היעד מועברת במחסנית.  פלט: המחרוזת שהוקלדה ע"י המשתמש פרט ל Enter. סוף המחרוזת יסומן ע"י הערך 0.  הדגל times\_up בשטח משותף יקבל את הערך 1 אם תם הזמן תוך כדי ההמתנה לקלט, אחרת ערכו יהיה 0. | כפי שנלמד בתרגול, בתוספת בדיקה של הדגל times\_up בזמן ההמתנה לפסיקה |
| inp | שגרת טיפול בפסיקת מקלדת |  | קלט: כתובת מערך היעד מועברת במחסנית.  פלט: המחרוזת שהוקלדה ע"י המשתמש פרט ל Enter. סוף המחרוזת יסומן ע"י הערך 0.  הדגל times\_up בשטח משותף יקבל את הערך 1 אם תם הזמן תוך כדי ההמתנה לקלט, אחרת ערכו יהיה 0. | כפי שנלמד בתרגול |
| printf | מקבלת כתובת מחרוזת ומדפיסה את המחרוזת ע"י מנגנון הפסיקות. השיגרה מסתיימת רק כאשר המחרוזת הודפסה במלואה. | pc | קלט: כתובת המחרוזת תועבר במחסנית. סוף המחרוזת יסומן ע"י הערך 0.  פלט: המחרוזת תודפס למסך | כפי שנלמד בתרגול |
| scanc | שגרת הטיפול בפסיקת מדפסת | pc | קלט: כתובת המחרוזת תועבר במחסנית. סוף המחרוזת יסומן ע"י הערך 0.  פלט: המחרוזת תודפס למסך | כפי שנלמד בתרגול |
| clock | שגרת הטיפול בפסיקת שעון | pc | * counter- בשטח הגלובאלי * rate- בשטח הגלובאלי * times\_up- בשטח הגלובאלי * duration- בשטח הגלובאלי | לולאה עם duration:   1. הגדלת counter ב-1 2. בדיקה אם עברה שנייה ע"י השוואת counter ל-rate. אם לא עברה שנייה- יציאה מהשגרה. אם כן:    1. איפוס counter    2. חיסור 1 מ-duration    3. בדיקה אם תם הזמן ע"י השוואת duration ל-0. אם לא- יציאה מהשגרה. אם כן:       1. השבתת IE של Clock       2. הדלקת הדגל times\_up       3. יציאה מהשגרה |