Exam 2 on Wednesday, April 10
Introduction to Computer Programming
CSCI-UA 2

Repetition Structures
For-Loops and While-Loops
Loops

Introduce nonlinearity into programs
Repeatedly execute blocks of code
For-loops
While-loops
For-Loops

Repeats a block of code a specified number of times

Like printing a list of numbers

```python
for i in range(n):
    print(i)
```
While-Loops

While loops are more flexible, but also more error-prone

Must initialize variable and set increment

```python
i = 0

while i < 10:
    print(i)
    i = i + 1
```
For-Loops and While-Loops

```
↓
initialization_block
↓
Is condition true?

(yes) ↓

(never) → body_block

↓

after_block
↓
```
String Formatting Expressions

Python defines the % operator to work on strings

When applied to strings, % provides a compact way to code multiple string substitutions

Operator on left is conversion target

On the right, provide the object you want Python to insert in its place

print('%d %s' % (i, 'str'))

Most common: %s %d %f
Break Statement

Lets you jump out of a loop from within the loop body

Allows you to skip over unnecessary statements

break

Should only be used when it makes your code simpler

Related to: continue
Introduction to Computer Programming
CSCI-UA 2

Functions
Encapsulating Logic
Functions

Groups of statements that can be run more than once in a program

Reusable chunks of code

Take input, provide output

Can be reused in a variety of contexts

Maximize code reuse, minimize code redundancy

Encapsulate logic, splitting complex systems into manageable parts
Calling Functions

`type(3.7)`

Function name
Arguments passed into function
Return Value

( ) tells Python to execute the function

Even if a function takes no input, the brackets are still required

Some functions do not return a value
Defining Functions

Just like writing a Python program but with some extra syntax

Function header:

• Begins with `def`
• Followed by name of function
• Function parameter list
• Ends with a `:`
Documentation String

Doc String

Optionally follows function header

Explains what function will do

May include example(s)

Almost always good to include for clarity and as a reminder
Function Body

All the code that follows the header

Simply an indented block of code with necessary statements

This code can use the variables from the function header

Function should return a value with the keyword return

After return, Python jumps out of the function and back to the program
Function Syntax

```python
def name(parameters):
    statements

A return value is not required

return None

Anything besides return are called “side effects”

A print() statement is an example of a side effect
```
Variables

Local and Global

Variable scope is something we must pay attention to when using functions. When and where is a variable accessible to your program? Local variables are only usable within the function they are local to. Global variables are usable by any function or code within your program.
Local Variables

def area(rad):
    import math
    return math.pi * rad ** 2

When this function ends, rad is automatically deleted because it is local.
Global Variables

```
rad = 5

def area():
    import math
    return math.pi * rad ** 2

def change_rad(new_rad):
    rad = new_rad

This program will not allow rad to be changed because it is global

To change the radius variable:
global rad
```
Main Function
main()

It is both common and a good idea to use a main function in your programs.

This is usually the starting point of a program and is run by typing: `main()`.

This simplifies rerunning programs and as well as passing input values.
Function Parameters

Used to pass input into a function

Python passes values by reference

Default values can be set for parameters

```python
def name(parameter = 'default')
```
Modules

A module is a group of related functions.

Different from a regular Python program in that it acts like a toolbox.

A module usually does not have a `main()` function.

To use a module, simply import it.
Importing Modules

Import functions individually:

```python
import module_name

module_name.function()
```

Import functions all at once:

```python
from module_name import *

function()
```
Namespaces

Functions within a module are available when you import them.

Modules form namespaces.

Different modules with the same function name will not clash in the same program.

module1.function(parameter)
module2.function(parameter)

The only time functions may conflict is when you import all using *
Strings
Strings and Characters

\[ s[i] \]
Strings

Strings are one of Python’s primary data types.

Strings can be used to represent anything that can be encoded as text.

Examples: symbols, words, text files, Python programs, and Web pages.

We’ve already been using strings quite a bit and now will focus on accessing and processing them.
Indexing

Strings are defined as ordered collections of characters

Therefore, we can access these characters by position with "indexing"

Offsets start at 0 and end at one less than the length of the string

Offset can also be specified negatively
<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>y</td>
<td>t</td>
<td>h</td>
<td>o</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>-6</td>
<td>-5</td>
<td>-4</td>
<td>-3</td>
<td>-2</td>
<td>-1</td>
<td></td>
</tr>
</tbody>
</table>
Characters

ord('a')

All characters have a corresponding character code

Python uses Unicode to assign character codes

Unicode is a common standard for encoding thousands of symbols and characters from many languages
Escape Characters

“Escape characters” are a special notation for whitespace and other unprintable characters

Used for characters without a standard symbol or symbols Python sets aside

Escape characters begin with a backslash

Escape sequences are considered one character
Slicing Strings

`s[begin:end]`

Slicing is for extracting substrings from a string

To slice a string, indicate the first character you want and one past the last character

First index location is always 0

Last index location is always the length of the string minus one
String Functions
s.function()
Regular Expressions

A “regular expression” is a special sequence of characters that helps you match or find other strings.

Python has a regular expressions module, “re,” in its standard library.

Used to efficiently and compactly describe a set of strings.

Regular expressions operators and functions can be combined to perform powerful string processing tasks.
Regular Expressions

Operators

xy?
Describes either x or xy (y is optional)

x | y
Describes x or y

x*
Describes an infinite set of strings

x+
Describes an infinite set of one or more strings

Use round brackets to indicate what substring an operator should apply to
Exam 2 on Wednesday, April 10