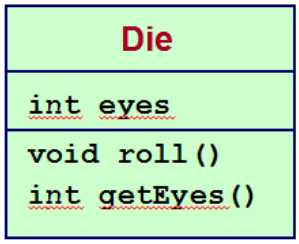
RAFLEBÆGER 1 (DIECUP 1)

I denne opgave tages udgangspunkt i BlueJ-projektet ***DieCup*** (zip). Start med at downloade projektet, og husk at pakke det ud, før I går i gang. Projektet indeholder et basalt skelet, og det er nu jeres opgave at udfylde hullerne i dette skelet. Ignorer ***TestServer*** klassen. Den skal først bruges i opgave 4.

Opgave 1

Klassen ***Die*** repræsenterer en spilleterning. Klassen har to metoder ***roll*** og ***getEyes***. Metoderne skal gøre det muligt, henholdsvis, at slå med termingen og at inspicere, hvor mange øjne terningen p.t. viser (dvs. har på den side, der vender opad). Det skal være muligt at inspicere terningen flere gange mellem hvert kast, dvs. at der er brug for en feltvariabel ***eyes*** til at huske det antal øjne, som terningen aktuelt viser.

Ovenstående kan sammenfattes i følgende UML klassediagram:



Færdiggør metoderne ***roll*** og ***getEyes***. Lav metoderne i små skridt. Start f.eks. med at implementere metoden ***roll***, så der altid slås en sekser. Når du har fået dette til at fungere, tages næste skridt i implementeringen.

For at opnå et realistisk element af tilfældighed i udfaldet af slagene, bruger vi klassen ***Random*** fra Java's klassebibliotek.

Øverst i ***Die*** klassen har vi importeret ***Random*** klassen ved hjælp af sætningen:

import java.util.Random;

Dette giver os adgang til at bruge ***Random*** klassen og de metoder, som klassen stiller til rådighed.

Som I kan se, har vi derefter erklæret en feltvariabel ***random*** af type ***Random***, og initialiseret ***random*** i ***Die*** klassens konstruktør ved hjælp af sætningen:

random = new Random();

Herved skabes et ***Random*** objekt, som bliver assignet til feltvariablen ***random***, som hermed bliver en reference til objektet.

Random klassen indeholder bl.a. nedenstående metode, der returnerer et tilfældigt heltal i intervallet [0,bound) (dvs. et af tallene 0, 1, 2, …, bound-1).

int nextInt(int bound)

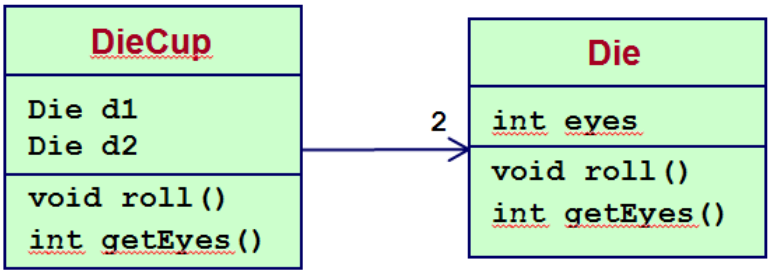
Ovenstående betyder, at udtrykket:

random.nextInt(6) + 1

returnerer et tilfældigt heltal i intervallet [1,6].

Opgave 2

Klassen ***DieCup*** repræsenterer et raflebæger indeholdende to terninger. Klassen har to feltvariabler, ***d1*** og ***d2***, som refererer til (peger på) hver sin terning (dvs. et objekt af typen ***Die***). Derudover har klassen to metoder ***roll*** og ***getEyes***. Metoderne skal gøre det muligt, henholdsvis, at slå med bægeret (med begge terninger) og at inspicere summen af de to terningers øjne.



Færdiggør ***DieCup*** klassens konstruktør, så den skaber to nye ***Die*** objekter ved hjælp af ***new*** operatoren. De to ***Die*** objekter assignes til henholdsvis ***d1*** og ***d2***.

Færdiggør metoderne ***roll*** og ***getEyes***.

Under implementationen af ***DieCup*** klassen skal I ikke tænke på, hvordan ***Die*** klassen er implementeret - kun hvilke metoder den stiller til rådighed.

Opgave 3

Vi vil gerne have raflebægeret til at huske antallet af øjne i det højeste slag, der er slået. Dette kan gøres ved at introducere en ny feltvariabel ***maxEyes***.

Modificér ***roll*** metoden, så ***maxEyes*** opdateres (når det er nødvendigt) og lav to nye metoder ***getMaxEyes*** og ***resetMaxEyes*** til, henholdsvis, at aflæse og nulstille ***maxEyes***.

Opgave 4

I skal nu afprøve det, som I har lavet i opgave 1-3, ved at kalde klassemetoden ***test*** i ***TestServer*** klassen med parameteren "DC1". Dette gøres ved at højreklikke på ***TestServer*** klassen, vælge ***test*** metoden og indtaste "DC1" i den dialogboks, der kommer op (husk anførelsestegnene). Metoden uploader jeres projekt til vores testserver.

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

Opgave 5

Et kast med to terninger, kan give 6 x 6 = 36 forskellige resultater. Summen 2 kan kun opnås på én måde (begge terninger skal vise 1). Det samme gælder for 12 (hvor begge terninger skal vise 6). Det betyder, at 2 og 12 begge har sandsynligheden 1/36 for at forekomme som sum. Regn ud, hvilken sandsynlighed de øvrige tal mellem 2 og 12 har for at forekomme som sum. Regn også ud, hvor meget summen i gennemsnit vil være. Skriv ovenstående i det dokument, der er i BlueJ's øverste venstre hjørne.

Lav 18 kast med raflebægeret, ved at kalde ***roll*** metoden i ***DieCup*** 18 gange, og notér for hvert kast summen af de to terningers øjne. Sammenlign jeres resultater med de sandsynligheder og gennemsnitsværdier, som I beregnede ovenfor.

Opgave 6 (til dem der har mod på mere)

Implementer nedenstående metode i ***DieCup*** klassen. Metoden skal foretage ***noOfRolls*** kast med raflebægeret og udskrive disse på terminalen (med en linje for hvert kast).

void multipleRolls(int noOfRolls)

I kan udskrive på terminalen ved hjælp af ***System.out.println*** metoden, der blev beskrevet i en af forelæsningerne. Til sidst udskrives det gennemsnitlige antal øjne i de foretagne kast, således at terminalen får et udseende, der svarer til nedenstående, hvor ***noOfRolls*** er sat til 5.

```
Throw no 1: 8
Throw no 2: 12
Throw no 3: 7
Throw no 4: 3
Throw no 5: 6
Average no of eyes: 7.2
```

Når man dividerer to heltal med hinanden runder Java ned til nærmeste heltal, dvs. at 36/5 evalurer til heltallet 7. Hvis man i stedet vil have et reelt tal, kan man skrive 1.0 * 36/5, hvilket evaluerer til det reelle tal 7.2.

Kald ***multipleRolls*** et antal gange med ***noOfRolls*** sat til 1000 og diskutér, hvordan resultaterne varierer, og hvordan de svarer til de beregninger, som I lavede i opgave 5. Hvis I ikke kan se alle linjerne vælges *Unlimited buffering* i den *Options* menu, som findes i terminalens øverste venstre hjørne.