CIS122 Intro to Programming

WELCOME!
Call me “Jason”

My background - Bachelors in Physics, currently finishing a Masters in Computer Science focusing on networking and security

I quite enjoy teaching and student interaction, feel free to email me, although most programming questions are better handled during office hours

OFFICE HOURS: M-TH 2-3 (right after class in KLA-B13)

Previously taught Chemistry and GTF’d for this class
Should You Be Here?

Other courses:
105- basic concepts in computing, no real programming

110/111- web focus, slower paced, use HTML/Javascript/CSS/XML

210/211/212- year long, faster paced, also gets into the theory a bit more

313/314/315- much more theory based, relatively little programming

330- C/C++ and UNIX

400 level- Networks, and databases, and automata, oh my!

We don’t assume any previous programming knowledge
You do need some basic algebra skills ( x=4, y = x+1, y = ?)
Structure of the Class

Class website can be found through http://www.cs.uoregon.edu/Classes/

10 weeks is not a lot of time, 8 weeks is really not a lot of time

Homework will be assigned on Fridays (except this first one) and due on Thursdays, find on class website

Weekly quizzes on Fridays

No midterms, the final is just a longer quiz but comprehensive

Final grade is ~66% quizzes/final and ~33% HW

Class attendance is not mandatory, you’re in college you can decide if you need to be here or not, just be aware that 8 weeks doesn’t leave much time to fix problems with grades
Programming is a skill, skills *need* practice

Pay attention to lecture and/or read the book (depending on what works best for you)

Do the homework, it’s not worth a lot of points but…

The quizzes should be quite easy if you’ve done well on the homework

Overall this should be an easy class to pass if you are simply willing to do the homework and show up for quizzes

Come to office hours, we have no lab this term

Try stuff on your own
Outside Sources

Why is HW worth relatively little while quizzes are worth 2x much?
Python

Named for Monty Python

Traits of Python
• Object Oriented
• General purpose
• High level
• Strongly typed
• Dynamically typed

In general terms Python is a very friendly language, easy to read, easy to write, and has many built in functions and libraries that are very useful, especially to beginning programmers. The cost? Speed.
TODO: get Python 3.x installed on your laptop/home computer ASAP (don’t get Python 2.x, enough differences to screw you up) https://www.python.org/download/

We do have a short homework this week (and quiz this Friday) Also you will want to be able to follow along in class, trying things out

Python is bundled with a very simple IDE (integrated development environment) called Idle which we will be using

Lets take a look at it

Shell vs. Editor
We *do* have a short homework this week (and quiz this Friday)

For HWs you will be turning in your editor file (*.py), nothing from the shell
WELCOME BACK!
What is programming?

Programming is computational problem solving.

• Start with a task (calculation to perform, idea to implement, domain to explore, etc.)

• apply a computational process

• Obtain desired outcome

TASK →

    COMPUTATIONAL PROCESS →

    DESIRED OUTCOME
A computational process (or *algorithm*) is a sequence of well defined operations that lead from an initial starting point to the desired final state.

“well defined” means there is no ambiguity

FYI
'Algorithm' stems from 'Algoritmi', the Latin form of Abū ʿAbdallāh Muḥammad ibn Mūsā al-Khwārizmī a Persian mathematician, astronomer and geographer. (wikipedia)
How many students in class

How can we solve that problem?
In class exercise agony

1. Everyone stands up
2. Assign yourself the number 1
3. While the # of standing students is > 1
   a) Partner with the closest student to you
   b) Add your numbers together
   c) One person is assigned the number
   d) Other person sits down
4. Report number of last standing person

This is an algorithm no different than you’d use in programming
Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing-agent. [CunySnyderWing10]
A computer program implements an algorithm
Computational Process

TASK → COMPUTATIONAL THINKING →
ALGORITHM → CODING →
PROGRAM → DESIRED OUTCOME

Coding is the process that leads from an algorithm (a well thought out step by step solution to the problem) to an executable program.

Programming then is computational thinking + coding
Primitives

Primitives are the basic elements of language that we use to construct expressions which make up programs.

The primitives of Python are objects, Objects can have value(s), a type, and an id:

- The value(s) are data we want the object to hold
- The type classifies what an object is, how it may be used
- The id lets us distinguish between objects that may be the same type and have the same value but are nevertheless distinct (ghosts in Pacman)
Let’s explore our first type in python: int

int stands for integer, which is a whole number (1, 2, 0, -4, 65, etc. but not 2/3 or .5)

ints have a number of associated operations you’d expect: +, -, *, / Also some you may not expect: ^, **, //, %, +=, -=, *=, /=,
We need one more thing before we delve into idle and start playing around

Print() is a function, we talk about functions next week but we need this one now. For the moment just accept that whatever you put in the parentheses for print will get *evaluated* and printed to the screen *in the shell*

i.e. if we write
print(3)
In our editor window and run it it should print the number 3 in the shell

Let’s try it
For the moment just accept that whatever you put in the parentheses for print will get *evaluated* and printed to the screen *in the shell*

Why did I say “evaluated” above?

Try `print(1+3)`

We now have everything we need to use python to make your laptop an incredibly expensive calculator
WELCOME BACK!
Things to try:
10 + 4
10 – 4
10 + -4
--7
+-7
4/2 (what happened?)

Remember to print these otherwise you won’t see the results.
Okay, one other function to try out this week.

If we print( type(2) ) it will print out the type of the number 2, which is int

Try print( type(4/2) )
So we have another type!

Floats are also numbers, like ints, but they look different. They aren’t necessarily whole numbers.

Why do we have floats and ints? Why not just one number type?
Other Oddities

Things to try:
2/3 vs 2//3 (what happened?)

10 % 5 and 10 % 3 (what happened?)

10^5 vs 10^10 (what happened?)

10 ** 2 (what happened?)
Why the oddities?

What if you want to know if a number is “divisible” by 4?

“divisible” meaning with no remainder, i.e. the result would be an integer.

If your number \( \% 4 \) is 0 then you know it is divisible by 4.

Try it

print( 12 \% 4 )
print( 13 \% 4 )
That “your number thing”

If your number % 4 is 0 then you know it is divisible by 4.

It might be nice if we had a convenient way to have place holders for “my number”

Hint- almost anytime you say “I wish Python had x,” it most likely does. I ♥ Python.
A variable is a basically a box with a name on it into which we can place objects.

```python
myVar1 = 2
myVar2 = 3
print( myVar1 + myVar2 )
```

This is why it is important that print evaluates what is in the parentheses.
Variables

Python visualizer:
http://www.pythontutor.com/visualize.html#mode=edit
You can name a variable (almost) anything you want!

“almost” explained:
- can contain number but cannot start with a number
- cannot contain symbols except “_” (typed with shift+-)
- must start with a letter
- uppercase and lowercase are fine
- spaces are verboten!

Legal:
- myVar1
- this_is_my_long_unwieldy_variable_name
- whenWillItBe5oClock

Illegal:
- variables are fun
- partyTime!
- 1stPlace
There’s one other type of variable name that is illegal; keywords.

Keywords are reserved names in Python. Here’s a list:

FALSE
None
TRUE
and
as
assert
break
class
continue
def
del
elif
else
except
finally
for
from
global
if
import
in
is
lambda
nonlocal
not
or
pass
raise
return
try
while
with
yield
Variable Names

Try to make your variable names as clear as possible

This will absolutely help you write, read, and debug your code

If you have a variable to represent the radius of a circle
rad = 5
radius = 5
circRad = 5
…are all good choices
r = 5
X = 5
funVariable2 = 5
…are *less good* choices.
Name Collisions

Temp = 100  # variable for temperature
Temp = 25   # temporary variable

Whoops.
The = sign has multiple meanings in math, in computer science we use it for just one thing:

\[ A = B \] means set \( A \) to have the \textit{value of} \( B \)

This is not a test of equality, nor is it persistent equality:

\begin{verbatim}
A = 5
B = A
A += 3
print(A)
print(B)
\end{verbatim}

Technically this is called “pass by value.” Entertain your friends, amaze your enemies!
CIS122 Summer

WELCOME BACK!
Where We Are

<table>
<thead>
<tr>
<th>Types</th>
<th>Functions</th>
<th>Flow Control</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>print()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>float</td>
<td>type()</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Well, you got to start somewhere
Questions?
What does this code do?

```python
a = 5
b = 3.14
c = b * a ** 2
print(c)
```

You can most likely figure it out, but....
Importance of Variable Names

This is much easier to understand:

circle_radius = 5
pi = 3.14
circle_area = pi * circle_radius ** 2
print(circle_area)

Not only is it easier to understand it’s easier to write and debug. I actually made an error when writing the version on the previous slide and had pi squared rather than the radius squared.
Increments/Decrements

Incrementing a variable

\[ X = 5 \]
\[ X = X + 1 \]
Increments/Decrements

Incrementing a variable

\[ X = 5 \]
\[ X = X + 1 \]

Identical code shorthand:

\[ X = 5 \]
\[ X += 1 \]
Increments/Decrement
Variables

Python visualizer:
http://www.pythontutor.com/visualize.html#mode=edit
Examples

Let’s do an example problem very similar to something you might do on homework.

Let’s say you have a 2 GB hard drive. That’s ~ 2000 MB. You want to install a bunch of software packages.

- You have 1 word processing program that takes 100 MB, and you want to reserve another 250 MB for that book you are going to start any day now.

- You have 6 games, each of which takes up 75 MB of memory.

- You also have Idle (*woohoo, Idle!* ) which takes up 35 MB.

- Your completely legal file sharing program is only 50 MB but you need another 800 for all your mp3s.

- Your OS takes up 500 MBs (stupid Windows) including a swap file of 200 MB.

- Does it all fit?
Unpacking Expressions

How do you figure out what
X = 1
X += X
X += X
X *= X
print( X)

prints?
Comments

The # symbol indicates to Python that everything after it is a comment.

Python will not try to read your comments at all.

Comments are very handy for recording information for other human beings and even yourself.