Intro to Python
Exercises and Projects

1 ONLINE REFERENCES 1

1.1 BASIC REFERENCES 1
1.2 INTERMEDIATE REFERENCES 1
1.3 ADVANCED REFERENCES 1

2 EXERCISES 2

2.1 IF - ELSE 2
2.2 LOOPS 2
2.3 FUNCTIONS 4
2.4 STRING / LISTS / TUPLES 4

3 PROJECTS 6

3.1 GUESS A RANDOM NUMBER 6
3.2 HANGMAN 6

1 Online references

1.1 Basic references

https://www.w3schools.com/python/

1.2 Intermediate references

https://www.pythontutorial.net/

https://www.programiz.com/python-programming/first-program

https://pynative.com/python/

1.3 Advanced references

https://realpython.com

Official tutorial
https://docs.python.org/3.11/tutorial/

2 Exercises

2.1 If - Else

Exercise solution: `solutions-if-else`

1. Write a program to check whether a number entered by a user is multiple of 6, and display a message correspondingly (for e.g. Number is / is not a multiple of 6).

2. Write a program to obtain the number of units from the user and calculate the electricity bill based on number of units according to the following criteria:

Number of units	**Charge**
First 100 units	No charge
Next 100 units	RM 5 per unit
After 200 units	RM 10 per unit

For e.g.
For 30 units, there is no charge
For 150 units, the charge is 250
For 300 units, the charge is 1500

3. A school uses the following mark ranges for determining a student's grade:

- Below 25 - F
- 25 to 44 - E
- 45 to 50 - D
- 51 to 60 - C
- 61 to 80 - B
- Above 80 - A

Obtain marks from a user and print the corresponding grade.

4. Get the ages of 3 people and determine the oldest among them. Note: There is an inbuilt function in Python that allows you to perform this immediately or you could you use the `list.sort` function. However, you should implement your solution with just `if-else` statements.

2.2 Loops

Exercise solution: `solutions-loop`

1. Write a program to accept a number n from a user and calculate the sum of all even numbers between 1 to n inclusive. For example, if for n = 7 the output should be 12 (2 + 4 + 6), for n = 10 the output should be 30 (2 + 4 + 6 + 8 + 10)

2. Write a program to accept a number n from a user and count the total number of digits in n. For example, if n = 246, the result is 3, and if the number is 73869, the result would be 5. You can use the floor division operator // in your answer (https://www.w3schools.com/python/python_operators.asp)

3. Write a program to display all prime numbers within a range of two numbers specified by the user. A Prime Number is a number that cannot be produced by multiplying other whole numbers. It is any natural number greater than 1 that is not a product of two smaller natural numbers

4. The Fibonacci Sequence is a series of numbers. The next number is found by adding up the two numbers before it. The first two numbers are 0 and 1.

   For example: 0, 1, 1, 2, 3, 5, 8, 13, 21. The next number in this series is 13 + 21 = 34.

   Write a program to accept a number n from a user and print out the Fibonacci sequence to n terms. For e.g. with n = 9, we have the same sequence as above, and with n = 5, we have 0, 1, 1, 2, 3

5. Write a program that obtains from the user a particular digit x and a number n and then calculates the sum of an increasing number of digits of x up to n terms. For example:

   When x = 2 and n = 5, the sum of the series is 2 + 22 + 222 + 2222 + 22222 = 24690
   When x = 4 and n = 3, the sum of the series is 4 + 44 + 444 = 492

6. Write a program to accept a number n from the user and print n rows of numbers in accordance to the pattern shown below:

For e.g. for n = 3

```
1
2 4
3 6 9
```

For e.g. for n = 5

```
1
2 4
3 6 9
4 8 12 16
5 10 15 20 25
```

7. Write a program to accept a number n from the user and prints rows with increasing / decreasing number of asterisks with the row of n asterisks right in the middle. For e.g.

For n = 3

```
*
*  *
*  *  *
*  *
*
```

For n = 5

```
*
*  *
*  *  *
*  *  *  *
*  *  *  *  *
*  *  *  *
*  *  *
*  *
*
```

2.3 Functions

Exercise solution: `solutions-basic-function`

Refactor the solution code for Q1, Q3, Q5 from the previous exercise as functions. Obtain the required input from the user and pass these inputs as arguments to the functions and then print out the return result from the function.

2.4 List / String / Tuple

Exercise solution: `solutions-list`

1. Write a function that accepts a list of strings and counts the number of strings where the string length is 2 or larger, and the first and last character in that string are identical.

2. Write a function that returns a list of strings whose lengths are longer than a value n from an original list of strings. The value of n is provided by the user.

3. Write a function that accepts a list of random numbers and removes the even numbers from that list and prints its content.

4. Write a function that accepts two lists and returns a list which contains all the common elements from the two lists. This will be an empty list if the two lists do not have any common elements.

For e.g. for the following two lists

a = [1, 2, 3, 4, 5, 6]

b = [5, 6, 7, 8, 9]

The function should return [5,6]

5. Write a function that accepts two lists and returns a list which contains all the different elements from the two lists. This will be an empty list if the two lists do not have any common elements.

For e.g. for the following two lists

a = [1, 2, 3, 4, 5, 6]

b = [5, 6, 7, 8, 9, 1, 2]

The function should return [3, 4, 7, 8, 9]

6. Write a function that accepts a list and determines whether the list contains only identical elements or not

For e.g.

['a', 'b', 'c'] -> false

[8,8,8] -> true

7. Consider a list that has multiple recurring numbers

[10, 20, 10, 30, 20, 20, 20, 20, 40, 40, 50, 50, 10]

Write a function that determines how many times each of these numbers occur (their frequency). The output can be in the form of a dictionary for this purpose, for e.g.

{10: 3, 20: 5, 30: 1, 40: 2, 50: 2 }

8. Consider a list of tuples as following below:

[(2, 5), (1, 2), (4, 4), (5, 3), (3, 1)]

Write a function that rearranges the tuples in the list in so that they are sorted in increasing order based on either the value of either the first or second element in each tuple (allowing the user to select which):

For e.g. sorting on the first element in the tuple

[(**1**, 2), (**2**, 5), (**3**, 1), (**4**, 4), (**5**, 3)]

For e.g. sorting on the second element in the tuple

[(3, **1**), (1, **2**), (5, **3**), (4, **4**), (2, **5**)]

3 Projects

3.1 Guess a random number

Exercise solution: `solutions-project-guess`

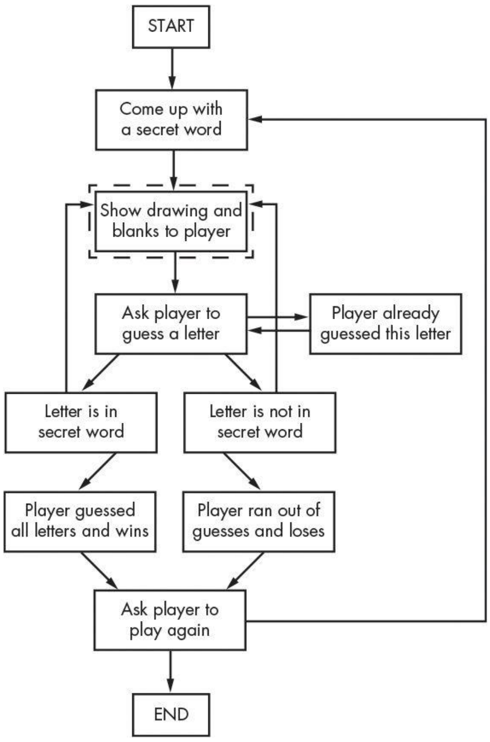
The program will select a random number between 1 and 20 and ask the user to guess it. After each guess, the computer will tell the user whether the guess higher or lower than the actual random number. The user wins if they can guess the number within a certain number of tries (e.g. 5)

To implement this, we need to import the random module in order to generate pseudo random numbers
https://www.programiz.com/python-programming/examples/random-number
https://www.w3schools.com/python/module_random.asp

3.2 Hangman

Exercise solution: `solutions-project-hangman`

The flowchart for program execution flow might look something like this:


START
Come up with a secret word
Show drawing and blanks to player
Ask player to guess a letter
Player already guessed this letter
Letter is in secret word
Letter is not in secret word
Player guessed all letters and wins
Player ran out of guesses and loses
Ask player to play again
END

To implement this, we need to import the random module in order to generate pseudo random numbers
https://www.programiz.com/python-programming/examples/random-number
https://www.w3schools.com/python/module_random.asp

Some hints:
You can hardcode the string sequences for the ASCII art for the hangman as string elements of a list
The possible words that the computer will randomly choose from can be a string elements of a list as well – you can choose some basic words to test the program
Modularize your program implementation by breaking down the functionality into discrete functions that are called from within a main loop that implements the core game logic. As an example:

- `def getRandomWord(wordList)` - Returns a random word from the list of possible words
- `def displayBoard(missedLetters, correctLetters, secretWord)` - Shows the correct hangman pic sequence, the correct letters guessed in the word sequence and also the letters missed
- `def getGuess(alreadyGuessed)` - Gets a valid letter to guess from the user (ensuring that it is not already in the list of letters already guessed

You can also define global variables that hold the list of letters guessed, the missed letters, the random word, etc and pass that to the appropriate functions
You will need to use the appropriate string methods as part of your core program logic