Laboration 6 – algoritmer, labyrint

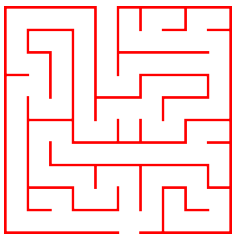
Mål: Du ska lösa ett problem där huvuduppgiften är att konstruera en algoritm för att hitta vägen genom en labyrint. Det ska du göra genom att skriva program som använder din `Turtle`-klass från laboration 5.

Förberedelseuppgifter

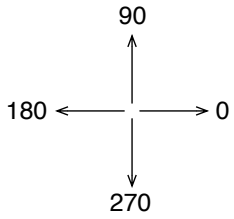
- Läs avsnittet Bakgrund.

Bakgrund

En labyrint består av ett rum med en ingång och en utgång. Alla väggar är parallella med x- eller y-axeln. Ingången finns alltid i den nedersta väggen, och utgången alltid i den översta (figur 1).



Figur 1: En labyrint.



Figur 2: Riktningar.

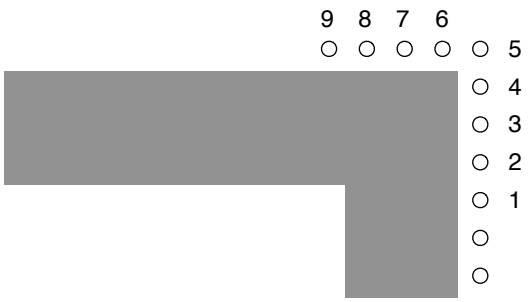
Ett sätt att hitta från ingången till utgången är att gå genom labyrinten och hela tiden hålla den vänstra handen i väggen. När man vandrar genom labyrinten är fyra riktningar möjliga. En riktning definieras som vinkeln mellan x-axeln och vandringsriktningen och mäts i grader (figur 2).

Labyrinten beskrivs av en färdigskriven klass `Maze`. Labyrintens utseende läses in från en fil när labyrintobjektet skapas. Klassen har följande specifikation:

```
/** Skapar en labyrint med nummer nbr. */
Maze(int nbr);

/** Ritar labyrinten i fönstret w. */
void draw(SimpleWindow w);

/** Tar reda på x-koordinaten för ingången. */
int getXEntry();

/** Tar reda på y-koordinaten för ingången. */
int getYEntry();

/** Undersöker om punkten x,y är vid utgången. */
boolean atExit(int x, int y);

/** Undersöker om man, när man befinner sig i punkten x,y och är på väg i
    riktningen direction, har en vägg direkt till vänster om sig. */
boolean wallAtLeft(int direction, int x, int y);

/** Som wallAtLeft, men undersöker om man har en vägg direkt framför sig. */
boolean wallInFront(int direction, int x, int y);
```

I metoderna `wallAtLeft` och `wallInFront` betraktas alla riktningar $R \pm n \cdot 360°, n = 1, 2, \ldots$ som ekvivalenta med R.

Väggarna ritas med lite tomrum mellan sköldpaddan och väggen, så att det blir lättare att skilja dem åt. Exempel där man vandrar runt ett hörn:



I punkterna 1–4 är riktningen 90. `wallAtLeft` ger i dessa punkter värdet `true`, `wallInFront` värdet `false`. I punkt 5 ger `wallAtLeft` värdet `false`, varför man svänger åt vänster (riktningen 90 ändras till 180). Man fortsätter sedan genom punkterna 6–9. I dessa punkter ger `wallAtLeft` värdet `true`, `wallInFront` värdet `false`.

I uppgiften ska du skriva en klass `MazeWalker` som låter en sköldpadda vandra i en labyrint. Låt klassen ha följande uppbyggnad:

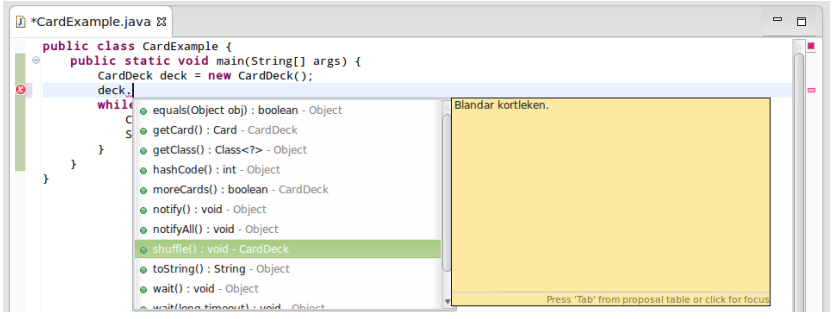
```
public class MazeWalker {
    private Turtle turtle;

    public MazeWalker(Turtle turtle) {
        // fyll i kod
    }

    /** Låter sköldpaddan vandra genom labyrinten maze, från
        ingången till utgången. */
    public void walk(Maze maze) {
        // fyll i kod
    }
}
```

Uppgifter

1. (valfritt) Om du vill kan du få lite mer hjälp av Eclipse när du programmerar. Ta fram fönstret med Eclipse-inställningarna:

   Window → *Preferences* (Windows och Linux, inklusive LTH:s Linuxdatorer)
   Eclipse → *Inställningar...* (Mac)

   Välj *Java* → *Editor* → *Content Assist* → *Advanced*. Klicka i *Java proposals* och *Template proposals*. Klicka *Apply*, därefter *OK*. Hädanefter kommer Eclipse att ge förslag beroende på sammanhanget. Följande bild visar hur det kan se ut när programmeraren skrivit ”`deck`”, tryckt punkt, och nu bläddrar bland förslagen:



2. Till denna uppgift finns inte något färdigt Eclipseprojekt. Du måste alltså skapa ett sådant:

   1. Klicka på *New*-ikonen i verktygsraden. Välj *Java Project*.
   2. Skriv namnet på projektet (t.ex. *MazeTurtle*).
   3. Klicka på *Finish*.
   4. Om Eclipse frågar om filen *module-info.java* ska skapas, så välj **"Don't Create"**. Någon sådan fil behövs inte.
      (Om du råkade trycka "Create" istället, och alltså skapade filen `module-info.java`, så ta bort den: högerklicka på filen i vänsterspalten och välj "Delete" i menyn. Annars kan du få mystiska kompileringsfel.)

3. Programmet som du skriver kommer att använda klasserna `SimpleWindow och Maze`. Dessa klasser finns i biblioteksfilen *cs_pt.jar*, som finns i projektet *cs_pt*. Programmet kommer också att utnyttja klassen `Turtle` från laboration 5.

   Du måste därför göra *cs_pt.jar* och projektet *Lab05* tillgängliga i ditt nya projekt.

   1. Högerklicka på projektet *MazeTurtle*, välj *Build Path* → *Configure Build Path*.
   2. Klicka på fliken *Libraries*. Om det finns en rad *Classpath*, markera då den.
   3. Klicka på *Add JARS...*, öppna projektet *cs_pt*, markera filen *cs_pt.jar* och klicka *OK*.
   4. Klicka nu på fliken *Projects* (istället för *Libraries*). Även här markerar du raden *Classpath* (om den finns).
   5. Klicka på *Add...*, markera *Lab05* och klicka *OK*.
   6. Klicka på *OK* för att lämna dialogen.

4. Skapa klassen `MazeWalker`. I metoden `walk` ska sköldpaddan börja sin vandring i punkten med koordinaterna `getXEntry()`, `getYEntry()` och gå uppåt i labyrinten. Alla steg ska ha längden 1.

   Lägg in en fördröjning efter varje steg så att man ser hur sköldpaddan vandrar. (Använd t.ex. metoden `delay` i `SimpleWindow`; ett exempel finns på s. 20 i detta kompendium.)

5. Skriv ett huvudprogram som kontrollerar att sköldpaddan kan vandra i labyrinterna. Numret på labyrinten ska läsas in från tangentbordet.

   Testa programmet. Det finns fem olika labyrinter, numrerade 1–5. Givetvis ska alla testas och fungera. För den sista labyrinten, som är ganska stor, kan du behöva justera (eller kanske helt ta bort) fördröjningen mellan varje steg.

Frivilliga extrauppgifter

6. Utforska källkoden för `Maze` som finns i projektet *cs_pt*. Kan du se hur du kan använda en egen karta[3] för labyrinten? (Tips: undersök `Maze`-klassens konstruktorer.)

7. Summera antalet steg sköldpaddan tar och antalet riktningsändringar som sköldpaddan gör och skriv ut dessa summor när sköldpaddan kommit i mål.

[3] Du kan exempelvis skapa en egen karta på `http://www.mazegenerator.se`. När du skapat en labyrint på sidan dyker det upp en nya ruta, med en rubrik i stil med "Labyrint med 20x20 fyrkantiga celler". I den rutan väljer du "PNG" i listan och klickar "Ladda ner". Om du använder en egen bild måste den ha helvit eller transparent bakgrund.