Index of /~cs61b/lectures/06

```
   Name                    Last modified     Size  Description
```

```
   Parent Directory       24-Sep-98 21:22      -
```

```
                     CS61b: Lecture 6
                    Friday, September 4
ANNOUNCEMENTS:
-------------
  Discussion 123 is at 5-6pm on Thursday (not 2-3pm).  It is in 320 Soda.
  Lab 24, 11am-1pm on Wednesday.  Not yet in Telebears, discussion TBD.
  Concurrent enrollment students see me after class.
  Because of the holiday, none of the Monday labs will meet
    The lab assignment online by Tuesday AM.  Monday labs need to do
    this during the week and have it checked off the following
    week **at the beginning of lab**.
  Project 1 and hw3 will go online soon.  Watch the home page and newsgroup.
    First step for pj1 -- find one or two project partners as soon as possible.
  Lab sections 24 should be added officially by next week.
    Note: section 24 is NOT full.
  Anyone who is still on the waiting list should attend the 8-10pm
   lab on Tuesday, or a "temporary overflow" lab from 1-3pm on Wednesday.

VECTORS
-------
So far, we have looked at individual objects, but no lists or other containers.
The Java library has a "Vector" class for holding other objects.

import java.util.*;

Vector v = new Vector();  // an empty vector
v.addElement(new Integer(1));
v.addElement(new Integer(2));
v.addElement(new Integer(1));

index    0      1     2
       -------------------
       |     |     |     |
       ---|-----|-----|---
          v     v     v
        -----  -----  -----
        | 1 |  | 2 |  | 1 |
        -----  -----  -----

    v.size()               // will return the size of v
    v.elementAt(2)         // return the value at index 2, which is the Integer 1
    v.setElementAt(new Integer(3), 2)
                           // replace object at index 2 with object containing 3
    v.addElement(new Integer(4))       // add an object containing 4 to the end

WHAT VECTORS CONTAIN
--------------------
Vector only contain objects, i.e., instances of classes, not primitive types.
```

The cay be heterogeneous, i.e., contain a bunch of types.
When you get something out of a vector, you may need to cast it:

```
      System.out.println("Element at index 2: " + v.elementAt(2)); // OK
      v.elementAt(1).equals(v.elementAt(2))  // OK (it's false)
      v.elementAt(1).intValue()              // doesn't compile
      ((Integer) v.elementAt(1)).intValue()  // OK
```

In the last line, the "(Integer)" part of the expression is called a cast. It is needed because Java doesn't (always) know what kind of object was stored in the vector.

ENUMERATIONS

A common thing to do on vectors is to examine all the elements. To do that (with vectors, or other containers) you build what is called an "Enumeration". The Enumerations remembers:

- what container it is looking at
- which was the last element
- whether there are any more

```
    Enumeration enum = v.elements();
      while (enum.hasMoreElements()) {
        System.out.println("Another element: " +  enum.nextElement());
    }
```

As with the "v.elementAt()" if you use "enum.nextElement()" you may need a cast.

Note: the vector v is not changed by the Enumeration.

WHILE LOOPS

In general, any boolean condition can be used in a while loop.
It will execute until the condition is false.

```
     int i = 0;
     while (i < v.size()) {
       System.out.println("Another element: " + v.elementAt(i));
       i = i + 1;
     }
```

PERFORMANCE OF VECTORS

In Scheme you often used linked lists, which had the same operations as vectors.
     '(),   (set-cdr! l (cons 4 '())),   (nth l)

You can build linked lists in Java too, which look essentially the same.

How are the two types different?
  1) What do they look like in memory? How to draw them?
  2) The elementAt() operation on vectors is constant time, O(1)
     The same operation on linked lists is linear time, O(n)

     What is the cost of the above while loop above, while (i < v.size())...
     with vectors vs. lists (assuming you changed the opns to be list opns)
  3) How long does it take to insert something into a vector? A list?

IMPLEMENTING A SET CLASS USING VECTORS

```
class Set {
  ...
  private Vector vector;
}
```

How are sets different than vectors:
  - they are unordered, so "elementAt" and "setElementAt" don't make sense
  - they do not contain duplicate elements

Consider the union operation:

```
        public Set union(Set s) {
                Set unionSet = (Set) s.copy();
                Enumeration enum = vector.elements();
                while (enum.hasMoreElements())
                        unionSet.addElement(enum.nextElement());
                return unionSet;
        }
```