



Programmering för Språkteknologer I

Markus Saers
 markus.saers@lingfil.uu.se
 9-2040
 stp.ling.uu.se/~markuss/vt10/pst1



Inlämningsuppgift 4

- Finns uppe!
- Inlämning senast en vecka efter tentan (9/6)



Idag

- Arrayer
 - En variabel som håller flera värden
- Gränssnitt
 - Ett sätt att garantera att objekt av en klass kan klara av en viss uppgift
- Exempeltenta/Inlupparna



Arrayer

- Låter oss samla flera värden av samma typ i en variabel
- Index används för att komma åt enskilda värden



Exempel

```
int[] a = new int[4];
a[0] = 1;
a[1] = 10;
a[2] = 100;
a[3] = 1000;
for (int i = 0; i < a.length; i++) {
    System.out.println(a[i]);
}
```

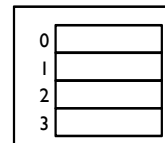
int[] a



Exempel

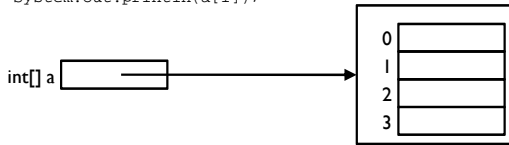
```
int[] a = new int[4];
a[0] = 1;
a[1] = 10;
a[2] = 100;
a[3] = 1000;
for (int i = 0; i < a.length; i++) {
    System.out.println(a[i]);
}
```

int[] a



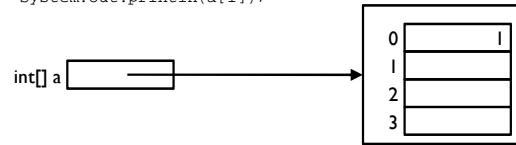
Exempel

```
int[] a = new int[4];
a[0] = 1;
a[1] = 10;
a[2] = 100;
a[3] = 1000;
for (int i = 0; i < a.length; i++) {
    System.out.println(a[i]);
}
```



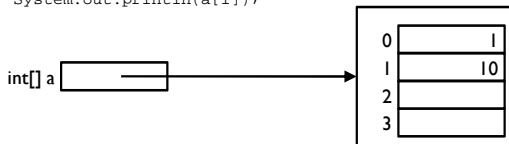
Exempel

```
int[] a = new int[4];
a[0] = 1;
a[1] = 10;
a[2] = 100;
a[3] = 1000;
for (int i = 0; i < a.length; i++) {
    System.out.println(a[i]);
}
```



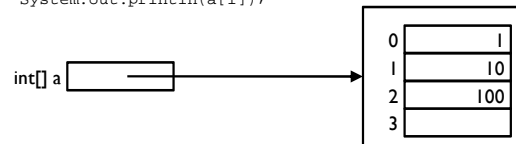
Exempel

```
int[] a = new int[4];
a[0] = 1;
a[1] = 10;
a[2] = 100;
a[3] = 1000;
for (int i = 0; i < a.length; i++) {
    System.out.println(a[i]);
}
```



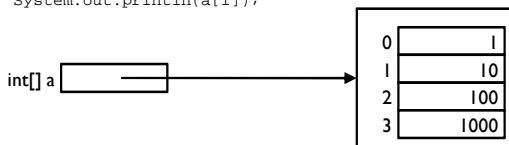
Exempel

```
int[] a = new int[4];
a[0] = 1;
a[1] = 10;
a[2] = 100;
a[3] = 1000;
for (int i = 0; i < a.length; i++) {
    System.out.println(a[i]);
}
```



Exempel

```
int[] a = new int[4];
a[0] = 1;
a[1] = 10;
a[2] = 100;
a[3] = 1000;
for (int i = 0; i < a.length; i++) {
    System.out.println(a[i]);
}
```



Arrayer

- Kan ha flera dimensioner
`int[][] b = new int[10][5];`
- Går att lagra värden av "mindre typ"
`short s = 12;`
`b[0][0] = s;`
- Det som kommer in till main-metoden är en array med `String` som motsvarar de argument som gavs vid kommandoprompten
- Kan användas i for-each-loopar
`for (int i : a) { /* gör något med i */ }`



Gränssnitt

- Gränssnittet för ett objekt utgörs av alla publika metoder
- I Java kan man ha frikopplade gränssnitt som beskriver en funktionalitet
 - Klasser kan sedan välja att implementera den funktionaliteten
 - Gör att objekt av klassen kan fungera i alla sammanhang där något som implementerar gränssnittet behövs
 - Viktig del i inlupp 4



Vad är ett gränssnitt?

```
public class StringBiGram {
    private String ord1;
    private String ord2;

    public StringBiGram() {
        ord1 = "";
        ord2 = "";
    }

    public int n() {
        return 2;
    }

    public String get(int position) {
        if (position == 0) return ord1;
        if (position == 1) return ord2;
        return null;
    }
}
```



Vad är ett gränssnitt?

```
public class StringBiGram {
    private String ord1;
    private String ord2;

    public StringBiGram() {
        ord1 = "";
        ord2 = "";
    }

    public int n() {
        return 2;
    }

    public String get(int position) {
        if (position == 0) return ord1;
        if (position == 1) return ord2;
        return null;
    }
}
```



Formaliserat gränssnitt

```
public interface StringNGram {
    /**
     * Returns the size of the n-gram.
     * @return the size of the n-gram.
     */
    int n();

    /**
     * Returns the String at the given position.
     * @param position The position to look at.
     * @return The String at the position.
     */
    String get(int position);
}
```



Klass som implementerar ett gränssnitt

```
public class StringBiGram implements StringNGram {
    private String ord1;
    private String ord2;

    public StringBiGram() {
        ord1 = "";
        ord2 = "";
    }

    public int n() {
        return 2;
    }

    public String get(int position) {
        if (position == 0) return ord1;
        if (position == 1) return ord2;
        return null;
    }
}
```



Varför?

- Man kan skriva komplicerade klasser som gör saker med objekt som implementerar ett bestämt gränssnitt
- Någon utifrån behöver bara skriva en egen klass som implementerar gränssnittet för att kunna använda de komplicerade klasserna!



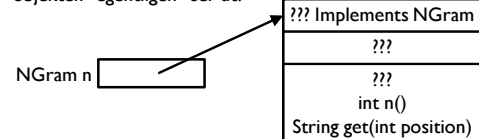
Inlupp 4

- Jag har skrivit ett gränssnitt för tecken-n-gram, samt några klasser som gör det möjligt för er att räkna frekvenser på tecken-n-gram
- Genom att implementera gränssnittet garanterar ni att er tecken-n-gram-klass kan användas för att räkna frekvenser på tecken-n-gram!



Hur går det till?

- Från mitt programs synvinkel:
 - Jag kan skapa variabler som håller objekt som implementerar gränssnittet `Ngram`
 - Dessa objekt är garanterade att ha metoderna `int n()` och `String get(int position)`
 - Så länge jag klarar mig med dessa spelar det ingen roll hur objekten "egentligen" ser ut!



Gränssnitt

- Ett välkommenterat `interface` beskriver hur de framtida byggstenarna ska se ut för att passa in
- Beskriver ett "kontrakt"
- Alla som följer kontraktet får vara med!
 - Kontrolleras till viss del av compilatorn