TDA 545: Objektorienterad programmering

Föreläsning 8:
Exempel och problemlösning

Magnus Myréen

Chalmers, läsperiod 1, 2015-2016

De tre senaste föreläsningarna

Läsanvisning: kap 2 & 13

- meddelanden och metoder
- informationsdöljande och inkapsling
- skapa och använda färdiga objekt !
- primitiva variabler kontra objektvariabler
- 3 tester på likhet
- metoder
- fält (arrays)
- att konstruera en klass
- Javadoc

Nu har vi en föreläsning som består av bara exempel, t.ex. hur man tolkar felutskrifter från kompilatorn.

Idag

Nu har vi en föreläsning som består av bara exempel, t.ex. hur man tolkar felutskrifter från kompilatorn.

Observera att när man utvecklar program så är inte alla lösningar "på vägen" korrekta.

Idag diskuterar vi (interaktivt!) problemlösning i programmering.

Denna föreläsning är repetition av tidigare material.

En tentauppgift med lösning: Skriv en metod

`void columnSort(int[][] x)`

som givet en matris x sorterar varje **kolumn**. Skriv också en metod `print(int[][] x)` som skriver ut en matris. Skriv sedan kod som skapar första matrisen nedan, anropar `columnSort` och sedan `print` som då skriver den andra matrisen nedan.

Exempel: om x är första 5x7 matrisen nedan så skall metoden `columnSort` arrangera matrisen som andra matrisen nedan.

12	6	7	17	18	19	8
0	14	8	15	5	3	2
7	2	1	6	9	18	21
1	5	9	3	7	11	2
89	12	6	1	0	19	27

skall bli:

0	2	1	1	0	3	2
1	5	6	3	5	11	2
7	6	7	6	7	18	8
12	12	8	15	9	19	21
89	14	9	17	18	19	27

Låter detta svårt?

Ifall det gör det lönar det sig att börja med att försöka lösa ett lättare problem.

Sortering

Hur sorterar man ett fält av ints?

```
int[] a = { 5, 1, 2, 6, 3, 9 };
```

Läs om bubble sort på nätet, t.ex. titta på en video om bubble sort.

https://www.youtube.com/watch?v=lyZQPjUT5B4

Sortering

Hur sorterar man ett fält av ints?

Denna kod byter två element i fältet.

```
int tmp0 = a[i];
int tmp1 = a[i+1];
a[i] = tmp1;
a[i+1] = tmp0;
```

Sortering

Hur sorterar man ett fält av ints?

Denna kod byter två element i fältet, ifall de inte är i ordning.

```
if (a[i] > a[i+1]) {
    int tmp0 = a[i];
    int tmp1 = a[i+1];
    a[i] = tmp1;
    a[i+1] = tmp0;
}
```

Sortering

Hur sorterar man ett fält av ints?

Denna kod bubblar upp det största elementet till slutet av fältet.

```
for (int i=0; i<a.length-1; i++) {
    if (a[i] > a[i+1]) {
        int tmp0 = a[i];
        int tmp1 = a[i+1];
        a[i] = tmp1;
        a[i+1] = tmp0;
    }
}
```

Sortering

Hur sorterar man ett fält av ints?

Denna kod sorterar fältet genom att köra a.length-1 “bubblingar”.

```
for (int j=0; j<a.length-1; j++) {
    for (int i=0; i<a.length-1; i++) {
        if (a[i] > a[i+1]) {
            int tmp0 = a[i];
            int tmp1 = a[i+1];
            a[i] = tmp1;
            a[i+1] = tmp0;
        }
    }
}
```

Sortering

Hur sorterar man ett fält av ints?

Denna kod sorterar fältet genom att köra a.length-1 “bubblingar”.

```
for (int j=0; j<a.length-1; j++) {
    for (int i=0; i<a.length-1; i++) {
        if (a[i] > a[i+1]) {
            int tmp0 = a[i];
            int tmp1 = a[i+1];
            a[i] = tmp1;
            a[i+1] = tmp0;
        }
    }
}
```

Hur vet vi om koden är korrekt?

Sortering

Hur sorterar man ett fält av ints?

Vi kan skriva ut lite debug data för att testa programmet.

```
System.out.print("a = { ");
for (int i=0; i<a.length; i++) {
  System.out.print(a[i] + " ");
}
System.out.println("}");
```

Exempel utskrift:

```
a = { 5 1 2 6 3 9 }
```

Sortering av ett fält

Hela koden:

```
public class Test {

    public static void print(int[] a) {
        System.out.print("a = { ");
        for (int i=0; i<a.length; i++) {
          System.out.print(a[i] + " ");
        }
        System.out.println("}");
    }

    public static void main(String[] args) {

        int[] a = { 5,1,2,6,3,9 };

        print(a);

        for (int j=0; j<a.length-1; j++) {
            for (int i=0; i<a.length-1; i++) {
                if (a[i] > a[i+1]) {
                    int tmp0 = a[i];
                    int tmp1 = a[i+1];
                    a[i] = tmp1;
                    a[i+1] = tmp0;
                }
            }
        }

        print(a);

    }
}
```

En tentauppgift med lösning: Skriv en metod

`void columnSort(int[][] x)`

som givet en matris x sorterar varje **kolumn**. Skriv också en metod `print(int[][] x)` som skriver ut en matris. Skriv sedan kod som skapar första matrisen nedan, anropar `columnSort` och sedan `print` som då skriver den andra matrisen nedan.

Exempel: om x är första 5x7 matrisen nedan så skall metoden `columnSort` arrangera matrisen som andra matrisen nedan.

12	6	7	17	18	19	8
0	14	8	15	5	3	2
7	2	1	6	9	18	21
1	5	9	3	7	11	2
89	12	6	1	0	19	27

skall bli:

0	2	1	1	0	3	2
1	5	6	3	5	11	2
7	6	7	6	7	18	8
12	12	8	15	9	19	21
89	14	9	17	18	19	27

Sortering av kolumner i matris

Hur representerar man matriser i Java?

```
int[][] a = {
    { 12,  6,  7, 17, 18, 19,  8 },
    {  0, 14,  8, 15,  5,  3,  2 },
    {  7,  2,  1,  6,  9, 18, 21 },
    {  1,  5,  9,  3,  7, 11,  2 },
    { 89, 12,  6,  1,  0, 19, 27 }
};
```

Sortering av kolumner i matris

Vi börjar med att skriva en print metod:

for går genom alla kolumner

for går genom alla element i raden

skriver ut ett element

byter till nästa rad

```
public static void print(int[][] x) {
    for (int i=0; i<x.length; i++) {
        System.out.print("Rad " + i + ": ");
        for (int j=0; j<x[i].length; j++) {
            System.out.print(x[i][j] + " ");
        }
        System.out.println("");
    }
}
```

Exempel utskrift:

```
Rad 0: 12 6 7 17 18 19 8
Rad 1: 0 14 8 15 5 3 2
Rad 2: 7 2 1 6 9 18 21
Rad 3: 1 5 9 3 7 11 2
Rad 4: 89 12 6 1 0 19 27
```

Sortering av kolumner i matris

Så här kan vi ändra koden för sortering av ett fält till kod som sorterar den första kolumnen.

```
for (int j=0; j<a.length-1; j++) {
    for (int i=0; i<a.length-1; i++) {
        if (a[i][0] > a[i+1][0]) {
            int tmp0 = a[i][0];
            int tmp1 = a[i+1][0];
            a[i][0] = tmp1;
            a[i+1][0] = tmp0;
        }
    }
}
```

Sortering av kolumner i matris

Med en for-loop kan vi få alla kolumner sorterade.

```
for (int k=0; k<a[0].length; k++) {
    for (int j=0; j<a.length-1; j++) {
        for (int i=0; i<a.length-1; i++) {
            if (a[i][k] > a[i+1][k]) {
                int tmp0 = a[i][k];
                int tmp1 = a[i+1][k];
                a[i][k] = tmp1;
                a[i+1][k] = tmp0;
            }
        }
    }
}
```

Sortering av kolumner i matris

… och så här sätter vi koden in i en metod.

```
public static void columnSort(int[][] a) {

    for (int k=0; k<a[0].length; k++) {
        for (int j=0; j<a.length-1; j++) {
            for (int i=0; i<a.length-1; i++) {
                if (a[i][k] > a[i+1][k]) {
                    int tmp0 = a[i][k];
                    int tmp1 = a[i+1][k];
                    a[i][k] = tmp1;
                    a[i+1][k] = tmp0;
                }
            }
        }
    }

}
```

Svar på tentafrågan

Så här kör vi programmet:

```
$ javac Test2.java
$ java Test2
```

```
Rad 0: 12 6 7 17 18 19 8
Rad 1: 0 14 8 15 5 3 2
Rad 2: 7 2 1 6 9 18 21
Rad 3: 1 5 9 3 7 11 2
Rad 4: 89 12 6 1 0 19 27

Rad 0: 0 2 1 1 0 3 2
Rad 1: 1 5 6 3 5 11 2
Rad 2: 7 6 7 6 7 18 8
Rad 3: 12 12 8 15 9 19 21
Rad 4: 89 14 9 17 18 19 27
```

Hela koden:

```
public class Test2 {

    public static void print(int[][] x) {
        System.out.println();
        for (int i=0; i<x.length; i++) {
            System.out.print("Rad " + i + ": ");
            for (int j=0; j<x[i].length; j++) {
                System.out.print(x[i][j] + " ");
            }
            System.out.println();
        }
    }

    public static void columnSort(int[][] a) {

        for (int k=0; k<a[0].length; k++) {
            for (int j=0; j<a.length-1; j++) {
                for (int i=0; i<a.length-1; i++) {
                    if (a[i][k] > a[i+1][k]) {
                        int tmp0 = a[i][k];
                        int tmp1 = a[i+1][k];
                        a[i][k] = tmp1;
                        a[i+1][k] = tmp0;
                    }
                }
            }
        }

    }

    public static void main(String[] args) {

        int[][] a = {
            { 12,  6,  7, 17, 18, 19,  8 },
            {  0, 14,  8, 15,  5,  3,  2 },
            {  7,  2,  1,  6,  9, 18, 21 },
            {  1,  5,  9,  3,  7, 11,  2 },
            { 89, 12,  6,  1,  0, 19, 27 }
        };
        columnSort(a);
        print(a);

    }
}
```

Implementera en kö

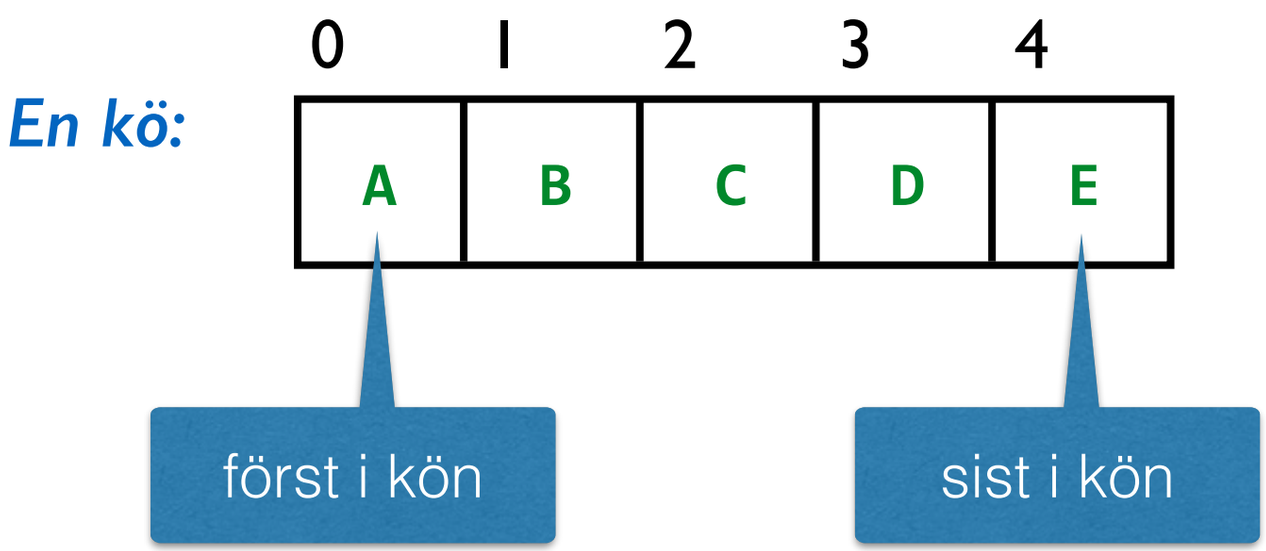
```
/** This class implements a queue of objects. New elements can join at
 * the back of the queue (method: put). Elements are removed from the
 * front of the queue (method: pop).
 */
public class Queue {

    /** Constructs an empty queue. */
    public Queue () {
    }

    /** Returns true if the queue is empty. */
    public boolean isEmpty () {
    }

    /** Returns the first element in the queue, if the queue is
     * non-empty. */
    public Object getFirst () {
    }

    /** Adds an element to the back of the queue. */
    public void put (Object o) {
    }

    /** Removes the first element from the queue, if the queue is
     * non-empty.. */
    public void pop () {
    }

}
```

Implementera en kö

Man måste alltid ha ***en idé*** för hur koden ska fungera ***innan man börjar skriva***.

Idé: En kö kan implementeras av en array (ett fält).

En kö:



Implementera en kö

```
public class Queue1 {

    private Object[] q;

    /** Constructs an empty queue. */
    public Queue1 () {
        q = new Object[0];
    }

    /** Returns true if the queue is empty. */
    public boolean isEmpty () {
        return (q.length == 0);
    }

    /** Returns the first element in the queue, if the queue is
     * non-empty. */
    public Object getFirst () {
        return q[0];
    }

    /** Adds an element to the back of the queue. */
    public void put (Object o) {
        Object[] tmp = new Object[q.length+1];
        for (int i=0; i<q.length; i++) {
            tmp[i] = q[i];
        }
        tmp[q.length] = o;
        q = tmp;
    }

    /** Removes the first element from the queue, if the queue is
     * non-empty. */
    public void pop () {
        if (!(isEmpty())) {
            Object[] tmp = new Object[q.length-1];
            for (int i=1; i<q.length; i++) {
                tmp[i-1] = q[i];
            }
            q = tmp;
        }
    }

    public String toString() {
        String str = "";
        for (int i=0; i<q.length; i++) {
            str = str + " " + q[i];
        }
        return str;
    }

    /** Test */
    public static void main(String[] args) {
        Queue1 t = new Queue1();
        t.put("A");
        t.put("B");
        t.put("C");
        System.out.println(t.toString());
        t.put("D");
        System.out.println(t.toString());
        t.pop();
        System.out.println(t.toString());
        t.pop();
        System.out.println(t.toString());
    }
}
```

internt tillstånd har arrayn som representerar kön

utgångsvärdet är en tom array

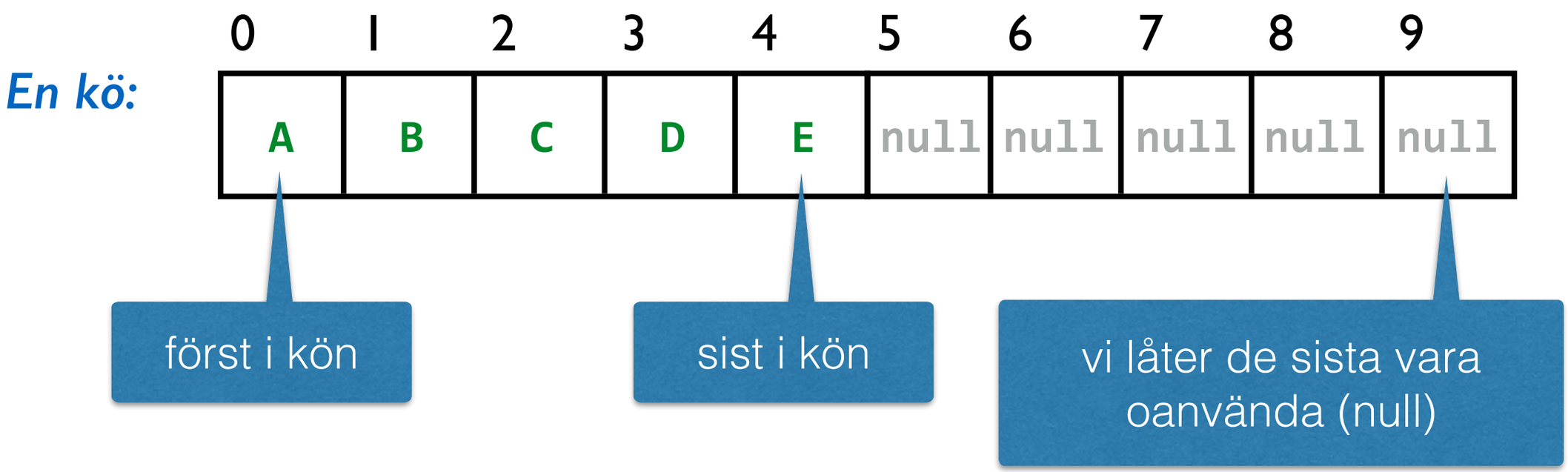
kopierar till ny array

kopierar till ny array

main testar koden

Bättre implementation (mindre kopering)

Ny idé: En kö kan implementeras av en array (ett fält).



Vi behöver ny en variabel som visar hur mycket av arrayn som vi använder.

Implementera en kö

```
public class Queue {

    private Object[] q;
    private int l;

    /** Constructs an empty queue. */
    public Queue () {
        q = new Object[10];
        l = 0;
    }

    /** Returns true if the queue is empty. */
    public boolean isEmpty () {
        return (l == 0);
    }

    /** Returns the first element in the queue, if the queue is
     * non-empty. */
    public Object getFirst () {
        return q[0];
    }

    /** Adds an element to the back of the queue. */
    public void put (Object o) {
        if (l < q.length) {
            q[l] = o;
            l = l+1;
        } else {
           Object[] tmp = new Object[q.length+10];
            for (int i=0; i<q.length; i++) {
                tmp[i] = q[i];
            }
            tmp[q.length] = o;
            q = tmp;
            l = l+1;
        }
    }

    /** Removes the first element from the queue, if the queue is
     * non-empty. */
    public void pop () {
        if (!(isEmpty())) {
            for (int i=1; i<l; i++) {
                q[i-1] = q[i];
            }
            l = l-1;
            q[l] = null;
        }
    }

    public String toString() {
        String str = "";
        for (int i=0; i<l; i++) {
            str = str + " " + q[i];
        }
        return str;
    }

    /** Test */
    public static void main(String[] args) {
        Queue t = new Queue();
        t.put("A");
        t.put("B");
        t.put("C");
        System.out.println(t.toString());
        t.put("D");
        System.out.println(t.toString());
        t.pop();
        System.out.println(t.toString());
        t.pop();
        System.out.println(t.toString());
    }
}
```

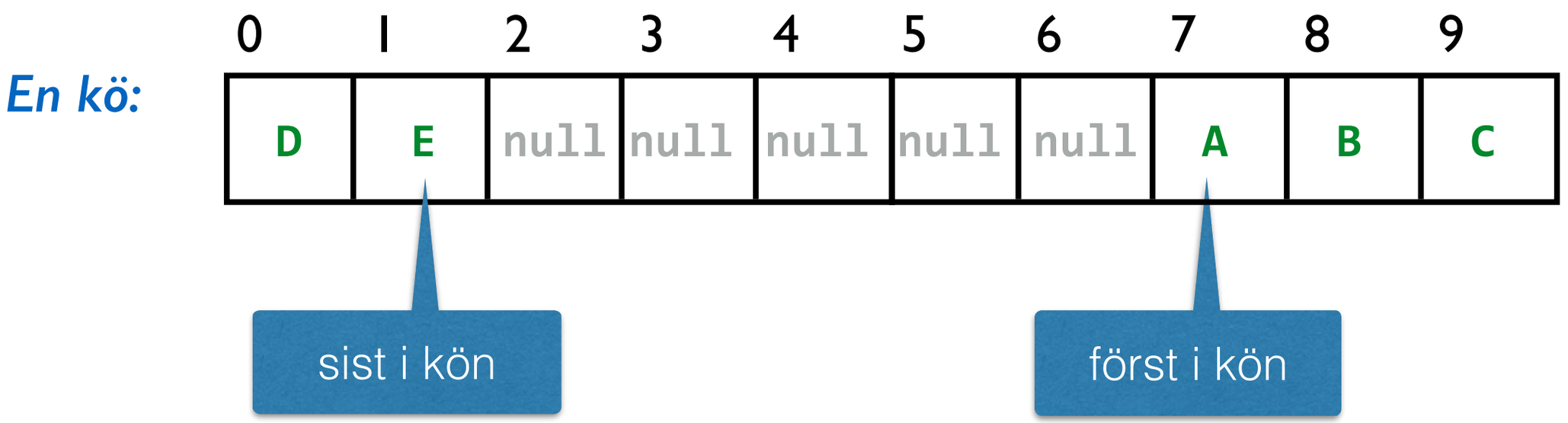
ny variabel l

kopierar *inte alltid*

main testar koden

Ännu bättre

Ny idé: samma som förr men låt början av kön åka runt.



Det fina med det här är att *vi aldrig behöver kopiera i pop-metoden*.

Implementeringstips: använda modulus, dvs %, för att komma åt elementen.

Personnummer

Skriv ett program som beräknar kontrollsiffran till ett inmatat personnummer (9 siffror).