Tutorial 1 – Breadth First

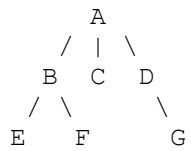
Question 1 — Basic Understanding

What data structure is primarily used in Breadth-First Traversal?

A. Stack
B. Queue
C. Array
D. Linked List

Question 2 — Trace the Output

Consider the following tree:



Using Breadth-First Traversal, what is the order of nodes printed?

Question 3 — Queue Simulation

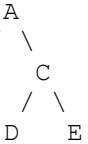
Given this BFS code:

```
Queue<Person> todo = new ArrayDeque<Person>();
todo.offer(root);

while (!todo.isEmpty()) {
    Person p = todo.poll();

    for (Person child : p.getChildren()) {
        todo.offer(child);
    }
}
```

If the tree is:



Show the contents of the queue after each iteration of the loop.

Queue Simulation

Start

Queue: _____

Iteration 1

Poll __

Add _____

Queue: _____

Iteration 2

Poll __

Queue: _____

Iteration 3

Poll __

Add _____

Queue: _____

Iteration 4

Poll __

Queue: _____

Iteration 5

Poll __

Queue: _____

Question 4 — Code Completion

Complete the missing lines in the BFS method.

```
public void printAll(Person root){

    if (root == null){
        return;
    }

    Queue<Person> todo = ____________;

    ____________;

    while (!todo.isEmpty()){

        Person p = ____________;

        UI.println(p.getName());

        for (Person child : p.getChildren()){
            ____________;
        }
    }
}
```

Question 5 — Explain the Algorithm

Explain in your own words how Breadth-First Traversal works.

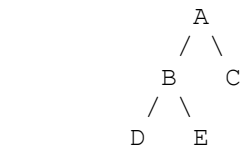
Your answer should include:

- the role of the queue
- what `offer()` does
- what `poll()` does
- why nodes are visited level by level

Answers:

Question 6 — Identify BFS vs DFS

Which traversal order represents Breadth-First Traversal for this tree?



A. A B D E C

B. A B C D E

C. D B E A C

D. A C B E D

Question 7 — Find the Error

The following BFS code contains TWO errors.

Identify and correct them.

```
Queue<Person> todo = new ArrayDeque<Person>();

while (!todo.isEmpty()) {
    Person p = todo.pop();

    for (Person child : p.getChildren()) {
        todo.push(child);
    }
}
```

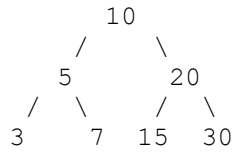
Question 8 — Time Complexity

What is the time complexity of Breadth-First Traversal on a tree with `n` nodes?

Explain why.

Question 9 — Dry Run

Given the tree:



Perform a Breadth-First Traversal manually and list:

1. The node visited at each step
2. The queue contents after each step

BFS Traversal

Start

Queue: _____

Visit __

Add _____

Queue: _____

Visit __

Add _____

Queue: _____

Visit __

Add _____

Queue: _____

Visit __

Queue: _____

Visit __

Queue: _____

Visit __

Queue: _____

Visit __

Queue: _____

Final BFS Order

_____

Question 10

Write a Java method that performs Breadth-First Traversal on a tree of `Person` objects and prints all names.

You may assume:

```
class Person {
    String getName();
    List<Person> getChildren();
}
```

Question 11 — BFS Search

Modify the BFS algorithm so that it returns `true` if a person with a given name exists in the tree.

```
public boolean contains(Person root, String target){




}
```

Question 12 — Count Nodes

Write a BFS method that counts how many nodes are in the tree.

Question 13 — Level Counting

Using BFS, write **a method** to determine how many levels exist in a tree.

Example:

```
Level 0: A
Level 1: B C
Level 2: D E F
```

Total levels = 3

Question 14

Why is a queue necessary for Breadth-First Traversal?

What would happen if a stack were used instead?

Question 15 — Compare BFS and DFS

Fill in the table:

Feature	BFS	DFS
Main Data Structure		
Traversal Style		

Finds shortest path in unweighted graph?		
Uses recursion commonly?		