FIT1045 Algorithmic Problem Solving – Workshop 2.

Objectives

The **objectives of this workshop** are:

- To get familiar with Python.
- To be able to perform iteration and selection in Python.
- To be able to perform basic string and numerical manipulation.
- To be able to import and use the math package and random package.
- To be able to do simple input/output.

Useful Material

Selection and Iteration: `https://docs.Python.org/3/tutorial/controlflow.html`

Strings: Some introductory information is available at `https://docs.Python.org/3/tutorial/introduction.html#strings` and more at `https://docs.Python.org/3/library/stdtypes.html#string-methods`.

Preparation

Task 0

We define an algorithm to be a process that meets the criteria given in lecture 1. For problems 2, 3, and 4 in Workshop 1, fill in the following table and decide if your solution to the task constitutes an algorithm.

Task	Finiteness	Definiteness	Input	Output	Effectiveness	Algorithm
1						
2						
3						

Note: If you have not completed the coin flipping exercise from Workshop 1, make sure you complete it before attending this weeks workshop.

Flipping Coins Continued...

In this task we will be continuing in our use of the random library from Workshop 1. This task will be divided up into 3 parts. It is important that you complete this task before continuing with the rest of the workshop.

Task 1(a)

Write a program that takes a number n as input from the user and simulates n coin flips printing the results each time. Using your knowledge of selection, improve upon the output suggested in Workshop 1, Task 2(b).

Example:

```
How times would you like to flip the coin? 5
The coin came up heads.
The coin came up tails.
The coin came up tails.
The coin came up heads.
The coin came up heads.
```

Task 1(b)

Modify the program from task 1(a) to store the number of heads and tails in variables. Once you have generated the variables, print the total number of heads and the total number of tails. Calculate the ratio of heads to the total coin flips.[1] Is this ratio what you would expect, what happens to this ratio as you change the probability of the coin coming up heads? What happens as the number of coins being flipped becomes large?

Task 1(c)

Up to this point you have been using the 'random.random()' function to generate a random number between 0 and 1. Using the documentation from the random library, use 'random.randrange(a)' to extend your program from part 2 so that it uses 3 sided coins.[2]

Note: What happens if you call 'random.randrange()'? Discuss these results with the person sitting next to you and don't forget to ask for help if you get stuck.[3]

The MU Game

Task 2

The MU game is based on the MU puzzle described in "*Godel, Escher, Bach: an Eternal Golden Braid*" by Douglas R. Hofstadter. To begin the game the user must supply a character string, which consists only of the characters M, U, and I. At each stage of the game the user may apply one of the following rules to change their current string into a new string.

Rule 1 If the current string's last character is I, then add U onto the end of the current string.

Rule 2 If the first character of the current string is M, and the rest of the string is denoted by R (so the current string looks like MR) then the new string is MRR.

Rule 3 If III is a substring of the current string, then replace one of the occurrences of this substring by U.

Rule 4 If UU is a substring of the current string, then delete one of the occurrences of this substring.

Note:
We have included some skeleton code on moodle to assist you with this task called *Workshop2Skeleton.py.* It is up to you to decide whether you wish to use this; you may start from scratch if you wish.

Examples for Rule 2:

Current String	New String
MIUI	MIUIIUI
MUM	MUMUM
MU	MUU

Examples for Rule 3:

Current String	New String
MIII	MU
MIIIUIII	MUUIII or MIIIUU
MIIII	MUI or MIU

[1]The ratio of two numbers, A and B, is given by $\frac{A}{B}$
[2]`https://docs.python.org/3/library/random.html`
[3]By this we mean call 'random.randrange()' without a parameter.

Examples for Rule 4:

Current String	New String
MUUIUU	MIUU or MUUI
UUU	U

In this workshop your goal is to write a Python program to allow the user to play the MU game. Your program should ask the user for a string. It will then need to check whether or not this string consists of the characters M, U, and I. The program should then repeatly ask the user which of the following commands they want the program to do:

- Apply Rule 1.
- Apply Rule 2.
- Apply Rule 3. For this command the user will also need to state which substring III to replace.
- Apply Rule 4. For this command the user will also need to state which substring UU to delete.
- Quit.

If the user does not quit, the program should apply the rule selected to the string, print the new string, and then ask the user again which command to do.

For example a sample run of your program may look like:

```
Enter a string: MI
Select a rule (1-4) or q to quit: 2
New string is MII
Select a rule (1-4) or q to quit: 2
New string is MIIII
Select a rule (1-4) or q to quit: 1
New string is MIIIIU
Select a rule (1-4) or q to quit: 3
State starting position of substring III you want to replace: 1
New string is MUIU
Select a rule (1-4) or q to quit: q
```

The MU puzzle is whether you can start the MU game with the string MI and end up with the string MU. In his book Douglas Hofstadter investigates whether or not this is possible.

Extension Questions

All questions contained here either extend on the current week or look into topics that will be introduced in depth in the future. They may require a certain level of self study to complete during thier prescribed lab. If you have completed all other work, we highly recommend that you have go at these questions. **These problems as they appear here are not examinable. Any examinable content will be covered in the main section of later workshops.**

Task 3: Rock, Paper, Scissors

Rock, paper, scissors is a game in which who people battle by choosing either rock, paper or scissors. Paper beats rock, rock beats scissors and scissors beats paper.

Write a program that randomly chooses either rock, paper or scissors and compares this to the user input. The program should then announce the winner or a draw if there is no winner. The game should keep going until the user enters ‘quit’.

Example:

```
What do you choose? Rock
Your opponent chose paper... You lose!
```