LECTURE 3.3. PRACTICE QUESTIONS

Q1. Prime numbers are natural numbers greater than 1 that are only divisible by 1 and themselves. The conditions in the two blocks of code below can (surprisingly) be used to check for prime numbers between 1-100. Note:*The two blocks of code are independent of each other.* Write **True** or **False** in the cells below if the corresponding line **executes** for the given inputs.

```
 1  prime     = True
 2
 3  if number <= 1:
 4      prime = False
 5
 6  if number > 2 and number % 2 == 0:
 7      prime = False
 8
 9  if number > 3 and number % 3 == 0:
10      prime = False
11
12  if number > 4 and number % 4 == 0:
13      prime = False
14
15  if number > 5 and number % 5 == 0:
16      prime = False
17
18  if number > 6 and number % 6 == 0:
19      prime = False
20
21  if number > 7 and number % 7 == 0:
22      prime = False
```

Line #	**number = 4**	**number = 5**	**number = 6**
1	True	True	True
3	True	True	True
4	False	False	False
6	True	True	True
7	True	False	True
9	True	True	True
10	False	False	True
12	True	True	True
13	False	False	False
15	True	True	True
16	False	False	False
18	True	True	True
19	False	False	False
21	True	True	True
22	False	False	False
Prime:	False	True	False

```
 1  if number <= 1:
 2      prime = False
 3
 4  elif number > 2 and number % 2 == 0:
 5      prime = False
 6
 7  elif number > 3 and number % 3 == 0:
 8      prime = False
 9
10  elif number > 4 and number % 4 == 0:
11      prime = False
12
13  elif number > 5 and number % 5 == 0:
14      prime = False
15
16  elif number > 6 and number % 6 == 0:
17      prime = False
18
19  elif number > 7 and number % 7 == 0:
20      prime = False
21
22  else:
23      prime = True
```

Line #	**number = 4**	**number = 5**	**number = 6**
1	True	True	True
2	False	False	False
4	True	True	True
5	True	False	True
7	False	True	False
8	False	False	False
10	False	True	False
11	False	False	False
13	False	True	False
14	False	False	False
16	False	True	False
17	False	False	False
19	False	True	False
20	False	False	False
22	False	True	False
23	False	True	False
Prime:	False	True	False

Q2.
A. (A or B) and (not A or not B) == A or B and not A or not B **TRUE** / **FALSE**
B. A or B and not A or not B:

Precedence	Left Operand	Operator	Right Operand
1	-	not	A
2	-	not	B
3	B	and	not A
4	A	or	B and not A
5	A or B and not A	or	not B

Q3. Given the day of the week and time (hours and minutes in military/24-hr time), set office hours to True or False.

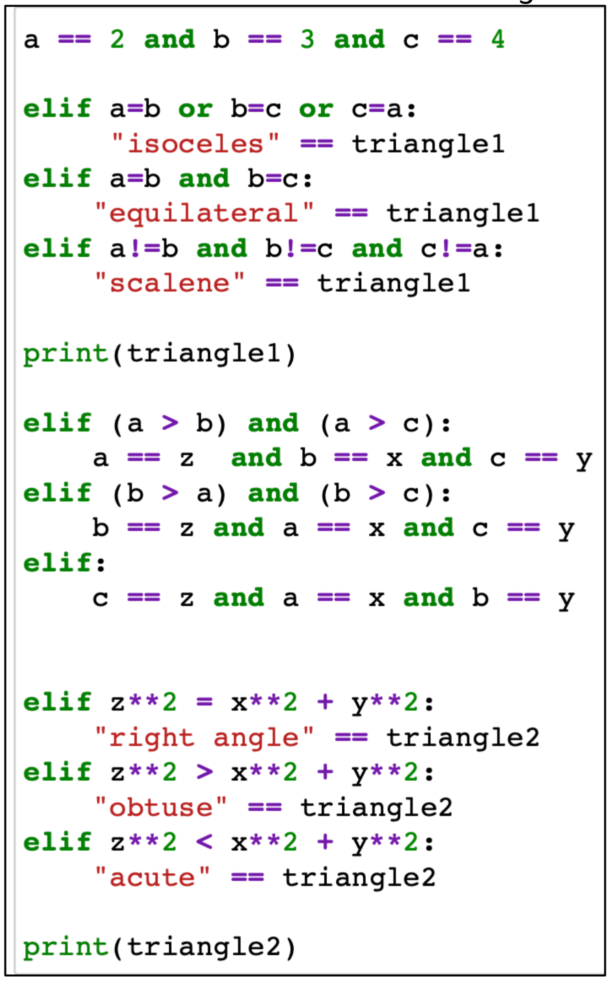
```
day = “Monday”
hours = 13
mins = 0

‘’’
You can also use nested ifs
With this solution, you have to be very careful about the parentheses
‘’’

if (day == “Monday”) and (hours >= 1 and mins >= 30) and (hours <= 16
and mins <= 30):
     office_hours = True
elif (day == “Thursday”) and (hours >= 9 and mins >= 30) and (hours <=
12 and mins <= 30):
     office_hours = True
else:
     office_hours = False
```

Q4. Given three sides of a triangle: a, b and c,

a. Print if the triangle is equilateral (all sides equal), or isosceles (two sides equal) or scalene (no sides equal).

b. Determine which side is the longest and call this side “z”, and call the other 2 sides “x” and “y”.
Print if the triangle is right ($z^2 = x^2 + y^2$), or obtuse ($z^2 > x^2 + y^2$) or acute ($z^2 < x^2 + y^2$).

Identify (underline) and fix all the syntactic and logical errors with the given code and re-write the correct code below:



```
a == 2 and b == 3 and c == 4

elif a=b or b=c or c=a:
     "isoceles" == triangle1
elif a=b and b=c:
    "equilateral" == triangle1
elif a!=b and b!=c and c!=a:
    "scalene" == triangle1

print(triangle1)

elif (a > b) and (a > c):
    a == z  and b == x and c == y
elif (b > a) and (b > c):
    b == z and a == x and c == y
elif:
    c == z and a == x and b == y


elif z**2 = x**2 + y**2:
    "right angle" == triangle2
elif z**2 > x**2 + y**2:
    "obtuse" == triangle2
elif z**2 < x**2 + y**2:
    "acute" == triangle2

print(triangle2)
```

```
a = 2
b = 3
c = 4

# equilateral condition must move up,
# if you want to keep the conditions
# from the sample unchanged

if a == b and b == c:
     triangle1 = “equilateral”
elif a==b or b==c or c==a:
     triangle1 = “isosceles”
elif a!=b and b!=a and c!=a:
     triangle1 = “scalene”

print(triangle1)

if (a > b) and (a > c):
     z = a
     x = b
     y = c
elif (b > a) and (b > c):
     z = b
     x = a
     y = c
else:
     z = c
     x = a
```

```
	y = b

if z**2 == x**2 + y**2:
	triangle2 = “right angle”
elif z**2 > x**2 + y**2:
	triangle2 = “obtuse”
else:
	triangle2 = “acute”

print(triangle2)
```