ITEC 1150 Week 8 Chapter 5

LAB PROJECTS

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.)

All assignments must meet the following requirements:

- All input must be validated to ensure the string from `input()` can be turned into a number without crashing.
- 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!**

General Requirements, continued

- MIPO:
  - Main
  - Inputs
  - Processing
  - Outputs
- This is the basic structure all our programs will now follow.
- Add additional functions as necessary, but the MIPO functions must exist and be used.
- Generic exception handling must be used to ensure input errors do not cause a crash.
- Programs must offer restart to the user when they are done.

Lab Section 1: Usernames and Full Names

- Starting with the `lab5_dictionary_starter.py` file in D2L, create a new program named `username_fullname.py`.
- The starter program allows a user to view, edit, and delete country codes and names. You must modify the starter program to manage usernames and full names instead of country codes and names.
- You must:
  - Rename all variables in the program as appropriate—there must be no reference to countries in the finished program!
  - Change capitalization methods as appropriate—usernames are lowercase, and full names are title case (`.title()`).
  - Replace the starter data (country codes and names) with a set of three usernames and full names.
  - Change all printed messages as appropriate.
- All existing features (view, edit, delete) must still work.
- You must add a **new** feature to allow the user to edit an existing user's full name.
- See the next slides for validation requirements and example output.

Lab Section 1 (cont.) – Validation Requirements

- Command menu input must be validated:

```
COMMAND MENU
view - View user name
add - Add a user
edit - Edit a user
del - Delete a user
exit - Exit program

Command: help
Not a valid command. Please try again.

Command:
```

- Input must be validated when interacting with dictionaries – no crashes!

```
Command: view
Usernames: hhassan jdoe jsmith
Enter username to view: hsmith
There is no user with that username.

Command:
```

Lab Section 1 (cont.) – Required Outputs

- When **view**ing, the user does not need to match the exact case of the username, and the full name is displayed with each part of the name capitalized:

```
Command: view
Usernames: hhassan jdoe jsmith
Enter username to view: jdoe
Full name: Jane Doe.
```

- When **add**ing, the added user's name is displayed with all parts of the name capitalized:

```
Command: add
Usernames: hhassan jdoe jsmith
Enter new username to add: sstudent
Enter full name: Sally Student
Sally Student was added.
```

- When **edit**ing, the username will be included in the response, printed lowercase, and the added user's name is displayed with all parts of the name capitalized:

```
Command: edit
Usernames: hhassan jdoe jsmith sstudent
Enter username to edit: sstudent
Enter the new full name of the user: Sally Strudel
sstudent has been updated to the full name Sally Strudel.
```

- When **del**eting, the deleted user's name is displayed with all parts of the name capitalized:

```
Command: del
Usernames: hhassan jdoe jsmith sstudent
Enter username to delete: sstudent
Sally Strudel was deleted.
```

Lab Section 2: Reading List

- Starting with the previous `username_fullname.py` program, create a new program named `reading_list.py`.
- This time, name the dictionary variable *readings* (instead of *users)*. It will store your favorite authors and book titles as key-value pairs.
- You must:
  - Rename all variables in the program as appropriate—there must be no reference to users in the finished program!
  - Change capitalization methods as appropriate—names are typically title case, as are book titles (`.title()`).
  - Replace the starter data with a set of three authors and book titles.
  - Change all printed messages as appropriate.
- Initialize the dictionary with 3 authors and book titles.
- Like the previous program, this one should display a command menu and let the user view, add, edit and delete dictionary entries.
- See the next slides for validation requirements and example output.

Lab Section 2 (cont.) – Validation Requirements

- Command menu input must be validated:

```
COMMAND MENU
view - View the reading list
add - Add a reading
edit - Edit a reading
del - Delete a reading
exit - Exit program

Command: edjit
Not a valid command. Please try again.

Command:
```

- Input must be validated when interacting with dictionaries – no crashes!

```
Command: view
Authors: Alice Smith, Bill Bower, Hassan Hassan
Enter the name of the reading author to view: alice bower
There is no reading with that author name.

Command:
```

Lab Section 2 (cont.) – Required Outputs

- When **view**ing, the user does not need to match the exact case of the author, and the authors and title are displayed with each word capitalized:

```
Command: view
Authors: Alice Smith, Bill Bower, Hassan Hassan
Enter the name of the reading author to view: bill bower
Title: Clojure Code.
```

- When **add**ing, the added title is displayed with each word capitalized:

```
Command: add
Authors: Alice Smith, Bill Bower, Hassan Hassan
Enter the author for the new reading: Carol Smith
Enter the title of the reading: kotlin koans
Kotlin Koans was added.
```

- When **edit**ing, the username will be included in the response, printed lowercase, and the title is displayed with each word capitalized:

```
Command: edit
Authors: Alice Smith, Bill Bower, Hassan Hassan, Carol Smith
Enter author to edit: bill bower
Enter the new title for Bill Bower: Jump Into Java
The reading for Bill Bower has been updated to the title Jump Into
Java.
```

When **del**eting, the deleted title is displayed with each word capitalized:

```
Command: del
Authors: Alice Smith, Bill Bower, Hassan Hassan, Carol Smith
Enter author of the reading to delete: alice smith
Python Primer was deleted.
```

Final Steps

Remember, all your programs must include multiple interactive functions, validation, exception handling and restart. And this time a dictionary!

Remember to use """doc string""" and short comments – every time!

Upload both programs to the Lab 5.0 by the deadline.

Questions or need help? Ask before the deadline!

Read Chapter 6 for next week.