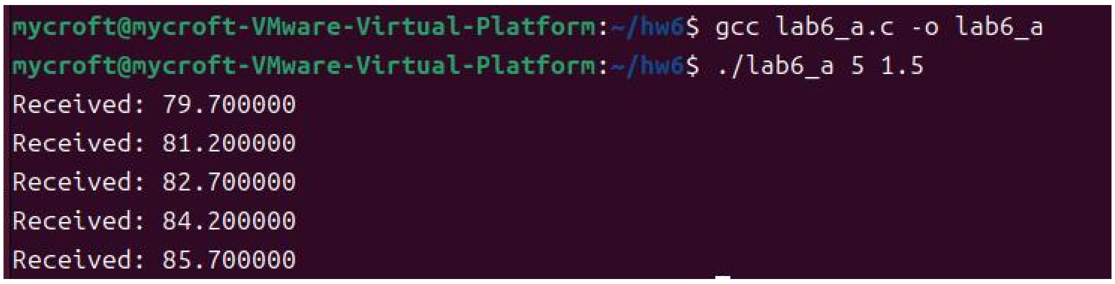
Lab6_a:

In Part 1, I reimplemented Assignment 5B using a TCP/IP socket instead of a pipe.
The parent process acts as a TCP server: it creates a socket, binds to port 50000 on 127.0.0.1, sleeps up to 6999 ms, then listens and accepts the connection.
The child process acts as a TCP client: it repeatedly attempts connect() until the server is ready, then sends n double values using write().
The parent receives each double with read() and prints the values as they arrive.

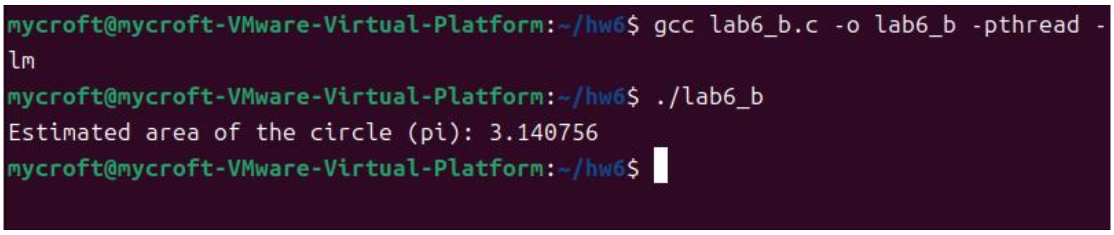
Output:



```
mycroft@mycroft-VMware-Virtual-Platform:~/hw6$ gcc lab6_a.c -o lab6_a
mycroft@mycroft-VMware-Virtual-Platform:~/hw6$ ./lab6_a 5 1.5
Received: 79.700000
Received: 81.200000
Received: 82.700000
Received: 84.200000
Received: 85.700000
```

Lab6_b:

I implemented a multi-threaded Monte-Carlo program using 4 threads. Each thread generates one million random points, checks whether each point falls inside the unit circle, and updates a shared counter protected by a semaphore. After all threads finish, the program uses the ratio of inside points to total points to estimate the value of π.



```
mycroft@mycroft-VMware-Virtual-Platform:~/hw6$ gcc lab6_b.c -o lab6_b -pthread -
lm
mycroft@mycroft-VMware-Virtual-Platform:~/hw6$ ./lab6_b
Estimated area of the circle (pi): 3.140756
mycroft@mycroft-VMware-Virtual-Platform:~/hw6$
```