RAFLEBÆGER 3 (DIECUP 3)

I denne opgave skal I endnu en gang arbejde videre med jeres raflebægerprojekt.

Start med at tage en kopi af jeres projekt fra sidste opgave. Så har I altid originalen at falde tilbage på, hvis noget går helt i koks.

Opgave 1

I skal nu generalisere situationen, således at raflebægre kan have et vilkårligt antal terninger (som alle har 6 sider).

Til dette formål erstattes de to feltvariabler ***d1*** og ***d2*** (af typen ***Die***) med en feltvariabel ***dice***, der er en arrayliste af typen ***ArrayList<Die>***.

Sproglig kommentar: På US engelsk (som vi benytter her) betyder *dice* terninger (flertalsform), mens *die* betyder terning (entalsform). På UK engelsk er *dice* både flertals- og entalsform.

Modificér konstruktøren for ***DieCup*** klassen, så den får en heltalsparameter ***noOfDice***, der bestemmer antallet af terninger. Brug en ***if*** sætning til at tjekke, at der er mindst en terning. Hvis det ikke er tilfældet, udskrives en passende besked på terminalen. Konstruktøren skal initialisere feltvariablen ***dice*** til at være en tom arrayliste, og derefter skabe og tilføje det specificerede antal ***Die*** objekter ved hjælp af en ***for*** løkke.

Modificér ***DieCup*** klassens metoder, således at de kan håndtere den nye situation, hvor feltvariablerne ***d1*** og ***d2*** er erstattet af feltvariablen ***dice*** af typen ***ArrayList<Die>***. Her kan I med fordel bruge ***for-each*** løkker.

Er der behov for at modificere ***Die*** klassen?

Opgave 2

Modificér metoderne ***test*** og ***testMultiple*** i ***TestDriver*** klassen, således at parametrene ***sides1*** og ***sides2*** erstattes af en ny parameter ***noOfDice*** (af type ***int***), der bestemmer antallet af terninger. Hvis du lavede spørgsmål 6 Raflebæger 2, skal du også modificere ***compareDieCups*** på tilsvarene vis.

Afprøv de nye metoder på et raflebæger, der har 4 terninger. Diskutér om resultaterne ser fornuftige ud.

Opgave 3

I skal nu afprøve det, som I har lavet i opgave 1, ved at kalde klassemetoden ***test*** i ***TestServer*** klassen med parameteren ”DC3-1”.

Brug testserveren med omtanke. Når I får en fejlrapport, bør I rette *alle* de fejl, der rapporteres og teste, at rettelserne er korrekte, før I atter forsøger at køre testserveren. Hvis I blot retter en enkelt fejl ad gangen (uden selv at teste om rettelsen ser ud til at fungere), kommer I let til at bruge alt for megen tid på at vente på, at testserveren genererer rapporter til jer (specielt hvis der er kø).

Hvis testserveren finder fejl, skal l gennemgå jeres kode og forsøge at rette dem.

Opgave 4

Før I laver denne opgave skal I tage en kopi af jeres projekt, således at I kan aflevere begge projekter til jeres instruktor.

I skal nu (endnu en gang) generalisere situationen, således at de enkelte terninger kan have et vilkårligt antal sider.

Modificér ***DieCup*** klassen konstruktør, så den får formen:

DieCup(ArrayList<Integer> newDice)

hvor parameteren ***newDice*** bestemmer antallet af terninger og antallet af sider for den enkelte terning. For hvert heltal i arraylisten laves en terning med det pågældende antal sider. Det betyder, at en arrayliste med elementerne 4, 6, 3 og 8 giver et raflebæger med 4 terninger, der tilsammen har 21 sider.

Brug en ***if*** sætning til at tjekke, at der er mindst en terning. Hvis det ikke er tilfældet, udskrives en passende besked på terminalen.

Er der behov for at modificere ***DieCup*** klassens metoder, således at de kan håndtere den nye situation? Er der behov for at modificere ***Die*** klassen?

Det er bøvlet at afprøve konstruktøren ved at højreklikke på et ***DieCup*** objekt og kalde ***new*** operatoren – idet der så skal tastes ret meget ind i dialogboksen. I stedet kan I afprøve konstruktøren via ***TestDriver*** klassen, hvor I opbygger en arrayliste på følgende måde:

```
ArrayList<Integer> newDice = new ArrayList<>();
newDice.add(4);
newDice.add(6);
newDice.add(3);
newDice.add(8);
```

Herefter kan I så bruge variablen ***newDice*** som parameterværdi i et kald af konstruktøren.

Opgave 5

I ***TestDriver*** klassen erstattes de eksisterende metoder med nedenstående metode:

void test4638(int noOfRolls)

Metoden skal skabe et raflebæger, der har 4 terninger med siderne 4, 6, 3 og 8, foretage ***noOfRolls*** kast med bægeret og dets fire terninger og udskrive resultatet af disse kast på terminalen på nedenstående form, hvor ***noOfRolls*** er sat til 5.

```
Throw no 1: 13
Throw no 2: 8
Throw no 3: 17
Throw no 4: 3
Throw no 5: 16
Average no of eyes: 11.4
```

Kald ***test4638*** et antal gange med ***noOfRolls*** sat til 1000. Diskutér om resultaterne ser fornuftige ud.

Opgave 6

I skal nu afprøve det, som I har lavet i opgave 4, fungerer korrekt. Kald klassemetoden ***test*** i ***TestServer*** klassen med parameteren ”DC3-4”.

Hvis testserveren finder fejl, skal l gennemgå jeres kode og forsøge at rette dem.

Opgave 7 (til dem der har mod på mere)

Implementér nedenstående metode i ***TestDriver*** klassen:

void splitDigits(int digits)

Metoden skal opsplitte parameterværdien i cifre, og udskrive hvert ciffer på en linje for sig på terminalen (startende med det sidste ciffer).

Til formålet kan det være nyttigt at bruge en ***while*** løkke samt ***/*** og ***%*** operatorerne. Den første operator er heltalsdivision, dvs. at ***26 / 10*** evaluerer til ***2***, mens ***347 / 10*** evaluerer til ***34***. Den sidste operator er modulo operatoren, dvs. at ***26 % 10*** evaluerer til ***6***, mens ***347 % 10*** evaluerer til ***7***. Aftest metoden på nogle simple eksempler.

Omdøb ovenstående metode til ***createArrayList*** og modificér den, således at de fundne cifre, i stedet for at blive udskrevet på terminalen, indsættes som elementer i en arrayliste af heltal, der returneres som metodens resultat.

Opgave 8 (til dem der har mod på mere)

Implementér nedenstående metode i ***TestDriver*** klassen. Hvis du lavede opgave 7 i Raflebæger 2, kan du nu med fordel modificere den metode, som du lavede der (og modificerede i opgave 2).

void compareDieCups(int dc1, int dc2, int noOfRolls)

Hvis I løste opgave 7 i Raflebæger 2, har I allerede løst store dele af indeværende opgave, og I kan derfor arbejde videre på det, som I allerede har lavet.

Metoden skal skabe to raflebægre, hvor det første har de terninger, der er angivet af cifrene i heltallet dc1, mens det andet har de terninger, der er angivet af cifrene i dc2. Dernæst foretages ***noOfRolls*** kast med de to raflebægre, og det udskrives, hvordan raflebægrene ser ud og hvor mange gange hvert af de to raflebægre vinder på nedenstående form:

DieCup 1 with [6, 6, 6] sides is highest: 37 no of times
DieCup 2 with [2, 8, 5, 3] sides is highest: 47 no of times
Same score in both: 16 no of times

Når ***arrayList*** er en arrayliste af heltal, kan du udskrive heltallene i ***arraylist*** på terminalen (adskilt med kommaer og omgivet af firkantede parenteser) ved blot at inkludere ***arrayList*** i et kald, af ***println*** metoden:

System.out.println(… + arrayList + …)

Undersøg hvem der oftest vinder. Et raflebæger med 3 terninger, der alle har 6 sider, eller et raflebæger med 4 terninger, der har henholdsvis 3, 5, 8, og 2 sider. Diskutér om resultatet er som forventet.