ITEC 1150
Chapter 2 part 2
Lab Projects

LOOPS & RANGE

Program Development Plan (PDP)

This is a step-by-step process for successfully constructing an application. Follow these steps and repeat until you have successfully completed the program.

This is a reminder to use the PDP process.

You do not need to turn in your PDP document, but going through the process will help you design your programs.

PDP template -

1. Problem definition
2. Analysis of variables & functions
3. Steps in algorithm
4. Code (separate .py file with added comments)
5. Screen snips from testing
6. Keep track of any wanted or needed improvements for a future version

General Requirements

All assignments must meet the following requirements:

The program must start with header at top and include appropriate comments throughout. Header example:

```
"""
Author: Erik Granse
Date: 2024-09-02
Description: Calculate and display student's average grades
"""
```

- Ensure the output is *information*; it needs to be a statement which explains the value being displayed (for example, “The average grade is 12.34”). Simply outputting “12.34” is not sufficient.

General Requirements (cont.)

All assignments must meet the following requirements:

- The data in the program must be stored in variables.
- The output **must** come from variables in the program
  - Do not simply hard code the output value in the `print()` statement.
  - Some data will be given to you, and some will be user input—any calculations that need to happen must be in your program. Don't calculate somewhere else and enter the value into your program.

General Requirements (cont.)

New requirements this week!

All assignments must meet the following requirements:

- Integer input must be validated to ensure the string from `input()` can be turned into an int without crashing (float validation is not yet required).
- All input must be validated to ensure it meets the requirements of the lab (for example, ensuring an age is >= 0 or a quiz score is between 0 and 10).
- If input is not valid, you must give a message to the user and allow them to try again until the input is valid.
- Exemptions to the above will be called out in the lab sections. **If not exempted, validation is required!**

Lab Section 1: Textbook

- Using the code from slide 9 of the lecture notes as a starting point, create a program named `textbook.py`. The program must:
  - Ask the user how many textbooks they need to buy.
  - Validate the number of textbooks:
    - Do not let the user continue until a valid integer > 0 is entered.
    - Give the user a message with instructions if they provide invalid input.
  - Ask the user for the price of each book, numbering the books starting from 1.
  - Provide a subtotal (the total price for all books entered so far) after each book is entered.
  - Provide the grand total of all books entered when the user has entered as many books as they specified.

Sample output:

```
How many textbooks do you have to buy? two
Please enter a positive whole number: 2
Enter price for book #1: 25.17
    Subtotal = $25.17.
Enter price for book #2: 34.98
    Subtotal = $60.15.
Grand total = $60.15.
```

Lab Section 2: Counting in Loops

- Create a program named **loop_counter.py**. The program must:
  - Print a statement indicating what the program will do next (see output example).
  - Use a loop to print the numbers from 0 through 5 inclusive.
  - Use a loop to print the numbers from 1 through 20 inclusive.
  - Use a loop to print the even numbers from 0 through 24 inclusive.
  - Use a loop to print the odd numbers from 37 through 53 inclusive.
  - Use a loop to print multiples of ten between 10 and 60 inclusive (10, 20, 30, 40, 50, 60).
  - Use a loop to print numbers counting down from 30 through 20 inclusive.

Several code samples in the lesson slides are close to what you need to produce this output.

Sample output:

```
Here are the
numbers from 1
through 5:
1
2
3
4
5

Here are the
numbers from 1
through 20:
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

Here are the
even numbers
from 0 through
24:
0
2
4
6
8
10
12
14
16
18
20
22
24

Here are the
odd numbers
from 37 through
53:
37
39
41
43
45
47
49
51
53

Here are the
multiples of 10
from 10 through
60:
10
20
30
40
50
60

Here is a
countdown from
30 through 20:
30
29
28
27
26
25
24
23
22
20
```

Lab Section 3: Loop Counter 2

- Create a program named `loop_counter_2.py`. Your program must:
  - Ask the user for a small number.
  - Ask the user for second, larger number.
  - In addition to requiring whole numbers for both inputs, you must ensure the second number is greater than the first number.
  - Use a loop and the range function to print the numbers between the first and second number *inclusive*.
  - Display the sum of all numbers printed.
  - *Hint 1: the two user-defined numbers will become the start and stop values for your range function.

Sample output:

```
Welcome to our counting program.
(It also adds up the digits counted!)
Please enter a small number, 0 or higher: two
Please enter a whole number only, 0 or higher: -1
Please enter a whole number only, 0 or higher: 2
Please enter a whole number larger than 2: 1
Please enter a whole number larger than 2: 5
2
3
4
5
The total of all the counted numbers is 14.
```

Lab Section 4: Quiz Average or Average Rainfall

- Choose one of the following two python projects to complete:
  - quiz_average.py or
  - average_rainfall.py:
- Those are shown on the next two slides

Lab Section 4: Quiz Average

- Choose one of quiz_average or average_rainfall – you do not need to do both
- Create a program named `quiz_average.py`. The program must:
  - Ask the user for the number of students in the class.
  - Ask the user for the number of quizzes taken by students in the class.
  - For each student, ask the user for the score of each quiz.
  - Calculate and display the total and average quiz score for each student.
- This program will require the use of nested loops. The outer loop will be for the students, and the inner loop will be for the quizzes.
- Hint: use small numbers like 2 students and 3 quizzes to test (otherwise, testing becomes very tedious).
- INPUT VALIDATION IS NOT REQUIRED for this section.

Sample output:

```
How many students in the class? 2
How many quiz scores to enter per student? 3
Enter scores for student 1:
   What was the score for quiz #1? 92
   What was the score for quiz #2? 88
   What was the score for quiz #3? 91
The total points for student 1 is 271.00.
The average score for student 1 is 90.33.
Enter scores for student 2:
   What was the score for quiz #1? 87
   What was the score for quiz #2? 81
   What was the score for quiz #3? 90
The total points for student 2 is 258.00.
The average score for student 2 is 86.00.
```

Lab Section 4: Average Rainfall

- Choose one of quiz_average or average_rainfall – you do not need to do both
- Create a program named **average_rainfall.py**. The program must:
  - Ask the user for the number of years in their study.
  - For each year, ask the user for the amount of rainfall in each of the 12 months.
  - At the end of each year, display the total and average rainfall for the year.
- This program will require the use of nested loops. The outer loop will be for the years, and the inner loop will be for the months.
- Hint: use small numbers like 2 students and 3 quizzes to test (otherwise, testing becomes very tedious).
- INPUT VALIDATION IS NOT REQUIRED for this section.
- Challenge: add a feature to show the total and average rainfall for the entire study period (multi-year figures).

Sample output:

```
How many years are in the rainfall sample? 2
Rainfall data for year #1:
Enter rain for month #1: 1
Enter rain for month #2: 2
Enter rain for month #3: 3
Enter rain for month #4: 1
Enter rain for month #5: 2
Enter rain for month #6: 3
Enter rain for month #7: 1
Enter rain for month #8: 2
Enter rain for month #9: 3
Enter rain for month #10: 1
Enter rain for month #11: 2
Enter rain for month #12: 3
Total rainfall for year #1: 24.00
Average monthly rainfall for year #1: 2.00
Rainfall data for year #2:
Enter rain for month #1: 3
Enter rain for month #2: 4
Enter rain for month #3: 5
Enter rain for month #4: 3
Enter rain for month #5: 4
Enter rain for month #6: 5
Enter rain for month #7: 3
Enter rain for month #8: 4
Enter rain for month #9: 5
Enter rain for month #10: 3
Enter rain for month #11: 4
Enter rain for month #12: 5
Total rainfall for year #2: 48.00
Average monthly rainfall for year #2: 4.00
Total rainfall, all years: 72.00
Average monthly rainfall, all years: 3.00
```

Upload Assignments & Plan for Ch 3 Homework

- As always…Use 2 types of comments """long comment/ Doc String""" at the top, and # short comments for each section.
- Work with and help other students, but never give out your code files!
- Upload your files to the Ch 2.2 Lab drop box:
  - Textbook.py, and
  - loop-1.py, and
  - loop-2.py, and
  - Your choice of quiz_avg.py OR avg_rainfall.py programs
- Complete your chapter 3 homework on functions before the deadline!
- Questions? Ask before the deadline!