Functions

Rose-Hulman Institute of Technology

Computer Science and Software Engineering

ROSE-HULMAN
INSTITUTE OF TECHNOLOGY

Goals Today

- Revisit functions with more details
- Make lots of progress on Cross Words

Why functions?

- A function allows us to group together several statements and give them a name by which they may be invoked.
  - *Abstraction* (easier to remember the name than the code)
  - *Compactness* (avoids duplicate code)
  - *Flexibility* (parameters allow variation)
- Example:
  - **def complain(complaint):**
    print("Customer:", complaint)

Functions in different realms

- Compare the mechanisms for *defining* and *invoking* functions in three different settings:
  - Standard mathematical notation
  - Maple
  - Python

Functions in Mathematics

- Define a function:
  - $f(x) = x^2 - 5$

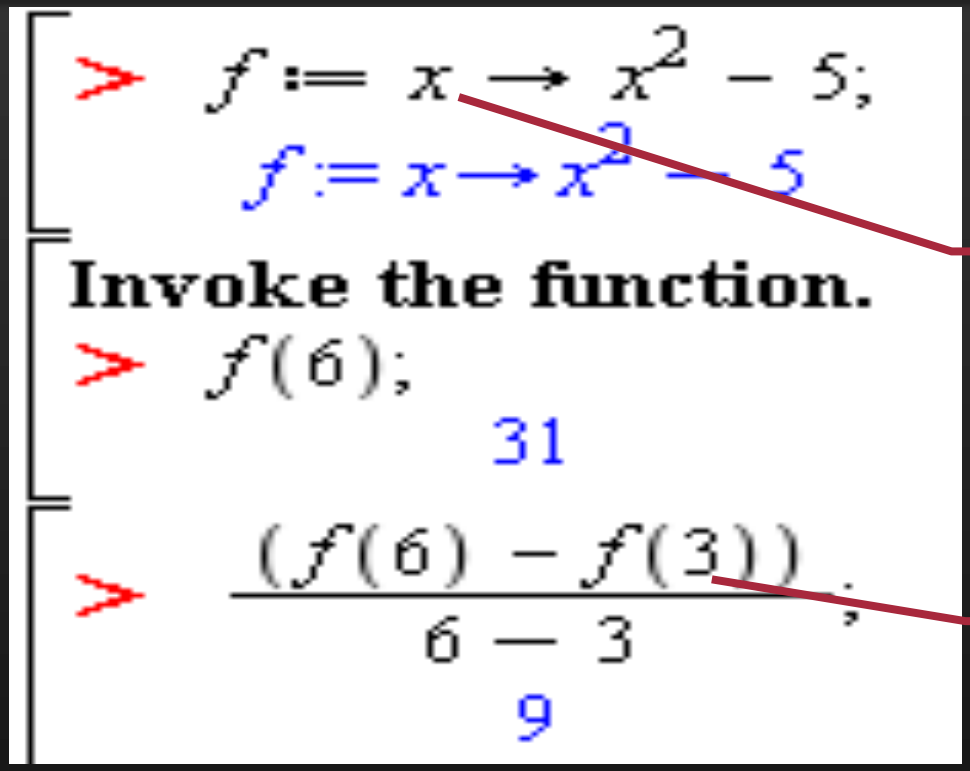
Formal Parameter. Gives a name to refer to the argument in the function's formula.

- Invoke (call) the function:
  - $\frac{f(6) - f(3)}{6 - 3}$

Two calls to function **f**. The first with *actual parameter* 6, and the second with 3.

- When the call f(6) is made, the actual parameter 6 is substituted for the formal parameter x, so that the value is $6^2 - 5$.

Functions in Maple



Formal Parameter. Gives a name to refer to the argument in the function's formula.

Two calls to function **f**. The first with *actual parameter* 6, and the second with 3.

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Functions in Python

```
>>> def f(x):
        return x*x - 5

>>> f(6)
31
>>> (f(6) - f(3)) / (6 - 3)
9
>>>
```

Formal Parameter. Gives a name to refer to the argument in the function's formula.

Two calls to function **f**. The first with *actual parameter* 6, and the second with 3.

- How would you evaluate **f(f(2))**?
- In Mathematics, functions calculate a value.
- In Python we can also define functions that instead *do something*, such as print some values.

Review: Parts of a Function Definition

Defining a function called “hello”

```
>>> def hello():
        print("Hello")
        print("I'd like to complain about this parrot")
```

Indenting tells interpreter that these lines are part of the hello function

Entering a blank line tells interpreter that we’re done defining the hello function

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Review: Defining vs. Invoking

- *Defining* a function says what it should do
- *Invoking* (calling) a function makes that happen
  - Parentheses tell the interpreter to invoke the function

```
>>> hello()
Hello
I'd like to complain about this parrot
```

  - Later we'll define functions with *parameters*

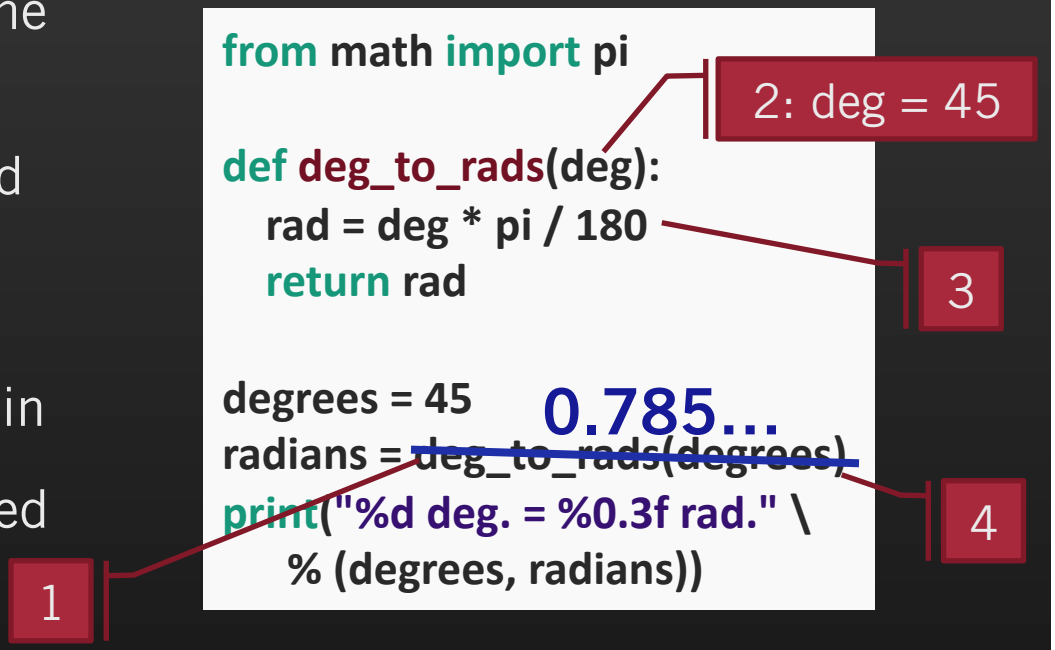
Q2

Review: Function with a Parameter

- def complain(complaint):
  ```
      print("Customer: I purchased this parrot not half " +
            "an hour ago from this very boutique")
      print("Owner: Oh yes, the Norwegian Blue. " +
            " What's wrong with it?")
      print("Customer:", complaint)
  ```
- invocation:
  - complain("It's dead!")

Python's 4 Steps for Invoking

1. Calling program pauses at the point of the call
2. Formal parameters get assigned the values supplied by the actual parameters
3. Body of the function is executed
4. Control returns to the point in calling program just after where the function was called



Functions can “do” or “return”

- Some functions just do things:
  - **hello()**
  - **complain(complaint)**
- Some functions return values:
  - **abs(-1)**
  - **x = math.cos(angle * math.pi / 180)**

Defining Functions that Return Values

- def square(x):
      return x * x
- def average(scores):
      sum = 0
      for s in scores:
      sum += s
      return sum / len(scores)

return statement

return statement

Why might it be better to return than print when a function performs a calculation?

Q3

Writing a distance() function

- **def distance(p1, p2):**

  '''Parameters are Points, returns distance between them. '''

- Should the function return anything?

Recall: If **p** refers to a **Point**, then use **p.getX()** to get its x-coordinate

Q4

If a Function Calls a Function …

- Trace what happens when the last line of this code executes
- Then do the similar one on the quiz

Guaranteed exam problem!

```
def g(a,b):
    print(a+b, a-b)

def f(x, y):
    g(x, y)
    g(x+1, y-1)

f(10, 6)
```

Q5

See Cross Words video linked from the schedule and assignment

Work on Homework 8/9

Pair Programming Time

Q6

ROSE-HULMAN INSTITUTE OF TECHNOLOGY