
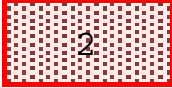
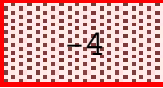
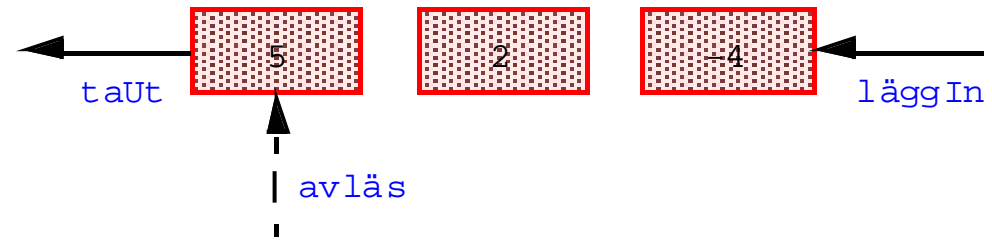


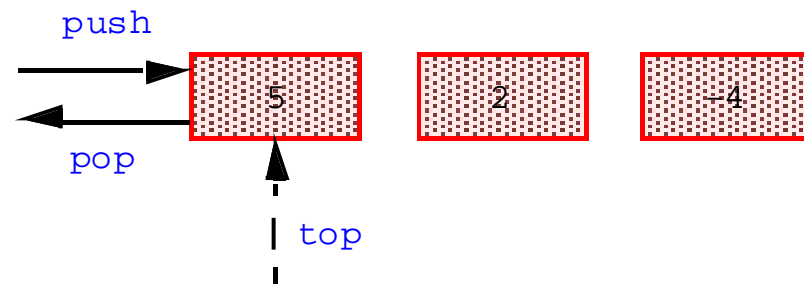
Lista

			
Index:	0	1	2

Kö



Stack



Standardklasser

```
import java.util.*;           // innehåller bl.a. listklasserna

List<String> ls = new LinkedList<String>();
List<Punkt> lp = new LinkedList<Punkt>();
```

Enklare:

```
List<String> ls = new LinkedList<>(); // från Java version 7, "diamant"
List<Punkt> lp = new LinkedList<>();
```

`List` är ett *generiskt* gränssnitt (interface). Bestämmer egenskaperna sett utifrån.
`LinkedList` är en *generisk* klass. Innehåller implementeringen.

Annan klass möjlig:

```
List<String> ls = new ArrayList<>();
List<Punkt> lp = new ArrayList<>();
```

Lägga in, ändra, avläsa och ta ut element:

```
lp.add(new Punkt(5, 7));  
ls.add("EEE");  
lp.add("ABC");          // FEL!!  
ls.add("VVV");          // "EEE", "VVV"  
ls.add("XXX");          // "EEE", "VVV", "XXX"  
ls.add(0, "JJJ");       // "JJJ", "EEE", "VVV", "XXX"  
ls.set(1, "AAA");       // "JJJ", "AAA", "VVV", "XXX"  
ls.remove(3);           // "JJJ", "AAA", "VVV"  
String t = ls.get(1);   // t får värdet "AAA"  
ls.add(t);              // "JJJ", "AAA", "VVV", "AAA"
```

Skapa kopia av en lista:

```
List<String> ls2 = new ArrayList<String>(ls);
```

Löpa igenom en lista:

```
for (int i = 0; i<ls.size(); i++) // sämre lösning
    System.out.println(ls.get(i));
```

```
for (String s : ls) // förenklat skrivsätt, bättre
    System.out.println(s);
```

Välja ut en del av en lista:

```
for (String s : ls.subList(1,3))
    System.out.println(s);
```

```
ls.subList(1, 3).clear(); // ls blir "JJJ", "AAA"
```

Köer

```
LinkedList<Person> kö = new LinkedList<>();  
kö.add(new Person("Mikael", "Olsson", 1990));  
Person p = kö.remove(0);
```

Extra metoder i klassen LinkedList:

```
kö.addLast(new Person("Mikael", "Olsson", 1990));  
Person p = kö.removeFirst();
```

Stackar

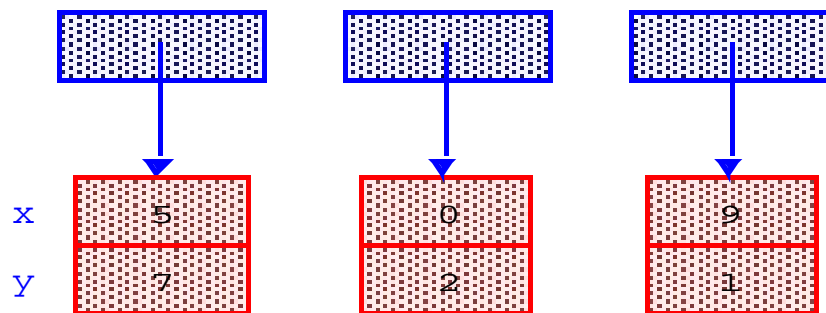
```
LinkedList<Person> stack1 = new LinkedList<>();  
stack1.addFirst(new Person("Mikael", "Olsson", 1990));  
Person p = stack1.removeFirst();
```

Alternativ, använd klassen `Stack`

```
Stack<Person> stack2 = new Stack<>();  
stack2.push(new Person("Mikael", "Olsson", 1990));  
Person p = stack2.pop();
```


Listor med objekt

```
List<Punkt> lp = new LinkedList<>();  
lp.add(new Punkt(5, 7));  
lp.add(new Punkt(0, 2));  
lp.add(new Punkt(9, 1));
```

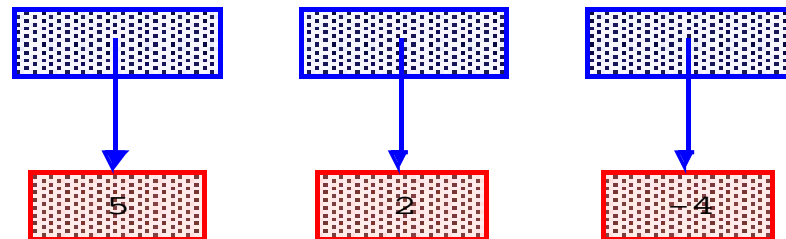


Listor med enkla element

```
List<int> intl = new LinkedList<>(); // FEL!!
```

Använd en *wrapper class*:

```
List<Integer> li = new LinkedList<>();  
li.add(9); // 9  
li.add(0, 2); // 2, 9  
li.add(0, 5); // 5, 2, 9  
li.add(2, 7); // 5, 2, 7, 9  
li.set(3, -4); // 5, 2, 7, -4  
li.remove(2); // 5, 2, -4  
  
int j = li.get(2); // j får värdet -4
```



Listor som parametrar

```
public static void printList(List<Integer> l) {  
    for (Integer i : l)  
        System.out.print(i + " ");  
    System.out.println();  
}
```

```
public static void nollBort(List<Integer> l) {  
    int k;  
    while ((k = l.indexOf(0)) >= 0)  
        l.remove(k);  
}
```