Burt Rosenberg

Problem Set 5

OUT: 30 OCTOBER, 1992
DUE: 9 NOVEMBER, 1992

Reading Assignment

Read:

- Chapter 4. This is general stuff about ADT's.
- Chapter 10, the section *Binary Trees*, and Chapter 11, the section *Terminology*.

Goals

Practice with trees.

Assignment

You are to write a program which prompts the user for an integer and then inserts it into a binary search tree. The tree begins empty, and after each insertion the program prints out the tree in two formats:

1. Such that the elements in the tree are printed in sorted order (in-order traversal of a tree).
2. Such that the structure of the tree is apparent, that is, pretty-print the tree (pre-order traversal of a tree).

After each integer is inserted the program loops back for another integer until the user enters -1, in which case the program exits.

You are given the following data structure definition to use for your tree:

```
Type
  TreeElemPntr = ^TreeElem ;
  TreeElem = record
    thing : integer ;
    leftchild, rightchild : TreeElemPntr
  end ;
```

```
TreeAnchor = record
  anchor : TreeElemPntr
end ;
Tree = ^TreeAnchor ;
```

And the following "main program" to test your subroutines:

```
var
  T : Tree ;
  x : integer ;
begin
  T := Tr_Create ;
  writeln('Type integers at the > prompt,') ;
  writeln('I will insert them into the tree.') ;
  writeln('The number -1 will end the program') ;
  write('>') ;
  readln(x) ;
  while x<>-1 do
  begin
    Tr_Insert(T,x) ;
    writeln('The elements in the tree in order:') ;
    Tr_Write(T) ;
    writeln('The structure of the tree:') ;
    Tr_PrettyPrint(T) ;
    write('>') ;
    readln(x)
  end
end.
```

Testing

How do you test that your program works? Carefully crafted programs have enough logical structure to allow a precise analysis of all possible computation paths. However, to err is human — a quality not shared by the computer. For this reason we also attempt many test cases. First, try all possible permutations of the integers 1, 2, 3. That is, input 1, 2, 3 in that order, then in the order 1, 3, 2, then 2, 1, 3, and so on. Then use several random digit

sequences: select numbers at random out of the phone book and treat them as 7 digit sequences. How sure are you now that your program works?

Example output

Here is the pretty-printed result of the number sequence 2, 8, 4, 2, 5, 7, 5. (To get the nice output, I used recursion, the data type *varying of char* and the concatentation operator +. For more information on these topics see on-line help: *help pascal data_types string_types* and *help pascal expressions operators string_operators*.)

```
The structure of the tree:
2
|‘---2
|    |‘----
|    ‘-----
‘----8
     |‘---4
     |    |‘----
     |    ‘----5
     |         |‘---5
     |         |    |‘----
     |         |    ‘-----
     |         ‘----7
     |              |‘----
     |              ‘-----
     ‘-----
>
```

Extra Credit

If -1 is input and the tree is not empty, do not exit the program. Instead call Tr_ChopDown to remove all elements from the tree using dispose on all new'ed TreeElem's. If the user enters -1 again, then exit. If the user enters something other than -1, proceed as usual: insert, print then loop back for the next integer.