

Laboration 2

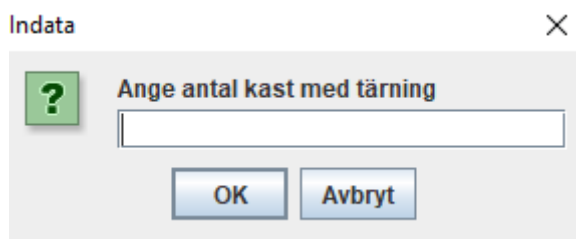
Problembeskrivning

Laborationens fokus gick ut på skapandet av en metod vars uppgift är att skapa ett godtyckligt antal slumpmässiga tal mellan 1-6. Den ska då sedan lagra dessa tal i en sträng som sedan returneras.

Metoden ska anropas från main och efter det, på valfritt sätt, bearbeta strängen som returnerades. Målet är en utskrift i form av en frekvenstabell över respektive siffra, detta ska då avslutningsvis visas i konsolfönstret.

Programbeskrivning

Den första delen handlar om det godtyckliga talet. Istället för att välja att låta talet – vars funktion är att motsvara "antal tärningskast" – vara hårdkodat blir det istället upp till användaren att utse ett tal. Detta tal måste dock uppfylla kriteriet att vara över noll och genom att samtidigt låta inmatningen skyddas av try catch kan heller användaren inte gå vidare genom att fylla i annat än siffror. Värt att notera att JOptionPane användes för använda dialogrutorna, därmed har vi då också importerat paketet innehållandes JOptionPane.



När väl användaren passerat input-delen kommer genast nästa del vilket handlar om anrop till första metoden som handlar om att returnera en sträng med ett antal (det tal användaren valde) slumpvalda siffror mellan 1 till 6. Vi slumpar genom att använda oss av en slumpgenerator vi kommer åt genom att instansiera ett objekt av klassen Random. För att få detta fungera behöver vi också importera "java.util.*;". En forloop på detta och vi är redo att lagra nummer för nummer i en sträng som då sedan returneras.

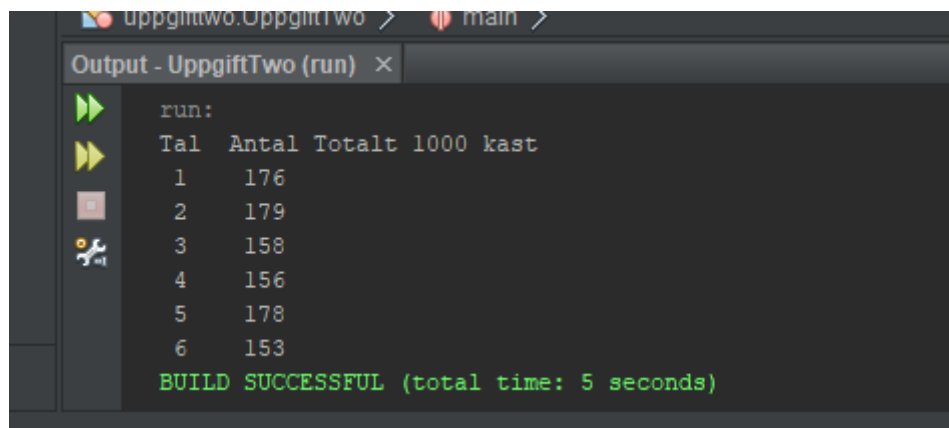
Nu när vi har strängen återkommer vi till där vårt anrop skedde och strängen lagras i en ny sträng som jag väljer att förse nästa metod med som anropas precis efter. Denna metod som jag väljer att kalla sortera har då en inparameter men kommer inte returnera något, varför den då blir en "voidmetod".

Inuti metoden initieras flera variabler av typen int, en för varje siffra som kan finnas i strängen. Då vi vet att det handlar om 1-6 är det alltså 6 stycken variabler. Med en forloop går vi igenom varje tecken och resultatet plussas på respektive variabel genom en if-sats.

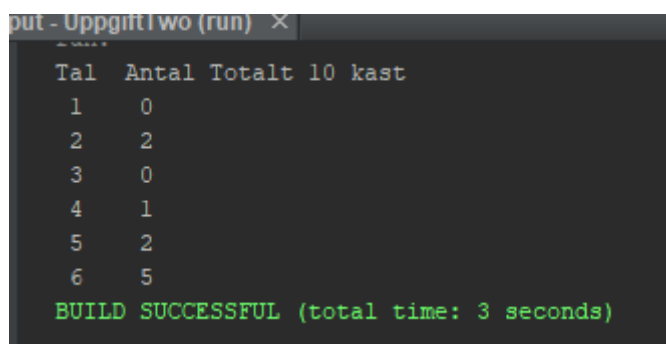
När hela längden på strängen har passerats kommer vi ur forloopen och till en konsolutskrift som skriver ut en variabel som även den var inuti forloopen. Denna variabel var av typen sträng och uppbyggd för att kunna genomföra en frekvenstabell på "ett något primitivt sätt".

Detta medför att det tidigare anropet kommer tillbaka med denna utskrift som resultat.

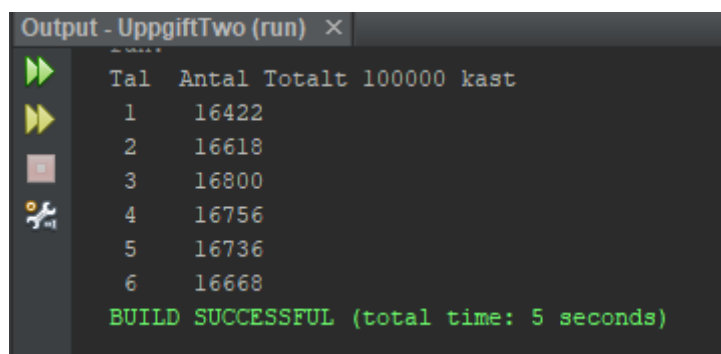
Här följer ett antal skärmdumpar på utskriften i konsollen:



```
run:
Tal  Antal Totalt 1000 kast
1    176
2    179
3    158
4    156
5    178
6    153
BUILD SUCCESSFUL (total time: 5 seconds)
```



```
Tal  Antal Totalt 10 kast
1    0
2    2
3    0
4    1
5    2
6    5
BUILD SUCCESSFUL (total time: 3 seconds)
```



```
Tal  Antal Totalt 100000 kast
1    16422
2    16618
3    16800
4    16756
5    16736
6    16668
BUILD SUCCESSFUL (total time: 5 seconds)
```

Egna reflektioner och slutsatser

En bra uppgift som innefattar många av de senaste momenten vi gått igenom. Detta samtidigt som uppgiften behandlar något som är enkelt att greppa. De skärmdumpar ovan jag valde att ha med skiljer sig till antalet tärningskast. Kanske inte helt förvånande kan vi notera att det skiljer sig markant mellan resultaten framförallt där tärningskasterna är få till antal (se bild två, 10 till antal). Denna skillnad är något som jämnar ut sig allt eftersom vi kastar tärning fler gånger – dock värt att skilja på utfallsskillnaden och den procentuella skillnaden. Det är den procentuella skillnaden som jämnar ut sig.

Jonas Hallström - 9008160252