Lab 01
Python Basics

A. Multiple Choice (42 points, 6 points each question)

1. In the Python statement `x = a + 5 - b`:
   `a` and `b` are ________, and `a + 5 - b` is ________.

   (a) operands, an expression (b) terms, a group
   (c) operands, an equation (d) operators, a statement

2. What is the value of the boolean expression `1.1+2.2==3.3`?

   (a) True (b) False

3. Suppose the following statements are executed:
   `a = 10; b = 20`
   What is the value of the expression `a and b`?

   (a) False (b) 10 (c) 20
   (d) True (e) 0 (f) 30

4. Which of the following operators has the lowest precedence?

   (a) `and` (b) `**` (c) `+`
   (d) `*` (e) `not` (f) `//`

5. What is the output of the `print()` function call?
   `s = 'foo'; t = 'bar'; print('barf' in 2 * (s + t))`

   (a) True (b) False

6. Suppose `s` is assigned as follows:
   `s = 'toomen'`
   All of the following expressions produce the same result except one. Which one?

   (a) `s[::-5]` (b) `s[::-1][::-5]` (c) `s[::5]`
   (d) `s[0] + s[-1]` (e) `s[::-1][-1] + s[len(s)-1]`

7. Which of the following are true of Python dictionaries:

   (a) All the keys in a dictionary must be of the same type.
   (b) A dictionary can contain any object type except another dictionary.
   (c) Items are accessed by their position in a dictionary.
   (d) Dictionaries are accessed by value.
   (e) Dictionaries are mutable.

B. Reading and Multiple selection (18 points, 6 points each question)

In the following article, you will explore the Python Enhancement Proposal 8, Python's style guide, with some code examples. Read the article and select all the answers to the questions. *How to Write Beautiful Python Code With PEP 8 (https://realpython.com/python-pep8/)*

1. Which of the following are true regarding multiple statements per line in Python:
   (a) Placing multiple statements on a single line is discouraged by PEP 8.
   (b) Only variable assignment statements may occur multiply on a single line.
   (c) Multiple statements on the same line are separated by the & character.
   (d) Specifying more than one statement on a line is typically not considered Pythonic, but may be acceptable if it enhances readability.

2. Which one of the following statements about block comments is true:
   (a) There is no way to create a multiline block comment in Python.
   (b) Block comments should always be specified using triple-quoted multiline string literals.
   (c) PEP 8 discourages that creating block comments using triple-quoted multiline string literal.
   (d) In general, block comments should be written with a `#` at the start of each line.

3. Which of the following conform to the PEP 8 recommendations for whitespace in expressions and statements:
   (a) `print(x) ; print(y)`
   (b) `d = {'foo': 100, 'bar': 200}`
   (c) `a.insert (1, 100)`
   (d) `x = a[1::2]`
   (e) `t = (100, )`

C. Programming Exercise (40 points, 10 points each question)

1. Write a function `draw_grid(m=2, n=3)` that draws a similar grid with `m` rows and `n` columns.

```
+ − − + − − + − − +
/     /     /     /
/     /     /     /
+ − − + − − + − − +
/     /     /     /
/     /     /     /
+ − − + − − + − − +
```

2. The mathematician Srinivasa Ramanujan found an infinite series that can be used to generate a numerical approximation of 1 / π:

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801}\sum_{k=0}^{\infty}\frac{(4k)!\,(1103+26390k)}{(k!)^4 396^{4k}}$$

Write a function called `estimate_pi()` that uses this formula to compute and return an estimate of π. It should use a while loop to compute terms of the summation until the last term is smaller than `1e-15` (which is Python notation for 10^{-15}). You can check the result by comparing it to `math.pi`.

3. Two words are a "reverse pair" if each is the reverse of the other. Write a function that finds all the reverse pairs in the word list using the following template:

```
def find_reverse_pair(word_list):
  reverse_pair_list = []
  #TODO
  return reverse_pair_list
```

4. Given a dictionary d and a key k, it is easy to find the corresponding value `v = d[k]`. This operation is called a **lookup**. But what if you have v and you want to find k? There is no simple syntax for **reverse lookup**. Implement a function for reverse lookup using the following template and note that there might be more than one key that maps to the value v.

```
def reverse_lookup(d, v):
  k = []
  #TODO
  return k
```