

# Övningstenta

Programmering för Språkteknologer I · VT 2010

Lärare: Markus Saers

*Frågorna är uppdelade i G-frågor (ommarkerade), samt VG-frågor (markerade med \*). För betyget G krävs godkänt svara på samtliga G-frågor. För betyget VG krävs godkänt svar på 3 av de 4 VG-frågorna. Det är alltid bättre att lämna in en dellösningen än ingen lösning, eftersom det i så fall kan det räcka med en komplettering istället för omtenta.*

## 1 Grundläggande typer

Lista de grundläggande typerna.

## 2 Kontrollstrukturer

### 2.1

Skriv Javakod som skriver ut alla tal mellan 0 och 30 (inklusive 30 självt) som är jämnt delbara med 3.

### 2.2

Skriv ett program som läser in en `String` från användaren, eller använder det första argumentet från kommandoraden, och sedan skriver ut alla delsträngar som är kortare än 6 tecken. Du avgör själv ifall du vill ha med de tomma strängarna, men du bör dokumentera ditt val.

### 2.3

Vad skriver följande program ut?

```
public class Kontroll {
    enum CardColor { HEARTS, SPADES, DIMONDS, CLUBS }
    public static String cardFace(int face) {
        switch (face) {
            case 11: return "Jack";
            case 12: return "Queen";
            case 13: return "King";
            case 14: return "Ace";
            default: return Integer.toString(face);
        }
    }
    public static void main(String[] args) {
        for (CardColor c : CardColor.values()) {
            for (int i = 2; i < 15; i += 1) {
                System.out.println(cardFace(i)
                    + " of "
                    + c.toString().toLowerCase()
                );
            }
        }
    }
}
```

## 3 Metoder

### 3.1

Deklarera (lämna implementeringen tom) en statisk subrutin som heter `aSubroutin` och tar tre parametrar: ett heltal och två `String`. Skriv en statisk funktion som heter `kvadratrot` och som använder `Math.sqrt` för att beräkna kvadratroten av sin `double`-parameter. Båda dina metoder ska vara synliga utifrån, du behöver inte sätta in dem i någon klass.

### 3.2 \*

Använd dig av programmeringsmetoden stegvis förfining för att lösa följande problem. Skriv en subrutin för varje steg du tar. Problemet du ska lösa är:

Gå igenom alla teckenbigram i en sträng som användaren ger. Räkna alla som börjar med en engelsk vokal (aeiouy), och håll reda på hur många av dessa som slutar på någon av konsonanterna `bdfhklt`. Räkna också hur många teckenbigram som börjar på någon av konsonanterna ovan. Producera en skriftlig rapport över samtliga tre räkneverk.

## 4 Objekt

### 4.1

Skriv en klass som representerar en punkt i en 2D-rymd. Den ska ha två privata attribut: `x` och `y`, och lämpliga `get`- och `set`-metoder för att läsa av och manipulera dem. Den ska ha tre konstruktorer, en som inte har några parametrar (sätt `x` och `y` till lämplig standardvärden), en som har två parametrar, en för `x`-värdet, och en för `y`-värdet (sätt objektets `x`- och `y`-värden till värdet på de inkommande parametrarna), samt en som har en parameter för ett objekt av samma klass (Använd det inkommande objektets `x`- och `y`-värden för att sätta det skapade objektets). Utöver detta ska den ha en `toString`-metod, som skapar en lämplig textrepresentation av punkten, samt en `move`-metod som tar två parametrar: en som beskriver hur långt parametern ska flyttas i `x`-led, och en som beskriver hur långt den ska flyttas i `y`-led. Du väljer själv vilken typ attributen `x` och `y`, men behöver motivera ditt val genom att ange ett lämpligt användningsområde.

### 4.2 \*

Lägg till två statiska attribut, `DEFAULT_X` och `DEFAULT_Y` som används för att ge standardvärden när den parameterlösa konstruktorn används. Det finns två strategier, antingen får de vara synliga och konstanta, eller så får de vara privata och ha lämpliga `get`- och `set`-metoder. Visa hur båda strategierna kan implementeras.

### 4.3 \*

Lägg till en metod som anpassar punkten till en rymd med givna dimensioner. Om punkten befinner sig utanför ska den flyttas så att den "wrappas" runt dimensionerna. Om dimensionerna bara tillåter `x`-värden mellan 1 och 100 och punkten befinner sig på position 125, ska den korrigeras så att den nu befinner sig på position 25. Metoden ska kunna ta fyra värden (högsta, vänstraste, lägsta och högraste), eller två punkter (högsta-vänstraste och lägsta-högraste) som parametrar. Använd överlagring, och var lat.

## 5 Objektorienterad programmering

### 5.1

Beskriv med ord och (om du vill) pseudokod hur ett program som räknar frekvens på olika ord-n-gram skulle kunna se ut. Du ska redogöra för vilka byggstenar (klasser/objekt) du tycker dig se i problemet, samt vad de ska kunna göra (omvandla en ström av ord till ord-n-gram, associera ett ord-n-gram med ett tal, osv). Du ska också redogöra för hur dessa byggstenar ska kopplas ihop för att skapa en ord-n-gram-frekvenslista givet en text.

### 5.2 \*

Gör ett bra job på 5.1. Om du löser ovanstående uppgift väl kan jag tänka mig att räkna den som en VG-uppgift.

## 6 Arrayer

### 6.1

Skriv en statisk metod som konkatenerar två teckenarrayer. Om du skrivit metoden

```
public static char[] conc(char[] a, char[] b)
```

och har två char []

```
char[] c1 = new char[]{'a', 'b'};
```

```
char[] c2 = new char[]{'c', 'd'};
```

så ska anropet `conc(c1, c2)` resultera i arrayen `['a', 'b', 'c', 'd']`.

### 6.2

Skriv en statisk metod som ger en multiplikationsmatris för en sifferarray. Matrisen ska vara lika stor på höjden och bredden som sifferarrayen är lång. Varje cell (på rad  $r$  och kolumn  $k$ ) ska ha värdet  $sa[r] * sa[k]$ , där  $sa$  är sifferarrayen. Med input `[1, 2, 3]` kommer alltså följande matris byggas upp:

```
1 2 3
2 4 6
3 6 9
```

Om man ger metoden en sifferarray med siffrorna 1 till 9 ger den en multiplikationstabell.