ITECH 1400 Foundations of Programming

SEMESTER 1, 2022

ASSIGNMENT 2. FISHING SIMULATION

35 MARKS

0. Introduction. In this assignment you are required to develop a program that simulates fishing: There are 6 fish species in the river which you may catch:

- *Australian Bass (Macquaria Novemaculeata) - commonly less than 4 Kg; excellent eating, when less than 2.5 Kg.*
- *Short Finned Eel (Anguilla Australis)- commonly less than 3 Kg; a good eating fish.*
- *Eel Tailed Catfish (Tandanus Tandanus) - Up to 6.8 Kg; excellent eating, when less than 4 Kg.*
- *Gippsland Perch (Macquaria Colonorum)- commonly less than 10 Kg; excellent eating when up to 6 Kg.*
- *Two more species you should add to this list* ***yourself****. Search the internet for the necessary details.*

Your program should be based on the following assumptions:

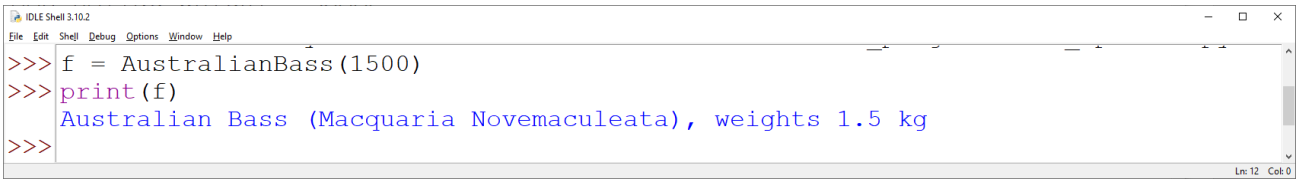
- Every second you catch a fish (perfect fishing).
- The chances (probabilities) to catch each of these six species are equal.
- Weights of fishes of the same species are distributed evenly and range from **0** to the **Maximal Weight**. The value of Maximal Weight for each of the species is given above. For example, Maximal Weight of Australian Bass is 4 Kg.
- Fishing starts with an empty basket which you should implement as a list. If you catch a fish with a weight greater than **500 g** or less than the recommended **excellent eating maximum**, you add it to the basket. Otherwise, release it. For example, only instances of Australian Bass with the weights between 0.5 Kg and 2.5 Kg should be added to the basket.
- Stop fishing immediately as soon as the total weight of all the fishes in the basket exceeds **25 Kg**.

To generate a random fish and weight, you are supposed to use the "**randrange**" function from the "**random**" package.
Fishes of different species must be implemented as objects of the corresponding classes.
For convenience, all weights appearing in the program body should be integers given in **grams** e.g. instead of 3 Kg you should use 3000g. However, when printing outputs on the screen you may use kilograms.

2. Develop a module named fish_species (file "fish_species.py"). This module should contain definitions of the following six classes: AustralianBass, ShortFinnedEel, EelTailedCatfish, GippslandPerch + 2 more classes for the species you add yourself.

class ***AustralianBass*** should contain the following members:

- Variables (Constants):
  - MAX_WEIGHT = 4000
  - MAX_EATING_WEIGHT = 2500
  - NAME = 'Australian Bass'
  - LATIN_NAME = 'Macquaria Novemaculeata'
- The **constructor** should define and initialise a single attribute named "**weight**". The attribute weight must get an integer value between 0 and MAX_WEIGHT.
- A method named "**is_good_eating**": returns True if the fish's weight is between 500 g and excellent eating weight (2500 g for Australian Bass).
- An overridden (redefined) method "**__str__**" that returns a nice, readable string representation of a fish object. Here's an example:



Other fish classes should have **similar**, mutatis mutandis, implementations.

[6×2.5 = 15 marks]

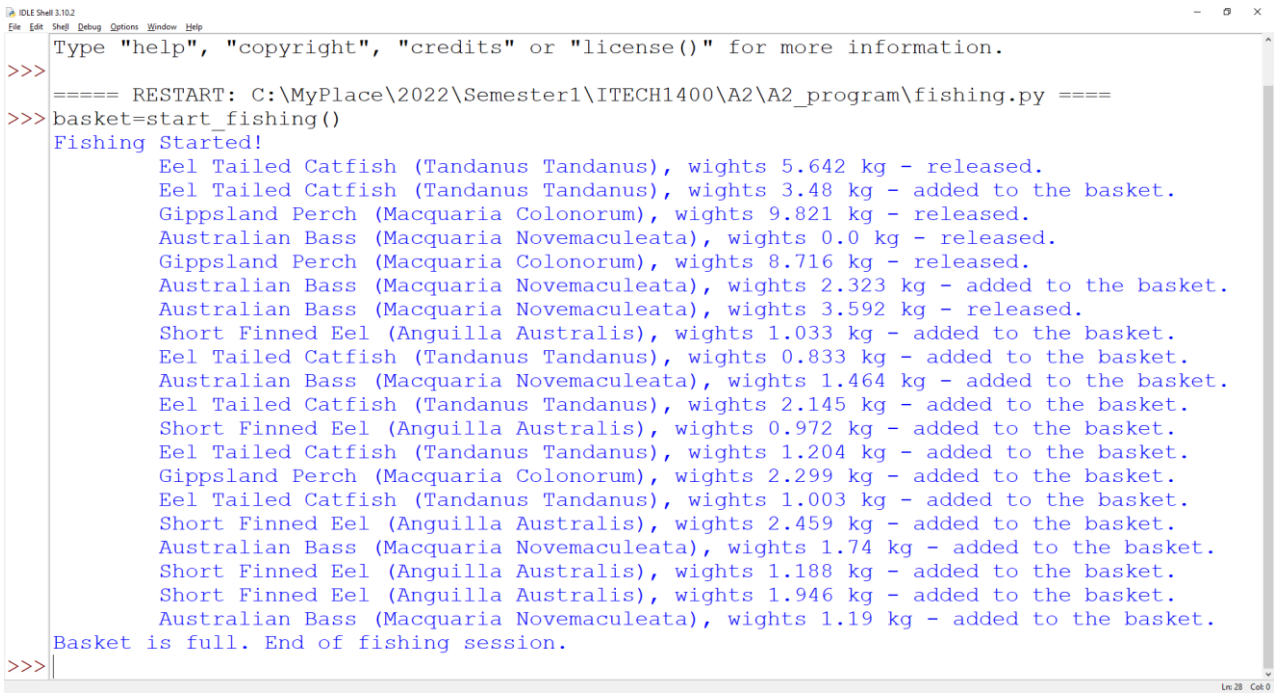
3. Develop a module named "fishing" (file "fishing.py"). This module should import the module "fish_species", so you can use the class definitions from it. In addition, in this module you should define the following functions:

3.1. **Function start_fishing().**

The function simulates fishing process in the following way:

- Every second a random fish is "caught". I.e., every second the program randomly chooses one of the 6 fish species, then randomly generates a weight within valid range (between 0 and the species' MAX_WEIGHT), and then creates the corresponding fish object.
- If the created fish object is_good_eating, the object is added to the basket (implemented as a list). Otherwise, the fish is released, i.e., is not added to the basket.
- Once total weight of fishes in the basket exceeds 25 Kg (25000 g), the basket is returned (fishing ends).

- Fishing results should be printed on the screen, one line per second.
- Here's a sample output:

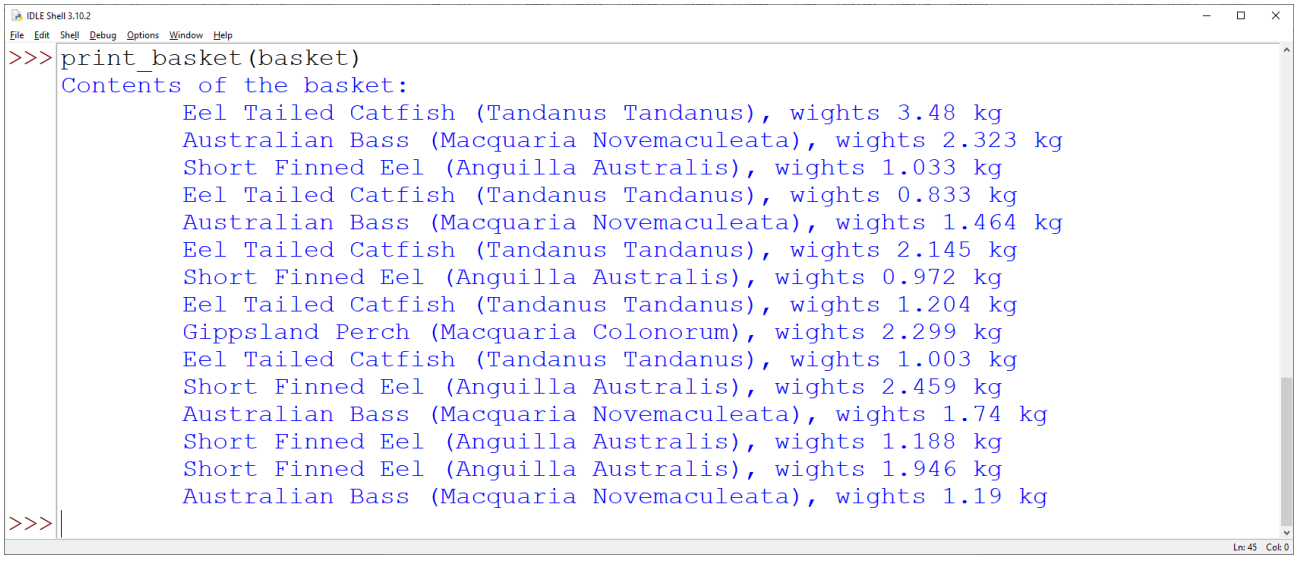


Note. To pause your function's execution for 1 second use **time.sleep(1)** function from the **time** package.

[10 marks]

3.2. Function print_basket(basket).

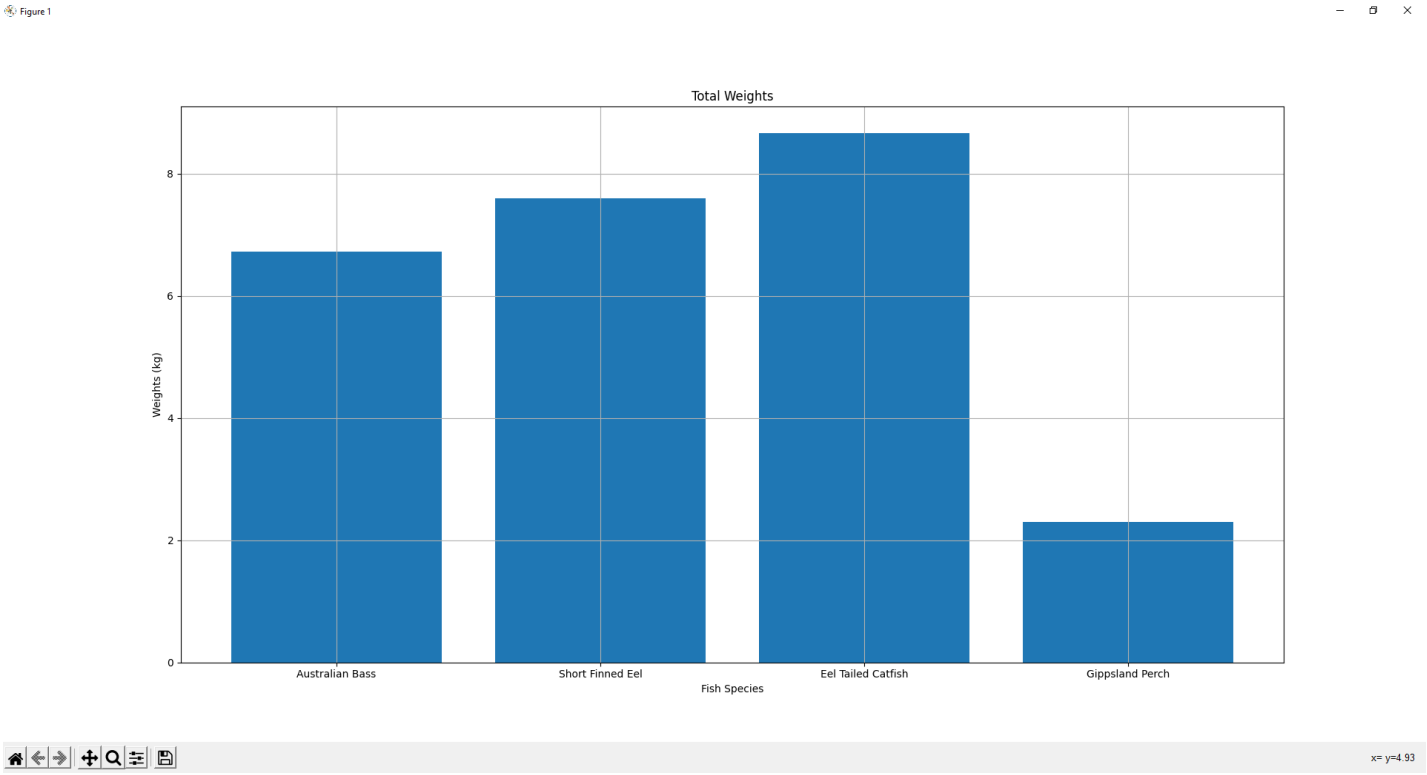
The function prints its argument's (basket's) content on the screen as shown below:



[2 marks]

3.3. Function plot_basket(basket).

The function plots a bar-chart that shows total weights of each of the species in the basket:



Note. In the examples above only four species of fishes are used. However, **in your program you should have 6 species!**

[3 marks]

3.4. Functions save_basket(basket, file_name) and load_basket(file_name).

In this task you must:

- search Python documentation to find out how to use the **pickle** package in order to save Python objects to files and load the saved objects back to programs.
- **save_basket(basket, file_name) –** using **pickle.dump** saves the basket to a **binary file** with the specified name.
- **load_basket(file_name) – using pickle.load** loads a saved object (basket) from the specified file.

[2.5 + 2.5 = 5 marks]

Allocated Marks: See Course Description
Due Date: See Course Description
Please refer to the Course Description for information relating to late assignments and special consideration.

Assignment Submission

You must supply your program source code files and your documentation (if any) as a single zip file named as follows:

<YOUR-NAME>_<YOUR-STUDENT-ID>.zip,

e.g., John_SMITH_30000000

Support documentation should be in PDF format.

Assignments will be marked on the basis of fulfilment of the requirements and the quality of the work.

In addition to the marking criteria, marks may be deducted for failure to comply with the assignment requirements, including (but not limited to):

- Incomplete implementation(s), and
- Incomplete submissions (e.g., missing files), and
- Poor spelling and grammar.

You might be asked to demonstrate and explain your work.
Submit your assignment (all program source files plus your pdf document) to the Assignment 2 Upload location on Moodle before the deadline.

Marking Criteria/Rubric

Student ID:_____**Student Name:** _____

Tasks	**Weight**	**Awarded**
Marks deducted for badly commented or badly written code	(-4)	
Module fish_species		
• class AustralianBass	2.5	
• class ShortFinnedEel	2.5	
• class EelTailedCatfish	2.5	
• class GippslandPerch	2.5	
• 2 additional fish classes	5	
Module fishing		
• Function start_fishing	10	
• Function print_basket	2	
• Function plot_basket	3	
• Function save_basket	2.5	
• Function load_basket	2.5	
Total	35+(-4)	

Appendix. Below is a simple Python program that illustrates the usage of the functions from **random**, **time** and **pickle** packages required in this assignment.

```
import random
import time
import pickle

l = []
for i in range(20):
    #generates a random number between 10 and 20
    random_number = random.randrange(10,21)
    l.append(random_number)
    print('Number {} added to the list.'.format(random_number))
    #pauses execution for 1 second
    time.sleep(1)

#opens/creates a binary file for writing
f = open('rand_numbers','wb')
#save list l to the file f
pickle.dump(l,f)
f.close()

#opens a binary file for reading
f1 = open('rand_numbers','rb')
#load the object(list l) from the file f1
l1 = pickle.load(f1)
print(l1)
f1.close()
```