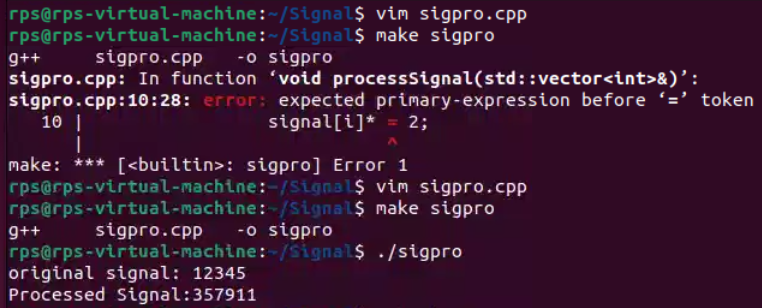
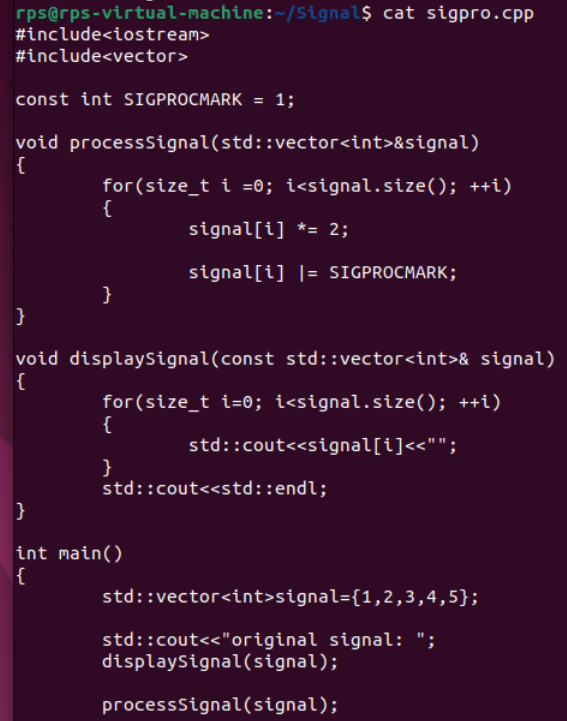
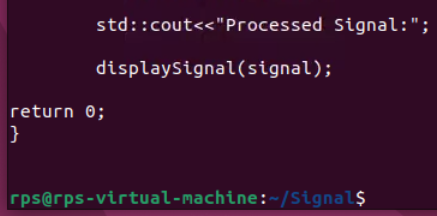
TASK 1:







**TASK 2:**

**Task is to implement a function to process a signal and mark the processed elements using a specific marker. The signal is represented as a vector of integers. You need to:**

**Define a marker value (SIGPROCMARK) to mark the processed signal elements.**

**Implement a function processSignal that processes each element of the signal by doubling its value and then marking it with SIGPROCMARK.**

**Implement a function displaySignal to print the signal values to the console.**

**Demonstrate the usage of these functions in a main function with an example signal.**

**Requirements:**

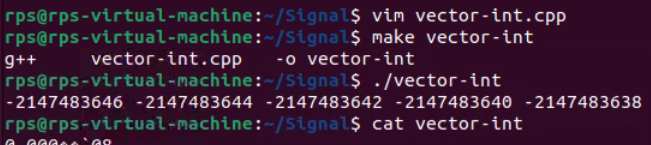
**The marker value should be defined as a constant.**

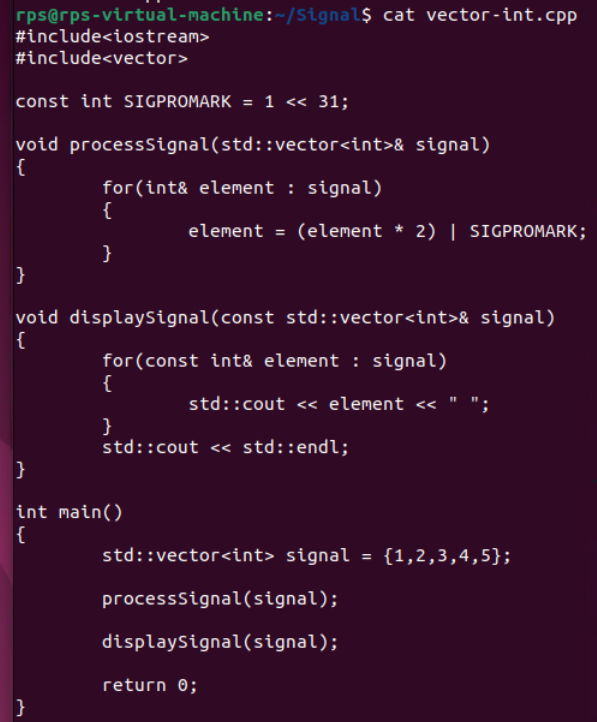
**The processSignal function should use bitwise operations to mark the processed elements.**

**The displaySignal function should print the signal values separated by spaces.**

**Input:**

**An example signal represented as a vector of integers, e.g., {1, 2, 3, 4, 5}.**

****

****

**TASK 3:**

**Signal Processing with Threshold Marking**

**You are tasked with extending the signal processing project to include a threshold marking mechanism. Your goal is to:**

**Define a marker value (SIGPROCMARK) to mark the processed signal elements.**

**Implement a function processSignalWithThreshold that processes each element of the signal by doubling its value only if it is greater than a given threshold, and then marking it with SIGPROCMARK.**

**Implement a function displaySignal to print the signal values to the console.**

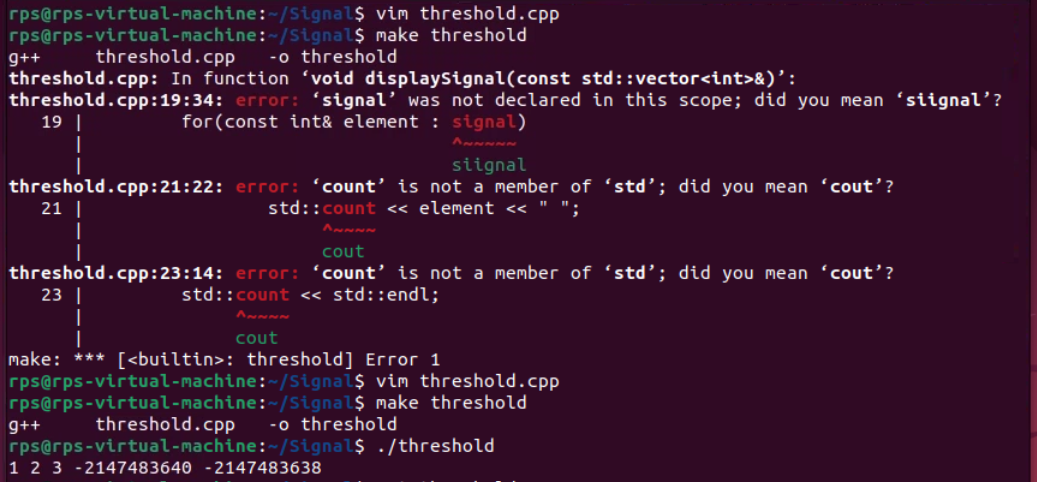
**Demonstrate the usage of these functions in a main function with an example signal and a threshold value.**

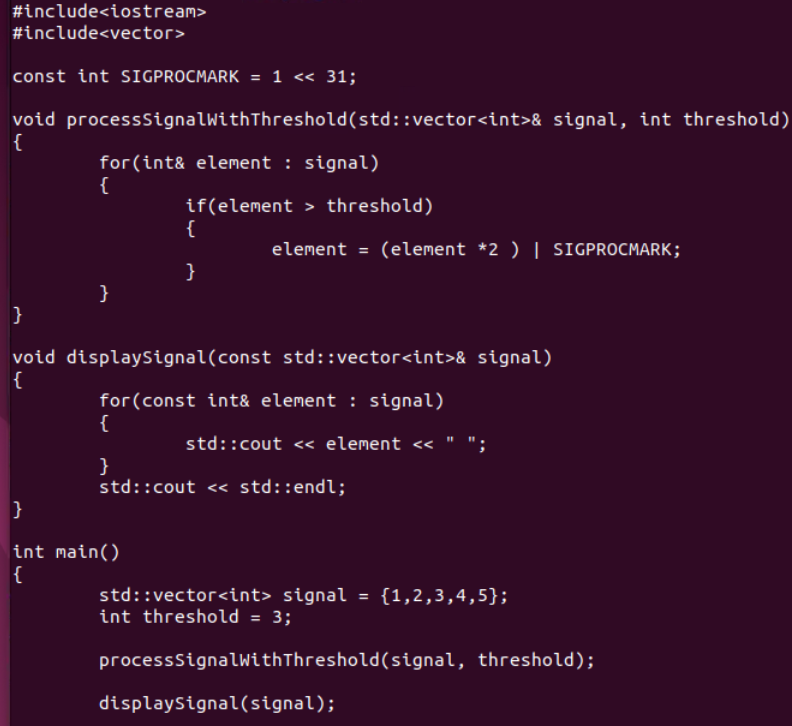
**Requirements:**

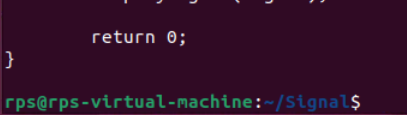
**The marker value should be defined as a constant.**

**The processSignalWithThreshold function should double the value of each element that exceeds the threshold and use bitwise operations to mark the processed elements.**

**The displaySignal function should print the signal values separated by spaces.**

****

****

****

**TASK 4:**

**Develop a C++ application that demonstrates effective signal handling using SIGALRM, SIGDEFAULT, and SIG\_IGN. The program should:**

**Set up a timer using alarm() to generate a SIGALRM signal after a specified interval.**

**Define a signal handler function to process the SIGALRM signal and perform specific actions, such as printing a message, updating a counter, or triggering an event.**

**Implement logic to handle other signals (e.g., SIGINT, SIGTERM) using SIGDEFAULT or SIG\_IGN as appropriate.**

**Explore the behavior of the application under different signal combinations and handling strategies.**

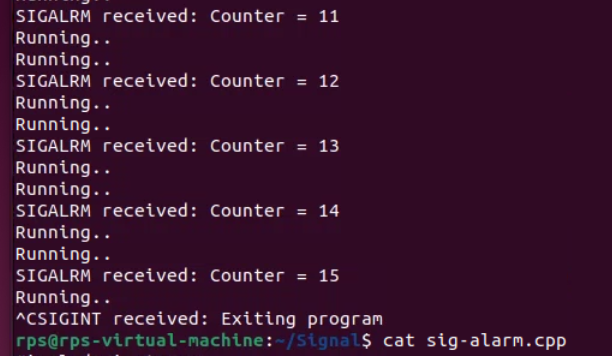
**Additional Considerations:**

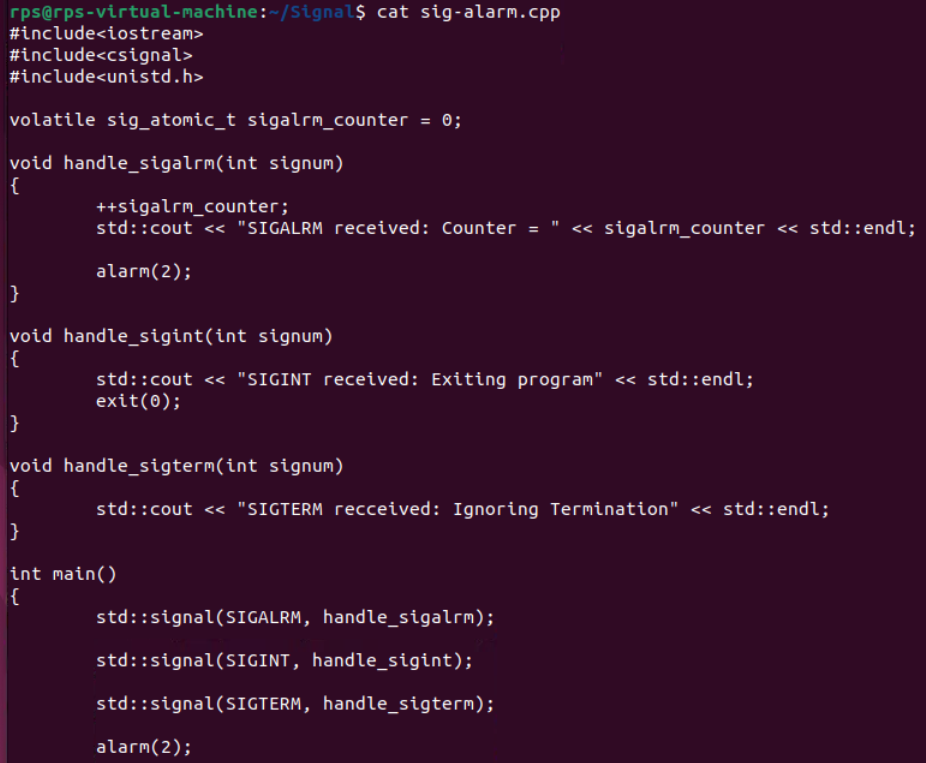
**Consider the impact of signal handling on program execution and potential race conditions.**

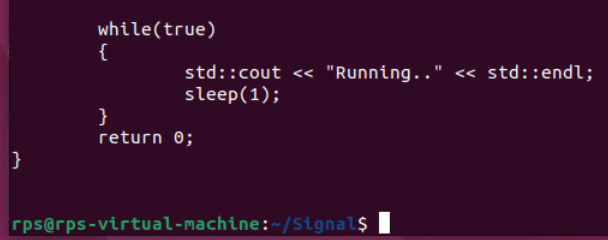
**Investigate the use of sigaction for more advanced signal handling capabilities.**

**Explore the application of signal handling in real-world scenarios, such as timeouts, asynchronous events, and error handling.**

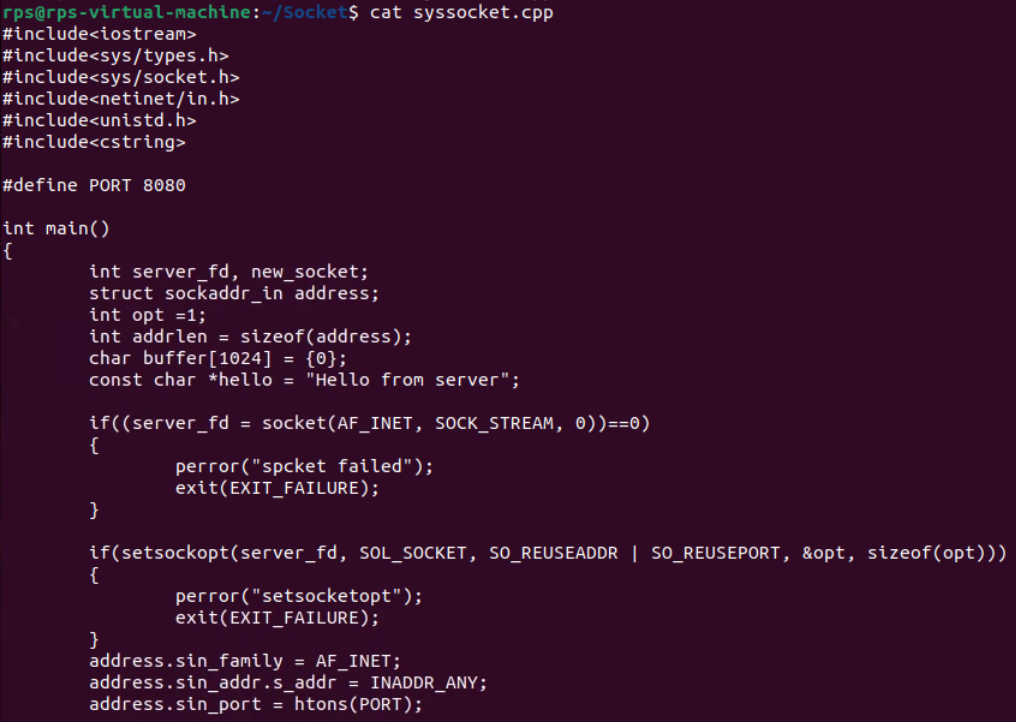
****

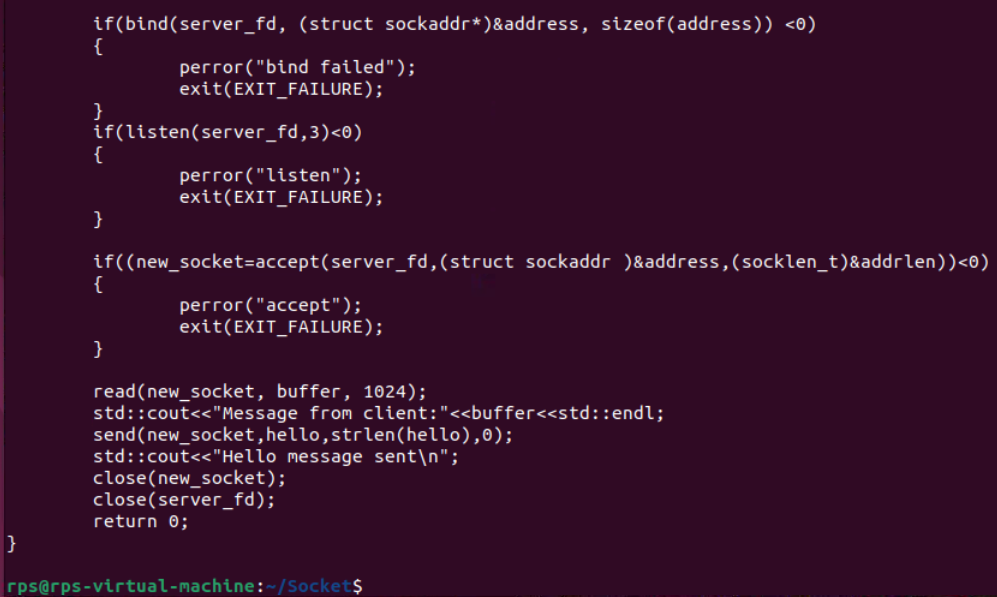
****

****

****

**TASK 5:**

****

****