```
voldemort@MSI:/mnt/c/Users/rkala/OneDrive/Desktop/Winter2025/Operating-Systems/Labs/Lab06$ gcc -o Lab06.o Lab06.c -lpthread
voldemort@MSI:/mnt/c/Users/rkala/OneDrive/Desktop/Winter2025/Operating-Systems/Labs/Lab06$ ./Lab06.o
Thread 1 Blocked
Thread 2 Blocked
Thread 3 Blocked
Thread 4 Blocked
Thread 5 Blocked
Thread 6 Blocked
Thread 7 Blocked
Thread 8 Blocked
Thread 9 Blocked
Thread 10 Blocked
Thread 10 Awakened
Thread 8 Awakened
Thread 9 Awakened
Thread 2 Awakened
Thread 6 Awakened
Thread 1 Awakened
Thread 4 Awakened
Thread 7 Awakened
Thread 3 Awakened
Thread 5 Awakened
```

Describe the benefits and drawbacks of fork versus thread.

Forking creates independent processes with their own memory space, providing isolation and reducing the risk of one process affecting another. However, it can be computationally expensive, as duplicating the parent process's memory can consume more resources, and communication between processes will require additional mechanisms. Forking is best suited for tasks that require full separation and independent execution.

Threads are lightweight and share the same memory space, which allows for faster communication and efficient resource usage. However, they come with the risk of memory leaks and race conditions if not carefully managed, since improper synchronization or memory handling can affect the entire process. Threads are ideal for tasks that require parallel execution within the same program but require careful handling to avoid errors.