MORE FUNCTIONS, DECISION STRUCTURES

CSSE 120—Rose Hulman Institute of Technology

Questions on concepts on Exam 1

- Last session's slides contained a list of concepts that will be on exam 1
- What questions on these concepts have come up so far as you study for the exam?
- Any other questions?
- Note: Today's material will not be on the exam.

Function Review

- Functions can take multiple parameters
  - distance(p1, p2)
- Functions can return values

```
def square(x):
        return x * x
```

- More about parameters (details on later slides):
  - What happens when we modify them?
  - What is an optional parameter?
- More about return values:
  - Can return multiple values

Passing parameters in Python

- What type of information do formal parameters receive?
- If we assign new values to formal parameters, does this affect the actual parameters?
- Consider this version of square:

```
def squareNext(x):
  x = x + 1
  return  x * x
```

Optional parameters

- A python function may have some parameters that are optional.

```
>>> int("37")
37
>>> int("37", 10)
37
>>> int("37", 8) # specify base 8
31
```

We can declare a parameter to be optional by supplying a default value.

```
>>> def printDate(month, day, year=2007):
        print month, str(day)+",", year


>>> printDate("January", 4, 2006)
January 4, 2006
>>> printDate("January", 4)
January 4, 2007
```

Multiple optional parameters

- If there are more than one, and it's not clear which argument you are providing, you can pass **variable=value:**

Note all 3 are optional:

```
>>> def printDate(month = 'January', day = 1, year=2007):
                print month, str(day)+',', year
```

```
>>> printDate()
January 1, 2007
>>> printDate(26)
26 1, 2007
>>> printDate(day=26)
January 26, 2007
```

Nice!

I wanted the 26th. Whoops!

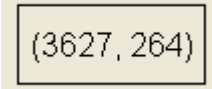
That's it.

Return Multiple Values

- A function can return multiple values
  - **def powers(n):**
    return n2, n**3, n**4**
- What's the type of the value returned by this call?
  powers(4)

Pair Programming: Three Squares

1. Checkout **Session08** project from your SVN repository
2. Work with another student on one computer
3. Run the threeSquares program to be sure it works. Put **both** students' names in the initial comment.
4. Add a function, **stats**, that takes a Rectangle, **r**, as a parameter and returns the area of **r**
5. modify the program so that it displays the area of each rectangle inside the rectangle
6. Finally, change **stats** to return the area and perimeter (see figure at right)
7. Commit your project back to your repository; also email **threeSquares.py** to your partner.



Example Display

Decision, Decisions

- Sometimes we want to alter the sequential flow of a program
  - What examples have we seen of this?
- Statements that alter the flow are called

  control structures
- *Decision structures* are control structures that allow programs to "choose" between different sequences of instructions

Simple Decisions

- The **if** statement
  - if <condition>:
    <body>
  - Semantics:
    "if the condition is true, run the body, otherwise skip it"
- Simple conditions
  - <expr> <relop> <expr>
  - Some relational operators:

Math	<	≤	=	≥	>	≠
Python	<	<=	==	>=	>	!=

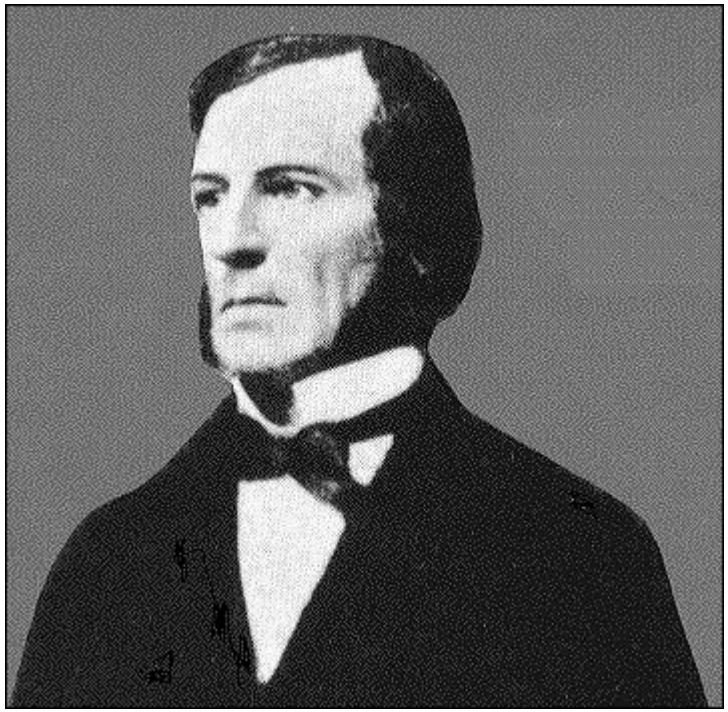
Class Exercise

- Define a function **grade(score)**
  - where score is an exam score
  - and result is "perfect", "passing", or "failing" based on the score

More on Comparisons

- Conditions are *boolean expressions*
  - They evaluate to True or False
- Try in IDLE:

```
>>> 3 < 4
>>> 42 > 7**2
>>> "ni" == "Ni"
>>> "A" < "B"
>>> "a" < "B“
```



George Boole

Having It Both Ways: if-else

- Syntax:

  if <condition>:

  <statementsForTrue>

  else:

  <statementsForFalse>

- Semantics:

  "If the condition is true, execute the statements for true, otherwise execute the statements for false"

A Mess of Nests

- Can we modify the **grade** function to return letter grades—A, B, C, D, and F?

Multi-way Decisions

- Syntax:

```
if <condition1>:
        <case 1 statements>
elif <condition2>:
        <case 2 statements>
elif <condition 3>:
        <case 3 statements>
…
else:
        <default statements>
```

reach here if condition1 is false

reach here if condition1 is false AND condition2 is true

reach here if BOTH condition1 AND condition2 are false

Cleaning the Bird Cage

- Advantages of **if-elif-else** vs. nesting
  - Number of cases is clear
  - Each parallel case is at same level in code
  - Less error-prone
- Fix **grade** function to use **if-elif-else** statement instead of nesting

Individual Exercise on Using if-else

- Finish the quiz first. Turn it in.

- Then open **countPassFail.py**.
- Define (in that file) a function **countPassFail(scores)** that
  - takes a list of exam scores
  - *returns* two values:
    - the count of passing scores in the list (those at least 60), and
    - the count of failing scores in the list
- Examples:
  - **print countPassFail([57, 100, 34, 87, 74])** prints **(3,2)**
  - **print countPassFail([59])** prints **(0,1)**
  - **print countPassFail([])** prints **(0,0)**
- Commit your project to your repository.