ENGG1330 Computer Programming

Tutorial 3

Objectives:

1. Write programs with conditional statements
2. Work out correct logical expressions in conditional statements

Tutorial participation (1%):

- t3_vpl_1
- Submission period: **within your OWN tutorial period**

Tutorial/take-home exercises (2%):

- Remaining problems in the worksheet
- Submission deadline: **noon, 22-FEB-2023 (Wednesday)**

t3_vpl_1. Write a program to receive 3 **non-zero integers** from the user's input. A pair of integers is picked from the 3 integers. This pair of numbers is called "magic" if their product is divisible by the remaining integer. Print the number of "magic" pairs that can be formed from these 3 integers.

Sample cases and screenshots

```
========== RESTART: /Users/cky/tutorial 3/t1.py ==========
3
4
5
0
>>>
========== RESTART: /Users/cky/tutorial 3/t1.py ==========
3
3
3
3
>>>
========== RESTART: /Users/cky/tutorial 3/t1.py ==========
1
9
9
3
>>>
```

t3_vpl_2. Recall the program in t2_vpl_5 of Tutorial 2, which converts the input in number of seconds to number of hours, minutes, and seconds. Extend this program such that the output conforms to the following rules.

a. Special case: when the input is 0, the output is "0 second = 0 second"
b. Except for the above special case, don't display the number '0' and its unit.
c. Add 's' to the unit only when the number is plural.

ATTENTION: The last output statement MUST end with a newline.

Sample cases and screenshots

```
========== RESTART: /Users/cky/tutorial 3/t2.py ==========
Please enter the number of seconds: 0
0 second = 0 second
>>>
========== RESTART: /Users/cky/tutorial 3/t2.py ==========
Please enter the number of seconds: 3661
3661 seconds = 1 hour 1 minute 1 second
>>>
========== RESTART: /Users/cky/tutorial 3/t2.py ==========
Please enter the number of seconds: 7259
7259 seconds = 2 hours 59 seconds
>>>
```

t3_vpl_3. Using if, else if and else statement, write a program to print students' grade based on the following grading criteria, assuming that the user input is an integer in the range of [0,100].

Grade	Grade boundaries
A	80
B	75
C	60
D	45
F	Otherwise

Sample cases and screenshots

Case	Input	Output
1	35	F
2	60	C
3	100	A
4	74	C

```
>>>
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_2.py ==========
35
F
>>>
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_2.py ==========
60
C
>>>
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_2.py ==========
100
A
>>>
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_2.py ==========
74
C
>>>
```

t3_vpl_4. Write a program to read 3 **positive integers** from the user. The 3 values represent the length of the three sides of a triangle. The program prints a message to tell whether the triangle is equilateral (all sides are equal), isosceles (only 2 sides are equal), scalene (all sides are unequal), or impossible (can't form a triangle). A triangle can be formed only if the sum of the length of any 2 sides is greater than the length of the 3rd side.

Sample cases and screenshots

Case	Input	Output
1	3 4 5	scalene
2	3 3 3	equilateral
3	5 5 2	isosceles
4	1 2 10	impossible

```
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_3.py ==========
3
4
5
scalene
>>>
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_3.py ==========
3
3
3
equilateral
>>>
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_3.py ==========
5
5
2
isosceles
>>>
========== RESTART: /Users/csvlee/Documents/1330/lab/tut3/t3_vpl_3.py ==========
1
2
10
impossible
>>>
```

t3_vpl_5. Write a program to read a sequence of 5 numbers from the user. If the sequence is strictly increasing, i.e., $N_{n+1} > N_n$ for all n, print "profit!!!". Otherwise, print the total sum of the sequence up to 1 decimal place followed by a comment:

a. If the first number is greater than the last number, the comment is "bad".
b. If the first number is smaller than the last number, the comment is "good".
c. If the first number is equal to the last number and more than half of the remaining 3 numbers are greater than the first number, the comment is "no change but good".
d. If the first number is equal to the last number and less than half of the remaining 3 numbers are greater than the first number, the comment is "no change but bad".

The sum and the comment should be joined by " - ", e.g., "10.5 - good".

Note:
Is there any missing case that cannot be handled by the rules listed in the question? What if the condition for "comment d" is changed to "If the first number is equal to the last number and more than half of the remaining 3 numbers are less than the first number"?

Sample cases and screenshots

```
========== RESTART: /Users/cky/tutorial 3/t5.py ==========
1
2
3
4
5
profit!!!
>>>
========== RESTART: /Users/cky/tutorial 3/t5.py ==========
1
2
3
4
1
11.0 - no change but good
>>>
========== RESTART: /Users/cky/tutorial 3/t5.py ==========
1.6
-5
1.3
-100
1.8
-100.3 - good
>>>
========== RESTART: /Users/cky/tutorial 3/t5.py ==========
1.6
-5
1.3
-100
1.6
-100.5 - no change but bad
>>>
```