CompSci 101
Part 1 of 5: main and import
KISS Principle

• Think of the non-computing context for any word/terms
• Keep It Simple, Stupid (US Navy-1960s)
  • Work smarter, not harder!!
• “Good programmers are simply good designers.”
  • -Dr. Washington
• Design first and always!
• Importance of reusability
main

• Python modules
  • *.py files

• Modules are either:
  • Stand-alone
    • Control the program execution
  • Imported
    • Used in another stand-alone module
Created on 1/25/2021

Author: alici

#

if __name__ == '__main__':
    pass

def output(value):
    print("Number = " + str(value))

if __name__ == '__main__':
    number = 10
    output(number)
CompSci 101
Part 2 of 5: main and import
import

• Real-world example:
  • Importing vs. exporting goods.

• What does import mean in Python?

• Why would you want/need to import a module?
  • KISS principle
  • Reusability
  • e.g., calculate square root, cosine, floor, etc.

• Python Standard Library
  • BOOKMARK THIS ON YOUR BROWSER!
    • https://tinyurl.com/kt4ogfu
Sample Code

Created 1/25/2021
@author: anw

```
import module_name

if __name__ == '__main__':
    print(module_name.function_name(arguments))
```
Can we only import modules from the Python Standard Library?

```python
def output_num(value):
    print('The value from your main code was ' + str(value))

def output(value):
    print('Number=' + str(value))

if __name__ == '__main__':
    number = 28
    output(number)

import mymodule

def output(value):
    print('Number=' + str(value))

if __name__ == '__main__':
    number = 28
    output(number)
    mymodule.output_num(number)
```

*Module named mymodule.py*

*Module named test.py*

**we want to import mymodule into here and use the output_num function**

*Updated test.py*

**imports mymodule to use output_num() function**
CompSci 101
Part 3 of 5:
random module
random module

• When would using random numbers be applicable?
  • Games: Rolling dice, spinning a wheel

• Rolling dice
  • Options: One die displays 1-6
  • How do we recreate this in Python?

• KISS/Reusability
  • import statement
  • random module (built-in Python functionality)
    • Python Standard Library-https://tinyurl.com/kt4ogfu
How does it work?

- **random() function**

```python
import random

def output(value):
    print("Number=
```  
```python
if __name__ == '__main__':
    number = random.random()
    output(number)
```

- **NOTE:** number is in [0.0, 1.0)

- **randint() function**

```python
import random

def output(value):
    print("Number=
```  
```python
if __name__ == '__main__':
    number = random.randint(1, 5)
    output(number)
```

- **NOTE:** number is in [1, 5]
CompSci 101
Part 4 of 5:
Relational and logic operators
Boolean Logic

<table>
<thead>
<tr>
<th>AND</th>
<th>x</th>
<th>y</th>
<th>xy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OR</th>
<th>x</th>
<th>y</th>
<th>x+y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Boolean values in Python

• True or False
  • Case matters!!

• Relational operators
  • x == y
  • x != y
  • x > y
  • x < y
  • x >= y
  • x <= y
Comparing Logical Expressions

• and, or, not

• Expression 1 and Expression 2
• Expression 1 or Expression 2
• not Expression 2

• Remember order of precedence
  • PEMDAS
  • Relational (==, !, >, <, >=, <=)
  • Logical (and, or, not)
CompSci 101
Part 5 of 5:
Conditionals
Conditionals
Conditionals: You can’t have it both ways!

• If condition is true ➔ action1
• Or else ➔ action2

```python
if condition1:
    block1
else:
    block2
```
What if there are more forks in the road?

• If condition is true → action1
• Or else
  • If new condition is true → action2
  • Or else → action3

```python
if condition1:
    block1
else:
    if condition2:
        block2
    else:
        block3
```
if...elif...else

• If condition is true → action1
• Or else if new condition is true → action2
• Or else → action3

```python
if __name__ == '__main__':
    num1 = 2
    if num1 == 5:
        print("The number is 5!")
    elif num1 < 5:
        print("The number is less than 5!")
    else:
        print("The number is greater than 5!")
```
Do you always need an elif or else?

- if condition is true ➔ action1
- ...remainder of program...

```python
if condition1:
    block1
```

Program code:
```python
if __name__ == '__main__':
    num1 = 2

    if num1 == 5:
        print('The number is 5!')
    num2 = num1 + 3
    print(num2)
```