Ubiquitous features of functional programming languages are first class functions and higher order functions. Can you have a useful functional PL without first class and higher-order functions. Defend your answer.

Every useful PL (at least every one with which I am familiar) that has first class functions also has garbage collection. What aspects of first class functions make automatic garbage collection a de facto requirement? (Explain)

Part A
Suppose that a language did not have short-circuit boolean evaluation. Write a function that would result in an error in such a language but would not result in an error in a language that did have short-circuit evaluation.
Part B
Rewrite the function from part A so that the function would not result in an error (in a language that does not have short-circuit boolean evaluation

In Kotlin, write a set of classes to implement a ternary (max of three children per parent) tree. The class set should include a “wrapper” class called Tree and a second class called Node. The set may include other classes as needed for your implementation. The Tree class is where an Add method should be written. Write a stub for the Add (the function name, parameters and return value) but do not write a method body for add. DO WRITE a recursive toString method that will print the entire ternary tree in depth first order.

This is a horrible recursive implementation of a fibonacci calculator. (0L means make a number whose value is 0 and whose type is Long) Why?

```
fun fibonacci(n: Long):Long = if (n==0L) { 0 }
else { if (n==1L) { 1 } else { fibonacci(n - 1)
+ fibonacci(n - 2) }}
```