# Software for Digital Innovation CIS4044-N Tutorial 3: Functions and Advanced Lists

## Before You Start

Ensure you have completed all tasks from Tutorial 2. Any issues seek help from your tutors.

**Hint:** In Python, if you create an empty function the program won’t work. Every function needs at least one statement. Programmers often define their functions first, then fill them in, however. You can use the pass keyword to get around this. For example:

def empty\_function():

    pass

Attempt to complete this set of tasks before your next tutorial.

## Introduction

This session aims to familiarise you with functions, as well as some more advanced list methods.

## Question 1: Shape Area Functions

You are tasked with drawing on work from tutorial 1 to write functions that calculate the area of a sphere, cylinder and cone.

1. First, create three functions:
   1. sphere(r) that takes the radius of the sphere as r and returns its area.
   2. cylinder(r,h) that takes the radius of the cylinder as r and its height as h, returning its area.
   3. cone(r,h) that takes the radius of the cone as r and its height as h, returning its area.
2. Formulae for all 3 can be found here: <https://www.bbc.com/bitesize/articles/ztqsdxs>
3. Load these functions into IDLE, and using F5 test each one out on the command line.

## Question 2: Recursive User Input

Create a menu system using conditionals to ask the user to specify which shape they would like to calculate the area of. If they select an invalid option, use recursion to ask the question again. If you cannot remember what recursion looks like, consult the lecture materials. Your menu should ask for input by number. For example:

>>> Please type 1 for sphere, 2 for cylinder and 3 for cone: \_

## Question 3: Exam Results

Write a program to provide some grade management software for a university. Download and edit the files results.py and datafile.py from Blackboard to proceed with this exercise. Do not be afraid to initially experiment with the files to see how they work!

1. Define a function that takes the results list as an argument and prints out a pass list, printing each student’s name, score and grading (use distinction 70%, merit 60% and pass 40% grading schema).
2. Define a function that takes the results list as an argument and prints out a list of all students that have failed with a mark below 40%, but where all students with a mark of 30-39 are entitled to a resubmission.
3. Define a function that takes the results list and returns the following performance statistics, printing them out after the two results lists:
   1. Highest mark and student
   2. Marks range (highest and lowest)
   3. Average mark overall
4. Define a function that returns how many students scored above the average and how many below, including failures and print these statistics.

## Question 4: Slicing

You are writing an adventure game and you are required to implement some of its functionality. For this question, you’ll need to read a tutorial on the Python random library such as this one: <https://pythonspot.com/random-numbers/>

1. Your adventurer has a backpack containing the following items:
   1. Water flask
   2. Cheese
   3. Gold coins
   4. Handkerchief
   5. Tinderbox
   6. Scrolls
   7. Dagger
   8. Rope
   9. Nuts
   10. Pipe
   11. Tobacco
   12. Wine skin
   13. Herbs
   14. Axe
2. Create a new Python file and populate a list called rucksack with these items as strings.
3. Now, sort your rucksack using the sort() function and print out a list of the items in it.
4. Now print how many items you have in your rucksack.
5. Your adventurer finds a treasure chest which contains gems and a necklace. Add the chest items to your rucksack and sort it again, printing its contents again.
6. Your adventurer goes to sleep and a thief steals 5 random items from your rucksack. Remove 5 random items and list your rucksack’s contents again.

## Question 5: Dice

Create a program to throw 2 dice 100 times. Record how often each number from 2 to 12 is thrown in a suitable list and then print out a graph of your results like so:

Distribution Chart

Score Rolls

2 1 \*

3 5 \*\*\*\*\*

4 11 \*\*\*\*\*\*\*\*\*\*\*

5 13 \*\*\*\*\*\*\*\*\*\*\*\*\*

6 15 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

7 16 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

8 18 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

9 8 \*\*\*\*\*\*\*\*

10 6 \*\*\*\*\*\*

11 4 \*\*\*\*

12 3 \*\*\*

## Question 6: Coin Tossing

Write a program to toss a coin.

1. Create a program to toss a coin 100 times, recording whether it lands heads or tails as “H” or “T” in a data list. Print your list in 5 rows of 20.
2. Now modify your program to find the longest continuous run of heads and the longest continuous run of tails.
3. Now modify your program to count how many times three heads were tossed in a row and output your result.

## Question 7: Comparing Many Numbers

Write a list management program.

1. Create a program to generate 20 random numbers from 0 to 99 and store them in a list.
2. Locate second largest and smallest numbers Now modify your program to determine the second largest and the second smallest number in the list and output your results.
3. Now modify your program to create a list of 20 random numbers, output the list, and then remove all entries greater than the third highest number in your list and output the resulting list after this operation.
4. Re-organise list data Create a program to generate a new list of 20 random numbers from 0 to 99.
5. Insert number 50 in the middle position of the list. All numbers in the first half of the list that are greater than 50 should be removed from the first half and appended to the second half.
6. Then do the same with all numbers in the second half of the list that are smaller than 50, appending them to the end of the first half of the list before the number 50.

## Document History

Revision 0 (09-Oct-20): This is the initial version of the 2020/21 exercise.