Part 1: Warm-up & Code Analysis (10 points)

Before you write any new code, analyze the following C++ program. Predict its exact output and write down your reasoning. This exercise is crucial for understanding how pointers affect variables inside and outside of functions.

```cpp
#include <iostream>

void processData(int val, int* ptr) {
    val += 10;
    *ptr += 10;
}

int main() {
    int alpha = 20;
    int beta = 20;

    std::cout << "Before function call:" << std::endl;
    std::cout << "alpha = " << alpha << std::endl;
    std::cout << "beta = " << beta << std::endl;

    processData(alpha, &beta);

    std::cout << "\nAfter function call:" << std::endl;
    std::cout << "alpha = " << alpha << std::endl;
    std::cout << "beta = " << beta << std::endl;

    return 0;
}
```

Your Task:

On a separate document (e.g., a Google Doc or a text file), answer the following questions:

1. The processData function takes two parameters: int val and int* ptr. Which of these is passed by value, and which is passed by reference?

In the processData function val is passed by the value, and int* ptr is passed by the pointer reference.

2. What will be the final value of alpha printed by the program? Why doesn't it change to 30?

Alpha is = 20. It does not change because the function is only modified by a local copy of the data, not the original variable in main.

3. What will be the final value of beta printed by the program? Why does its value change?

   Beta is = 20. It changes because the function is used by a pointer to have access and to modify beta’s actual memory address.