CSc 110, Autumn 2016

Lecture 6: Parameters

Adapted from slides by Marty Stepp and Stuart Reges

---

Redundancy Club

Listen up! The first rule of Redundancy Club is you do not talk about Redundancy Club.
The second rule of Redundancy Club is you do NOT talk about Redundancy Club.
Promoting reuse

• Programmers build increasingly complex applications
  • Enabled by existing building blocks, e.g. methods

• The more general a building block, the easier to reuse

• **Abstraction**: focusing on essential properties rather than implementation details

• Algebra is all about abstraction
  • Functions solve an entire class of similar problems
Redundant recipes

• Recipe for baking **20** cookies:
  • Mix the following ingredients in a bowl:
    • 4 cups flour  
    • 1 cup butter  
    • 1 cup sugar  
    • 2 eggs  
    • 40 oz. chocolate chips ...  
  • Place on sheet and Bake for about **10** minutes.

• Recipe for baking **40** cookies:
  • Mix the following ingredients in a bowl:
    • 8 cups flour  
    • 2 cups butter  
    • 2 cups sugar  
    • 4 eggs  
    • 80 oz. chocolate chips ...  
  • Place on sheet and Bake for about **10** minutes.
Parameterized recipe

• **Recipe for baking 20 cookies:**
  • Mix the following ingredients in a bowl:
    • 4 cups flour
    • 1 cup sugar
    • 2 eggs
    • ...

• **Recipe for baking N cookies:**
  • Mix the following ingredients in a bowl:
    • \( \frac{N}{5} \) cups flour
    • \( \frac{N}{20} \) cups butter
    • \( \frac{N}{20} \) cups sugar
    • \( \frac{N}{10} \) eggs
    • 2N oz. chocolate chips ...
  • Place on sheet and Bake for about 10 minutes.

• **parameter:** A value that distinguishes similar tasks.
Redundant figures

• Consider the task of printing the following lines/boxes:

*************

********

************************

*************
*        *
*************

*************
*     *
*************

********
*     *
*     *
**********
A redundant solution

```python
def main():
    line_of_13()
    line_of_7()
    line_of_35()
    box10x3()
    box5x4()

def line_of_13():
    for i in range(1, 14):
        print("*", end="")
    print()

def line_of_7():
    for i in range(1, 8):
        print("*", end="")
    print()

def line_of_35():
    for i in range(1, 36):
        print("*", end="")
    print()
...
```

- This code is redundant.
- Would variables help?
  Would constants help?
- What is a better solution?
  - `line` - A function to draw a line of any number of stars.
  - `box` - A function to draw a box of any size.
Parameterization

- **parameter**: A value passed to a function by its caller.

- Instead of `line_of_7, line_of_13`, write `line` to draw any length.
  - When *declaring* the function, we will state that it requires a parameter for the number of stars.
  - When *calling* the function, we will specify how many stars to draw.
Declaring a parameter

Stating that a function requires a parameter in order to run

```
def <name> (<name>) :
    <statement>(s)
```

• Example:
```
def say_password(code):
    print("The password is: " + code)
```

• When say_password is called, the caller must specify the code to print.
Passing a parameter

*Calling a function and specifying values for its parameters*

\(<\text{name}>\) (\(<\text{expression}>\))

• Example:

```
say_password(42)
say_password(12345)
```

Output:

```
The password is 42
The password is 12345
```
Parameters and loops

• A parameter can guide the number of repetitions of a loop.

chant(3)

def chant(times):
    for i in range(0, times):
        print("Just a salad...")

Output:
Just a salad...
Just a salad...
Just a salad...
How parameters are passed

• When the function is called:
  • The value is stored into the parameter variable.
  • The function's code executes using that value.

```python
def chant(times):
    for i in range(0, times):
        print("Just a salad...")

chant(7)
```

```python
chant(3)
chant(7)
```
Common errors

• If a function accepts a parameter, it is illegal to call it without passing any value for that parameter.
  
  \[\text{chant()} \quad \# \text{ ERROR: parameter value required}\]

• The value passed to a function must be of a type that will work.
  
  \[\text{chant}(3.7) \quad \# \text{ ERROR: must be of type \texttt{int} if it is used as a range bound}\]

• Exercise: Change the \texttt{stars} program to use a parameterized function for drawing lines of stars.
Stars solution

# Prints several lines of stars.
# Uses a parameterized method to remove redundancy.
def main():
    line(13)
    line(7)
    line(35)

# Prints the given number of stars plus a line break.
def line(count):
    for i in range(0, count):
        print("*", end="")
    print()
Multiple parameters

• A method can accept multiple parameters. (separate by , )
  • When calling it, you must pass values for each parameter.

• Declaration:
  ```python
def <name>(<name>, ..., <name>):
    <statement>(s)
```

• Call:
  ```python
  <name>(<exp>, <exp>, ..., <exp>)
  ```
Multiple parameters example

```python
def main():
    printNumber(4, 9)
    printNumber(17, 6)
    printNumber(8, 0)
    printNumber(0, 8)

def printNumber(number, count):
    for i in range(0, count):
        print(number, end="")
    print()

Output:
444444444
171717171717
00000000

• Modify the stars program to draw boxes with parameters.
Stars solution

# Prints several lines and boxes made of stars.
# Third version with multiple parameterized methods.

def main():
    line(13)
    line(7)
    line(35)
    print()
    box(10, 3)
    box(5, 4)
    box(20, 7)

    # Prints a box of stars of the given size.
    def box(width, height):
        line(width)
        for line in range(0, height - 2):
            print("*", end="")
            for space in range(0, width - 2):
                print(" ", end="")
            print("*")
        line(width)

    # Prints the given number of stars plus a line break.
    def line(count):
        for i in range(0, count):
            print("*", end="")
        print()
Strings as parameters

```python
say_hello("Allison")

teacher = "Bictolia"
say_hello(teacher)

def sayHello(name):
    print("Welcome, " + name)

Output:
Welcome, Allison
Welcome, Bictolia

• Modify the stars program to use string parameters. Use a function named repeat that prints a string many times.
Stars solution

# Prints several lines and boxes made of stars.
# Fourth version with String parameters.

def main():
    line(13)
    line(7)
    line(35)
    print()
    box(10, 3)
    box(5, 4)
    box(20, 7)

# Prints a box of stars of the given size.
def box(width, height):
    line(width)
    for line in range(height - 2):
        print("*", end="")
        repeat(" ", width - 2)
        print("*")
    line(width)

# Prints the given number of stars plus a line break.
def line(count):
    repeat("*", count)
    print()

# Prints the given String the given number of times.
def repeat(s, times):
    for i in range(0, times):
        print(s, end="")
Value semantics

- **value semantics**: When numbers and strings are passed as parameters, their values are copied.
  - Modifying the parameter will not affect the variable passed in.

```python
def strange(x):
    x = x + 1
    print("1. x = " + x)

x = 23
strange(x)
print("2. x = " + x)
```

Output:
```
1. x = 24
2. x = 23
```
def main():
    x = 9
    y = 2
    z = 5

    mystery(z, y, x)

    mystery(y, x, z)

def mystery(x, z, y):
    print(str(z) + " and " + str(y - x))