ITEC 1150

FINAL PROJECT

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 requirements on the right:

- The program must start with the long """doc string""" at top and include appropriate comments throughout. Docstring example:

```
"""
Author: Erik Granse
(erik.granse@minnstate.edu)
Date: 2021-01-27
Description: Gathers info about
coffee sales and displays a sales
report
"""
```

- 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, continued

All assignments must meet the requirements on the right:

All assignments must meet the following requirements:

- All input must be validated for both type (int, float, etc.) and 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!

Final Project Requirements

- The general requirements slides have been simplified for the final project. Please review them as they are not optional.
- There is only one program due for the final project.
- This program is a Pizza Ordering system, similar to the Sandwich Maker project from Chapter 8.
- Unlike the Sandwich Maker, your Pizza Ordering program will download prices and ingredients from the internet and will have to work with that file to guide the user through the order.
- **No ingredients, prices, or categories will be hard-coded into your program.**
- https://itec-minneapolis.s3.us-west-2.amazonaws.com/ingredients.json is the URL of the file you will use to develop your program; grading will be done with a file that has the same structure, but which will have different values.

User Menu

- Your program must display a menu to the user allowing them to:
  - Add a pizza
  - View their order
  - Submit the order
- The menu must be created with PyInputPlus.

Adding A Pizza

- When adding a pizza, the program needs to display the number of the pizza being ordered.
- Your program must download the `ingredients.json` file from the internet! Do not download it yourself and save it with your program.
- The URL is https://itec-minneapolis.s3.us-west-2.amazonaws.com/ingredients.json
- All the options for the pizza must come from the file downloaded from the internet—there must be no categories, choices, or toppings hard-coded in your program!
- See the following slides for details on the input file and further explanation of how it is to be used to help the user order a pizza.

Adding a Pizza: Input File Structure

An example file is shown on the right. The actual file will have many more choices, but that should not matter to your program!

```
{
    "base_options": [
        {
            "category": "crust",
            "options": {
                "Thin": 10.99,
                "Deep dish": 12.99
            }
        },
        {
            "category": "cheese",
            "options": {
                "Three-cheese blend": 0,
                "Fresh mozzarella": 3.99
            }
        }
    ],
    "toppings": {
        "Grilled chicken": 2.25,
        "Onions": 1.19,
        "Black olives": 1.99
    }
}

```

Adding a Pizza: Base Options

- '`base_options`' is a list of options that make up the base of the pizza (crust, sauce, cheese, etc.).
- Each `base_option` has a category and several options within it.
- Your program will loop through each of the base options and ask the user to select the category by giving the options as a menu.
- Using the example on the right, your program should produce output like this for base options:

```
What kind of crust do you want?
1. Thin
2. Deep dish
1

What kind of cheese do you want?
1. Three-cheese blend
2. Fresh mozzarella
1
```

```
{
    "base_options": [
        {
            "category": "crust",
            "options": {
                "Thin": 10.99,
                "Deep dish": 12.99
            }
        },
        {
            "category": "cheese",
            "options": {
                "Three-cheese blend": 0,
                "Fresh mozzarella": 3.99
            }
        }
    ],
    "toppings": {
        "Grilled chicken": 2.25,
        "Onions": 1.19,
        "Black olives": 1.99
    }
}
```

Adding a Pizza: Toppings

- '`toppings`' is simpler than base options.
- Each `topping` will be a simple yes/no question.
- Your program will loop through each of the toppings and ask the user if they want the topping.
- Using the example on the right, your program should produce output like this for toppings:

```
Do you want Grilled chicken? y
Do you want Onions? y
Do you want Black olives? y
```

```
{
    "base_options": [
        {
            "category": "crust",
            "options": {
                "Thin": 10.99,
                "Deep dish": 12.99
            }
        },
        {
            "category": "cheese",
            "options": {
                "Three-cheese blend": 0,
                "Fresh mozzarella": 3.99
            }
        }
    ],
    "toppings": {
        "Grilled chicken": 2.25,
        "Onions": 1.19,
        "Black olives": 1.99
    }
}
```

Viewing the Order

- Viewing the order must display each pizza (numbered), along with all options and toppings for that pizza.
- Each option and topping must have its price displayed.
- The total for each individual pizza must be displayed.
- The grand total for the whole order must be displayed.
- All prices must be in a right-aligned column with a dollar sign and decimal point.

```
Order Summary:
Pizza #1:
Thin                          $ 10.99
Traditional red               $  0.00
Tuscan blend                  $  1.99
Sausage                       $  2.49
Grilled chicken               $  2.25
Black olives                  $  1.99
Green peppers                 $  1.99
Pineapple                     $  1.79
Mushrooms                     $  1.99
Total for pizza #2:           $ 25.48
---------------------------------------
Pizza #2:
Deep dish                     $ 12.99
Traditional red               $  0.00
Three-cheese blend            $  0.00
Sausage                       $  2.49
Onions                        $  1.19
Black olives                  $  1.99
Green peppers                 $  1.99
Mushrooms                     $  1.99
Total for pizza #3:           $ 22.64
---------------------------------------
Grand total:                  $ 48.12
=======================================
```

Placing the Order

- When placing the order, it must be displayed as in Viewing the Order.
- The full order must be written to a file in JSON format—this is to simulate sending the order to the store over the internet.
- A message should be shown that the order was placed, and then the program should exit.

```
…
Green peppers                    $  1.99
Mushrooms                        $  1.99
Total for pizza #3:              $ 22.64
----------------------------------------
Grand total:                     $ 48.12
========================================

Your order has been placed! Thanks for ordering with us!
```

Guidance and Design Suggestions

THIS IS A COMPLICATED PROJECT. I HOPE THESE TIPS WILL HELP YOU BE SUCCESSFUL!

Suggested Data Structures

- I would suggest using a list to store all the pizzas. A list will work well with `json.dumps()` for saving the order when the user places the order.
- For individual pizzas, I would recommend using a dictionary with `ingredient : price` as the `key : value` pairs.
  - Remember that the difference between base options and toppings is only important for how the user chooses them—once a base option or a topping is selected, it's just an ingredient on the pizza.

Interpreting The Ingredients File

- The structure of the file used for grading will be the same as the example in these slides, even if the options are different.
- Remember that after you download the file and read it as JSON, the result is just a dictionary (a slightly complicated dictionary, but a dictionary all the same).
- The first two things you need to get from the ingredients dictionary are the **list** of base options and the **dictionary** of toppings.
- Each base option is a dictionary, with a category key and an options key that returns a dictionary of option : price.
- Toppings are a dictionary of topping : price
- Review Week 8 (Chapter 5) if you need to remember how to get all the keys from a dictionary.

Timeline and Suggestions

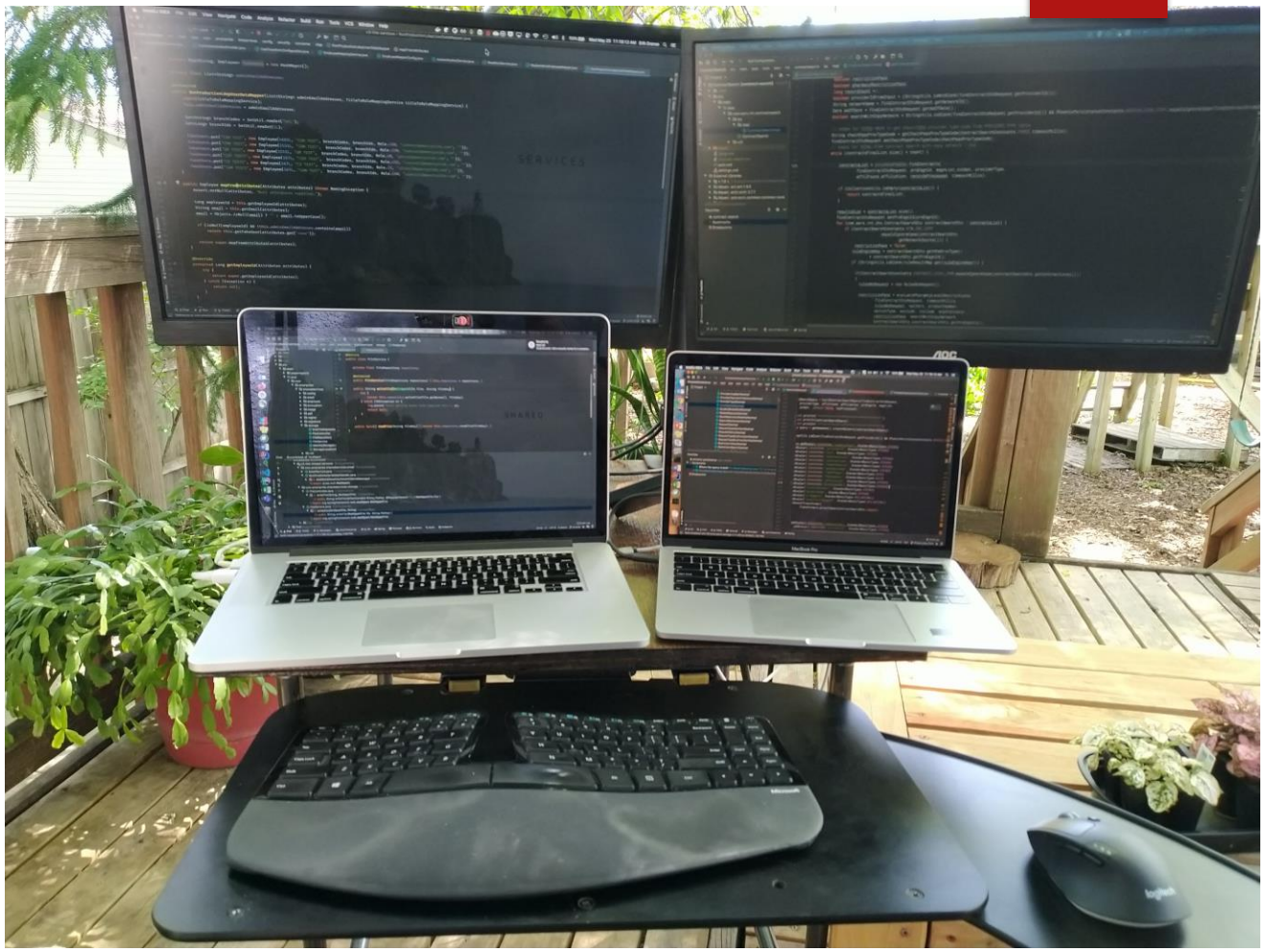
Office Hours on December 13 and December 18 (a Wednesday) will be used for questions and help on this final project.

It is due December 20 at 11:59 PM—this is the last day of the semester, so no late work can be accepted past this time!

I recommend setting a personal due date of December 16 (Monday) so that you have time to submit and receive feedback before the final due date.

Thank You

- I hope you've enjoyed this introduction to programming and wish you success as you continue your journey!



A glimpse into the exciting life of a working programmer