Laboration 9 – TurtleRace, del 1: ArrayList

Mål: Du ska träna på att använda listor för att hålla reda på objekt. Du ska även förstå grundläggande användning av arv.

Förberedelseuppgifter

- Läs i läroboken: Downey & Mayfield, kapitel 14.
- Läs avsnittet Bakgrund.

Bakgrund

Under den här laborationen ska du skriva ett program där du låter sköldpaddor tävla mot varandra i en kapplöpning.

Vi använder klassen `SimpleWindow` på ett annorlunda sätt än tidigare: vi har här specialiserat den genom subklassen `RaceWindow`. Ett `RaceWindow` fungerar som ett vanligt `SimpleWindow` (eftersom det ärver från `SimpleWindow`) men med tillägget att en kapplöpningsbana ritas upp då fönstret skapas.

Uppgiften går ut på att skriva ett program som simulerar en (slumpmässig) kapplöpning mellan sköldpaddor (`Turtle`-objekt). En sköldpadda tar sig fram genom simulerade tärningskast (dvs. slumpmässiga steg framåt mellan 1 och 6), med ett ”kast” i varje runda. Samtliga sköldpaddor ska lagras i en `ArrayList`.

En klass `RaceTurtle` ska representera en sköldpadda som kan simulera en kapplöpning. `RaceTurtle` ska ärva från klassen `Turtle` och därmed kan en `RaceTurtle` göra allt som en vanlig `Turtle` kan. Därtill ska en `RaceTurtle` innehålla en slumptalsgenerator och ett startnummer. RaceTurtle ska ha en metod `raceStep()` och en metod `toString()`. Metoden `raceStep()` ska beskriva hur sköldpaddan tar ett löpsteg.

`RaceTurtle` ska alltså ärva från Turtle och ha följande specifikation:

```
/**
 * Skapar en sköldpadda som ska springa i fönstret w och som har start-
 * nummer nbr. Sköldpaddan startar med pennan nere och nosen vänd åt höger.
 */
RaceTurtle(RaceWindow w, int nbr);

/**
 * Låter sköldpaddan gå framåt ett steg. Stegets längd ges av ett
 * slumptal (heltal) mellan 1 och 6.
 */
void raceStep();

/**
 * Returnerar en läsbar representation av denna RaceTurtle,
 * på formen "Nummer x" där x är sköldpaddans startnummer.
 */
String toString();
```

Datorarbete

1. Inkludera din `Turtle` från laboration 5 genom att:
   1. Högerklicka på projektet *Lab09*, välj *Build Path* → *Configure Build Path*.
   2. Klicka på fliken *Projects*.
   3. Klicka på *Add...*, markera *Lab05* och klicka *OK*.
   4. Klicka på *OK* för att lämna dialogen.

2. Implementera klassen `RaceTurtle` enligt beskrivning och specifikation ovan. När du implementerar konstruktorn för `RaceTurtle` måste du beräkna dess position. Det finns två hjälpmetoder i `RaceWindow`, som beroende på startnummer returnerar en lämplig x- respektive y-position. På så sätt får alla sköldpaddor en egen bana.

3. I klassen `RaceTurtleTest` finns en main-metod som skapar en sköldpadda och låter denna genomföra ett eget lopp utan motståndare. Klassen kan användas som ett första test av din `RaceTurtle`. Avkommentera och studera koden – det är meningen att du ska förstå vad den gör. Kör sedan programmet för att se om sköldpaddan går i mål som förväntat.

4. Gör en ny klass `TurtleRace` för att genomföra ett lopp enligt följande (det är lämpligt att inspireras av testprogrammet):
   - Skapa åtta sköldpaddor och lagra dessa i en `ArrayList`. Låt dem ha sin kapplöpning i ett och samma `RaceWindow`.
   - Efter loppet ska första, andra och tredje plats skrivas ut (enligt exemplet nedan). Det är ett krav att utskriften använder sig av `toString`-metoden.

     ```
     På plats 1: Nummer 5
     På plats 2: Nummer 6
     På plats 3: Nummer 1
     ```

   - **Tips:** När en sköldpadda går i mål kan du ta ut den ur den vanliga listan, och sätta in den i en separat lista. När den vanliga listan är tom har alla sköldpaddorna gått i mål, och då görs utskriften ovan.
   - Observera att en for-each-loop inte tillåter att vi ändrar en listas innehåll i loopen.
   - Små orättvisor i resultatberäkningen är acceptabla. Kanske upptäcker du att din lösning gynnar sköldpaddor med lägre startnummer över de med högre startnummer. Förklara för din handledare vari orättvisan består, och diskutera gärna hur man kan åtgärda den.
   - Det är därmed inte heller nödvändigt att hantera delade placeringar.

5. Provkör ditt sköldpaddslopp. Använd en fördröjning, t. ex. `RaceWindow.delay(10)`, så man hinner se hur loppet fortskrider. Det kan även vara trevligt att låta användaren starta loppet med ett musklick, dvs `w.waitForMouseClick()` när sköldpaddorna är uppställda på startlinjen. Kontrollera att rätt placeringar skrivs ut.

Checklista

I den här laborationen har du övat på att

- använda listor för att lagra objekt,
- ärva egenskaper från en superklass,

- implementera och använda `toString`-metoden, och
- använda statiska hjälpmetoder.