Namespaces 1

- Function definition creates a new *namespace* (also called *local scope*)
- Variables created inside this scope are not available from outside the function definition
- Also the function parameters are only visible inside the function definition
- Variables that are not defined inside any function are called *global variables*
Global variables are readable also in local scopes, but an assignment creates a new local variable without rebinding the global variable.

If we are inside a function, a local variable hides a global variable by the same name:

```python
i=2
def f():
    i=3  # this creates a new variable,
         # it does not rebind the global i
    print i  # this will print 3
```
Namespaces 3

- If you really need to rebind a global variable from a function, use the `global` statement

- Example

```python
i=2
def f():
    global i
    i=5
f()
print i  # will print 5
```
An example of nested functions:

```python
def f():  # outer function  
    b=2  
    def g():  # inner function  
        # b=3  # this would create a local variable  
        print b  # this works  
        g()  
        print b  # will print 2  
    f()  
```

There is no way we can rebind the variable b of function f from function g.