חומרים שהוכנו על-ידי מורי הניסוי תש"ע

להוראת "יסודות מדעי המחשב"

סביבת קארל, פרק 3 עולם משתנה

מותאם להוראת הפרק הראשון (הכרות עם עצמים)

כתיבה ועריכה:

שירלי רוזנברג-כהן

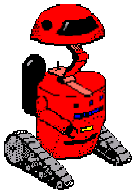
מעובד על פי הספר "מבוא למדעי המחשב – סביבת קארל הרובוט"

שפותח ע"י צוות מגוון במחלקה להוראת הטכנולוגיה והמדעים, הטכניון

דפי העבודה מבוססים על התוכנה Object Karel for Karel++ מהדורת 2005

ניתן להוריד את התוכנה ללא תשלום בכתובת

http://pclc.pace.edu/~bergin/temp/findkarel.html

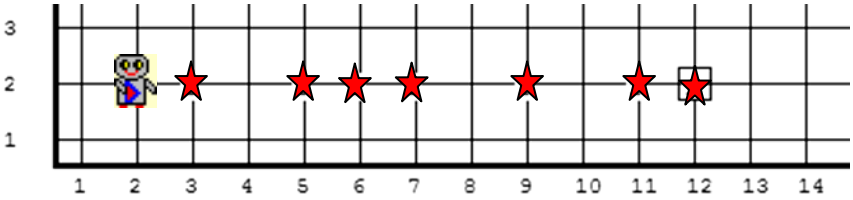


רובוטים בעולם משתנה

עד כה כתבנו תוכניות עבור רובוטים לעולם מאד מסוים. קבענו את מצבם ההתחלתי ואת העולם עליו הם יעבדו. אולם לא יכולנו לגרום לרובוט לעבוד על עולם אחר. בעולם אחר מהמתוכנן הרובוט היה יכול להגיע למצב שגיאה. למשל: אם נתנו לרובוט הודעה pickbeeper ולא היה זמזם בצומת – הרובוט עצר והודיע על שגיאה.

הרובוט זקוק ל"תבונה" נוספת על מנת שיוכל לעבוד בעולמות שונים, שיוכל לבדוק את סביבתו ועל סמך הממצאים להחליט מה יהיה צעדו הבא. לצורך כך יש לרובוט חושים והוא משתמש בהם על מנת לבדוק את העולם ולקבל החלטות.
ההוראות המאפשרות לקארל לבדוק ולהחליט נקראות **הוראות תנאי.**

דוגמא א:
נרצה שקארל יבדוק עבור עשרה צמתים אם יש בצומת זמזם, ורק אם התשובה חיובית - יאסוף אותו.




```
loop (10)
{
        Karel.move();
        if  ( Karel.nextToABeeper() )
        {
                Karel.pickBeeper();
        }
}
```

צורתה הכללית של ההוראה:

```
if <תנאי לבדיקה>
{
        <הוראות לביצוע >
}
```

ההוראה מתבצעת כך:
ראשית קארל בודק אם התנאי מתקיים או לא, אם התנאי מתקיים (כלומר ערכו הוא אמת – True) אזי קארל מבצע את ההוראה לביצוע. אם התנאי אינו מתקיים (כלומר ערכו שקר – False), קארל אינו מבצע את ההוראה שבתוך הסוגריים המסולסלים של if . בכל מקרה קארל ממשיך ומבצע את ההוראות 10 פעמים.

התנאים הקיימים בשפה:

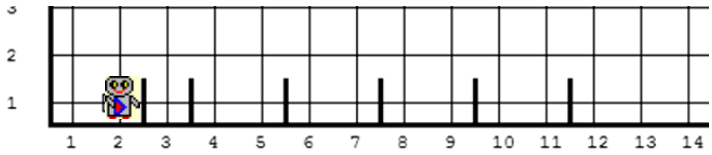
הזכרנו בעבר כי לרובוט יש חושים, בעזרתם הוא יכול לראות קירות, לקבוע מה כיוון פניו, לשמוע זמזמים בצומת בה הוא עומד ולבדוק אם יש זמזמים בשקו. התנאים מחולקים לפי קטגוריות אלה.

שים לב: התנאים מופיעים בזוגות, ולכל תנאי קיים תנאי "הפוך".
סימן הקריאה (!) מסמן את הפעולה הלוגית not.

nextToABeeper()	יש זמזם בצומת	**!nextToABeeper**	אין זמזם בצומת
anyBeepersInBeeperBag()	יש זמזם בשק	**!anyBeepersInBeeperBag()**	אין זמזם בשק
frontIsClear()	פנוי מלפנים	**!frontIsClear()**	חסום מלפנים
facingNorth()	פונה צפונה	**!facingNorth()**	לא פונה צפונה
facingSouth()	פונה דרומה	**!facingSouth()**	לא פונה דרומה
facingEast()	פונה מזרחה	**!facingEast()**	לא פונה מזרחה
facingWest()	פונה מערבה	**!facingWest()**	לא פונה מערבה
nextToARobot()	בצומת רובוט נוסף	**!nextToARobot()**	אין רובוט נוסף

דוגמא ב:

קארל אמור להתאמן בריצת משוכות לאורך 10 צמתים. מיקום המשוכות הוא אקראי ומשתנה מידי פעם.
כמובן קארל יקפוץ רק כאשר קיימת משוכה, וכאשר אין משוכה יתקדם רגיל.



נכתוב הוראה שתגרום לקארל לדלג כאשר הוא רואה משוכה או להמשיך ישר כאשר אין משוכה:
ההוראה תראה כך:
אם אין משוכה? – התקדם
אחרת – קפוץ מעליה.

זוהי הוראת תנאי מורחבת, המורה על ביצוע הוראות מסוימות כאשר התנאי אינו מתקיים.

המבנה הכללי שלה הוא:

```
if  (< תנאי לבדיקה > )
{
        < הוראות לביצוע כאשר התנאי מתקיים >
}
else
{
        <  הוראות לביצוע כאשר התנאי לא מתקיים >
};
```

הקטע הרלבנטי בתכנית יהיה:

```
if ( Karel.frontIsClear )
{
        Karel.move();
}
else
{
          <קבוצת הוראות לדילוג מעל למשוכה >
}
```

תרגיל 1:
השלם את התכנית.

דוגמא ג:

קארל יצא לחופשה מעבודתו, ומחליפו התבקש לחלק טפסים ל-12 עובדי המשרד. התברר כי חלק מהעובדים קיבלו שני טפסים, חלקם לא קיבלו טפסים, וחלק קיבלו טופס אחד כנדרש. קארל התבקש לטפל בעניין ולדאוג כי לכל עובד יהיה טופס (זמזם) אחד בדיוק.
שני מתכנתים ניסו לטפל בבעיה.

מתכנת א כתב את הקטע הבא:

אם אין זמזם בצומת? – הנח זמזם.
אחרת (כלומר יש לפחות זמזם אחד בצומת)
אסוף זמזם
אם אין זמזם בצומת? - הנח זמזם.
(אם יש זמזם – אין צורך לעשות דבר)

```
loop (12)
{
        if ( !Karel.nextToABeeper())
        {
                Karel.putBeeper();
        }
        else
        {       Karel.pickBeeper();
                if ( !Karel.nextToABeeper())
                {
                        Karel.putBeeper();
                }
        }
}
```

המשך בעמ' הבא ←

מתכנת ב כתב את הקטע הבא:

```
loop(12)
{
        if ( Karel.nextToABeeper())
        {
                Karel.pickBeeper();
        }
        if ( !Karel.nextToABeeper())
        {
                Karel.putBeeper;
        }
}
```

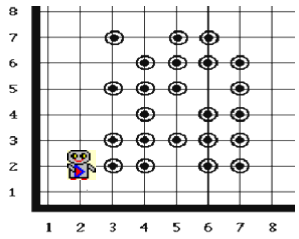
אם יש זמזם בצומת? אזי:
אסוף זמזם.
אם אין זמזם בצומת? אזי:
הנח זמזם.

תרגיל 2:
האם שני האלגוריתמים מבצעים את המטלה כראוי?
אם לא – הסבר את השגיאה, אם כן- איזה מהם טוב יותר? למה?

תרגיל 3:
א. כתוב קטע תכנית המורה לקארל להסתובב שמאלה כאשר הדרך מקדימה חסומה, אחרת ימשיך ישר.
ב. כתוב קטע תכנית המורה לקארל להסתובב שמאלה כאשר הדרך מקדימה חסומה. אם גם הדרך שמאלה חסומה יפנה ימינה.
ג. כתוב קטע תכנית המורה לקארל להסתובב אחורה כאשר הדרך לפניו ומצדדיו חסומה.

תרגיל 4:
כתוב קטע תכנית שבו קארל יסתובב ימינה ויתקדם רק אם הימין פנוי (אין קיר), אם הימין אינו פנוי – קארל יתקדם ישר.

תרגיל 5:
כתוב תכנית לאיסוף זמזמים מריבוע בגודל 5X6 כך שהרובוט יאסוף זמזמים גם כאשר הריבוע אינו מלא ויש צמתים שאין בהם זמזמים.
הרץ את התכנית על שני עולמות שונים.



פישוט הוראות תנאי:
כאשר יש בתכנית הוראות תנאי מורכבות יתכן שנתקשה בהבנת התכנית. נשתדל איפוא לפשט את הוראות התנאי עד כמה שאפשר.
לקארל לא משנה אם ההוראה פשוטה או מסובכת כל עוד היא תקינה ומבצעת את הנדרש (למרות שהוא מתרגז כאשר עליו לבצע הרבה פעולות סרק) , אולם מבחינתנו כקוראי התכנית יש חשיבות גדולה לבהירות הקריאה.

דוגמא ד: הפרדה של הוראות המתבצעות בכל מקרה.

במקום:

```
if ( Karel.nextToABeeper() )
{
        Karel.pickBeeper();
        Karel.move();
}
else
{       Karel.putBeeper();
        Karel.move();
}
```

נוכל לכתוב כך:

```
if ( Karel.nextToABeeper() )
{
        Karel.pickBeeper();
}
else
{       Karel.putBeeper();
}
Karel.move();
```

שים לב כי לא תמיד אפשר להפריד הוראה ולהוציא אותה מחוץ להוראת התנאי.
אם ההוראה המקורית היתה:

```
if ( Karel.nextToABeeper() )
{       Karel.move();
        Karel.pickBeeper();
}
else
{       Karel.move();
        Karel.turnLeft()
}
```

לא נוכל לפשט אותה כך:

```
Karel.move();
if ( Karel.nextToABeeper() )
{
        Karel.pickBeeper();
}
else
{       Karel.turnLeft();
}
```

מדוע?

דוגמה ה: ביטול בדיקה מיותרת.

במקום

```
if ( Karel.facingWest() )
{
        Karel.move();
        if ( Karel.facingWest() )
        {
                Karel.turnLeft()
        }
}
```

נוכל לכתוב כך:

```
if ( Karel.facingWest() )
{
        Karel.move();
        Karel.turnLeft()
}
```

דוגמה ו: הוראת תנאי מיותרת.

במקום

```
if ( Karel.facingWest() )
{
        Karel.move();
}
else
{
        Karel.move();
}
```

נוכל לכתוב פשוט:

```
Karel.move();
```

תרגיל 6:

לפניך 3 הוראות תנאי. האם הן שקולות? נמק.
(כאשר קארל מבצע שתי הוראות שונות ובכל המקרים תוצאות הביצוע זהות, נאמר שההוראות שקולות)

```
if ( Karel.frontIsClear() )
{
        Karel.move();
        Karel.putBeeper();
 }
else
{
        Karel.turnLeft();
 };
```

```
if ( !Karel.frontIsClear() )
 {
        Karel.turnLeft();
  }
else
{
        Karel.move();
        Karel.putBeeper();
 };
```

```
if ( Karel.frontIsClear() )
{
        Karel.move();
        Karel.putBeeper();
}
if ( !Karel.frontIsClear() )
{
        Karel.turnLeft;
};
```

תרגיל 7:

לפניך 3 הוראות תנאי מורכבות. כיצד יגיב קארל בביצוע כל אחת מההוראות?
כתוב את ההוראות מחדש, כך שיהיה פשוט ומובן יותר מהי המשימה לביצוע.

א

```
if ( Karel.facingWest() )
{
        Karel. move();
        Karel.turnLeft();
        if ( Karel.facingNorth() )
        {
                Karel.move();
         }
         Karel.turnLeft();
         Karel.turnLeft();
}
else
{       Karel.move();
        Karel.turnLeft();
        Karel. move();
        Karel.turnLeft();
        Karel.turnLeft();
}
```

ב

```
if ( Karel.facingNorth() )
{
        Karel.turnLeft();
        Karel.pickBeeper();
}
else
{
        Karel.turnLeft();
        Karel. putBeeper();
};
```

ג

```
if ( Karel.nextToABeeper() )
{
        Karel. move();
        Karel.turnLeft();
        if ( Karel.nextToABeeper() )
        {
                Karel. pickBeeper();
         }
}
else
{       Karel. move();
        Karel.turnLeft();
        Karel.turnLeft();
        Karel.turnLeft();
        if ( Karel.nextToABeeper() )
        {
                Karel.pickBeeper();
        }
}
```

כתיבת שאלות חדשות לרובוט:

כאשר יצרנו רובוטים משוכללים, יכולנו ללמד אותם פעולות חדשות. כעת נוכל ללמד את הרובוטים המשוכללים לשאול שאלות חדשות שהתשובה עליהן היא 'אמת' או 'שקר' – ערך בוליאני.
למשל: נרצה ללמד את הרובוט החדש לשאול האם הימין חסום.

נוכל לכתוב את הפעולה כך:
הסתובב ימינה.
אם פנוי מלפנים? – החזר 'אמת'
אחרת החזר 'שקר'

```
bool SmartRobot :: rightIsClear()
{
        turnRight();
        if (frontIsClear())
        {       return true;
        }
        else
        {       return false;
        }
}
```

דרך קצרה יותר לכתוב זאת:
הסתובב ימינה.
החזר "פנוי מלפנים?".

```
bool SmartRobot :: rightIsClear()
{
        turnRight();
        return frontIsClear();
}
```

כיצד מתבצעת הפעולה החדשה :
המילה bool בכותרת הפעולה מציינת כי הפעולה מחזירה ערך בוליאני (אמת או שקר).
כאשר הרובוט נתקל במילה **return** הוא מחזיר את הערך הבוליאני שנוצר, ויוצא מביצוע הפעולה.
שים לב: כל פעולה שתינתן לרובוט אחרי שניתנה לו הפעולה return לא תתבצע.

תרגיל 8:
הבעיה בהוראת התנאי החדשה שכתבנו שהרובוט נשאר כאשר פניו ימינה מהעמדה המקורית שלו. שנה את הפעולה כך שהרובוט יחזיר ערך בוליאני וגם יחזור לכיוונו המקורי.

תרגיל 9:
מה מחזירה הפעולה הבאה?
ובאיזה מצב מסיים הרובוט יחסית למצב שהיה בו בהתחלה?

```
bool SmartRobot :: mystery()
{
        if ( !frontIsClear() )
        {
                turnLeft();
                if ( !frontIsClear() )
                {
                        turnLeft();
                        turnLeft();
                        if ( !frontIsClear() )
                        {
                                return true;
                        }
                }
        }
        return false;
}
```

תרגיל 10:
כתוב פעולה בוליאנית המחזירה 'אמת' אם יש בצומת שני זמזמים בדיוק, אחרת תחזיר 'שקר'.

הוראת חזרה בתנאי:

קארל יצא לטייל ברחוב. הוא יודע שבמרחק מה ממנו, בכיוון הליכתו, נמצאת חומה. אולם הוא לא יודע בדיוק היכן היא נמצאת. משימתו של קארל היא להגיע עד החומה ולעצור.
אם ננסה לתת לקארל סדרת הוראות לביצוע המשימה ניוכח שאין לנו הכלים לעשות זאת, לא ידוע לנו מראש כמה צמתים על קארל להתקדם כדי להגיע לחומה. החומה יכולה להיות במרחק שני צעדים או במרחק עשרים צעדים או בכל מרחק אחר.

```
while <תנאי לבדיקה>
{
        <הוראות לביצוע>
};
```

הוראת החזרה המותנית **while** משלבת בין היכולת לחזור על ביצוע הוראות (של הוראת loop), ובין היכולת לבדוק את מצב העולם (של הוראת if).

למשל, לצורך ביצוע המטלה המוזכרת ברצוננו לתת לקארל את ההוראה :

כל עוד פנוי מלפנים?
התקדם.

```
while ( karel.frontIsClear() ) do
{
        Karel.move();
}
```

איך מתבצעת הוראת while?
תחילה קארל בודק את התנאי. אם התנאי מתקיים יבצע קארל את הוראת הביצוע, ואחר כך יחזור ויבצע את כל הלולאה מחדש (יבדוק את התנאי, אם הוא מתקיים יבצע את ההוראות בגוף הלולאה, ושוב יבדוק את התנאי ויבצע את ההוראות בגוף הלולאה, ושוב...)
אם התנאי אינו מתקיים, קארל מדלג על הוראות הביצוע בגוף הלולאה ועובר לבצע את ההוראות המופיעות לאחר הלולאה.

דוגמה ז: ריקון צומת מזמזמים

קארל מתבקש לאסוף זמזמים בצומת. לא ידוע כמה זמזמים יש בצומת.

כל עוד יש זמזם בצומת?
אסוף זמזם.

```
while ( Karel.nextToABeeper() )
{
        Karel.pickBeeper();
};
```

דוגמא ח: קישוט חדר

קארל כבר מתורגל לעזור בקישוט אולמות. וכולם מבקשים זאת ממנו, בחדרים בגדלים שונים.
עלינו לכתוב תכנית כללית שמנחה את קארל לתלות בלון בכל פינה בחדר, גם כאשר גודל החדר לא ידוע.
בחדר 4 פינות וקארל נמצא בתוך החדר בפינה ימנית תחתונה ופניו צפונה.
האלגוריתם לפתרון יהיה:

חזור 4 פעמים:
לך עד לפינה הבאה
הנח זמזם
פנה ימינה

```
task
{
        Robot Karel(1,1,north,4);
        loop (4)
        {
                //  go to next corner
                while (karel.frontIsClear();
                {
                        Karel.move();
                }
                Karel. putBeeper();
                Karel.turnRight();
        }
}
```

תרגיל 11:

לפניך שני קטעים. הסבר את ההבדל ביניהם.

```
if ( Karel.frontIsClear() )
{
        Karel. move();
};
```

```
while ( Karel.frontIsClear() )
{
        Karel.move();
}
```

תרגיל 12:

לפניך שני קטעים. מה הם מבצעים? האם ניתן לפשט אותם? נמק.

```
while ( !Karel.nextToABeeper() )
{
        Karel.move();
}
if (Karel.nextToABeeper() )
 {
        Karel.pickBeeper();
}
else
{
        Karel.move();
}
```

```
while ( !Karel.nextToABeeper() )
{
        if ( Karel.nextToABeeper() )
        {
                Karel.pickBeeper();
        }
        else
        {
                Karel.move();
        };
}
```

תרגיל 13:

כתוב קטע תכנית המורה לקארל להגיע לצומת (1,1) (פינה שמאלית תחתונה) מכל מקום בעולם. הנח כי אין מכשולים בדרך.

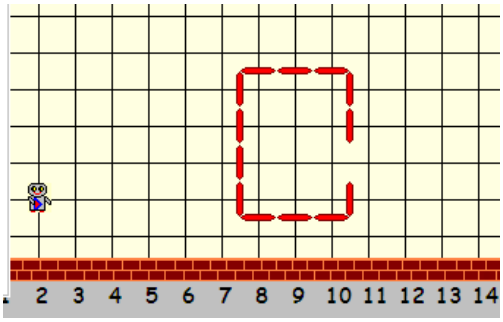
תרגיל 14:

קארל נמצא בחדר החסום על-ידי ארבע קירות. גודל החדר אינו ידוע. כתוב תכנית המורה לקארל לאסוף את כל הזמזמים בחדר.

תרגיל 15:

א. כתוב פעולה בשם stickToRight עבור טיפוס רובוט חכם SmartRobot המלמדת אותו ללכת צמוד לקיר ימני. כאשר אין קיר כזה הרובוט יפנה ימינה.

ב. רובוט נמצא ליד בנין. מקומו של הבנין וגודלו אינם ידועים. כתוב תכנית המורה לרובוט למצוא את פתח הכניסה לבנין תוך שימוש בפעולה שכתבת בסעיף הקודם.
<u>הנחות:</u>
הרובוט עומד ברחוב בו נמצא הבנין.
הפתח יכול להיות בכל אחד מקירות הבנין.



חשיבה מהנה!