Mittuniversitetet
MID SWEDEN UNIVERSITY

Inlämningsuppgift 7

Datateknik GR(A), Java I, 7,5 högskolepoäng

Syfte:	Att lära dig skapa och använda arrayer och ArrayList av olika typer. Att kunna sortera elementen i en array av typen int.
Att läsa:	Lektion 7
Uppgifter:	2
Inlämning:	Inlämningslåda 7 i Moodle

Uppgift 1

Översikt

Du ska skriva en klass med vilken vi kan generera användaridentiteter för anställda på ett företag. Syftet är att du ska lära dig använda arrayer och `ArrayList` för att lagra flera värden av samma typ.

Uppgift

På många arbetsplatser blir de anställda tilldelade ett användarnamn för att kunna logga in på sina datorer. En metod för att generera dessa användarnamn är att ta de tre första bokstäverna i förnamnet och de tre första bokstäverna i efternamnet och låta dessa tillsammans bilda användarnamnet. En person med namnet Sven Andersson får då användarnamnet sveand (enbart små bokstäver).

Din uppgift är att skriva en klass med namnet `Usernames` som ska innehålla en statisk metod med namnet `create`, som skapar ett användarnamn enligt ovan. Metoden ska ta en array av typen `String` som argument, och returnera en `ArrayList` innehållandes en sträng för varje genererat användarnamn. Arrayen som skickas som argument till metoden antas innehålla strängar med de anställdas namn enligt formen "förnamn efternamn". D.v.s. förnamnet följt av ett mellanslag följt av efternamnet.

Den `ArrayList` med användarnamn som genereras ska ha lika många element som den ursprungliga arrayen och namnen ska vara i samma ordning som elementen i den ursprungliga arrayen. Ingen kontroll behöver göras så att dubbletter av användarnamn undviks. Tänk på att för- och efternamn kan bestå av färre än tre bokstäver och det användarnamn som genereras innehåller då alla bokstäver i för- och/eller efternamnet.

Klassen ska även innehålla en statisk metod med namnet `list` som tar två argument. Det första argumentet är en array av typen `String` som innehåller de ursprungliga namnen (för- och efternamn). Det andra argumentet är en `ArrayList` av typen `String` och innehåller de genererade användarnamnen. Metoden ska skriva ut innehållet i båda dessa enligt formen ”förnamn efternamn (användarnamn)”.

Ex) Om vi har följande array och skickar den som argument till metoden enligt:

```
String[] names = {"Bo Ek", "Erik Edström", "Jörgen Svensson"};
ArrayList<String> usernames = Usernames.create(names);
Usernames.list(names, userNames);
```

Skrivs följande ut på skärmen:

```
Bo Ek (boek)
Erik Edström (erieds)
Jörgen Svensson (jörsve)
```

Skriv även en testklass som demonstrerar alla metoder i klassen `Usernames`. Den array du skickar som argument till `create` ska innehålla minst 10 namn.

Uppgift 2

Översikt

Du ska i denna uppgift skriva en klass med vilken vi kan kasta fem tärningar och sen kontrollera om vi har fått Yatzy (alla fem tärningar visar samma värde). Syftet är att du ska lära dig hantera en arrayer och att sortera elementen i en array.

Uppgift

Du har säkert kommit i kontakt med tärningsspelet Yatzy[1] tidigare. I det spelet gäller det att samla poäng genom att kasta fem tärningar och få ihop olika kombinationer, som t.ex. kåk (två av en siffra och tre av en annan), två par eller yatzy. Att få yatzy innebär att alla tärningar visar samma siffra.

Din uppgift är att skriva en klass med namnet `Yatzy` med vilken man ska kunna kasta de fem tärningarna och därefter kontrollera om det blev yatzy eller inte. Tanken är att denna klass ska innehålla en metod för varje typ av kombination som är möjlig att få poäng för, men att du endast ska implementera metoden som kontrollerar yatzy.

Klassen ska innehålla en instansvariabel med namnet `dices`. Detta är en array med plats för fem heltal. Det är i denna array tärningarnas värden sparas efter ett kast. Värdet för tärning 1 lagras i element 1 (index 0), värdet på tärning 2 lagras i element 2 (index 1) osv.

I klassen ska det finnas en metod med namnet `rollDisces` som inte tar några parametrar. När denna metod anropas ska nya värden för alla fem tärningarn slumpas och lagras i arrayen `dices`.

Det ska finnas ännu en metod med namnet `rollDices` (det vill säga en överlagrad metod), men som tar fem parametrar. Alla parametrar är av typen `boolean` och anger om en viss tärning ska kastas eller inte. `true` anger att tärningen ska kastas medan `false` anger att tärningen inte ska kastas. Ex. Om första parametern är `true` ska tärning 1 kastas (ges ett nytt slumpat värde), om andra parametern är `false` ska tärning 2 inte kastas osv.

När ett nytt objekt av klassen skapas ska alla tärningar automatiskt kastas en gång.

Metoden som kontrollen om tärningarnas värden är en yatzy eller inte ska heta `isYatzy` och ska inte ta några parametrar. Denna metod ska returnera en `boolean` vars värde är `true` om det är enyatzy och `false` om det inte är en yatzy.

När man i detta spel kontrollerar alla poänggivande kombinationer kan det underlätta om tärningskastet är sorterad i sjunkande ordning (tärningar med högst värde först). För att kontrollera om vi har ett par (minst två tärningar lika) kan vi då t.ex. sortera tärningarna och jämföra om första tärningen har samma värde som andra tärningen.

[1] https://sv.wikipedia.org/wiki/Yatzy

Om så är fallet har vi ett par (som också är det högsta möjliga paret eftersom tärningarna är sorterade i sjunkande ordning). Om dessa två tärningar inte är lika jämför vi tärning 2 och tärning 3, sen tärning 3 och tärning 4 etc.

Du ska skriva en metod med namnet `sort` som sorterar tärningarna i sjunkande ordning. Metoden ska inte ta några parametrar och inte heller returnera något.

Du ska till sist överskugga metoden `toString` så att denna returnerar en sträng med alla tärningars värden. Exempel på strängar som kan returneras:
"4 5 1 1 3", "1 2 3 4 6", "3 3 3 3 3".

Skriv en testklass där du skapar ett objekt av klassen `Yatzy` och därefter demonstrerar alla metoder i klassen på lämpligt sätt. Exempel på körning:

```
Rolling all the dices...
3 4 6 4 2

Rolling dices 4 and 5…
3 4 6 4 1

Sorting the dices...
6 4 4 3 1

Checking for yatzy...
Sorry, 6 4 4 3 1 in not yatzy!
```

Skriv ytterligare en testklass där du i en loop gör 10 000 000 kast (tio miljoner). Detta värde ska vara enkelt att ändra om vi vill prova med ett annat antal kast. När alla kast gjorts ska det skrivas ut på skärmen hur många gånger det blev yatzy och hur många procent av totala antalet kast detta är. Procenten ska skrivas ut med maximalt 4 decimaler. Du ska även mäta den tid det tar att göra alla kast och kontroll av yatzy.

Som jämförelse kan nämnas att den teoretiska chansen att få yatzy på ett kast är:

$$\left(\frac{1}{6}\right)^5 \cdot 6 = \frac{1}{6^4} = \frac{1}{1296} \approx 0{,}0007716 \approx 0{,}07716\%$$

Exempel på tre olika körningar:

```
Rolling the dices 10000000 times... 836 ms
Number of yatzy: 7622 (0,0762%)

Rolling the dices 10000000 times... 1304 ms
Number of yatzy: 7876 (0,0788%)

Rolling the dices 10000000 times... 885 ms
Number of yatzy: 7391 (0,0739%)
```

Skapa en jar-fil som innehåller alla klasser (.class) du skrivit för denna uppgift. När jar-filen körs är det den andra testklassen som ska startas.