תרגיל בית מספר 3

נושא: סימולציה של כלי-רכב באמצעות MVC
תאריך הגשה: מוצ"ש, 25/06/2022, 23:59
הגשה אפשרית בצמדים

בהצלחה רבה!

תיאור התרגיל

תרגיל זה עוסק בסימולציה תלויית-זמן של כלי-רכב, והיא מיועדת לעשות שימוש בפרדיגמת **Model-View-Controller**. אתם נדרשים לתת פתרון מונחה-עצמים בשפת C++ תחת אילוצים לשימוש בתבניות-תכן (*design patterns*).
ראשית תינתן הצגה כללית של מרכיבי עולם הסימולציה ותיאור אחריות המחלקות השונות:

המרחב הגיאוגרפי

המרחב הגיאוגרפי מוגדר באמצעות מרחב אוקלידי דו-מימדי; יחידת המידה הינה מאה קילומטר (***100km***), וההנחה היא כי המרחב אינו סופי (אין אכיפה של תנאי שפה, מעבר למגבלות האחסון של הטיפוס `float`). *צפון* מוגדר להיות הכיוון החיובי של ציר y *(0 מעלות)*, *מזרח* הוא הכיוון החיובי של ציר x *(90 מעלות)*, *דרום* הינו הכיוון השלילי של ציר y *(180 מעלות)*, ו*מערב* הכיוון השלילי של ציר x *(270 מעלות)*. מיקום או יעד של כלי-רכב יוגדר באמצעות צמד קואורדינטות, אך כיוון נסיעה יכול להיות מוגדר באמצעות זווית.

מחסן (warehouse)

מחסן הינו אובייקט סימולציה בעל מיקום קבוע ותכולה של ארגזי סחורות (ללא מגבלת אחסון), והוא נושא באחריות לפרוק ולהעמיס משאיות. קיומו נקבע בשלב איתחול הסימולציה.

משאית (truck)

כלי-רכב זה מעביר ארגזי סחורות ממחסן למחסן ע"פ פקודות העמסה ופריקה. בהגיעה למחסן המיועד לפריקה, המשאית תפרוק את מספר הארגזים שבפקודת הפריקה. יצירת המשאית מתרחשת בזמן איתחול הסימולציה, שם גם נקבע הסבב שלה בין המחסנים.

ניידת משטרה (state trooper)

כלי-רכב זה מפטרל בין המחסנים השונים במרחב הגיאוגרפי, ובהשלמת מסלול חוזר לנקודת היציאה. יצירת הניידת מתרחשת במהלך הסימולציה, שם גם נקבעים המהלכים שלה.

אופנוע-שודדים (chopper)

כלי-רכב זה תוקף ושודד משאיות; הוא איננו רשאי לחנות במחסנים. יצירת האופנוע מתרחשת במהלך הסימולציה, שם גם נקבעים המהלכים שלו.

מודל (model)

זהו אובייקט יחיד אשר נדרש להיות מוגדר בתבנית Singleton; אחריותו לנהל מעקב אחר עולם הסימולציה על כל היבטיו, החל מניהול הזמן וכלה באחסון האובייקטים. בפרט, עליו להחזיק באמצעות מצביעים את כל כלי הרכב והמחסנים המשתתפים בסימולציה, ולספק שירותי גישה אליהם. בנוסף, האובייקט אחראי לספק שירותי עדכון לאובייקט התצוגה (view).

תצוגה (view)

אובייקט זה הינו בעל אחריות אחת והיא הצגת מפת העולם באמצעות גרפיקה מבוססת-`ascii`.
כל אובייקט מיוצג במפה באמצעות שני התווים הראשונים של שמו. **עדכון המידע מבוסס על אינטראקציה עם המודל**.

יחידת-בקרה (controller)

זהו אובייקט יחיד שאחריותו לנהל את האינטראקציה מול המשתמש, ולנתב את הקלטים המתקבלים עבור המודל. אחריות זו כוללת גם את ניהול השגיאות בקלטי המשתמש.

נושאים מתקדמים בתכנות מונחה עצמים, אביב 2022

איפיון מפורט של מרכיבי הסימולציה

מדובר בסימולציה בה צעדי הזמן נשלטים באמצעות פקודות בדידות, כאשר העולם "קופא" בכל צעד זמן, עד למעבר הזמן הבא. מעבר של צעד זמן מתרחש ע"פ פקודה של המשתמש, מייצג שעה עגולה אחת, וכולל עדכון של כל העצמים בעולם הסימולציה.

כלי-רכב (כללי)

אובייקט כלי הרכב הכללי (Vehicle) יכול להימצא במצבים הבאים:

- מצב עצירה ("Stopped"); הוא מצבו ההתחלתי של כל כלי-רכב (מלבד משאית שאותחלה עם פקודות מסע)
- מצב חניה ("Parked") במחסן נתון
- מצב תקוע ("Off road") במיקום נתון (משאית, לאחר שנשדדה)
- מצב תנועה אל מיקום / תנועה אל מחסן / תנועה אל כיוון ("Moving to…")

כמו כן, הוא מצופה לתמוך בפונקציונליות הבאה:

- עצירה: פקודה זו תבטל את יעדיו של כלי-הרכב ותעדכן את מצבו ל-"Stopped".
- קביעת **יעד** נסיעה: פקודה זו תקבע את יעד נסיעה של כלי-הרכב באמצעות צמד קואורדינטות ותעדכן את מצבו ל - "Moving to <destination position>"
- קביעת **כיוון** נסיעה: פקודה זו תקבע את כיוון הנסיעה של כלי-הרכב באמצעות **זווית** ותעדכן את מצבו ל- "Moving on course ...."
- הצגת סטטוס: הדפסת מצב כלי-הרכב הכוללת את שמו, מיקומו, ואת סטטוס הנסיעה שלו (ראו דוגמאות מפורשות בהמשך).

משאית

בנוסף לפונקציונליות הנזכרת, המשאית מצופה לתמוך בפונקציונליות הבאה:

- הגדרת סבב בעת איתחול הסימולציה באמצעות קובץ קלט; קובץ הקלט יגדיר את שם המשאית (שם הקובץ) ואת פקודות הפריקה המתוכננות שלה. המחסנים המופיעים בקובץ הקלט מצופים להתאים לרשימת המחסנים שהוגדרו בסימולציה.
- כל קובץ קלט מגדיר משאית (ע"פ שמו) וכן קובע את הסבב שלה בסימולציה. קובץ קלט חוקי יכיל בשורה הראשונה שלו שם מחסן ("מקור הארגזים") ושעת יציאה חוקית ביחס לתחילת הסימולציה, מופרדים בפסיק. לאחר מכן, כל שורת קלט חוקית תכלול את שם המחסן בו יתבצעו עצירה/פריקה, שעת ההגעה, מספר הארגזים הנפרקים, ושעת היציאה – מופרדים באמצעות פסיק (','):

  `<node>,<arrival_time>,<case_quantity>,<departure_time>`

  לדוגמא, קובץ בשם `Godzilla.txt` מגדיר משאית בשם Godzilla, ותוכנו כולל את הפרטים הבאים על הסבב שלה:

```
Frankfurt,00:00
Lille,10:30,67,11:15
Antwerp,15:45,23,18:30
Hamburg,19:00,15,27:00
```

- סטטוס קונקרטי: בנוסף לשמה, מיקומה וסטטוס הנסיעה שלה, המשאית תציין את מספר הארגזים שהיא נושאת (ניתן להוסיף מידע טקסטואלי אחר). למשל,

```
Truck Godzilla at (37.14, 10.00), Heading to Lille, Crates: 105
```

ניידת משטרה

ניידת משטרה מפטרלת בין המחסנים השונים באופן אוטומטי. בהינתן פקודת נסיעה למחסן מסויים (**destination**), הניידת תיסע למחסן זה, משם תמשיך למחסן הקרוב ביותר אליו, ממנו אל המחסן הקרוב ביותר שטרם ביקרה בו, וכך הלאה. במקרה של שוויון מרחקים בין מחסנים קרובים שטרם ביקרה בהם, ייבחר המחסן הראשון לפי סדר אלפביתי. מטרתה לבקר בכל מחסן פעם אחת בלבד, ולאחר שעשתה זאת תשלים את מסלולה בחזרה אוטומטית למחסן הראשון בו ביקרה ותסיים את הפטרול. הניידת יכולה גם לקבל פקודות נסיעה פרטניות (course/position).

כל ביקור של ניידת במחסן אינו אורך זמן. כל ניידת מפטרלת במהירות קבועה של 90 km/h, והיא מצופה לתמוך בפונקציונליות הבאה:

- יצירת ניידת חדשה באמצעות שני הפרמטרים הבאים:
  i. שם הניידת (מחרוזת אלפביתית עם לכל היותר 12 תווים)
  ii. שם מחסן לציון המיקום ההתחלתי
- פקודה בשם **destination** לקביעת מחסן היעד (ארגומנט: שם מחסן חוקי)
- פקודות בשם **course** לקביעת כיוון הנסיעה (ארגומנט: זווית במעלות) וכן **position** לקביעת יעד הנסיעה (2 ארגומנטים: צמד קואורדינטות).

נושאים מתקדמים בתכנות מונחה עצמים, אביב 2022

אופנוע-שודדים

כלי-רכב זה מאופיין ע"י מהירותו (מהירותו המקסימלית של האופנוע היא 170 km/h) וע"י טווח התקיפה שלו (איתחול כל האופנועים הוא 2 ק"מ; הערך המקסימלי האפשרי הינו 20 ק"מ). אובייקט זה נדרש לתמוך בפעולות הבאות:

- יצירת אופנוע חדש באמצעות הפרמטרים הבאים:
  - i. שם האופנוע (מחרוזת אלפביתית עם לכל היותר 12 תווים)
  - ii. צמד קואורדינטות לציון המיקום ההתחלתי
- פקודות בשם **course** לקביעת כיוון הרכיבה (2 ארגומנטים: זווית במעלות ומהירות הרכיבה) וכן **position** לקביעת יעד הרכיבה (3 ארגומנטים: צמד קואורדינטות ומהירות הרכיבה).
- תקיפה: פעולת התקיפה צריכה להיות מכוונת כלפי משאית אשר נמצאת בטווח התקיפה של האופנוע בתחילת זמן הפעולה (צעד הזמן הבא בסימולציה), אחרת הפעולה תסתיים בכישלון. אם יעד התקיפה נמצא בטווח, הפעולה מתבצעת מיידית (כלומר בצעד הזמן הנוכחי, שהוא בהכרח בשעה עגולה), ובמהלכה נבדקת נוכחות של ניידות משטרה בקרבת המשאית המותקפת. **התקיפה תצליח רק אם לא קיימת ניידת משטרה בשעת התקיפה ברדיוס של 10 ק"מ ממיקום המשאית**.
  - במקרה של משאית, אם התקיפה צלחה, המשאית תאבד את כל מטענה (מספר הארגזים שלה יתאפס), פקודות המסע שלה יבוטלו מיידית, והיא תעבור למצב "Off road". בפועל, אובייקט זה יושבת ויחדל לתפקד לאחר התקיפה. אם התקיפה כשלה, המשאית תשמור על מצבה לפני התקיפה, ותמשיך בסבב שלה כמתוכנן.
  - במקרה של אופנוע –בין אם התקיפה צלחה או לא, האופנוע יעבור למצב "Stopped" במיקום השוד. אם התקיפה צלחה, טווח התקיפה של האופנוע יגדל בק"מ אחד; אם התקיפה כשלה, הטווח שלו יקטן בק"מ אחד.

איפיון המודל

המודל נדרש לאחסן את כל העצמים של עולם הסימולציה.

המודל נושא באחריות לעדכון מצבם של כל האלמנטים בעולם הסימולציה במעברי זמן בדידים של שעה אחת. כלומר, על המודל לבצע את כל החישובים והעדכונים המתרחשים בכל מעבר זמן, אשר קבוע להיות יחידת זמן של שעה אחת.

על בנאי המודל ליצור בכל מקרה מחסן בשם `Frankfurt` במיקום (`40,10`), עם תכולת ארגזים `100000`.

איפיון יחידת הבקרה

יחידת הבקרה מנהלת את האינטראקציה מול המשתמש באמצעות קריאת פקודות, ובהתאם, מאפשרת שליטה במודל ובתצוגה. עיבוד טקסטואלי של פקודה מתבצע ע"פ הסדר הבא:

המילה הראשונה חייבת להיות המחרוזת `exit`, שם של כלי-רכב קיים, או פקודת מודל או תצוגה.

- באם המחרוזת הינה `exit`, על התכנית לסיים את הסימולציה ואת ריצתה בשלב זה.
- באם המחרוזת זהה לשם המלא של כלי-רכב, המילה שלאחריה נדרשת להיות פקודה חוקית שתעובד בהתאם.
- אחרת, המחרוזת נדרשת בהכרח להיות פקודה חוקית עבור המודל או התצוגה.

<u>בכל הפקודות, תו ההפרדה הסטנדרטי בין הארגומנטים הינו רווח בודד</u>.

<u>פקודות יחידת הבקרה עבור התצוגה</u>:

פקודה בשם **default** לשחזור פרמטרי ברירת המחדל של המפה (ללא ארגומנטים)

פקודה בשם **size** לקביעת גודל המפה (ארגומנט יחיד, שלם חיובי s, המגדיר את מספר העמודות והשורות: $30 \geq s > 6$)

פקודה בשם **zoom** לקביעת קנה-המידה של המפה (ארגומנט יחיד, ממשי חיובי, לקביעת היחס של 100 ק"מ לתא שטח במפה)

פקודה בשם **pan** לקיבוע ראשית הצירים של המפה (צמד ארגומנטים ממשיים)

פקודה בשם **show** לבקשה מיחידת התצוגה לצייר את המפה המעודכנת

<u>פקודות יחידת הבקרה עבור המודל</u>:

פקודה בשם **status** בעקבותיה על כל האובייקטים בסימולציה לתאר את מצבם העדכני (ללא ארגומנטים)

נושאים מתקדמים בתכנות מונחה עצמים, אביב 2022

פקודה בשם **go** לעדכון כל האובייקטים ביחידת זמן בדידה של שעה (ללא ארגומנטים)

פקודה בשם **create** ליצירת כלי-רכב חדש (ניידת משטרה או אופנוע שודדים, 3 או 4 ארגומנטים בהתאמה); באם כבר קיים כלי-רכב בשם זה, הדבר נחשב לשגיאה:

- שם כלי-הרכב (מחרוזת אלפביתית עם לכל היותר 12 תווים)
- סוג כלי-הרכב: State_trooper / Chopper
- ציון המיקום ההתחלתי: שם מחסן חוקי עבור ניידת, צמד קואורדינטות עבור אופנוע (בסוגריים, מופרדים ביניהם בפסיק)

<u>(הפקודות הבאות יינתנו לאחר שם כלי-רכב):</u>

פקודה בשם **course** לקביעת כיוון הנסיעה עבור כלי-רכב נתון (ארגומנט: זווית במעלות; במקרה של אופנוע, לאחריה תימסר מהירות הרכיבה)

פקודה בשם **position** לקביעת יעד הנסיעה עבור כלי נתון (ארגומנטים: צמד קואורדינטות; במקרה של אופנוע, לאחריהם תימסר מהירות הרכיבה)

פקודה בשם **destination** לקביעת מחסן היעד (עבור ניידת משטרה; ארגומנט: שם מחסן חוקי)

פקודה בשם **attack** לקביעת משאית לתקיפה בצעד הזמן הבא (ארגומנט יחיד: שם משאית חוקי)

פקודה בשם **stop** לעצירת הנסיעה (ללא ארגומנטים); **פקודה זו תגרום לביטול כל פקודות המסע העתידיות**, באם קיימות

איפיון התצוגה

אובייקט התצוגה מספק פלט מרחבי בתצורת מפה עליו פרושים איברי הסימולציה.
התצוגה מורכבת ממטריצה ריבועית, אשר בברירת המחדל ובתחילת הסימולציה מאותחלת להיות בגודל 25X25; (גודלה לפחות 6, ולכל היותר 30).
תא מייצג טווח ערכים של קואורדינטות מרחביות (x,y), כאשר גודלו של כל תא מייצג את קנה-המידה; ברירת המחדל היא 2.0.
ראשית הצירים, אשר יכולה להיות כל נקודה מרחבית, מוגדרת להיות הקואורדינטה בפינה השמאלית התחתונה של המפה.
מבחינת ייצוג, כל תא במפה מיוצג ע"י שני תווים, כאשר תא ריק מיוצג ע"י נקודה ' . ' ולאחר מכן רווח ' ' :

<u>שימו לב כי יחידת המידה המרחבית היא 100 ק"מ, בעוד שיחידת מידת המהירות של כלי-הרכב היא קמ"ש.</u>

פורמט קלט

התכנית תקבל כקלט קובץ אחד ויחיד המתאר את המחסנים במרחב הגיאוגרפי (לאחר דגל `-w`), וכן לפחות קובץ קלט אחד המתאר סבב של משאית (לאחר דגל `-t`). לאחר קומפילציה של התכנית לכדי קובץ הרצה בשם `simDepot`, כך ניתן יהיה להריץ אותה במקרה של עבודה דרך טרמינל:

```
$ ./simDepot -w depot.dat -t <truckfile1> [<truckfile2> <truckfile3> ...]
```

1. המחסנים במרחב הגיאוגרפי, אשר יוזנו לתכנית בתחילתה בקובץ הקלט (`depot.dat` בדוגמא שלעיל), יתוארו באמצעות שמם, ומיקומם הגיאוגרפי (באמצעות צמד קואורדינטות בסוגריים).
להלן דוגמא לתוכן של קובץ קלט תקין:

```
Lille, (10.00, 10.00), 10000
Antwerp, (20.00, 20.00), 20000
Hamburg, (0.00, 30.00), 50000
```

2. סבב המשאית יתואר לפי הפורמט שניתן לעיל.
3. במקרה של קלט לא חוקי (למשל, שגיאת קבצים, הזנת תווים/שדות לא חוקיים, וכיוצא באלה), התכנית תסיים את ריצתה בשלב זה ותדפיס הודעת שגיאה מתאימה לערוץ השגיאות הסטנדרטי.
4. בהנחה שהקלט של המשתמש חוקי, על התכנית לפתוח ערוץ קלט סטנדרטי מול המשתמש לשם קליטת פקודות.

נושאים מתקדמים בתכנות מונחה עצמים, אביב 2022

קונסול לקליטת פקודות

היות ומדובר בסימולציה עם מעברי זמן בדידים ונשלטים, על התכנית להדפיס בכל שלב את צעד הזמן הנוכחי.
הפניה אל המשתמש בזמן X לקבלת פקודה תתרחש בפורמט הבא:

```
Time X: Enter command:
```

ניתן להזין מספר רב של **פקודות controller** בכל צעד זמן, ורק הפקודה `go` תביא להתקדמות הזמן (סיום צעד הזמן הנוכחי) תוך חישוב כל העדכונים הנדרשים בשעת המעבר.

להלן רצף חוקי של פקודות קונסול לאחר איתחול התכנית באמצעות קבצי הדוגמא `depot.dat, Godzilla.txt`:

```
Time 0: Enter command: create Sylvester Chopper (14.00, 14.00)

Time 0: Enter command: create Nadakhan Chopper (24.00, 24.00)

Time 0: Enter command: create Cooper State_trooper Frankfurt

Time 0: Enter command: Sylvester course 180 100

Time 0: Enter command: Nadakhan course 270 100

Time 0: Enter command: Cooper destination Hamburg

Time 0: Enter command: go

Time 1: Enter command: status

Warehouse Frankfurt at position (40.00, 10.00), Inventory: 999895

Warehouse Antwerp at position (20.00, 20.00), Inventory: 0

Warehouse Lille at position (10.00, 10.00), Inventory: 0

Warehouse Hamburg at position (0.00, 30.00), Inventory: 0

Truck Godzilla at (37.14, 10.00), Heading to Lille, Crates: 105

Chopper Sylvester at (14.00, 13.00), Heading on course 180.00 deg, speed 100.00 km/h

Chopper Nadakhan at (23.00, 24.00), Heading on course 270.00 deg, speed 100.00 km/h

State_trooper Cooper at (39.19, 10.40), Heading to Hamburg, speed 90.00 km/h
```

קובץ טקסטואלי של פקודות אלו, הכולל הצגת מפה באמצעות `show`, נתון לכם כנספח במוודל בשם `console.dat`. בנוסף, מצורף קובץ טקסטואלי המדגים את פקודות התצוגה עבור אותה סימולציה בזמן 0 - כנספח במוודל בשם `view_console.dat`.

הנחות

- שם מחסן הוא מחרוזת אלפביתית עם 12 תווים לכל היותר.
- תנועת כלי-הרכב תמיד תתרחש בקו ישר בין נקודות מקור ליעד.
- האלמנטים בסימולציה הינם בעלי גודל אפס, ולפיכך אין להתחשב בתסריט של התנגשות בין עצמים סמוכים במרחב.

אילוצים ודגשים

- תבניות-תכן: אובייקט המודל נדרש להיות סינגלטון בעל גישה גלובלית; יצירת כלי-הרכב נדרשת להסתמך על תבנית *factory* כלשהי.
- המימוש נדרש להיות בשפת C++, עם מתן עדיפות לתחביר התקן החדש.
- בתרגיל זה עליכם להשתמש במצביעים החכמים של C++0x בכל שימוש במצביעים.
- יש לבדוק תקינות קלטים; במקרה של אי-תקינות, יש להפעיל מנגנון חריגות ולהציג הודעות שגיאה מתאימות.

נושאים מתקדמים בתכנות מונחה עצמים, אביב 2022

- עליכם לשאוף ליעילות ביצועים, ובפרט, לממש *move semantics* עבור מחלקות שעשויות להפיק מכך תועלת.
- עליכם לוודא כי התכנית עוברת קומפילציית `g++ -std=c++11` התואמת את הקומפיילר שעל שרת המכללה <u>ללא כל שגיאות או אזהרות כלשהן</u>, ורצה בהצלחה. לשם כך, ניתן לבדוק קומפילציה עם הבודק האוטומטי בקוד 27.
- עליכם לתעד את הקוד באמצעות הערות המתארות בקצרה את המחלקות והפונקציות השונות.

הגשה

- עליכם להגיש במערכת Moodle קובץ ארכיב מטיפוס zip בלבד, ששמו כולל את קוד הקורס (`'27'`), שם התרגיל (`'ex3'`) ותעודת הזהות של הסטודנט/ית המגיש/ה, מופרדים בקו תחתי בפורמט הבא: **`27_ex3_studID.zip`, או בפורמט `27_ex3_ID1_ID2.zip` במקרה של הגשת צמד.**
- על ארכיב zip זה להכיל את כל קבצי המקור (ממשק/מימוש) הנדרשים לקומפילציה, והוא רשאי להכיל תיעוד טקסטואלי; מבחינת טיפוסי קבצים, עליו לכלול רק קבצים עם סיומות **`*.txt *.h *.cpp`**
- לדוגמה: על סטודנט שמספר הזיהוי שלו הינו 012345678 להגיש ארכיב בשם `27_ex3_012345678.zip` הכולל את כל קבצי המקור של הפרוייקט, *ללא תיקיות כלשהן*, ורשאי להכיל קובץ טקסטואלי לתיעוד.

אי-הקפדה על ההנחיות, תגרור הורדה בציון התרגיל. <u>לא תתקבלנה הגשות באיחור!</u>

Bibliography

1. E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995
2. B. Stroustrup, The C++ Programming Language, 4th ed., Addison-Wesley, 2013