Assignment 2

Conditions, Loops, and Loot Boxes

SUBMISSION REQUIREMENTS: Submit a single zip file called **assignment2.zip**. It must contain all of your question code. Information regarding deductions for late submissions, invalid submissions, and grade disputes are available on the cuLearn assignment submission page.

This assignment has **30 marks**.
A full marking scheme is available on the cuLearn submission page.
Bonus marks will not exceed 100% but will be applied both parts of the assignment.

MARKING NOTES: A full marking scheme with notes, deductions, and policies is available on the cuLearn assignment submission page.

Assignment Overview

Before a game designer sits down to make a game, they always think - “But how can I make the **most** money from my users, with the least amount of effort?” In this assignment, we won’t try to make the game, but instead we’ll make the part of the game that *really* makes all the money - Loot boxes!

> **What is a loot box?**
> Loot boxes are a way of selling items to users. Users will purchase a “loot box” and opening the box provides a random item. Typically these boxes produce one or more items, with some items being more rare than others. Really, it’s quite similar to gambling and there is a lot of debate around if they are ethical or not. I am not endorsing the use of loot boxes in games!
>
> Note: You may be familiar with “gacha games”. These are more or less the same concept. Feel free to theme accordingly!

We are going to make a text-based store to purchase loot boxes. There will be three tiers of loot boxes: Common, Rare, and Epic. Each loot box will drop 3 items with different probabilities, depending on the type of box. At the end, you will open the boxes and see what you get!

The Tools You’ll Use (These tools are not mandatory):

- We use loops any time we want to repeat an instruction multiple times
- We use while loops any time we want to loop until some event
- We use for loops when we know how many times we want to loop
- We can use the string function, **.format()**, to print nice strings (info in Tutorial 2!)
    - You can format a float to should two decimal places with `{:.2f}`
- We can import the **random** module to randomly generate loot from boxes
- We can use the string's **isdigit()** function to check if a string is a positive integer *before* trying to convert it
- **\t** can be used in a string to tab forward

Theme: Feel free to use any theme you would like! Give your game a name, interesting box names, have fun with it. You are allowed to use the names I've provided here or change them out. It can even be a game that exists, or based on another existing piece of media.

Marking: A marking scheme will be released shortly after the specification PDF goes live. Check the submission page periodically for the uploading marking scheme.

Tips:

- Before trying to program anything, make sure to read **the entire assignment**, make sure you **fully understand** what it is asking, and use flowcharts and pseudocode to help plan your logic.
- Submit your solutions periodically to make sure that you'll still get grades if something goes wrong with your final submission.
- Always verify your submission after uploading it to make sure you submitted the correct files
- Make use of office hours! The best way to receive help is to attend TA or instructor office hours, or if that isn't possible, seeking clarifications on the cuLearn forums. Always make sure that the question hasn't already been asked.

Problem 1: A Single Purchase

Submission: Save your file as **a2q1.py** and add it to the zip submission.
Goals: Learn how to use while loops to validate user input, make use of string formatting, and perform simple math calculations.

This is the starting point for our text-based purchasing system for the game *Raven Runner* (pick your own game if you'd like!). You will build the menu that allows the user to buy one or many of a single type of loot box.

1. The program starts by asking for the player's username
2. The program shows a menu to the user to purchase one of three loot boxes.
    - **Display:** The menu must display the number for the option, the rarity, the cost, and the name of the box.
3. The user enters a number that corresponds with the loot box they would like to purchase
    - **Invalid Inputs:** The user *is allowed* to type incorrect inputs. If they do not type a valid number, it should print an error message and show the menu again. You can use a **while loop** to help with this.
4. The user is then asked how many of those loot boxes they would like to purchase
5. The user is given a personal message, thanking them for their business (using their name), and a receipt is shown for their order.
    - **Receipt Info:** The receipt should show the loot box name, the number of boxes they purchased, and the total cost of the purchase.

Example (user input is **bold**):

```
HELLO, GAMER! Welcome to the Raven Runner Loot Box Purchasing System.
First, what's your player name? Connor

Please select a loot box from the menu below:
      1. [Common]  The GeeGee (1.50)
      2. [Rare]    The Raven (3.00)
      3. [Epic]    The Three-Eyed Raven (7.99)
> 4
Error: That was not a valid selection. Please enter a number between 1-3
Please select a loot box from the menu below:
      1. [Common]  The GeeGee (1.50)
      2. [Rare]    The Raven (3.00)
      3. [Epic]    The Three-Eyed Raven (7.99)
> Common
Error: That was not a valid selection. Please enter a number between 1-3
Please select a loot box from the menu below:
      1. [Common]  The GeeGee (1.50)
      2. [Rare]    The Raven (3.00)
```

```
      3. [Epic]   The Three-Eyed Raven (7.99)
> 1
How many GeeGees ($1.50) would you like to purchase? 4
Thanks, Connor! Here is your receipt:
--------------------------------
4x    GeeGees ($1.50)
--------------------------------
Total Cost: $6.00
Thank you! Good luck, gamer!
```

Important Notes:

- You an not ever assume the user will enter correct inputs
- Your output should be nicely formatted

Problem 2: Bulk Purchases

Submission: This should be a **separate file** from problem one, save to the file **a2q2.py**
Goals: Use multiple variables to track independent information, make use of while loops to process until the user wants to quit, begin writing reusable code.

Firstly, **copy your a2q1.py** solution into a new file. The TAs will need both solutions. Save it as **a2q2.py**.

In this problem, you will modify your previous solution so that we can purchase multiple loot boxes at once.

1. Our menu will now need a “Complete Purchase” option to stop purchasing (you can rename this, as long as it is clear)
2. Our menu will continue prompting for additional purchases until the selects the Complete Purchase option
3. We will need to track how many of **each type of loot box** we are purchasing - one variable per type
4. If the user selects the same box a second or third time, it should be **added to the previous amount**
5. The receipt should contain a list of each box they have purchased; if they did not purchase a type of box, it should not appear on the receipt

Tips: You will want variables for each type of box with information like the cost, the amount you’ve purchased, and possibly even the name of the box. For clarity, try defining these variables together and early in the program.

Tip: You can use global constant values to give names to the selections in your code! Example:

```
COMMON = 1
RARE = 2
EPIC = 3
QUIT = 4
```

Example:

```
HELLO, GAMER! Welcome to the Raven Runner Loot Box Purchasing System.
First, what's your player name? Connor

Please select a loot box from the menu below:
      1. [Common] The GeeGee (1.50)
      2. [Rare]   The Raven (3.00)
      3. [Epic]   The Three-Eyed Raven (7.99)
      4. Complete Purchase
> 1
How many GeeGees ($1.50) would you like to purchase? 2
Please select a loot box from the menu below:
      1. [Common] The GeeGee (1.50)
      2. [Rare]   The Raven (3.00)
      3. [Epic]   The Three-Eyed Raven (7.99)
      4. Complete Purchase
> 2
How many Ravens ($3.00) would you like to purchase? 4
Please select a loot box from the menu below:
      1. [Common] The GeeGee (1.50)
      2. [Rare]   The Raven (3.00)
      3. [Epic]   The Three-Eyed Raven (7.99)
      4. Complete Purchase
> 3
How many Three-Eyed Ravens ($7.99) would you like to purchase? 1
Please select a loot box from the menu below:
      1. [Common] The GeeGee (1.50)
      2. [Rare]   The Raven (3.00)
      3. [Epic]   The Three-Eyed Raven (7.99)
      4. Complete Purchase
> 4
Thanks, Connor! Here is your receipt:
---------------------------------
1x    GeeGees ($1.50)
4x    Ravens ($3.00)
1x    Three-Eyed Ravens ($7.99)
---------------------------------
Total Cost: $22.99
Good luck, gamer!
```

Problem 3: Open Those Boxes

Submission: This should be a **separate file** from problem two, save to the file **a2q3.py**
Goals: Use nested for loops to repeat multi-step tasks a set number of times, understand how to use cumulative probability in conditions

Firstly, **copy your a2q2py** solution into a new file. The TAs will need both solutions. Save it as **a2q3.py**.

Now it's time to open our boxes!

After the purchase is complete and the receipt is shown, we will use one **for loop** for each type of box and open them. We will generate **3 items per box** - use a nested for loop for this. Each box will have different likelihoods of different boxes.

- A common box has an 80% chance to be common, 15% rare, 5% epic
- A rare box has a 50% chance of common, 40% rare, 10% epic
- An epic box has a 30% chance of common, 50% rare, 20% epic

When opening boxes, your program should:

1. For each box of one type…
    a. For each of the three items to generate…
        i. Generate a random number between 0 and 1.
        ii. If it is less than or equal to the first probability (eg. 0.8), it's a common
        iii. Otherwise, if it is less than the first probability plus the second (eg. 0.8+0.15 = 0.95), it's a rare
        iv. Otherwise, it is an epic.
        v. Let the user know which item they got.

Tips:

- For each **type** of box, you will need one for loop for each box of that type, and one for loop for each item in that box
- You can use `random.random()` to get a float between 0 and 1

Bonus (2 marks): Write your box opening code as a **function**, named `open_box()` which takes two parameters: The amount of boxes to open, and the type of box being opened. It does not need to return anything. It should **only contain the nested for loop for opening boxes once**. Use variables to determine the correct probabilities.

Example on Next Page

Example, starting from the receipt:

```
Thanks, ! Here is your receipt:
---------------------------------
4x   GeeGees ($1.50)
1x   Ravens ($3.00)
4x   Three-Eyed Ravens ($7.99)
---------------------------------
Total Cost: $36.46
Good luck, gamer!


Time to Open Boxes!
------------------------
Opening common box 0
     It's a common item!
     It's a common item!
     It's a rare item!
Opening rare box 0
     It's a common item!
     It's a rare item!
     It's a rare item!
Opening epic box 0
     It's a rare item!
     It's an EPIC ITEM!
     It's a rare item!
Opening epic box 1
     It's a rare item!
     It's a common item!
     It's a rare item!
Opening epic box 2
     It's a common item!
     It's a common item!
     It's a common item!
Opening epic box 3
     It's a common item!
     It's a rare item!
     It's a rare item!
```

Recap

Your zip file should contain your **a2q1.py, a2q2.py** and **a2q3.py**, files.
Submit your **assignment2.zip** file to cuLearn.
Make sure you download the zip after submitting and verify the file contents.
Late submissions will receive a 2.5%/hour deduction up to an 8 hour cut-off period

---