Computer Programming
Spring 2020

CSI2100-01 Lab 2

Bernd Burgstaller
Yonsei University

Overview

- Programming Problems (from our textbook)

  → Because of our coronavirus precautions, you are provided with an excerpt of the relevant sections of the textbook together with this lab assignment.

- Deliverables, due-date and submission

Programming Problems

Problem 1: Problem M2 on page 37 of the textbook.

- Please read the explanations on page 32 about the given Python code carefully.
  - You only have to hard-code the asked values. Do not modify the computations in any other way.
  - Please make sure that your program displays the result with 1 digit after the floating-point, even if that digit is 0.
- You do **not** have to submit test-cases.
- Below you find an example of how your program should work.
- Please find the source-code of the original Drake program on YSCEC. Use this code for your modifications (you do not have to type in the Drake program from the textbook).

```
Welcome to the SETI program
This program will allow you to enter specific values related to
the likelihood of finding intelligent life in our galaxy. All
percentages should be entered as integer values, e.g., 40 and not .40

How many planets per star do you think can support life?: 4
How long do you think civilizations on other worlds last?: 80000

Based on the values entered ...
there are an estimated 2240000.0 potentially detectable civilizations in our galaxy
```

Programming Problems (cont.)

Problem 2: Chapter 1, p. 36, Exercise P3

You do not have to consider faulty input. The integer base will be an integer >= 0, and the exponent will be an integer (may be negative).

Note: this program needs to ask two times for input from the user. The input prompt with the second input() command depends on the input from the first input() command. This is highlighted in the following example:

```
What base? 10
What power of 10? 4
10 to the power of 4 is 10000
```

Because the input() commands accepts only one argument, you need to build the string argument for the second input() command yourself, **ahead** of calling input().

Please have a look at Slide 26 in Lecture 2 on ``Data and Expressions''. The "+" operator allows you to concatenate two or more strings.

- to build this string, you obviously need the string for the ``base'' argument. It is therefore not a good idea to do ```base = int(input('What base? '))`''. Rather, store the base string and convert it to int subsequently (see next slide).

Programming Problems (cont.)

```
base_str = input ('What base? ')
base = int(base_str)
...
```

Problem 3: Chapter 1, p. 36, Exercise P4

You do not have to consider faulty input. Assume that the user only enters zeroes and ones, one per line.

Problem 4: Chapter 2, p. 76, Exercise P1

Example:

```
Number 1: 22
Number 2: 5
Result: 4.40
```

Note: you do not have to consider faulty user input. You do not have to consider zero for the second number. Numbers may be negative.

Programming Problems (cont.)

Problem 5: Chapter 2, p.76, Problem M4

Example:

```
This program will convert degrees Celsius to degrees Fahrenheit
Enter degrees Celsius: 22.099
22.099 degrees Celsius equals 71.8 degrees Fahrenheit
```

1) You should assume that the user input is of type **float** (no faulty user input). Use the float() function to convert the user's input string to a floating-point value. Numbers may be negative.
2) You should output the degrees Fahrenheit with **one digit** after the decimal point. See the lecture slides on the "format" function on how to do this.
3) Note: the source-code in Figure 2-22 of the textbook contains a typo. A corrected version has been provided with the Lab2 spec on YSCEC. Please use the corrected version as your starting point.
4) Please print the program greeting as stated in the above example ('`This program will convert degrees Celsius to degrees Fahrenheit`').

Marking Criteria and Plagiarism

- Marking Criteria
  - Score is only given to programs that compile and produce the correct output with Python 3.
  - Points are deducted for programs that produce warnings.
  - Please pay particular attention to the requested output format of your programs. Deviating from the requested output format results in points deductions.

- Plagiarism (Cheating)
  - This is an individual assignment. All submissions are checked for plagiarism.
  - Once detected, measures will be taken for all students involved in the plagiarism incident.

Deliverables

- Please prepare the files for the programming problems.
- The names of the files, their due-date and the archive file-name is given in the below table.
- Putting files into archives has been explained with the Lab 1 specification.

Problem	File name	Due	Archive name
1	lab2_p1.py	Wednesday April 1 23:00	lab2_<student id>.zip
2	lab2_p2.py		
3	lab2_p3.py		
4	lab2_p4.py		
5	lab2_p5.py		