הנדסת תוכנה 094129
תרגיל בית 1

תכנות מודולרי, עבודה עם vector ו- string
תאריך אחרון להגשה: 06\05\2018 עד השעה 23:55
מתרגל אחראי: תאהר מוסא

נושא התרגיל: בניית iRobot מתקדם

תיאור התוכנה:

חברת iRobot החליטה כי היא רוצה לשכור את שירותיכם לפיתוח דור ההמשך לסדרת הרובוטים המוצלחת שלה - irobot boomba. החברה גילתה כי הרבה יותר חסכוני ויעיל לייצר מספר רובוטים קטנים שיבצעו את עבודת הניקיון בו זמנית מאשר רובוט אחד גדול.

החברה פיתחה אבטיפוס ראשוני הכולל מספר חיישנים מתקדמים, מיפוי סביבה חכם ואלגוריתם ניווט מתקדם (מבוסס SLAM). כעת, iRobot מבקשת מכם לפתח ממשק חכם לרובוטים הללו אשר יאפשר גם שליטה מרחוק באופן שיתואר בהמשך.

להלן תיאור המרכיבים של המערכת:

- **מפה**

כפי שצוין לעיל, לרובוט מיפוי סביבה חכם המאפשר לו לזהות היכן בחדר מסוים יש לכלוך והיכן לא, ובכך לייעל את הניווט שלו בסביבה. המיפוי מיוצג באמצעות מפה דו מימדית, כאשר תא עם הערך '0' משמעו נקי ו-'1' מלוכלך.
את יכולות התנועה של boomba נמדל בעזרת שמונה כיוונים: למעלה (U), למטה (D), שמאלה (L), ימינה (R); ולארבעת האלכסונים: למעלה ימינה (UR), למעלה שמאלה (UL), למטה ימינה (DR), למטה שמאלה (DL).

לדוגמא נתונה מפה 7X7 הבאה:

	0	*1*	*2*	*3*	*4*	*5*	*6*
0	0	1	1	1	1	1	1
1	1	0	0	0	1	0	1
2	0	0	1	1	1	0	1
3	1	0	0	0	0	0	1
4	1	1	1	0	1	0	1
5	1	0	0	0	0	0	1
6	1	0	1	1	1	1	1

נניח כפי שתואר לעיל רובוט יחיד בשם "boom" נמצא בתא (2,0). אם יקבל פקודה ימינה (R) ינוע לתא (2,1). אם בהימצאו בתא (2,1) יקבל את הפקודה למעלה ימינה (UR), "boom" ינוע לתא (1,2).
כעת אם בהימצאו בתא (1,2) יקבל את הפקודה למטה (D) יעבור לתא (2,2).
בנוסף, במידה ו- "boom" יקבל את הפקודה "clean" הוא ינקה את התא בו הנמצא. כלומר ערך התא יהפוך ל-0 (לא משנה ערכו הקודם).

המפה מקודדת כמטריצה (מערך דו מימדי) מסוג int בגובה S_HIGHT וברוחב S_WIDTH ובשם gmap.
לדוגמא, המפה למעלה תוגדר כך:

```
#define S_HIGH (7) // define the numbers of rows
#define S_WIDTH (7) // define the numbers of columns


int gmap [S_HIGH][S_WIDTH] =
{
      { 0 , 1 , 1 , 1 , 1 , 1 , 1},
      { 1 , 0 , 0 , 0 , 1 , 0 , 1},
      { 0 , 0 , 1 , 1 , 1 , 0 , 1},
      { 1 , 0 , 0 , 0 , 0 , 0 , 1},
      { 1 , 1 , 1 , 0 , 1 , 0 , 1},
      { 1 , 0 , 0 , 0 , 0 , 0 , 1},
      { 1 , 0 , 1 , 1 , 1 , 1 , 1}
};
```

- אפשר להניח כי בלתי אפשרי לשנות את גודל המפה בזמן ריצה
- יש לאתחל את פתרון התרגיל עם המפה הנ"ל

הקלט לתכנה:

התכנה קולטת את נתוני הפקודות מערוץ הקלט הסטנדרטי (cin). הקלט מחולק לפקודות, וכל פקודת תופסת שורה אחת בדיוק בקלט. סדר ביצוע הפקודות חייב להיות סדר הופעתן בקובץ. המילה הראשונה בפקודה הינה תמיד השם שלה, ולאחר מכן פרמטרים לפקודה, מופרדים ברווחים. הפקודות האפשריות הינן (יש חשיבות לגודל האות):

מס'	תיאור פקודה	מבנה שורת הפקודה	פרמטרים
1	הזז רובוט	Move NNN W	NNN- שם הרובוט בעל שלושה תווים בדיוק. W- כיוון התנועה מתוך הכיוונים: L, R, U, D, UR, UL, DR, DL
2	לכלך	AddDirt X Y	לכלך את המפה בשורה ה-x ובעמודה ה-y (x,y מספרים שלמים אי שליליים)
3	נקה	Clean NNN	NNN- שם הרובוט בעל שלושה תווים בדיוק. נקה את המפה במיקום הנוכחי של הרובוט.
4	הצג רובוט חדש	Place NNN X Y	NNN- שם הרובוט. צור רובוט חדש בשם שלושה תווים בדיוק והצב אותו במיקום ה-(x,y) במפה.
5	מחק רובוט	Delete NNN	NNN- שם הרובוט. מחק רובוט NNN מהמפה.
6	הזזה מרובה	MoveMulti NNN W1, W2 …end	NNN- שם הרובוט. W1, W2 … - כל כיוון כמו פקודה מס' 1. מס' הכיוונים אינו ידוע מראש. **שימו לב** שסוף הפקודה מסומן ע"י המחרוזת הקבועה end

התנהגות הרובוטים:

- לכלוך, או הזזה של רובוט במקום לא מוגדר אינם משנים את מצב התכנית. מחיקת רובוט לא קיים אינה משנה את מצב התכנית.
- בעקבות תקלה בחיישני הרובוט, במידה ורובוט יצא מגבולות המפה הקשר עם הרובוט מופסק לצמיתות. לכן, מאותו הרגע והלאה מיקום הרובוט יהיה בנקודה (-1,-1) והרובוט יפסיק להגיב לפקודות מס' 1 3 ו-6. לאחר שינוי המיקום נדפיס הודעה לגבי מיקומו (-1,-1). אך מרגע איבוד הקשר ועד סוף חיי הרובוט לא יודפסו הודעות לגבי מיקומו.
- ניתן להניח כי הכיוונים יהיו אך ורק אחד מהשמונה הנתונים.
- ניסיון ליצור רובוט חדש בתא לא חוקי (מחוץ למפה) יסתיים ללא יצירת הרובוט.
- ניסיון ללכלך תא בו ישנו רובוט (או רובוטים) יסתיים ללא הצלחה.
- ניסיון חוקי (במקום חוקי - לפי המוזכר לעיל) ליצור רובוט קיים, יזיז את הרובוט למקום החדש, גם אם הרובוט היה מחוץ לגבולות המפה.
- מותר ליצור יותר מרובוט אחד בכל תא ומס' הרובוטים שיבנו אינו ידוע מראש.
- לכל רובוט יש מיכל אבק. מצב מיכל האבק מוגדר להיות בסולם מ-0 עד 5. כאשר 0 משמעו שהמיכל ריק (נקי) ו-5 המיכל מלא (מלוכלך) והרובוט לא יכול יותר לנקות. כל רובוט חדש נוצר עם מיכל ריק.
- כאשר המיכל מתמלא, בהינתן פקודת ניקיון הרובוט לא ינסה לנקות אלא יגיע אל לנקודת הניקיון (0,0) וירוקן את המיכל.
- לכל רובוט יש דירוג. רק ובמידה והרובוט ינקה תא מלוכלך הוא יקבל נקודה.
- שלב הקלט יסתיים כאשר תתקבל הפקודה ctrl+z או סוף הקובץ, במקרה והקלט הופנה מקובץ.

הנחות שמותר לכם להניח לגבי הקלט:

- ניתן להניח כי הקלט מכיל רק פקודות בפורמט הנ"ל.
- ניתן להניח שלא מנסים להסיר רובוט שלא קיים בבסיס הנתונים.
- ניתן להניח שכל המחרוזות המוזכרות לעיל מכילות רק אותיות באנגלית (גדולות או קטנות) ומספרים. ובפרט, שאינן מכילות רווחים.
- בסיום כל פקודת הזזה או פקודת הוספת רובוט מוצלחות יש לקרוא לפקודת ההדפסה:

PrintRobotPlace(robotName, locX, locY)

עם הפרמטרים הרלוונטיים. לצורך העניין, פקודת הזזה מוגדרת כהצלחה כל עוד הרובוט נשאר בגבולות המפה. פקודת הוספה מוגדרת כמוצלחת אם הרובוט נמצא על המפה ומיקומו השתנה לאחר הפקודה.

- בסיום כל פקודת ניקיון מוצלחת יש לקרוא לפקודת ההדפסה: PrintClean(robotName, locX, locY)
- בסיום הקלט יש להדפיס את טבלת הדירוגים (עבור כלל הרובוטים) ע"י פקודת ההדפסה:

PrintTable(robots, ranks, tankCounter)

הפלט של המערכת:

ההדפסות של המערכת יתבצעו בעזרת קריאות לפונקציות המוגדרות בקובץ Printer.h, המצורף לתרגיל זה.
בסופו של דבר , בהינתן המפה למעלה, על המערכת עבור הקלט:

```
Place Rb5 3 3
Move Rb5 L
Move Rb5 U
Clean Rb5
Move Rb5 L
Move Rb5 L
Move Rb5 L
Move Rb5 L
```

וגם עבור הקלט:

```
Place Rb5 3 3
MoveMulti Rb5 L U end
Clean Rb5
MoveMulti Rb5 L L L L end
```

התכנית תדפיס:

```
Robot: Rb5 at: 3,3
Robot: Rb5 at: 3,2
Robot: Rb5 at: 2,2
Robot: Rb5 clean: 2,2
Robot: Rb5 at: 2,1
Robot: Rb5 at: 2,0
Robot: Rb5 at: -1,-1
==========
Rank Table:
Robot: Rb5, rank: 1, dirt-tank: 1
```

(אין רווחים לאחר התו האחרון)

דרישות המימוש:

1. **המימוש חייב להכיל לפחות את הקבצים הבאים:**
   א. main.cpp - כפי שניתן על ידי צוות הקורס בקובץ המצורף.
   אין לשנות קובץ זה ,והמימוש שלכם חייב להשתמש בפונקציית main המוגדרת בו!
   ב. Printer.h - כפי שניתן על ידי צוות הקורס.
   אין לשנות קובץ זה ,וכל ההדפסות שלכם חייבות להתבצע בעזרת הפונקציות המוגדרות בו!
   אסור לכם להדפיס בשום דרך אחרת!
   ג. Interface.h – ממשק השליטה ברובוטים.
   אין לשנות קובץ זה!
   שימו לב: בקובץ Interface.cpp לפני קליטת הקלט מהמשתמש יש לכתוב את הפקודה הבא:
   std::cout << "Start entering commands:" << std::endl;
   בכדי להתריע למשתמש שכעת עליו להזין את הקלט.
   השתמשו בדיוק בשורה זאת על מנת שהתרגיל יעבור בבדיקה.
   ד. RobotDB.h – מכיל ממשק של מודול שמממש את אוסף הרובוטים, ז"א שומר את הנתונים במבנה נתונים פנימי וממשש פעולות עליו.
   ה. Map.h - מכיל ממשק של מודול שמממש את ממשק המפה, ז"א שומר את נתוני ופעולות המפה.
   ו. קבצי מימוש של המודולים הנ"ל (קבצי ה-cpp).
   ז. קבצי קוד אחרים, במידה ותמצאו זאת לנכון.
   ח. **שימו לב:** מצורף קובץ map.cpp ובו הגדרת המפה כפי שראיתם לעיל ומימוש הפונקציות הבאות:
      i. InMapLimit – בודקת בהינתן (x,y) מסוימת אנו בגבולות המפה.
      ii. addDirt – מוסיפה לכלוך למפה במיקום ה- (x,y).
   השתמשו במימוש זה כדוגמא לשאר הפקודות ובפרט לצורת התיעוד המצופה ממכם.
2. חובה לתעד את הקוד באנגלית (אין צורך לדאוג לרמת האנגלית) והתיעוד חייב להופיע בתוך הקוד, לפני הפונקציה\טיפוס\משתנה שהתיעוד מתייחס אליהם. נא תעדו ע"פ ההסברים באתר הבא:
   http://www.edparrish.net/common/cppdoc.html
3. **התיעוד יכיל:**
   א. בראש כל מודול תיאור קצר(כמה שורות) של ייעוד המודול והקשר שלו למודולים האחרים.
   ב. לפני כל פונקציה תיאור בן שתי שורות של ייעוד הפונקציה, וכן תיאור של כ"א מהפרמטרים שלה (שורה אחת לכל פרמטר) וערך ההחזר שלה. תיעוד הפרמטרים\ערך החזר צריך להכיל הסבר לגבי תוכן המשתנה ,לא הטיפוס שלו.
   ג. לפני כל משתנה גלובלי תיאור בשורה של הייעוד שלו .
4. **יש לתת שמות בעלי משמעות לכל משתנה!** למען הסר ספק:
   א. השמות i,j,k,l,m עבור משתני לולאה שהם אינדקסים במערך הם כן בעלי משמעות.
   ב. השם it עבור iterator והשם cit עבור const_iterator הם כן בעלי משמעות ,לצורך מילוי הוראה זו.
   ג. פרט למצוין לעיל ,שם שהוא ראשי תיבות <u>לא</u> נחשב לבעל משמעות.
5. אין בתרגיל זה דרישה של יעילות זמן. בפרט, מבנה הנתונים vector בפירוש אינו היעיל ביותר מבחינה זו.
6. עבודה עם (בפרט מעבר על) מיכלים סדרתיים תיעשה אך ורק בעזרת איטרטור המתאים ביותר.
7. הקדישו תשומת לב ומחשבה לאופן בו הפונקציות שלכם מקבלות פרמטרים .השתמשו בהעברה ע"י references במקום העברה by value במקרים בהם הדבר אפשרי וחוסך העתקות. חובה להשתמש ב - const לפי הנלמד בקורס!

<u>הוראות הגשה:</u>

1. התרגיל להגשה בזוגות **בלבד**.
2. הקוד חייב להיכתב על פי מוסכמות כתיבת הקוד (Coding Conventions) בקורס. קוד שלא עומד במוסכמות לא יזכה במלוא הניקוד.
3. **חובה** לקמפל את התרגיל בעזרת סביבת eclipse. תוכנית אשר תצליח להתקמפל בסביבה אחרת ולא בסביבה זו **<u>תחשב כקוד שלא עובר קומפילציה</u>**.
4. ההגשה חייבת להכיל **קובץ ZIP יחיד בלבד** (ולא קובץ RAR וכדומה) המכיל:
   - תיקיה בשם code ובה כל קבצי **קוד המקור (h/cpp)**, ללא קבצי הרצה.
5. שם הקובץ חייב להיות hw1_xxxxxxxxxx_yyyyyyyyyy.zip, כאשר xxxxxxxxxx ו-yyyyyyyyyy הם מספרים תעודות הזהות של המגישים, כולל ספרת ביקורת
6. ההגשה היא אלקטרונית **בלבד**, דרך אתר ה-moodle של הקורס. תרגילים שיוגשו בכל דרך אחרת **לא ייבדקו**.
7. אין להגיש את אותו הקובץ פעמיים. התרגיל יוגש על ידי <u>אחד</u> מבני הזוג.
8. שימו לב שההגשה תיחסם בדיוק בשעה 23:55. מומלץ להגיש לפחות שעה לפני המועד האחרון.
9. תרגיל בית שלא יוגש על פי הוראות ההגשה - **לא ייבדק**.

בהצלחה!