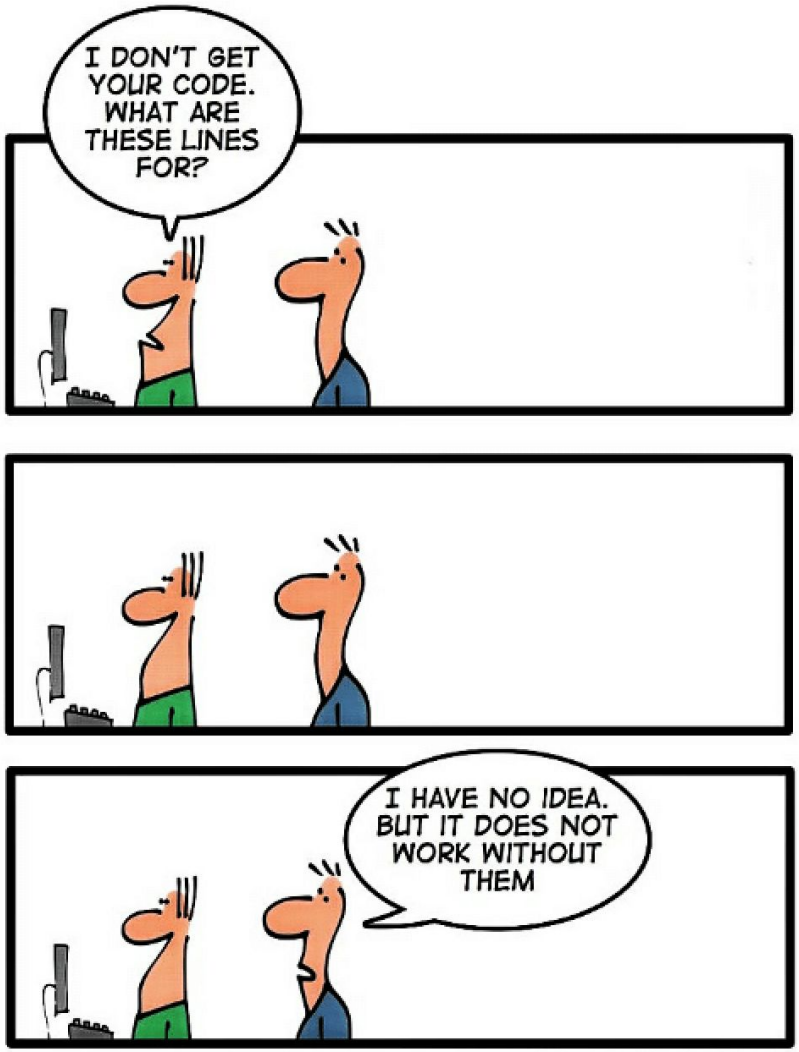
CS 110 Functions, Parameters, Arguments

Benjamin Dicken



Announcements

- Exam Grades Posted
  - Not releasing answer key, but you may visit an office hour if you want to go over any of the problems
- Programming Assignments 4 and 5
- D2L grades
- New groups start Wednesday

Arguments and Parameters

- It is possible to send values to functions when called

- When the function is defined, must specify one or more **parameter variables**
- When the function is **called**, must specify one or more **arguments**

```
size = int(input('Size:'))

def repeat():
    index = 1
    while index <= size:
        print('|     |')
        index += 1

def print_row():
    print('+-----+')

def print_eyes():
    print_row()
    print('| 0 0 |')

def print_shape():
    print('| \_/ |')
```

```
print_eyes()
repeat()
print_shape()
repeat()
print_row()
```

```
def print_info(name):
    print('Hi', name)
    print('How are you?')

more code . . .
print_info('Joe')
more code . . .
print_info('Joseph')
more code . . .
```

```
def print_taxes(salary):
    if salary < 10000:
        print('taxes are', (salary * 0.15))
    elif salary < 50000:
        print('taxes are', (salary * 0.20))
    elif salary < 150000:
        print('taxes are', (salary * 0.30))

print_taxes(27000)
print_taxes(150000)
print_taxes(1000)
```

salary is only available within this function

```
def print_taxes(salary):
    if salary < 10000:
        print('taxes are', (salary * 0.15))
    elif salary < 50000:
        print('taxes are', (salary * 0.20))
    elif salary < 150000:
        print('taxes are', (salary * 0.30))

money = int(input('Enter your salary: '))

print_taxes(money)
```

What would print out?:

```
def print_taxes(salary):
    if salary < 10000:
        print('taxes are', (salary * 0.15))
    elif salary < 50000:
        print('taxes are', (salary * 0.20))
    elif salary < 150000:
        print('taxes are', (salary * 0.30))

index = 3
while index > 0:
    money = int(input('Enter your salary: '))
    print_taxes(money)
    index -= 1
```

Inputs: 5000
15000
123456

```
def print_class(units):
    if units > 90:
        print('senior')
    elif units > 60:
        print('junior')
    elif units > 30:
        print('sophomore')
    elif units >= 0:
        print('freshman')
    else:
        print('WAT')

print_class(20)
print_class(-5)
print_class(100)
```

What will this produce?

What will this produce?

```
def print_status(units):
    if units >= 12:
        print('full-time')
    else:
        print('part-time')

def print_class(units):
    if units > 90:
        print('senior')
    elif units > 60:
        print('junior')
    elif units > 30:
        print('sophomore')
    else:
        print('freshman')
```

```
semester = int(input('Semester Units: '))
total = int(input('Total Units: '))
print_status(semester)
print_class(total)
```

What will this produce?

```
def print_school_info(semester_units, total_units):
    if semester_units >= 12:
        print('full-time')
    else:
        print('part-time')
    if total_units > 90:
        print('senior')
    elif total_units > 60:
        print('junior')
    elif total_units > 30:
        print('sophomore')
    else:
        print('freshman')
```

```
semester = int(input('Semester Units: '))
total = int(input('Total Units: '))
print_school_info(semester, total)
```

validate_name.py

```
first = input('Enter first name: ')
if not first.isalpha() or len(first) > 15 or not first[0].isupper():
    print('Invalid first name.')
    exit()

middle = input('Enter middle initial: ')
if not middle.isalpha() or len(middle) > 1 or not middle[0].isupper():
    print('Invalid middle initial.')
    exit()

last = input('Enter last name: ')
if not last.isalpha() or len(last) > 30 or not last[0].isupper():
    print('Invalid last name.')
    exit()

print('Valid name!')
```

Activity

validate_name.py

```
first = input('Enter first name: ')
if not first.isalpha() or len(first) > 15 or not first[0].isupper():
    print('Invalid first name.')
    exit()

middle = input('Enter middle initial: ')
if not middle.isalpha() or len(middle) > 1 or not middle[0].isupper():
    print('Invalid middle initial.')
    exit()

last = input('Enter last name: ')
if not last.isalpha() or len(last) > 30 or not last[0].isupper():
    print('Invalid last name.')
    exit()

print('Valid name!')
```

What is redundant? What is different?

validate_name.py

```
first = input('Enter first name: ')
if not first.isalpha() or len(first) > 15 or not first[0].isupper():
    print('Invalid first name.')
    exit()

middle = input('Enter middle initial: ')
if not middle.isalpha() or len(middle) > 1 or not middle[0].isupper():
    print('Invalid middle initial.')
    exit()

last = input('Enter last name: ')
if not last.isalpha() or len(last) > 30 or not last[0].isupper():
    print('Invalid last name.')
    exit()

print('Valid name!')
```

What is redundant? What is different?

```
first = input('Enter first name: ')
if not first.isalpha() or len(first) > 15 or not first[0].isupper():
    print('Invalid first name.')
    exit()

middle = input('Enter middle initial: ')
if not middle.isalpha() or len(middle) > 1 or not middle
    print('Invalid middle initial.')
    exit()

last = input('Enter last name: ')
if not last.isalpha() or len(last) > 30 or not last[0].isupper():
    print('Invalid last name.')
    exit()

print('Valid name!')
```

Write function `validate_input`, Call it three times

What is redundant? What is different?

Is this better?

```
def validate_input(input_type, length):
    value = input('Enter ' + input_type + ': ')
    if not value.isalpha() or len(value) > length or not value[0].isupper():
        print('Invalid ' + input_type + '.')
        exit()

validate_input('first name', 15)
validate_input('middle initial', 1)
validate_input('last name', 30)

print('Valid name!')
```

What will it print?

```
def process_numbers(first, second, third):
    if first >= second >= third or second >= first >= third:
        print(first + second)
    elif first >= third >= second or third >= first >= second:
        print(first + third)
    else:
        print(second + third)

process_numbers(30, 10, 20)
process_numbers(10, 20, 30)
process_numbers(20, 30, 10)
```

Function Comments

- Important to document
  - What each function does
  - The expected ***type*** and ***purpose*** of each parameter variable
    - Otherwise, how would you know what to pass in?
- This should be done with a multi-line string

Function Comments

```
def process_numbers(first, second, third):
    if first >= second >= third or second >= first >= third:
        print(first + second)
    elif first >= third >= second or third >= first >= second:
        print(first + third)
    else:
        print(second + third)
```

Function Comments

```
def process_numbers(first, second, third):
    '''
    This function accepts three numeric values and will print out
    The sum of the largest two.
    first: should be an integer number
    second: should be an integer number
    third: should be an integer number
    '''
    if first >= second >= third or second >= first >= third:
        print(first + second)
    elif first >= third >= second or third >= first >= second:
        print(first + third)
    else:
        print(second + third)
```

Remember this?

```
def validate_input(input_type, length):
    value = input('Enter ' + input_type + ': ')
    if not value.isalpha() or len(value) > length or not value[0].isupper():
        print('Invalid ' + input_type + '.')
        exit()

validate_input('first name', 15)
validate_input('middle initial', 1)
validate_input('last name', 30)

print('Valid name!')
```

Write the function comment
Be detailed!

```
def validate_input(input_type, length):
    value = input('Enter ' + input_type + ': ')
    if not value.isalpha() or len(value) > length or not value[0].isupper():
        print('Invalid ' + input_type + '.')
        exit()
```

Write the function comment

```
def validate_input(input_type, length):
    '''
    Asks the user for an input and exits if the input is not
    alphabetical, capitalized, and of the correct length.
    input_type: a string label for the input prompt and error message
    length: an int, representing the max length of the input string
    '''
    value = input('Enter ' + input_type + ': ')
    if not value.isalpha() or len(value) > length or not value[0].isupper():
        print('Invalid ' + input_type + '.')
        exit()
```

main()

- ***No code should be without a function*** ***
- Standard practise in programming to have a **main()** function, which should be the first function to be called
- From this function other functions can be called
  - and others from those, and other from those, and so on
  - The **main** naming convention is standard practise across several programming languages, not just python

*** *with a few exceptions*

main()

- Exceptions to the last slide
  - Comments
  - `import` statements
  - Global variables (if allowed)
  - Constants (if allowed)
  - The call to **`main()`**

Activity

Use a main function

```
def validate_cap_alpha_input(input_type, length):
    '''
    Asks the user for an input and exits if the input is not
    alphabetical, capitalized, and of the correct length.
    input_type: a string label for the input prompt and error message
    length: an int, representing the max length of the input string
    '''
    value = input('Enter ' + input_type + ': ')
    if not value.isalpha() or len(value) > length or not value[0].isupper():
        print('Invalid ' + input_type + '.')
        exit()

validate_cap_alpha_input('first name', 15)
validate_cap_alpha_input('middle initial', 1)
validate_cap_alpha_input('last name', 30)

print('Valid name!')
```

Use a main function

```
def validate_cap_alpha_input(input_type, length):
    '''
    Asks the user for an input and exits if the input is not
    alphabetical, capitalized, and of the correct length.
    input_type: a string label for the input prompt and error message
    length: an int, representing the max length of the input string
    '''
    value = input('Enter ' + input_type + ': ')
    if not value.isalpha() or len(value) > length or not value[0].isupper():
        print('Invalid ' + input_type + '.')
        exit()

def main():
    validate_cap_alpha_input('first name', 15)
    validate_cap_alpha_input('middle initial', 1)
    validate_cap_alpha_input('last name', 30)
    print('Valid name!')

main()
```