[220 / 319] Using Functions

Meena Syamkumar
Andy Kuemmel
Cole Nelson

Readings:
Parts of Chapter 3 of Think Python,
Chapter 5.1 to 5.4 of Python for Everybody

Due: Quiz1

Learning Objectives

How to call functions

- input/output
- terminology: call / invoke, parameter, argument, keyword argument, return value
- control flow

Function usage examples

- input()
- print(), along with keyword arguments “end” and “sep”
- type cast functions: int(), bool(), float(), str()

Using functions from built-in module:

- round(), abs()
- keywords: import, from
- attribute operator: “.”
- help: inspect a module

we'll learn about how to give functions input by passing arguments (e.g., 2) to parameters (e.g., moves)

Main Code:

1. Put 2 in the "moves" box
2. Perform the steps under "Move Code", then continue to step 3
3. Rotate the robot 90 degrees to the right (so arrow points to right)
4. Put 3 in the "moves" box
5. Perform the steps under "Move Code", then continue to step 6
6. Whatever symbol the robot is sitting on, write that symbol in the "resut" box

today we'll learn how to use functions in Python

we'll also learn how to ask functions questions and get answers called return values

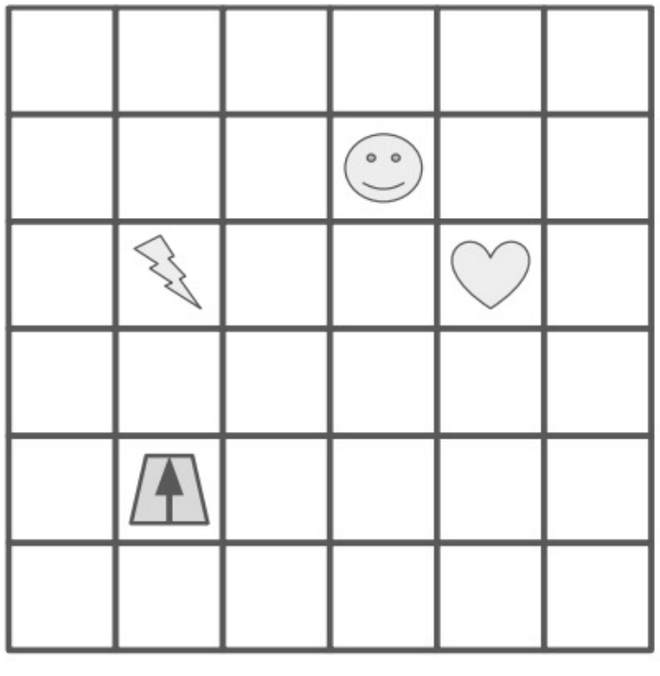
Move Code:

A. If "moves" is 0, stop performing these steps in "Move Code", and go back to where you last were in "Main Code" to complete more steps
B. Move the robot forward one square, in the direction the arrow is pointing
C. Decrease the value in "moves" by one
D. Go back to step A

"Move Code" is a function

next lecture, we'll learn how to write our own new functions

Functions are like "mini programs", as in our robot worksheet problem



Terminology / Vocabulary

- **function definition:** a grouping of lines of code; a way for us to tell our program to run that entire group of code
- **call / invoke:** a statement in Python code that instructs the program to run all the lines of code in a function definition, and then come back afterward
- **parameter**: variable that receives input to function
- **argument**: value sent to a function (lines up with parameter)
- **keyword argument**: argument explicitly tied to a parameter
- **return value**: function output sent back to calling code

next lecture, we'll learn how to write our own new functions

Calling/Invoking a Function in Python

```
print("hello")
result = f(x)
```

ALWAYS: function's name

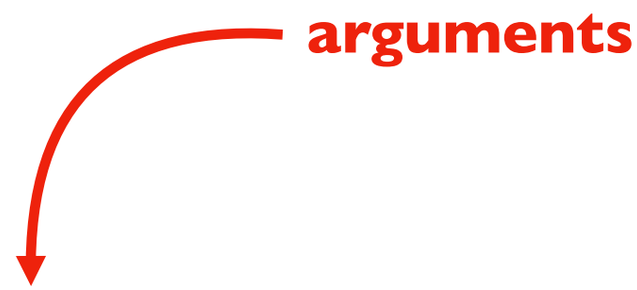
Calling/Invoking a Function in Python

```
print(“hello”)
result = f(x)
```

ALWAYS: function’s name

ALWAYS: followed by parentheses

Calling/Invoking a Function in Python



print(“hello”)

result = f(x)

ALWAYS: function’s name

ALWAYS: followed by parentheses

SOMETIMES: with one or more arguments

Calling/Invoking a Function in Python

print(“hello”)

result = f(x)

return value

ALWAYS: function’s name

ALWAYS: followed by parentheses

SOMETIMES: with one or more arguments

SOMETIMES: producing a result

Calling/Invoking a Function in Python

```
print("hello", "world")
x = input()
```

ALWAYS: function's name

ALWAYS: followed by parentheses

SOMETIMES: with one or more arguments

SOMETIMES: producing a result

Notebook examples