Name:_____ **Partner:** _____

Python Activity 45: Trees

Learning Objectives
Students will be able to:
Content:
- Define a **tree**
- Identify the **root** and **leaf** of a tree
- Explain the meaning of a tree's `__slots__`
- Explain how a tree data structure represents a 20 Questions game

Process:
- Write code that adds nodes to a tree
- Write code that iterates through the tree's values.

Prior Knowledge
- Python concepts from Activities 1-19, Linked Lists, Recursion

Conceptual Model:
Twenty Questions is a game in which a "Knower" thinks of a noun, and the "Guesser(s)" have to guess the word the Knower is thinking of by asking fewer than 20 yes/no questions. Here is a sample dialogue of one such instance of the game:

Knower: Let's play Twenty Questions. I have a noun in mind.
Guesser: Is it alive?
Knower: No.
Guesser: Is it food?
Knower: Yes.
Guesser: Is it sweet?
Knower: No.
Guesser: Is it a pretzel?
Knower: Yes, you win!

a. How many questions does the Guesser ask? _____

b. What is the Guesser's first question? _____

c. What is the Guesser's last question? _____

d. How is a question distinguishable from a guess?

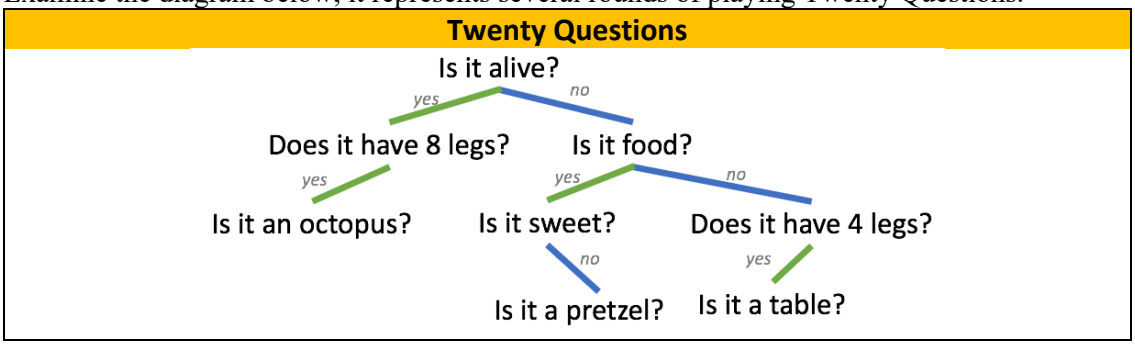
_____

e. How many guesses and how many questions are asked in this single game?

Guesses: _____ Questions: _____

Critical Thinking Questions:

1. Examine the diagram below, it represents several rounds of playing Twenty Questions.



   a. How many possible answers does each question have? _____

   b. On what side of the questions do "yes" responses appear? _____

   c. If this diagram represents 3 games of 20 Questions, what is the first question asked?

   _____

   d. There are three final guesses represented in the diagram, what are they?

   _____

   e. What physical object does the data structure in the diagram resemble? (This could be its own game of Twenty Questions!)

   _____

FYI: ***Trees*** are data structures that simulate a hierarchical tree structure, represented as a set of linked Tree nodes.

FYI: A ***leaf*** is a node of the tree that has no children. A ***root*** is the top node of the tree that points to other nodes (children), but none of these ***child*** nodes point to the root.

   f. There are three leafs in the above Twenty Questions diagram. Which questions are they?

   _____

2. The following code creates the topmost sub-tree of the Twenty Questions tree diagram:

```
t2      = Tree('Does it have 8 legs?')
t3      = Tree('Is it food?')
mytree  = Tree('Is it alive?', t2, t3)
```

   a. What does the first parameter of a new ***Tree*** instance represent?

   _____

   b. What does the second parameter of a new ***Tree*** instance represent?

c. What does the third parameter of a new ***Tree*** instance represent?

d. Write a line of code to add `'octopus'` to the correct location in the Tree, where in the sample code would you need to place it?

3. On the left is sample code, on the right is its output when executed:

```
print(mytree.value)             'Is it alive?'
print(mytree.left.value)        'Does it have 8 legs?'
print(mytree.left.left.value)   'Is it an octopus?'
print(mytree.left.right)        None
```

a. What would happen if we replaced the first line with:
`print(mytree.right.value)` ?

b. Assuming we implemented the diagram from question 1, and `mytree.value == 'Is it alive?'` what would the following line output?
`print(mytree.right.right.left.value)`

c. How does `Tree.left` differ from what's stored in `Tree.right` for our 20 Questions game?

d. Why does `print(mytree.left.right)` output `None`?

e. What might the following line refer to (according to the diagram)?:
`mytree.right.right.right`

f. Write a method, `isLeaf`, that takes in a `Tree` as a parameter and determines if it is a ***leaf***.

4. Examine the following example code:

```
    def mystery(self):
         if not self.right:
             return self
    else:
             return self.right.mystery()
```

b. What does the following line do?: `if not self.right`

_____

c. For this recursive method, what is the base case / stopping condition?

_____

d. For this recursive method, how is the longer journey broken down/shortened?

_____

e. For 20 Questions, what will this `mystery` method return?

_____

Application Questions: Use the Python Interpreter to check your work

1. Write a recursive method of `Tree` that returns the left most leaf of *any* `Tree.` In our 20 Questions example, that would be the "Is it an octopus?" node.

```
def leftmost(self):
```

_____

_____

_____

_____

_____

_____

_____

_____

_____

2. Write the `__str__(self)` method for our Tree class so that it prints the values of all the child nodes of *any* Tree, not only the Tree's values (*Hint:* `LinkedList.__str__` is similar):

```
def __str__(self):
```

_____

_____

_____

_____

_____

3. Write a recursive method of `Tree` that returns True if the given value, `v`, exists as a value within an unsorted `Tree, False` if not contained in the `Tree.`

```
def contains(self, v):
```

_____

_____

_____

_____

_____

_____

_____

_____

_____