Lecture 7 - Control Flow / If Statements

1. We are working on the grading.
2. Assignment 2 - too hard.

Using GIT and Github.com

We are basically following most of the steps in https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners.

https://www.freecodecamp.org/news/how-to-undo-a-git-add/

1st Test Question

```
 1:
 2: def feet_to_inches ( feet ):
 3:     conv = 12
 4:     inches = feet * conv
 5:     return ( feet )
 6:
 7: # Automated Test
 8: if __name__ == "__main__":
 9:     n_err = 0
10:     x = feet_to_inches ( 1 )
11:     if x != 12:
12:         n_err = n_err + 1
13:         print ( "Error: Test 1: conversion not working, expected {} got {}".format
14:     x = feet_to_inches ( 0 )
15:     if x != 0:
16:         n_err = n_err + 1
17:         print ( "Error: Test 2: conversion not working, expected {} got {}".format
18:
19:     if n_err == 0 :
20:         print ( "PASS" )
21:     else:
22:         print ( "FAILED" )
23:
```

1. (10pts) The above code has something wrong with it. When the test is run it says “FAILED”. Correct the code.

Where we are so far...

1. Installs - usually hard and unpleasant.
2. Using some files - where - paths - directories/folders.
3. Fixing some code - changes break thins - then you have to fix it.
4. Testing. Thomas Piketty, "Capital in the Twenty-First Century". https://www.reuters.com/article/idUS268051827620140527 "The Financial Times has launched a critique of the data behind the French economist's bestseller "Capital in the Twenty-First Century." ... he has also fallen prey to sloppy spreadsheets."
5. Functions

```
def fucntion_name ( input1, input2 ):
do_someting...
return ( output )
```

6. Input a number - differences between strings and numbers and integers and floats

"if" / True / False

```
if n_err == 0 :
    print ( "PASS" )
else:
    print ( "FAILED" )
```

```
    if expression-evaluates-to-true:
            Do the true side
    elif expression-2-evaluates-to-true:
            More stuff if expression-2 is true
    else:
            if non of the above are true, do this.
```

Operators that commonly go into expressions in if:

```
==              compare for equality
!=              not equal
<               less than
>               greater than
<=              less than or equal
>=              greater than or equal
```

An Example:

The ski area sells tickets and gives a discount based on age. Adult tickets age 18-69 are $59, Youth 5-12 are $40, Teen are $52, Children 4 and under are free, seniors 70 and older are free.

```
 1:
 2: print ( "Input Age\n=> ", end="" )
 3: age_str = input()
 4: age = int(age_str)
 5:
 6: ticket_price = 0
 7: if age <= 4:
 8:     ticket_price = 0
 9: elif age >= 5 and age <= 12:
10:     ticket_price = 40
11: elif age >= 13 and age <= 17:
12:     ticket_price = 52
13: elif age >= 18 and age <= 70:
14:     ticket_price = 59
15: else:
16:     ticket_price = 0
17:
18: print ( "Ticket Price ${}.00 dollars".format(ticket_price) )
```

Order of Evaluation

```
 1:
 2: print ( "Input Age\n=> ", end="" )
 3: age_str = input()
 4: age = int(age_str)
 5:
 6: ticket_price = 0
 7: if age <= 4:
 8:     ticket_price = 0
 9: elif age <= 12:
10:     ticket_price = 40
11: elif age <= 17:
12:     ticket_price = 52
13: elif age <= 70:
14:     ticket_price = 59
15: else:
16:     ticket_price = 0
17:
18: print ( "Ticket Price ${}.00 dollars".format(ticket_price) )
```

Common Errors - leaving out cases in the logic.

```
 1:
 2:
 3: print ( "Input Age\n=> ", end="" )
 4: age_str = input()
 5: age = int(age_str)
 6:
 7: ticket_price = 59
 8: if age <= 4:
 9:     ticket_price = 0
10: elif age <= 12:
11:     ticket_price = 40
12: elif age <= 17:
13:     ticket_price = 52
14:
15: print ( "Ticket Price ${}.00 dollars".format(ticket_price) )
```

“and” and “or”

When we have “if” the expression is true or false as values.

There are operators that work on Boolean values. These are *or*, *and*, *not* and an exclusive or operator, `^`.

```
a = 2
b = 3

r = ( a == 2 ) and ( b == 3 )
```

Truth Tables

And:

A	B	A and B
False	False	False
False	True	False
True	False	False
True	True	True

Or:

A	B	A or B
False	False	False
False	True	True
True	False	True
True	True	True

Exclusive Or:

A	B	A ^ B
False	False	False
False	True	True
True	False	True
True	True	False

Not:

A	not A
True	False
False	True

calling functions

You can also make a function (def) that returns a True/False value and use that in an if.

```
def isRed ( r ):
        if r == "Red":
                return True
        if r == "red":
                return True
        return False


if isRed("green"):
        print ( "Its Is Red" )
else;
        print ( "Its Is *NOT* Red" )
```

Copyright