Functions

Textbook Readings

- Chapter 6: Dragon Realm
  - The `time.sleep()` function
  - Creating our own functions with the `def` keyword.
  - The `return` keyword, Parameters and Arguments.
  - The `and`, `or`, and `not` Boolean operators.
  - Variable scope (Global and Local)
  - Flow charts

- Chapter 7: Using the Debugger

- Chapter 8: Flow Charts

- Chapter 9: Hangman
  - The `append()` list method
  - `lower()`, `upper()` and `split()` string methods
  - The `range()` function and for loops.
  - The `list()` function
  - The `key()` and `values()` dictionary methods.
Python Function: `def`

```
def chooseCave():
  function name

Function Call
```

Function: No Arguments, No Returns

```
def showMessage():
  print("Hello")
  print("Control Starts Here!")
  showMessage()

Function Call
```

Function Body

- Can be any number of statements.
- All statements in the function body must be indented.
Function: One Argument, No Returns

```python
def showMessage(price):
    if (price < 100):
        print("It is a deal.")
    else:
        print("What ??!!!")
        print(""")

while (True):
    x = input("How much does this cost?")
    x = int(x)
    showMessage(x)
```

Function: One Argument, One Return

```python
def addTip(price):
    return price + price*20/100

while (True):
    x = input("How much is the bill?")
    x = int(x)
    print("Total cost is: ", addTip(x))
```
Why is this an Error?

```python
def takeDamage(damage):
    health = health - damage

health = 100
takeDamage(20)
takeDamage(15)

print(health)
```

```
line 2, in takeDamage
    health = health - damage
UnboundLocalError: local variable 'health' referenced before assignment
```

Keyword: `global`

```python
def takeDamage(damage):
    global health
    health = health - damage

health = 100
takeDamage(20)
takeDamage(15)

print(health)
```

```
takeDamage(10)
print(health)
```

```
Output:
65
55
```
def addCheepTip(price):
    return price + price*10/100

print(addCheepTip(30))
print(addCheepTip(50))
print(addCheepTip(70))

a) 3.0  b) 5.0  c) 30.0  d) 0.3  e) 33.0
    5.0  8.0  50.0  0.5  55.0
    7.0  15.0 70.0  0.7  77.0

def getGrade(score):
    grade = "F"
    if (score >= 90):
        grade = "A"
    if (score >= 80):
        grade = "B"
    if (score >= 70):
        grade = "C"
    else:
        grade = "D"
    return grade

print(getGrade(88))
print(getGrade(75))

a) F  b) B  c) C  d) D  e) 88 75
Quiz: Python: def and while

```python
def quiz(n):
x = 0
i = 0
while (i < n):
x = x + i
i = i + 1
print(x)
quiz(6)
```

The output of the Python code above is:

a) 21  b) 16  c) 15  d) 6  e) 0

---

Quiz: Python: def and while

```python
def bob(a, b, c):
    if (a > b and a > c): print(a)
    elif (b > a and b > c): print(b)
    else: print(c)

bob(2, 4, 6)
bob(32, 71, 13)
```

The output of the Python code above is:

a) 6  b) 2  c) 6  d) 4  e) 4
71 32 71 6 4
13 71 32 71