**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.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 |

A picture containing graphical user interface

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **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 |

A picture containing table

Description automatically generated

**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

Text

Description automatically generated**Q4.**  Given three sides of a triangle: a, b and c,

* 1. Print if the triangle is equilateral (all sides equal), or isosceles (two sides equal) or scalene (no sides equal).
  2. 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 (z2 = x2 + y2), or obtuse (z2 > x2 + y2) or acute (z2 < x2 + y2).

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

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)