Local and Global Variables
The mental model 1.0 explains why variables defined inside a function cannot be used in the main program.

What about variables defined in the main program? Can they be used inside a function?

```python
def foo(x):
    var1 = "hello"
    return var1 + x + y

y = "good"
print foo("bye")
```

*y is a *global* variable that is defined in the main program, but can be used in the function that is called after it is defined.*
Here is a “more correct” version of item (4)

Whenever we access a variable inside `foo`, `foo`’s dictionary is looked up. If a variable is not found in `foo`’s dictionary, then Python looks up the dictionary of the main (calling) program.

This allows a function access to “global” variables.
Local variables override global variables

```python
def foo(x):
    y = "hello"
    return x + y

y = "good"
print foo("bye")
print y
```

- This mechanism also gives local variables precedence.
- In the above example, the variable `y` is found in `foo`’s dictionary and that is the variable that is accessed in `foo`. 

This is a different, local y. During the function, all mention of y refers to this local y.

y is a global variable
def foo(x):
    global y
    y = "hello"
    return x + y

y = "good"
print foo("bye")
Print y

- **global** is a Python keyword.
- If it were not for the `global y` statement, the variable `y` being mentioned inside `foo` would have been defined in `foo`'s dictionary and would be local to `foo`. 

We are now explicitly declaring that the `y` we want to access inside `foo()` is the global variable `y`
A variable cannot be both local and global in the same function

```
def foo():
    if y == "hello":
        print "Hello to you as well!"
    y = "hi"
    print y
y = "hello"
foo()
```

- We need yet another version of our mental model!

Here `y` is a global variable

And here `y` is a local variable
Here is an “even more correct” version of item (4)

When Python starts executing a function, the statements of the function are first examined to get the names of variables that might be assigned a value in the function. If a variable \( x \) might be assigned in the function, but is not explicitly \texttt{global}, then it is local. If a variable is not local by this criterion, then by default it is global.
I would discourage the use of global variables, both implicit and explicit.

Communication between functions or between the main program and a function should be explicit – via parameters/arguments and returned values.