Exercises 2 – 6 are programming problems. Submit them in a single file named `a07.rkt` electronically per the instructions for your course.

1. Study Hailperin, Sections 8.1, 8.2, 8.3.

2. Do Hailperin, Exercise 7.15.

3. Do Hailperin, Exercise 8.1.
   Output an error message using the display function (not the error function) if the tree is empty. Note that the tree is a binary *search* tree.

4. Do Hailperin, Exercise 8.4. Do not use `append`.

5. Do Hailperin, Exercise 8.6.
   The first parameter should be the number and the second parameter should be the binary search tree.

6. Do Problem Flatten.
   Procedure `flatten` should take a list and return a list with all the elements in the same order but with no nested parentheses. If the parameter is not a list, output an error message using the display function (not the error function). Here are some test cases for flatten.

```
> (flatten 'a)
Bug: parameter is not a list
> (flatten '())
()
> (flatten '(()))
()
> (flatten '(a))
(a)
> (flatten '(a b c))
(a b c)
> (flatten '(((a))))
(a)
> (flatten '(a b (c d (e (((f))) () g) h) i))
(a b c d e f g h i)
```

   You can write `flatten` with a single `cond` using functions `display`, `not`, `list?`, `null?`, and `append`, as well as the usual Scheme operators.