ITEC 1150
Chapter 16
Lab Projects

CSV & JSON DATA

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: Data Collection

MIPO not required for this section

- Create a program named **data_collection.py**. The program must:
- Store contact information in a CSV file.
- Use PyInputPlus to create a menu which will allow the the user to:
  - View: read the contents of the user file and display all users as shown in the sample on the next slide.
    - The program must not crash if the file does not exist yet!
  - Add:
    - Prompt the user to enter contact information, including name, email and phone number.
    - Each field must be entered in a separate input
  - Exit: exits the program.
  - Accept contact information from the user and store it in a CSV file.

Lab Section 1: Data Collection (cont.)

- Note that adding a user uses three input prompts, not just one—this is required!
- The output when viewing must be in columns with headers.

```
Please select one of the following:
1. View
2. Add
3. Exit
2
Enter the contact's name: Daffy Duck
Enter the contact's email: dduck@wb.com
Enter the contact's phone: 612-555-1234
Please select one of the following:
1. View
2. Add
3. Exit
2
Enter the contact's name: Mickey Mouse
Enter the contact's email: mmouse@disney.com
Enter the contact's phone: 612-555-9876
Please select one of the following:
1. View
2. Add
3. Exit
1
Name                    Email                   Phone
Daffy Duck              dduck@wb.com            612-555-1234
Mickey Mouse            mmouse@disney.com       612-555-9876
Please select one of the following:
1. View
2. Add
3. Exit
3
Thanks for using the program!
```

Lab Section 2: Weather Forecast

MIPO not required for this section

- Create a program named **weather_json.py**. The program must:
  - Read the data the lab_16_forecast.json file from D2L and convert it from a JSON string to a format that can be used in Python.
  - Print out each forecast period and detailed forecast as shown in the output below:

```
Tuesday             Rain likely, mainly before 10am.  Cloudy, with a temperature falling to around 44 by 5pm. Breezy, …
Tuesday Night       A 20 percent chance of snow after 5am.  Mostly cloudy, with a low around 33. West southwest wind 10…
Wednesday           Snow likely, mainly after noon.  Mostly cloudy, with a high near 36. West wind around 15 mph, with …
Wednesday Night     A 30 percent chance of snow, mainly before midnight.  Mostly cloudy, with a low around 28. Northwest
Thursday            Partly sunny, with a high near 38. Northwest wind around 15 mph, with gusts as high as 25 mph.
Thursday Night      Mostly cloudy, with a low around 28. Northwest wind 10 to 15 mph.
Friday              Partly sunny, with a high near 36. Northwest wind around 10 mph.
Friday Night        Mostly cloudy, with a low around 26. Northwest wind 5 to 10 mph.
Saturday            Partly sunny, with a high near 34. Northwest wind around 5 mph.
Saturday Night      Mostly cloudy, with a low around 28. North northeast wind around 5 mph becoming east southeast in the…
Sunday              Mostly cloudy, with a high near 39. South southeast wind 5 to 10 mph.
Sunday Night        A slight chance of rain and snow.  Mostly cloudy, with a low around 28. South southeast wind around 10…
Monday              Mostly cloudy, with a high near 34. West northwest wind around 10 mph.
```

The output requirement is identical to Section 2 of last week.

The data needs to be presented in two columns!

Lab Section 2 Hints

- The main objective is to understand the structure of a JSON file and how to use data provided in that format.
- Very little code will be necessary (probably less than 10 lines)
- The first thing you will need to identify in the data is the name of the key that holds the list of forecast periods.
- Each forecast period is a dictionary, so the next thing you will need to identify are the names of the keys that hold the name of the forecast period and the forecast description.
- Once you know those things, reading in the file and looping through the field you need should be easy.

Final Steps

Comment your code – don't forget a good doc_string!

Save your programs for future reference.

Submit your code files (data files are not necessary) before the deadline.

Questions or need help? Ask before the deadline!