General Instructions

Make sure that you read and then follow the instructions for each task carefully.

Please make sure that you save all of your work to a safe place and that you make **regular back-ups**.

You should begin all tasks with the following steps unless otherwise stated:

- Create a new Python file with the *Project Name* specified in the file.
- It is suggested that you save the file with your completed code (and/or any other files specified) in a folder named for the tutorial week (*week01, week02,* etc.), which should itself be in a root folder called *sdam*. However, you can use whatever method you wish to organise your work – just make sure that each file can be located quickly and efficiently based on the week number and the project name.

If you are in any doubt, please check with your tutor.

Coffee Pseudo-Code
Project Name: make_coffee

Beginner

Create a text file called *make_coffee_pseudo_code.txt*.

Using the notes from the lecture, write the pseudo-code to represent the process of making a cup of coffee.

You can make the pseudo-code as simple or as complex as you like as long as it:

- adequately represents the process of making a cup of coffee, and
- makes use of the pseudo-code notation that was described in the lecture.

Portfolio

The task contribution to your portfolio is:

- The text file containing your pseudo-code *make_coffee_pseudo_code.txt*

Thunderbirds Are Go
Project Name: thunderbird

Beginner

You need to write a program to prompt the user to enter a number between 1 and 4 inclusive.

Your program should then output the following in response:

- If user enters 1, output “Thunderbird 1 pilot is Scott Tracy”
- If user enters 2, output “Thunderbird 2 pilot is Virgil Tracy”
- If user enters 3, output “Thunderbird 3 pilot is Alan Tracy”
- If user enters 4, output “Thunderbird 4 pilot is Gordon Tracy”
- If the user enters any other number, output “Have you never watched Thunderbirds!”

In writing your program you have to complete the following

- Write pseudo-code for the problem and save it in a file called *thunderbird_pseudo_code.txt*.
- Based on the pseudo-code you have produced, write the Python code and save it in a file called *thunderbird.py*.

- Test your program with an input number from the permitted range of numbers – take a screenshot of the output and save it as *thunderbird_valid.jpg*
- Test your program with an input number outside of the permitted range of numbers – take a screenshot of the output and save it as *thunderbird_invalid.jpg*

Portfolio	The task contribution to your portfolio is: • The pseudo-code file for the task • The Python source code file for the task • *thunderbird_valid.jpg*, *thunderbird_invalid.jpg* showing output from *thunderbird.py* when tested

Restaurant menu
Project Name: restaurant

Beginner

Download the *restaurant.py* file from this link restaurant.py and place it in the correct folder of your portfolio.

You are to complete the restaurant program so that:

- A user can select either "1: Lasagne" or "2: Stir Fry" as their main dish
- If "1: Lasagne" is selected as the main, the user can select either "1: Garlic Bread" or "2: Fries" as their side dish
- If "2:Stir Fry" is selected as the main, the user can select either "1: Noodles" or "2: Crispy Seaweed" as their side dish
- The order is displayed showing the main dish and the side dish that the user selected.

Create a design of the above program using stepwise refinement – save it as *restaurant_stepwise.txt*.

Complete your program in Python and save it as *restaurant.py*.

Complete a test plan in Word for your program and save it as *menu_test.docx*.

Test your program with each permitted input and with an incorrect selection for the main dish and then an incorrect selection for the side dish. Your test plan should include:

- Lasagne with Garlic Bread – take a screen shot of the output and store it as lasagne.jpg
- Stir Fry with Noodles – take a screen shot of the output and store it as stir_fry.jpg
- Enter an invalid number for main dish – take a screen shot of the output and store it as main_error.jpg
- Enter Lasagne and then enter an invalid selection for a side dish – take a screen shot of the output and store it as side_error.jpg.

Portfolio	The task contribution to your portfolio is: • *restaurant_stepwise.txt* • The Python source code file for the task

- *menu_test.docx* test plan
- *lasagne.jpg*, *stir_fry.jpg*, *main_error.jpg* and *side_error.jpg* showing output from *restaurant.py* when tested

A Simple Calculator
Project Name: simple_calc

Intermediate

Using stepwise refinement, design a program to prompt the user for two positive integer values and then an operator (+, -, /, or *).

Your program should then output the full sum as text with the answer.

For example, if the user enters 5 2 * your program should output:

"Your sum is 5 * 2 and the answer is 10"

Save your design as *simple_calc_stepwise.txt*.

Code your application in Python based on your stepwise refinement design.

Complete a test plan in Word for your program that should ensure you test a sum using each one of the permitted operators and an incorrect entry for the operator. Save your test plan as *simple_calc_test.docx*.

When complete, test your program using the data in your test plan and save screenshots for each test save the screenshots as *add.jpg*, *subtract.jpg*, *divide.jpg*, *multiply.jpg* and *invalid_operator.jpg*.

Portfolio

The task contribution to your portfolio is:

- *simple_calc_stepwise.txt*
- The Python source code file for the task
- *simple_calc_test.docx* test plan
- *add.jpg*, *subtract.jpg*, *divide.jpg*, *multiply.jpg* and *invalid_operator.jpg* showing output from *simple_calc.py* when tested

Character Generator
Project Name: character_gen

Expert

You need to write a program that generates five attributes for a character (randomly):

1. Strength
2. Intelligence
3. Wisdom
4. Dexterity
5. Constitution

All attributes should be in the range 3 to 18.

The character can be one of the following classes:

- Warrior
- Wizard
- Thief
- Necromancer

Prompt the user to select which class they want their character to be.

The minimum ability scores for each class are as follows:

Class	Strength	Intelligence	Wisdom	Dexterity	Constitution
Warrior	15	-	-	12	10
Wizard	-	15	10	10	-
Thief	10	9	-	15	-
Necromancer	10	10	15	-	-

If the character does not have any of the necessary minimum scores for the attributes required by the selected character class they can trade points from non-required attributes at the rate of 2 to 1 but cannot take an attribute below 3.

For example, a character is created with the following scores and the player wants to be a warrior:

Class	Strength	Intelligence	Wisdom	Dexterity	Constitution
Base score	12	8	13	8	11
Warrior	15	-	-	12	10
Deficit	-3	-	-	-4	-
Surplus	-	5	10	-	-

Therefore the player has 5 + 10 = 15 points to exchange. They require 3 points for strength and 4 points on their dexterity; therefore (3 * 2) + (4 * 2) – 14; so they have enough points to be a warrior. Your program should prompt the user to select which attributes they want to sacrifice, how many points they want to exchange and which attribute they want to assign them to.

In the example, the user takes 4 points from intelligence and assigns 2 to strength (strength: 14), then they take 2 from wisdom and assign to strength (strength: 14) finally they take 8 more from wisdom and assign to dexterity (dexterity: 12) and their final character attributes will be:

Class	Strength	Intelligence	Wisdom	Dexterity	Constitution
Amended score	15	4	3	12	11

If the player did not have sufficient point to exchange then the program should tell them they cannot be that character class and will have to start over again by re-running the program.

Design, build and run your program. You should use stepwise refinement for your design.

Provide a screenshot of a user trying to create a character successfully using the points redistribution process (rather than a character that has the necessary ability scores on initial creation) call your screenshot *character_gen.jpg*.

Portfolio

The task contribution to your portfolio is:

- *character_gen_stepwise.txt*
- The Python source code file for the task

- character_gen.jpg showing output from character_gen.py when tested

All portfolio requirements for this tutorial	
Beginner	*make_coffee_pseudo_code.txt* *thunderbird_pseudo_code.txt* *thunderbird.py* *thunderbird_valid.jpg* *thunderbird_invalid.jpg* *restaurant_stepwise.txt* *restaurant.py* *menu_test.docx* *lasagne.jpg* *stir_fry.jpg* *main_error.jpg* *side_error.jpg*
Intermediate (opt)	*simple_calc_stepwise.txt* *simple_calc.py* *simple_calc_test.docx* *add.jpg* *subtract.jpg* *divide.jpg* *multiply.jpg* *invalid_operator.jpg*
Expert (opt)	*character_gen_stepwise.txt* *character_gen.py* *character_gen.jpg*