Hayden Nix
Lab 1

2.

a. The use of *pi* on line 4 is bound on line 3 to the value 3.14159. This binding is inside the scope of the circumference function where the line 4 use is, so it shadows the global assignment.

   The use of *pi* on line 7 is bound on line 1 to the value of 3.14. This binding is outside the scope of the circumference function, so this use relies on the global binding.

b. The use of x on line 3 is bound on line 1 to the value of 3. This binds x for its use as an argument in the function *f*, which is the same value used on line 3.

   The use of x on line 6 is bound on line 5. This is the catch all case, which refers to "whatever else" the case variable may be bound to. That variable is then used in the scope of that catch all case.

   The use of x on line 10 is also bound on line 5. Because it is not in the scope (curly braces) of the binding on line 8, it uses the same catch all case binding as above.

   The use of x on line 13 is bound on line 1 to a value of 3. This use is not in the scope of *f* function and therefore uses the global binding about it.

3.

Yes, *g*() is a well-typed function, returning a *( ( Int, Int ), Int )*. Based on the binding of *a* and *b* on line 2, *a* is type *Int* and *b* is type *(Int, Int)*.

g: *( ( Int, Int ), Int )* because
- b: *( Int, Int)* because
  - x: *Int*
  - 3: *Int*
- a + 2: *Int* because
  - a: *Int*
- 1: *Int*