```
 1: // $Id: testtrees.java,v 1.2 2014-02-13 12:53:40-08 - - $
 2:
 3: import static java.lang.System.*;
 4:
 5: class testtrees {
 6:
 7:    static class printer implements visitor<String> {
 8:       public void visit (String item) {
 9:          out.printf ("%s%n", item);
10:       }
11:    }
12:
13:    static class find_longest implements visitor<String> {
14:       String longest = "";
15:       public void visit (String item) {
16:          if (longest.length() < item.length()) longest = item;
17:       }
18:    }
19:
20:    public static void main (String[] args) {
21:       String[] arguments = new String [args.length];
22:       for (int itor = 0; itor < args.length; ++itor) {
23:          arguments[itor] = "args[" + itor + "]=\"" + args[itor] + "\"";
24:       }
25:       tree<String> the_tree = new tree<String> (arguments);
26:       the_tree.visit (new printer ());
27:       find_longest longest = new find_longest();
28:       the_tree.visit (longest);
29:       out.printf ("The longest string is \"%s\"%n", longest.longest);
30:    }
31:
32: }
```

```
 1: // $Id: tree.java,v 1.1 2014-01-16 17:43:14-08 - - $
 2:
 3: class tree<item_t> {
 4:
 5:    private class node {
 6:       item_t item;
 7:       node left;
 8:       node right;
 9:    }
10:
11:    private node root;
12:
13:    public tree (item_t[] argitems) {
14:       if (argitems.length == 0) return;
15:       java.util.ArrayList<node> nodes = new java.util.ArrayList<node>();
16:       for (int itor = 0; itor < argitems.length; ++itor) {
17:          node tmp = new node ();
18:          tmp.item = argitems[itor];
19:          nodes.add (tmp);
20:          int parent = (itor + 1) / 2 - 1;
21:          if (parent < 0) continue;
22:          node parentnode = nodes.get (parent);
23:          if (itor % 2 == 1) parentnode.left = tmp;
24:                        else parentnode.right = tmp;
25:       }
26:       root = nodes.get (0);
27:    }
28:
29:    private void visit_rec (node a_node, visitor<item_t> vis) {
30:       if (a_node == null) return;
31:       visit_rec (a_node.left, vis);
32:       vis.visit (a_node.item);
33:       visit_rec (a_node.right, vis);
34:    }
35:
36:    public void visit (visitor<item_t> the_visitor) {
37:       visit_rec (root, the_visitor);
38:    }
39:
40: }
41:
```

```
1: // $Id: visitor.java,v 1.1 2014-01-16 17:43:14-08 - - $
2:
3: interface visitor<item_t> {
4:    public void visit (item_t item);
5: }
6:
```

```
 1: #!/usr/bin/perl
 2: # $Id: pxref.perl,v 1.1 2014-01-16 17:43:14-08 - - $
 3: use strict;
 4: use warnings;
 5:
 6: $0 =~ s|^(.*/)?([^/]+)/*$|$2|;
 7: my $status = 0;
 8: END {exit $status}
 9: $SIG{'__WARN__'} = sub {print STDERR "$0: @_"; $status = 1};
10: $SIG{'__DIE__'} = sub {warn @_; exit};
11:
12: my $sep = "\n" . ":" x 65 . "\n";
13: for my $filename (@ARGV ? @ARGV : "-") {
14:    open my $file, "<$filename" or do {warn "$filename: $!\n"; next};
15:    print "$sep$filename$sep\n";
16:    my %xref;
17:    while (defined (my $line = <$file>)) {
18:       m/^\d*$/ or push @{$xref{$_}}, $. for split m/\W+/, $line
19:    }
20:    close $file;
21:    print "$_ @{$xref{$_}}\n" for sort keys %xref;
22: }
23:
```

```
 1: #!/bin/sh -x
 2: # $Id: mk,v 1.2 2014-01-16 17:46:25-08 - - $
 3: JAVA=*.java
 4: SRC="$JAVA *.perl $0"
 5: cid + $SRC
 6: javac $JAVA
 7: echo Main-class: testtrees >Manifest
 8: jar cvfm testtrees Manifest *.class
 9: rm Manifest
10: chmod +x testtrees
11: mkpspdf Listing.ps $SRC
```