

CIS 122

Recap

Midterm Details

- Monday July, 23
- 50 minutes
- Study guide on course website
 - Resources page
- You are allowed to bring a note sheet
 - 1 sheet of paper
 - Double sided

Midterm Details

- What should you study?
- Homework assignments
 - Know how they work
 - Know why they work
- Study guide
 - Make sure you're familiar with the terms
 - Know how to use them
- In class quizzes
 - Look them over
 - Slides are all online

Types

- What types have we seen so far?
 - Ints
 - Floats
 - Strings
 - Booleans
 - (don't worry about tuples)

Types - Ints

- Whole numeric values
- Can perform arithmetic operations
 - Addition
 - Subtraction
 - Multiplication
 - Division
- Any integer operation always returns an integer
 - Careful when dividing
 - Always truncates down

Types - Floats

- Fractional numeric values
 - Any number with a decimal point
- Can do anything ints can do
- Any operation involving a float returns a float
 - $5 / 2 = 2$
 - $5.0 / 2 = 2.5$
- Need a float fast?
 - Multiply by 1.0
 - $42 * 1.0 = 42.0$

Types - Strings

- Sequences of characters
 - Surrounded by quotes
 - "HAPPY BIRTHDAY"
- Not just letters
 - Numbers
 - Punctuation
 - White space
- How long are these strings?
 - "Count me!"
 - " "
 - ""

Types - Strings

- What can we do with strings?
 - Basic operations
- String addition (concatenation)
 - "abc"+"def"
- String multiplication
 - "hip " * 3

Types - Strings

- What can we do with strings?
 - String indexing
- `s[i]` = ith character of s (starting from 0)
 - `"abcdef"[3]`
- `s[-i]` = ith character from the right (starting from 1)
 - `"abcdef"[-3]`

Types - Strings

- What can we do with strings?
 - String slicing
- `s[i : j]` = substring of `s`
 - Starting from `s[i]`
 - Up to but not including `s[j]`
 - `"abcdef"[2 : 4] = "cd"`
- If we leave out a number, it defaults to the end
 - `"abcdef"[2 :] = "cdef"`
 - `"abcdef"[: 4] = "abcd"`

Types - Booleans

- Only two values
 - True
 - False
- Comparisons
 - $3 \leq 4$
 - `'a' != 'b'`
- Boolean logic
 - and
 - or
 - not

Types

- What questions should you expect?
 - Evaluate this expression (as python would)
- Some sample expressions
 - $1 + 2 * 3$
 - "sequence" [3]
 - $3 < 4$ and True

Variable Assignment

- We can assign values to variables
 - Assignment operator (=)
 - Variable on the left
 - Value on the right
- `x = 5`
- `myString = "puppy"`
- `isItRainingToday = False`

Variable Assignment

- Variables can be reassigned
 - New value replaces old value
 - Variables on LHS = names
 - Variables on RHS = values

- $x = 5$
- $x = 6$
- $x = x + 1$

Conditional Logic

- Conditional code execution
 - `if`, `elif`, `else`

```
if x == 0:  
    print "x is zero"  
elif x==1:  
    print "x is one"  
else:  
    print "I don't know what x is"
```

Conditional Logic

- What questions should you expect?
 - What happens when we run this code?
 - What is the value of x afterwards?

```
x = 0
```

```
if x < 0:
```

```
    x = x + 1
```

```
elif x != 2:
```

```
    x = x * 2
```

```
else:
```

```
    x = 5
```


Functions

- Function Components
 - Definition
 - Name
 - Arguments
 - Body
 - Docstring
 - Return Value

```
def plusOne(myNum):  
    """Adds one to myNum"""  
  
    myLargerNum = myNum + 1  
    return myLargerNum
```

Functions

- Function Components

- Definition

- Name

- Arguments

- Body

- Docstring

- Return Value

```
def plusOne(myNum):  
    """Adds one to myNum"""
```

```
    myLargerNum = myNum + 1  
    return myLargerNum
```

- What questions should you expect?

- Tell me what this function does (high level description)

- Write a function to perform a simple task

- Stack diagrams

Functions - Stack Diagrams

```
def plusOne(myNum):  
    newNum = myNum + 1  
    return newNum
```

```
def myFunc(x,y):  
    z = plusOne(x)  
    ans = y*z  
    return ans
```

```
a = myFunc(2,3)
```

```
__main__  
plusOne → <func>  
myFunc → <func>  
a → 9
```

```
myFunc  
x → 2  
y → 3  
z → 3  
ans → 9
```

```
plusOne  
myNum → 2  
newNum → 3
```

Recursion

- Recursive Functions
 - Just like normal functions
 - Except they call themselves
- Structure
 - Base Case
 - Recursive Step
- What questions should you expect?
 - Implement this recursive problem
 - I'll give you a base case and recursive step

Turtle

- Importing Modules
 - `import turtle`
- Basic turtle functions
 - `turtle.forward(dist)`
 - `turtle.backward(dist)`
 - `turtle.left(angle)`
 - `turtle.right(angle)`
- What sort of question should you expect?
 - Something tied into a previous topic
 - I won't ask you to draw a fractal