CSI 33 LECTURE NOTES (Ojakian)

Topic 1: First Day, and Basic Comparison of Python and C++

OUTLINE

(References: superficial view of ch 8)

1. Basic comparison of Python and C++
2. Focus on similarity for now ...

---

1. Starting up ...
   (a) Get Codelab and Dropbox set up.
   (b) You should have Python installed. Also install the C++ Code::Blocks IDE.
   (c) Start HW 0 in Dropbox.

2. Plan for this topic
   (a) **We'll proceeed to discuss Python and C++, emphasizing their similarities at first ...**
   (b) Do lots of small programs.
   (c) Work towards a bigger program: Quiz program of chapter 1 (page 38) exercise 8. I will do it in C++. For HW you will write a Python version.

3. Compiled versus Interpreted
   (a) `system(''pause'')` - pause at end of run.
   (b) Do MinGW compilation (with g++ NAME.cpp -o NAME)

4. Libraries and modules

   Your programs in this course should generally work without using outside libraries, etc.

   (a) Can import modules into Python. For us we will mostly need to import nothing
   (b) For C++, can include libraries.
      i. We will almost always want to include `iostream` for input and output.
      ii. Often include `string` for working with strings.
      iii. Sometimes include `typeinfo` to get the types of data.
      iv. Sometimes include `vector` or `list` when needed.

5. Data types
   (a) Statically Typed versus Dynamically Typed - see table on p. 265 in book for C++ types
      i. Declare variables first (not done in Python).
      ii. Then define (... or simultaneously do both)

iii. String: not a built in C++ type (need to include “string”)

iv. Get type info:

   A. In Python: Use command `type(BLAH)`

   B. In C++: Include `typeinfo` and use `typeid(BLAH).name()` to get string description.

v.

   PROBLEM 1. *Write Python and C++ examples. Get the types. And try changing the types.*

(b) Static versus Dynamic fits with Compiled versus Interpreted.

   i. Variable in C++: reserves a spot of a certain size for the data - so it needs to know how much space is needed (and thus its type)

   ii. Variable in Python: type can change

6. User input and output

(a) Python: `print` and `input`

(b) C++: `cout` and `cin`

(c) The complication: C++ input ...

   i. C++ `cin` ignores whitespace requires correct type.

   ii. C++ keeps characters in the stream

   PROBLEM 2. *Write a C++ program that uses* `cin` *to get a first, then a second number, then print them out. Mess it up by flooding the first* `cin` *request.*

(d) BUT! ... Can use `getline` combined with following conversion commands

   i. `stoi`, `stof`: string to int, string to float (etc.)

   ii.

   PROBLEM 3. *Do the last problem again, but now use* `getline`.
   There are other issues, but at least it gives you the whole line of text, and the next getting of input starts fresh. For example, could use the `find` method on strings, etc. to make this nicer.

7. Things which are basically the same (except for syntax) between Python and C++

(a) Expressions - numerical and boolean; and type conversion

   PROBLEM 4. *Consider the Python and C++ expressions in the two “Topic 1 Basics” programs. What is printed?*

(b) Decision statements

PROBLEM 5. *In Python and C++, write a program that reads in an integer value (understood to be how old you are), and then prints out your generation.*

- *Baby Boomers: Baby boomers were born between 1946 and 1964. They're currently between 56-74 years old (71.6 million in U.S.)*
- *Gen X: Gen X was born between 1965 and 1980 and are currently between 40-55 years old (65.2 million people in U.S.)*
- *Gen Y: Gen Y, or Millennials, were born between 1980 and 1994. They are currently between 24-39 years old (72.1 million in the U.S.)*
- *Gen Z: Gen Z is the newest generation to be named and were born between 1996 and 2015. They are currently between 5-24 years old (nearly 68 million in U.S.)*

(c) Loops

PROBLEM 6. *In Python and C++, write a program that reads in an integer input and finds the sum of the numbers from 1 up to and including this number.*

PROBLEM 7. *In Python and C++, write a program that takes input till 0 is inputed, then prints the sum of all the given numbers.*

8. <u>Containers</u>

(a) In Python: list, set, dict

(b) In C++ (from standard library): vector, set, unorderd map

(c) Python is Heterogenous. C++ is Homogeneous. Why? - again: C++ has fixed spots for things, unlike Python.

PROBLEM 8. *Look at programs in "Topic 1 Containers" in Python and C++. What do they print?*

(d) 2 typical kinds of for-loops:

i. Basic: Integer ranges over some specified values. Do the body of the loop for each specified value in order.

ii. Range-based: Range over the items in a container type. Do the body of the loop for each items in the container.

PROBLEM 9. *Write a program in C++ to take in user input of floats, till enter is inputed. Then return the mean and the standard deviation.*

9. <u>Functions</u>

(a) C++: specify return type. Python: not.

(b) C++: specify parameter types. Python: not.

(c) C++: Can declare before defined.

PROBLEM 10. *In both Python and C++ write a main and a function call, putting the definition before and after the main to see what happens.*

(d) Similar operation of how values sent to function change and do not change

PROBLEM 11. *Look at code in "Topic 1 Functions" in Python and C++ to determine what they print.*

PROBLEM 12. *Write a function in C++ that takes as input: a vector of integers and a single integer. Calling the function returns nothing, but changes the vector so that the integer is added to each entry.*

10. File Reading
    (a) In Python
        i. Open file: [FILE VAR] = open("[FILE NAME]", LETTER) where LETTER is string 'r' for reading and 'w' for writing.
        ii. Read: [FILE VAR].read() (reads entire file to string)
        iii. Write: print("ANY STRING", file = [FILE VAR])
            PROBLEM 13. *Write a function which reads in a file, then writes the vocabulary to another file (use Python split and set).*
    (b) In C++
        i. Include "fstream"
        ii. Open:
            A. For reading do - `ifstream [FILE VAR]("[FILE NAME]");`
            B. For writing do - `ofstream [FILE VAR]("[FILE NAME]");`
        iii. Read: Like `cin` :
            A. One way: [FILE VAR] >> [string variable] >> etc...
            B. Another way: getline([FILE VAR], [STRING VAR])
        iv. Write: Like `cout`: [FILE VAR] << [string]
            PROBLEM 14. *Do a basic example of file reading and writing in C++.*
    (c)
        PROBLEM 15. *Do the Quiz Program in C++ (from Chapter 1, page 38).*