EMAT10007 – Introduction to Computer Programming

Exercises – Week 3.
Loops

Getting Started: Pycharm IDE

Open PyCharm on linux lab computers

- Scroll down to bring up log in screen and log in with your UoB user name and password.
- Click activities (top left corner) to bring up the side panel.
- Click the grid of 9 dots to bring up applications.
- Choose JetBrains PyCharm
- When prompted about the user agreement click accept and read

Create a new project and Python file

- Click New project or File >> New project >> Pure python
- Unselect 'Create a main.py welcome script'
- Note the file location:
  /home/**UoB_username**/PycharmProjects/**your_projectname**/venv
  where **UoB_username** is your UoB username and rename **your_projectname** to be a name of your choice e.g. EMAT10007_exercises
- Right click on the folder icon with project name next to it (top left of window).
- Choose new >> python file
- Give your file a name e.g. week_1_exercises.py

Write and run code

Type some code and click the green play arrow at the top to run.

Save your project

File >> Save all to save your wor

Open a project you created previously

Click File >> Open >> /home/**UoB_username**/PycharmProjects/**your_projectname**/venv, Open >> New window

Rules for naming variables

- Variable names may contain letters or numbers
- Variable names must begin with a letter
- Variable names are case sensitive (`time` is not the same as `Time`)
- Some `keywords` are reserved by the Python language and cannot be used as variable names. For a full list of keywords reserved by Python, enter the following run the following comand in the editor you are using:

  ```
  help("keywords")
  ```

- Use a consistent naming convention:
  - `snake_case`: lower case letters, words separated by underscore (_)
  - `camel_Case`: first letter of each word capitalised, excluding first word
  - `Pascal_Case`: first letter of each word capitalised

Exercise 1 - For loops

`for` loops are used for **definite iteration**, when the number of repetitions is specified explicitly in advance.

1. Use a for loop to cast each value in the sequence [1.5, 1.0, 2.1, 3.8] as an integer and print each integer value.
2. Find the mistake(s) in the following program, which is meant to sum the first 10 multiples of 5:

```
total = 0
for i in range(1,10)
total = total + 5 * i
```

Fix the program so that the final value of `total` is 275, and print the value of `total`.

3. Compute the factorial of 10 = 3,628,800. Recall that the factorial of an integer n is defined as $n! = n \times (n-1) \times (n-2) \times \ldots 2 \times 1$.
4. Using a `for` loop and the `break` keyword, determine how many positive cubic numbers are less than 2,000. Recall that cubic number is a number of the form n^3 where n is an integer. Answer = 12
5. Use the `zip` function to sum each pair of elements, taking one element from sequence a = [1.4, 2.2, 2.1, 3.8] and one element from sequence b = [0.1, 1.1, 2.1, 1.2], in the order that they appear in the sequence, and print the result of each addition.
6. Create a variable and assign it a string value. Using the `zip` function and the `range` function write a loop which prints both each letter and its position in the string.
   Hint: The Python `len()` function returns the length of a string.
   Note: *The Python function,* `enumerate` *can be used to achieve this operation and avoids the need to define the range of values needed for the counter*
   https://www.w3schools.com/python/ref_func_enumerate.asp
   Now edit the code to use the `break` keyword to terminate the loop prematurely if the letter is 'e' *before* printing the letter and its position

Exercise 2 - While Loops

`while` loops are used for **indefinite iteration**, the code block repeatedly executes until some condition is met.

1. Using a while loop determine how many positive cubic numbers are less than 2,000. Recall that cubic number is a number of the form n^3 where n is an integer. Answer = 12
2. Write a program that finds the smallest power of 2 that is greater than 100 = 128
3. The Python function `input()`:
   - displays a string given in the parentheses () to the user
   - accepts typed input from the user this types input within the program as a string

   Write a program that prompts the user for a password until a value that matches the password `my_password123` is given
4. Find the mistake(s) in the following program, which is meant to Find the greatest power of 4 that is smaller than 200. Answer = 64:

```
exponent = 0
power = 4 * exponent

while power > 200:
    result = power
    power = 4 * exponent

print('largest power = ', result)
```

Exercise 3 - Choosing an appropriate loop type

1. Write a program that calculates how many years it would take for the value of a savings account to exceed £400, if the initial (and only) deposit made is £100 and the annual interest is 5%. Answer: 29 years.
2. A ball is dropped (initial velocity $u = 0ms^{-1}$) and falls towards the ground with acceleration due to gravity of $a = 9.81ms^{-2}$. It is assumed that no other forces act on the ball so the distance travelled by the ball, d (m), at time t (s), can be found by:

$$d = ut + \frac{1}{2}at^2$$

   Print the distance from the start position the ball has fallen at 0.2 s intervals for 2 s, assuming the ball does not reach the ground within this time.
   Answer = 0.0 m, 0.196 m, 0.785 m, 1.766 m, 3.139 m, 4.95 m, 7.063 m, 9.614 m, 12.557 m, 15.892 m, 19.62 m.
3. The value of π can be approximated using the Leibniz formula:

$$\pi_N = \sum_{n=0}^{N} \frac{8}{(4n+1)(4n+3)}$$

   where N is a large number. Taking the limit as $N \to \infty$ produces the exact value of π, but this requires evaluating an infinite number of terms, which is impossible on a computer. Therefore, we can only approximate the value of π by using a finite number of terms in the sum. Use this formula to compute approximations to π by taking $N = 100$ (answer = 3.1366 to 5 s.f.), $N = 1,000$ (answer=3.1411 to 5 s.f), and $N = 10,000$ (answer = 3.1415 to 5 s.f.).

Exercise 4 - Nested loops

1. Use two `for` loops to compute the double sum as 5,275.

$$S = \sum_{i=1}^{10} \sum_{j=0}^{5} j^2(i+j)$$

2. A prime number is a natural number greater than 1 that is not a product of two smaller natural numbers. In other words, a prime number cannot be written as a product of two natural numbers that are both smaller than it. Write a program that prints all prime numbers between 1 and 150.
   Hints:
   - Remember the modulo operato, `%`, gives the remainder when one number is divided by another.
   - Use two nested loops to cycle through each value, then cycle through the series of possible factors.

   Answer = 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149.

Exercise 5 - Text based adventure game

Use the control structures you have learnt so far to build a text-based adventure game (example below). When you have built your game, get the person sitting next to you to play it. You can also send it to me at hemma.philamore@bristol.ac.uk :)
Example game:

```
print('You enter the castle.')
print('You see a mysterious figure appear in the print distance.'
print('Where would you like to go?')

# Ask user to choose direction
user_input = input('Choose direction (Options: right/left/backward)')

# Wait until valid response given by user
while user_input != 'right' and user_input != 'left' and user_input != 'backward' :
    user_input = input('Please enter a valid option: right/left/backward')

# Act on user input
if userInput == 'right':
    # Ask user to choose direction
    print('You see a portal)
    user_input = input('Enter the portal? (Options: yes/no)')

    # Wait until valid response given by user
    while user_input != 'yes' and user_input != 'no':
        user_input = input('Please enter a valid option: yes/no')

    # Act on user input
    if userInput == 'yes':
        print('The portal takes you to a black hole!)
        print('you lose the game!)

    else:
        if ...

elif userInput == 'left':
  print("You find that this door opens into a dead end.")

else:
  if ...
```