CSE344 Systems Programming

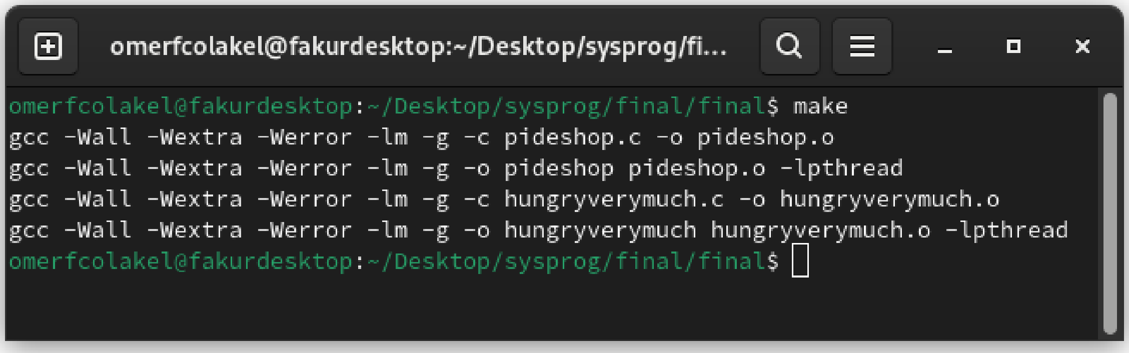
Final Report

Ömer Faruk Çolakel

200104004043

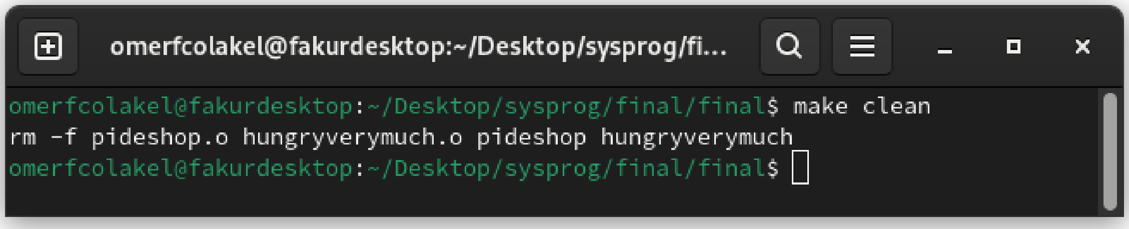
How to Compile, Run and Clean

- Compile



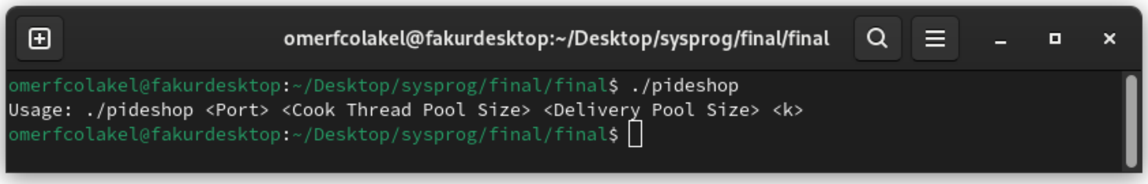
```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ make
gcc -Wall -Wextra -Werror -lm -g -c pideshop.c -o pideshop.o
gcc -Wall -Wextra -Werror -lm -g -o pideshop pideshop.o -lpthread
gcc -Wall -Wextra -Werror -lm -g -c hungryverymuch.c -o hungryverymuch.o
gcc -Wall -Wextra -Werror -lm -g -o hungryverymuch hungryverymuch.o -lpthread
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

- Clean



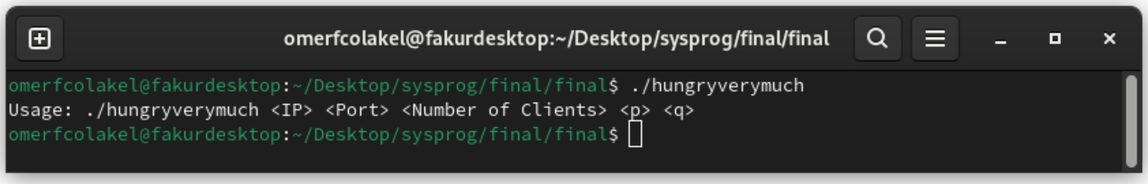
```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ make clean
rm -f pideshop.o hungryverymuch.o pideshop hungryverymuch
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

- Run (Server - PideShop)



```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./pideshop
Usage: ./pideshop <Port> <Cook Thread Pool Size> <Delivery Pool Size> <k>
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

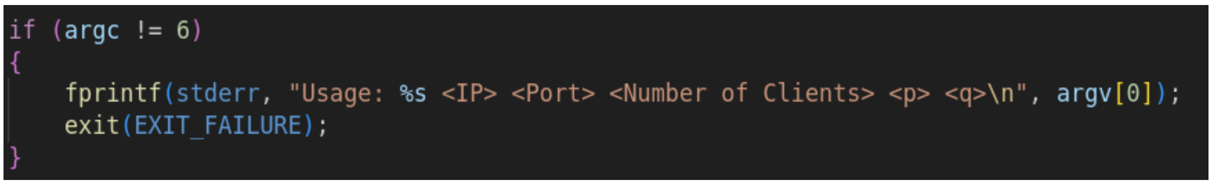
- Run (Client - HungryVeryMuch)



```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch
Usage: ./hungryverymuch <IP> <Port> <Number of Clients> <p> <q>
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

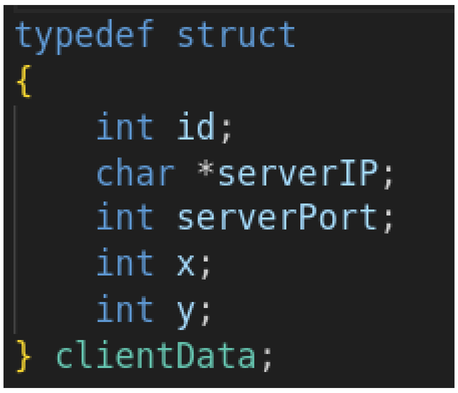
You can run the programs only when you give them the correct amount of inputs.

hungryverymuch.c



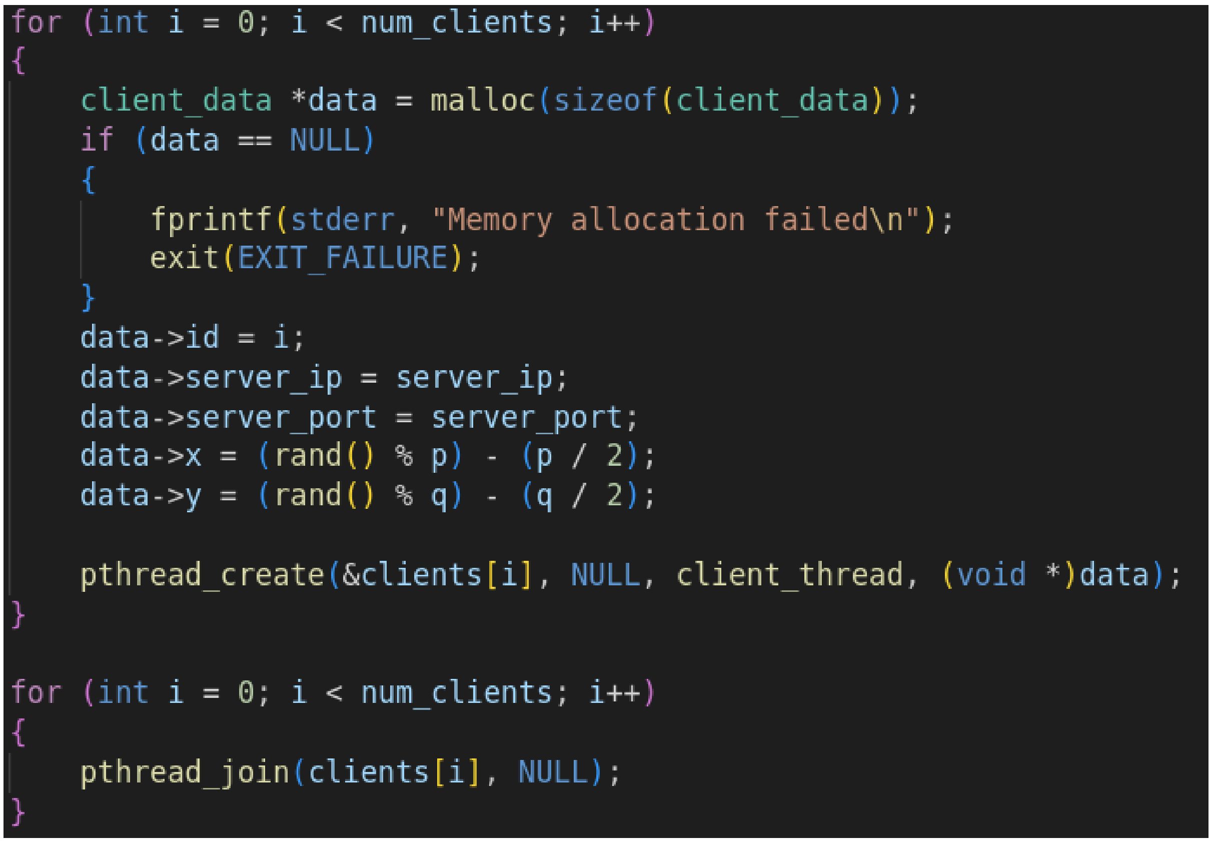
```c
if (argc != 6)
{
    fprintf(stderr, "Usage: %s <IP> <Port> <Number of Clients> <p> <q>\n", argv[0]);
    exit(EXIT_FAILURE);
}
```

The purpose of this file is to create clients for the pide shop. It takes the ip address of the server, port number, number of clients, p and q values. If there are more or less command line arguments than necessary, it prints the usage and exits. There is an interrupt signal handler called handleSigInt. It stops all the threads and sends “CANCEL” to the server. It also deallocates all the resources.



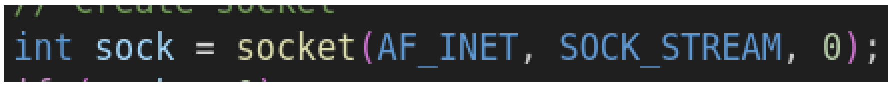
```c
typedef struct
{
    int id;
    char *serverIP;
    int serverPort;
    int x;
    int y;
} clientData;
```

The program creates numClients number of threads. So each thread represents a client. Each thread has its own clientData. ID is unique for each thread while serverIP and serverPort are the same. The x and y values are determined randomly according to the p and q values. The pide shop is assumed to be at (0,0). p and q assumed to be the x and y dimensions. Therefore; -p/2 < x < p/2 and -q/2 < y < q/2. The x and y values are determined randomly in the main before creating the thread. By using pthread_join, the program waits for all threads to finalize, to end the program. So, the program only ends when all the orders are delivered.



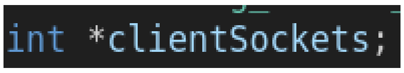
```c
for (int i = 0; i < num_clients; i++)
{
    client_data *data = malloc(sizeof(client_data));
    if (data == NULL)
    {
        fprintf(stderr, "Memory allocation failed\n");
        exit(EXIT_FAILURE);
    }
    data->id = i;
    data->server_ip = server_ip;
    data->server_port = server_port;
    data->x = (rand() % p) - (p / 2);
    data->y = (rand() % q) - (q / 2);

    pthread_create(&clients[i], NULL, client_thread, (void *)data);
}

for (int i = 0; i < num_clients; i++)
{
    pthread_join(clients[i], NULL);
}
```

In the main function, we have created clientThread threads. Each thread creates its own socket to connect to the server. If the parameters for the server are wrong, there will be no connection and threads will exit; therefore ending the program.



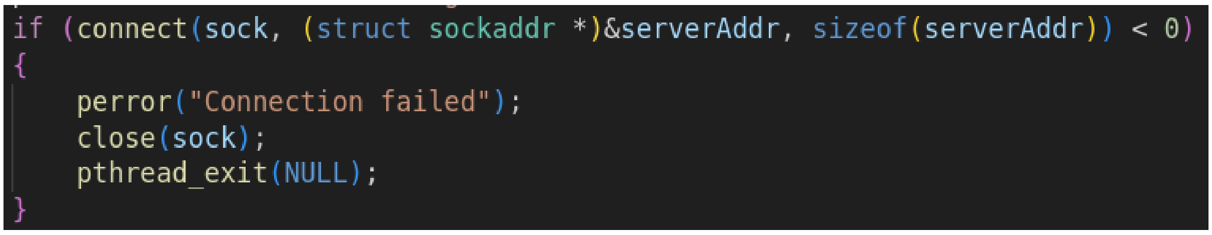
```c
int sock = socket(AF_INET, SOCK_STREAM, 0);
```

AF_INET is used to use IPV4 protocol while connecting, SOCK_STREAM specifies the type of the socket and 0 sets it to default which is TCP.



```c
int *clientSockets;
```

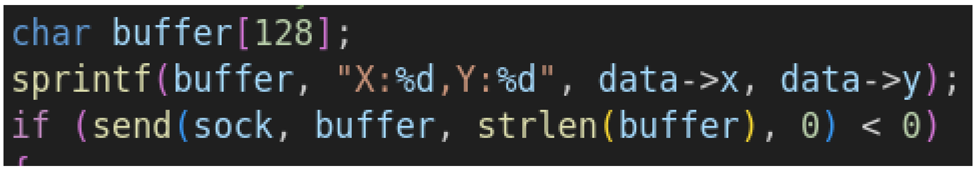
Client sockets are kept in this array. This enabled us to send a “CANCEL” message to the server. At least this was my method for order cancellation but I failed. So it is not working. You can see it being used in the signal handler.



```c
if (connect(sock, (struct sockaddr *)&serverAddr, sizeof(serverAddr)) < 0)
{
    perror("Connection failed");
    close(sock);
    pthread_exit(NULL);
}
```

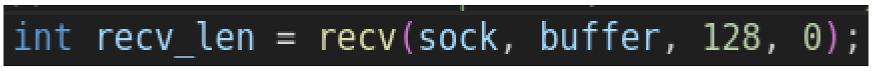
connect function is used to establish a connection to a socket, sock is the socket fd, serverAddr is the server IP address. Again if there was an error, the thread would exit.

Then the x and y values are sent to the server for it to process.



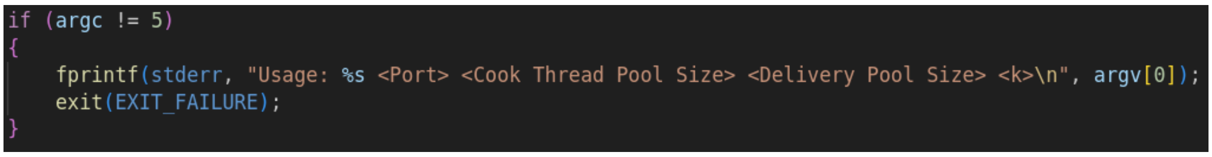
```c
char buffer[128];
sprintf(buffer, "X:%d,Y:%d", data->x, data->y);
if (send(sock, buffer, strlen(buffer), 0) < 0)
```

Finally the client waits for a response from the server, prints it into the terminal and then exits. If there is something wrong, an error message is printed. Either way, it exits the thread.



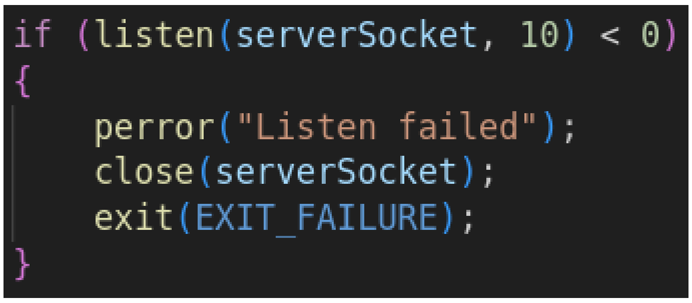
```c
int recv_len = recv(sock, buffer, 128, 0);
```

pideshop.c



```c
if (argc != 5)
{
    fprintf(stderr, "Usage: %s <Port> <Cook Thread Pool Size> <Delivery Pool Size> <k>\n", argv[0]);
    exit(EXIT_FAILURE);
}
```

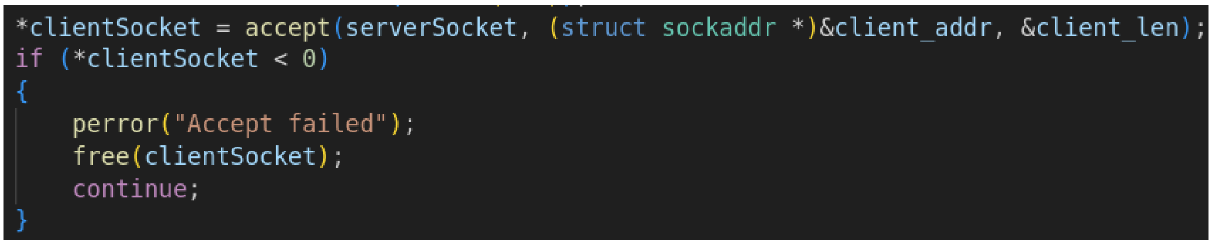
Pide shop requires port number, cook count, delivery count and k. If there are less or more than 5 arguments, it exits. Then it creates a socket like we did in the hungryverymuch. If the port number is used then the binding fails and the program exits. It also appends or creates a logger file called “serverLog.txt”.



```c
if (listen(serverSocket, 10) < 0)
{
    perror("Listen failed");
    close(serverSocket);
    exit(EXIT_FAILURE);
}
```

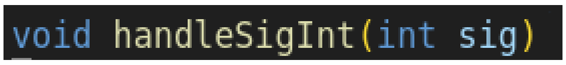
After binding, the server starts to listen from the socket. ‘10’ is the backlog parameter. If the server is slow to process connections, the queue might fill up. Every connection after 10 is refused.

After that, the server prints its ip address so the clients can use it to connect. Then in for loops, the cook and delivery threads are created. The counts of them are based on the number given as command line arguments. Every delivery thread gets its own index also. More on that later.



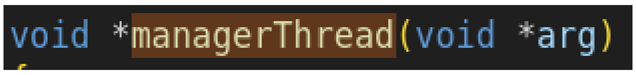
```c
*clientSocket = accept(serverSocket, (struct sockaddr *)&client_addr, &client_len);
if (*clientSocket < 0)
{
    perror("Accept failed");
    free(clientSocket);
    continue;
}
```

The accept is used to accept connections from the client and create a socket descriptor for it. After that the manager thread is created to handle the client.



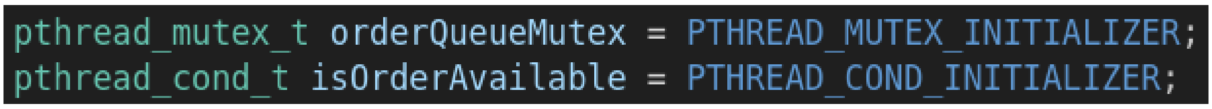
```c
void handleSigInt(int sig)
```

The handleSigInt function is designed to handle the SIGINT signal (usually triggered by pressing Ctrl+C). When this signal is received, the function first prints a termination message with the signal number. It then checks which delivery thread has delivered the most orders by iterating through the deliveredCount array, finding the maximum value. It prints out this maximum value and the index of the corresponding thread. Next, it sets a stop flag to indicate that the server should terminate, locking and unlocking a mutex to ensure thread safety. The server socket is then closed, followed by printing another termination message and logging it. Finally, it closes the log file and exits the process.



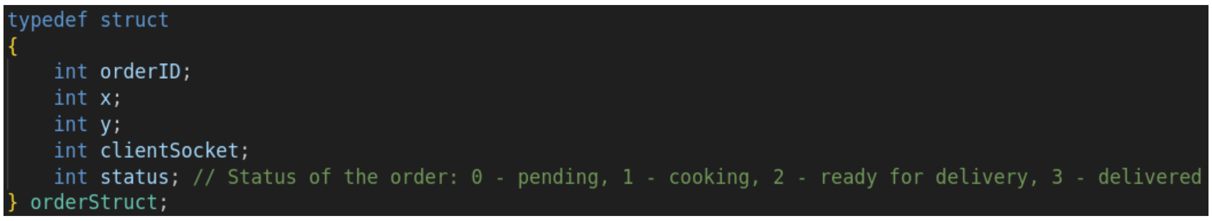
```c
void *managerThread(void *arg)
```

The managerThread function handles the communication with a client. It starts by getting the client's socket descriptor from the passed argument and freeing the memory allocated for that argument. Then, it tries to receive data from the client into a buffer. If it successfully receives data, it parses the data to get two integers, x and y. An orderStruct is then created with these values, along with a unique order ID, the client socket, and a status of 0(will be mentioned later). This order is added to a shared order queue, protected by a mutex lock to prevent race conditions. After adding the order, it signals a condition variable to indicate that a new order is available. The function logs the receipt of the order and the connection of the client, including the current client count. If no data is received, it closes the client socket and prints an error message. Finally, the thread exits. It writes some info to both terminal and logger.



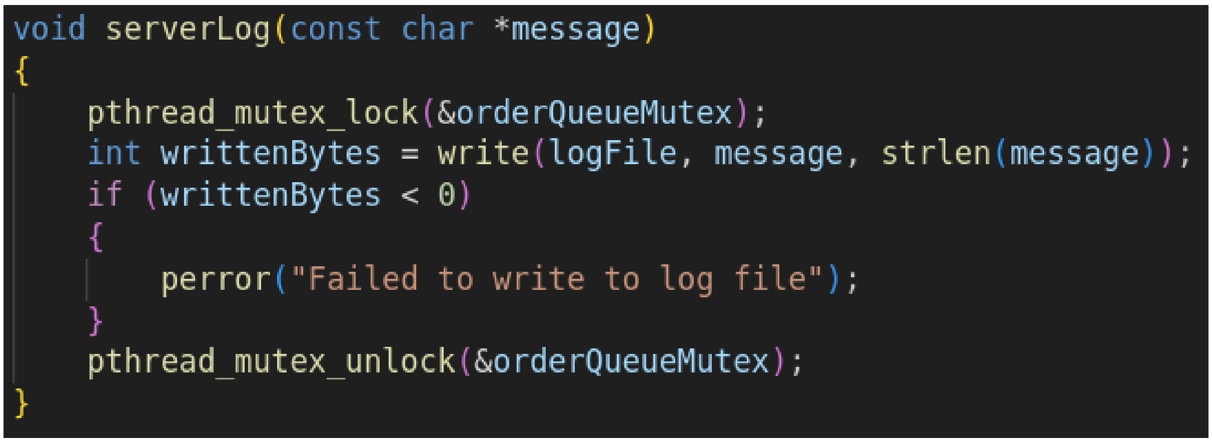
```c
pthread_mutex_t orderQueueMutex = PTHREAD_MUTEX_INITIALIZER;
pthread_cond_t isOrderAvailable = PTHREAD_COND_INITIALIZER;
```

orderQueueMutex is used to prevent two threads writing or reading from the order queue at the same time. isOrderAvailable is used to signal that an order is ready for a process. Both of them are used in the manager thread.



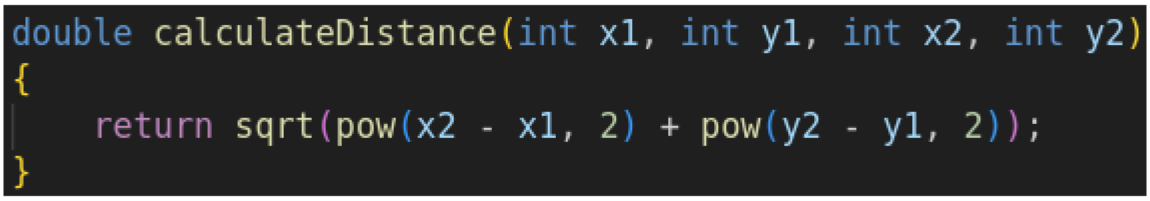
```c
typedef struct
{
    int orderID;
    int x;
    int y;
    int clientSocket;
    int status; // Status of the order: 0 - pending, 1 - cooking, 2 - ready for delivery, 3 - delivered
} orderStruct;
```

orderStruct represents an order. The status meanings are given in the image. Each order has its own unique ID. x and y are taken from the client.



```c
void serverLog(const char *message)
{
    pthread_mutex_lock(&orderQueueMutex);
    int writtenBytes = write(logFile, message, strlen(message));
    if (writtenBytes < 0)
    {
        perror("Failed to write to log file");
    }
    pthread_mutex_unlock(&orderQueueMutex);
}
```

serverLog handles writing a message to the logger. There are a lot of messages to write to the logger at the same time, therefore we used the orderQueueMutex to prevent race conditions.



```
double calculateDistance(int x1, int y1, int x2, int y2)
{
    return sqrt(pow(x2 - x1, 2) + pow(y2 - y1, 2));
}
```

calculateDistance is a very basic Euclidean distance calculator for the delivery process.

```
void *cookThread()
```

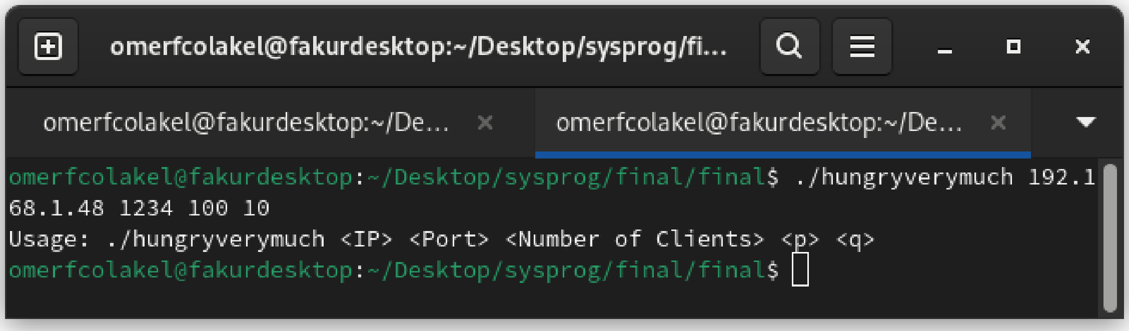
The cook thread runs continuously until the stop flag is set. It waits for orders to be ready by using condition variable isOrderAvailable and orderQueueMutex. Once an order is available, it dequeues it from the orderQueue, decreases orderCount and sets status of the order to 1. Then simulates the preparation and cooking time. Preparation time is between 1 and 5 seconds while cooking time is half of that. It decreases shovel count before cooking and increases it after cooking. And if the count of available shovels is 0, it waits for one of them to be available. The availability is signaled with isShovelAvailable and uses mutexes to change the shovels variable. After preparation and cooking, the status is set to 2 and the order is added to deliveryQueue, also increasing deliveryCount. The orderQueueMutex is used to prevent errors. Finally a message is printed to the logger and terminal andisDeliveryReady condition is signaled.

```
void *deliveryThread(void *arg)
```

The delivery thread is used to deliver the orders. Now normally every delivery should carry up to three orders but I couldn’t implement it so, they carry only one. I have talked about threadIndex earlier. The reason why I used it is that there is an array of integers that holds the number of orders that each delivery delivered. At the end of the program the best delivery is printed to the terminal alongside its order count. The threadIndex is the index of that thread in the array. This function works in a loop until the stop is set. It waits for a delivery to be ready (isDeliveryReady condition variable). As soon as there is one lock the orderQueueMutex and removes the order from the deliveryQueue. Using the calculated distance and k (speed of the delivery) calculates the time to sleep. After the delivery, it sends a message to the client to let it know that the order is delivered. Then it prints a message to both the logger and the terminal. Finally, it increases the number of orders that it delivers using the threadIndex.

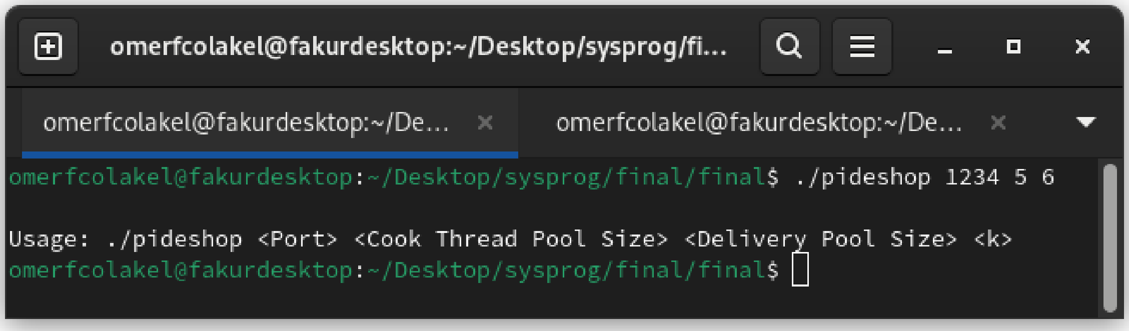
Test Cases

1. Invalid number of parameters (Client - Server)



omerfcolakel@fakurdesktop:~/Desktop/sysprog/fi...
omerfcolakel@fakurdesktop:~/De...
omerfcolakel@fakurdesktop:~/De...

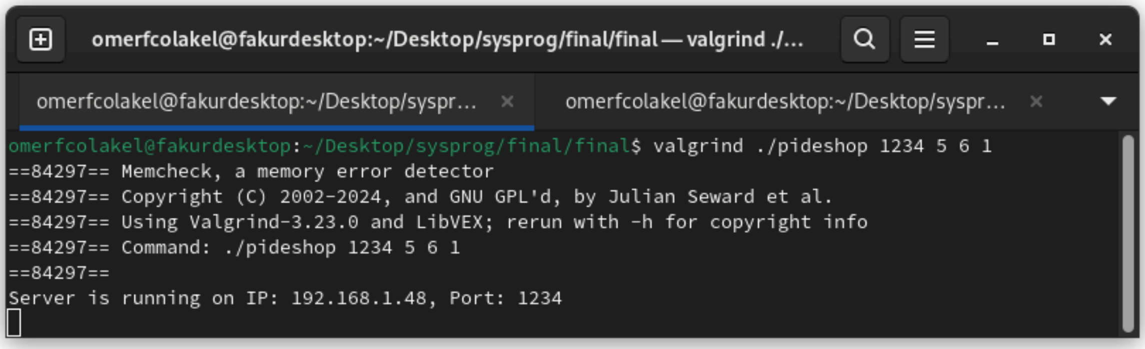
```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.1
68.1.48 1234 100 10
Usage: ./hungryverymuch <IP> <Port> <Number of Clients> <p> <q>
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```



omerfcolakel@fakurdesktop:~/Desktop/sysprog/fi...
omerfcolakel@fakurdesktop:~/De...
omerfcolakel@fakurdesktop:~/De...

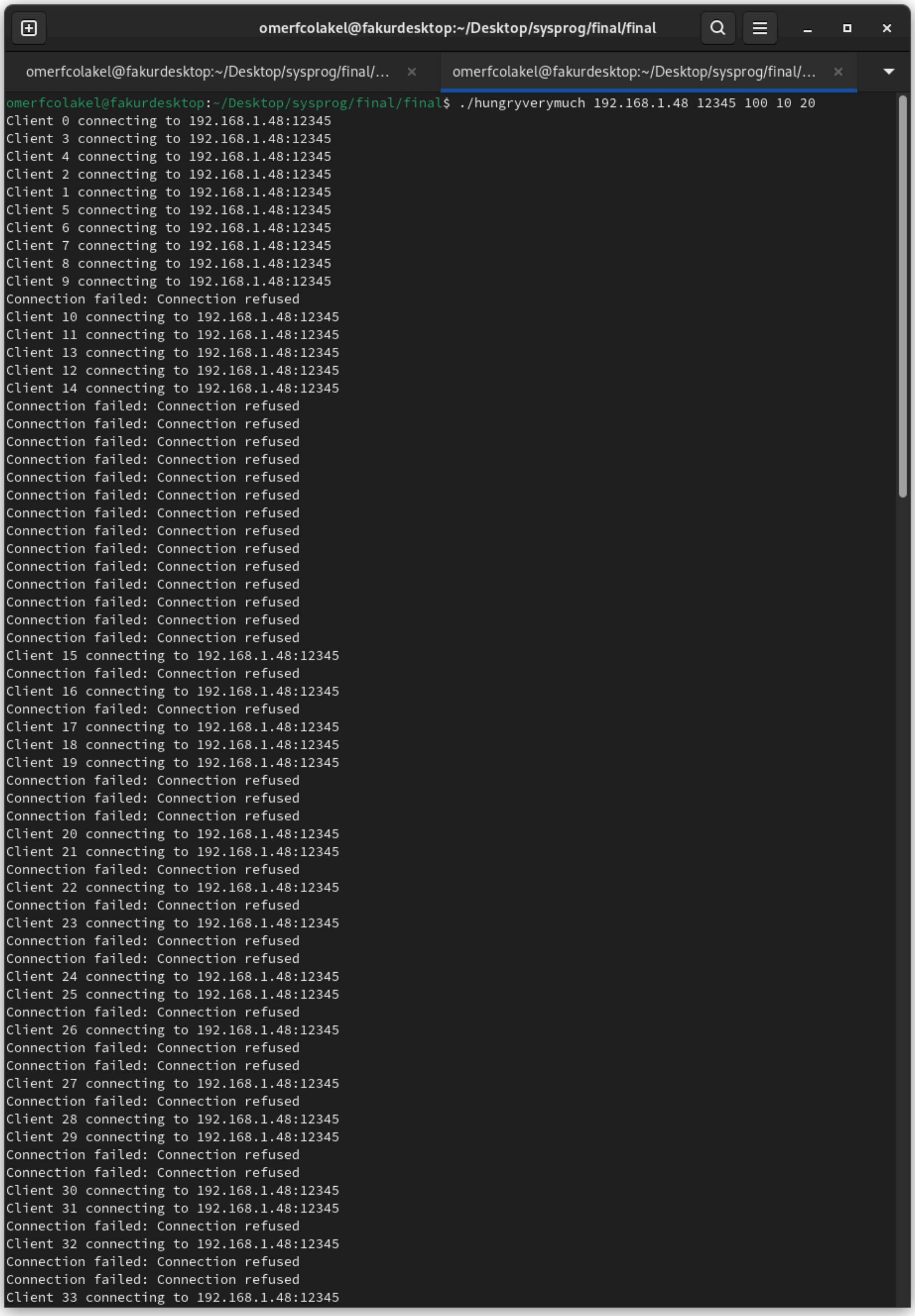
```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./pideshop 1234 5 6

Usage: ./pideshop <Port> <Cook Thread Pool Size> <Delivery Pool Size> <k>
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

2. Invalid port number or IP address for client (Client - Client)

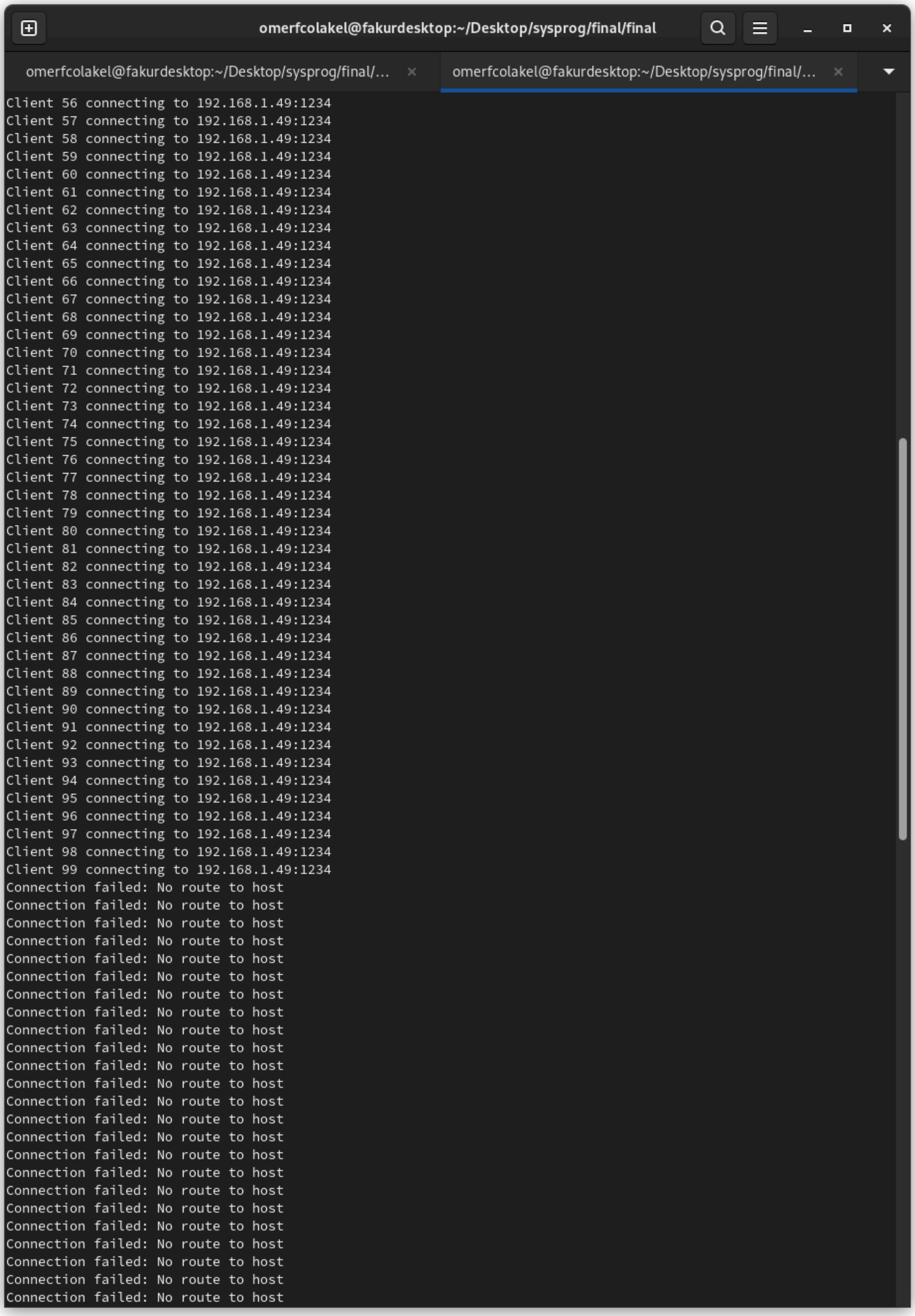


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final — valgrind ./...
omerfcolakel@fakurdesktop:~/Desktop/syspr...
omerfcolakel@fakurdesktop:~/Desktop/syspr...

```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ valgrind ./pideshop 1234 5 6 1
==84297== Memcheck, a memory error detector
==84297== Copyright (C) 2002-2024, and GNU GPL'd, by Julian Seward et al.
==84297== Using Valgrind-3.23.0 and LibVEX; rerun with -h for copyright info
==84297== Command: ./pideshop 1234 5 6 1
==84297==
Server is running on IP: 192.168.1.48, Port: 1234
```


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.168.1.48 12345 100 10 20
Client 0 connecting to 192.168.1.48:12345
Client 3 connecting to 192.168.1.48:12345
Client 4 connecting to 192.168.1.48:12345
Client 2 connecting to 192.168.1.48:12345
Client 1 connecting to 192.168.1.48:12345
Client 5 connecting to 192.168.1.48:12345
Client 6 connecting to 192.168.1.48:12345
Client 7 connecting to 192.168.1.48:12345
Client 8 connecting to 192.168.1.48:12345
Client 9 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 10 connecting to 192.168.1.48:12345
Client 11 connecting to 192.168.1.48:12345
Client 13 connecting to 192.168.1.48:12345
Client 12 connecting to 192.168.1.48:12345
Client 14 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Client 15 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 16 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 17 connecting to 192.168.1.48:12345
Client 18 connecting to 192.168.1.48:12345
Client 19 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Connection failed: Connection refused
Connection failed: Connection refused
Client 20 connecting to 192.168.1.48:12345
Client 21 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 22 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 23 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Connection failed: Connection refused
Client 24 connecting to 192.168.1.48:12345
Client 25 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 26 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Connection failed: Connection refused
Client 27 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 28 connecting to 192.168.1.48:12345
Client 29 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Connection failed: Connection refused
Client 30 connecting to 192.168.1.48:12345
Client 31 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Client 32 connecting to 192.168.1.48:12345
Connection failed: Connection refused
Connection failed: Connection refused
Client 33 connecting to 192.168.1.48:12345
```

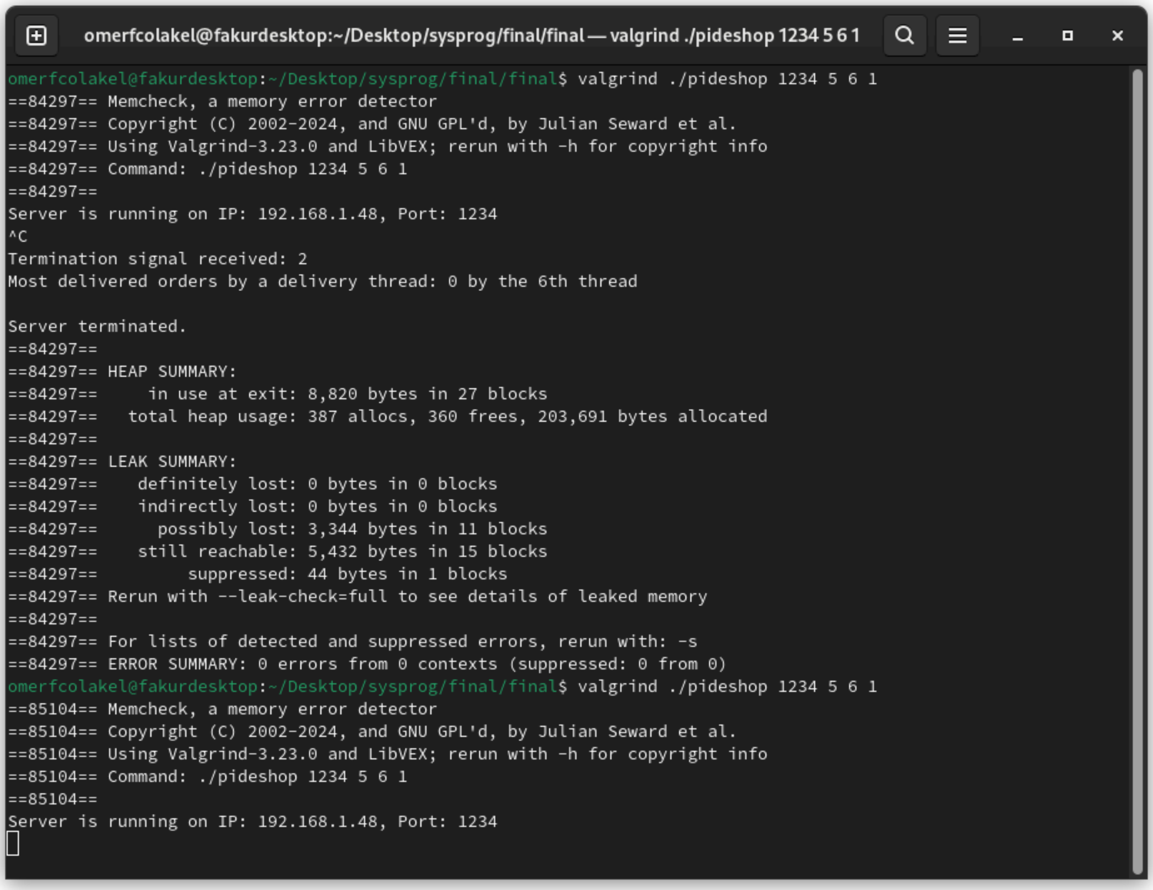

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

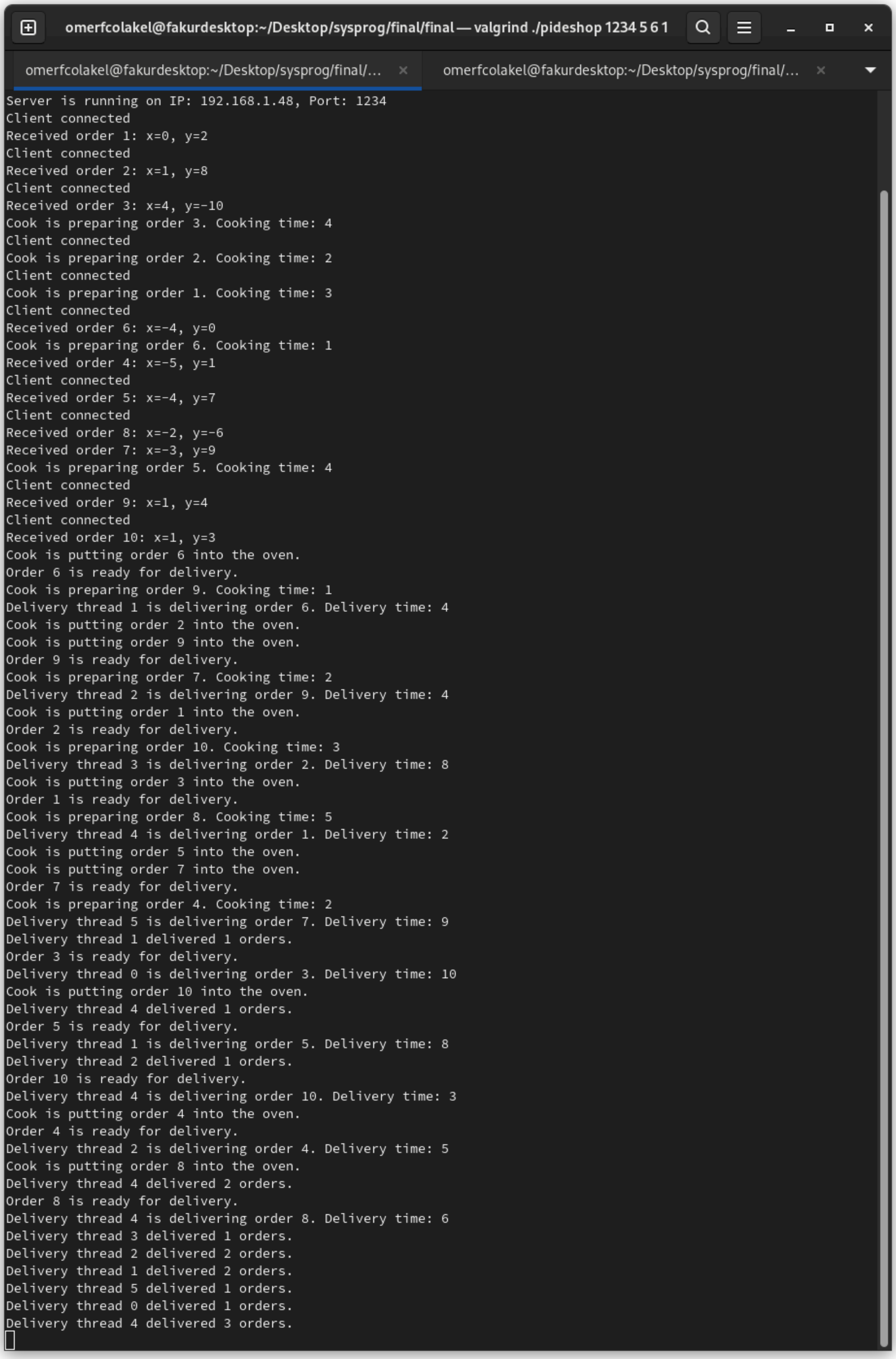
```
Client 56 connecting to 192.168.1.49:1234
Client 57 connecting to 192.168.1.49:1234
Client 58 connecting to 192.168.1.49:1234
Client 59 connecting to 192.168.1.49:1234
Client 60 connecting to 192.168.1.49:1234
Client 61 connecting to 192.168.1.49:1234
Client 62 connecting to 192.168.1.49:1234
Client 63 connecting to 192.168.1.49:1234
Client 64 connecting to 192.168.1.49:1234
Client 65 connecting to 192.168.1.49:1234
Client 66 connecting to 192.168.1.49:1234
Client 67 connecting to 192.168.1.49:1234
Client 68 connecting to 192.168.1.49:1234
Client 69 connecting to 192.168.1.49:1234
Client 70 connecting to 192.168.1.49:1234
Client 71 connecting to 192.168.1.49:1234
Client 72 connecting to 192.168.1.49:1234
Client 73 connecting to 192.168.1.49:1234
Client 74 connecting to 192.168.1.49:1234
Client 75 connecting to 192.168.1.49:1234
Client 76 connecting to 192.168.1.49:1234
Client 77 connecting to 192.168.1.49:1234
Client 78 connecting to 192.168.1.49:1234
Client 79 connecting to 192.168.1.49:1234
Client 80 connecting to 192.168.1.49:1234
Client 81 connecting to 192.168.1.49:1234
Client 82 connecting to 192.168.1.49:1234
Client 83 connecting to 192.168.1.49:1234
Client 84 connecting to 192.168.1.49:1234
Client 85 connecting to 192.168.1.49:1234
Client 86 connecting to 192.168.1.49:1234
Client 87 connecting to 192.168.1.49:1234
Client 88 connecting to 192.168.1.49:1234
Client 89 connecting to 192.168.1.49:1234
Client 90 connecting to 192.168.1.49:1234
Client 91 connecting to 192.168.1.49:1234
Client 92 connecting to 192.168.1.49:1234
Client 93 connecting to 192.168.1.49:1234
Client 94 connecting to 192.168.1.49:1234
Client 95 connecting to 192.168.1.49:1234
Client 96 connecting to 192.168.1.49:1234
Client 97 connecting to 192.168.1.49:1234
Client 98 connecting to 192.168.1.49:1234
Client 99 connecting to 192.168.1.49:1234
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
Connection failed: No route to host
```

3. Server runs again after closing with interrupt signal


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final — valgrind ./pideshop 1234 5 6 1

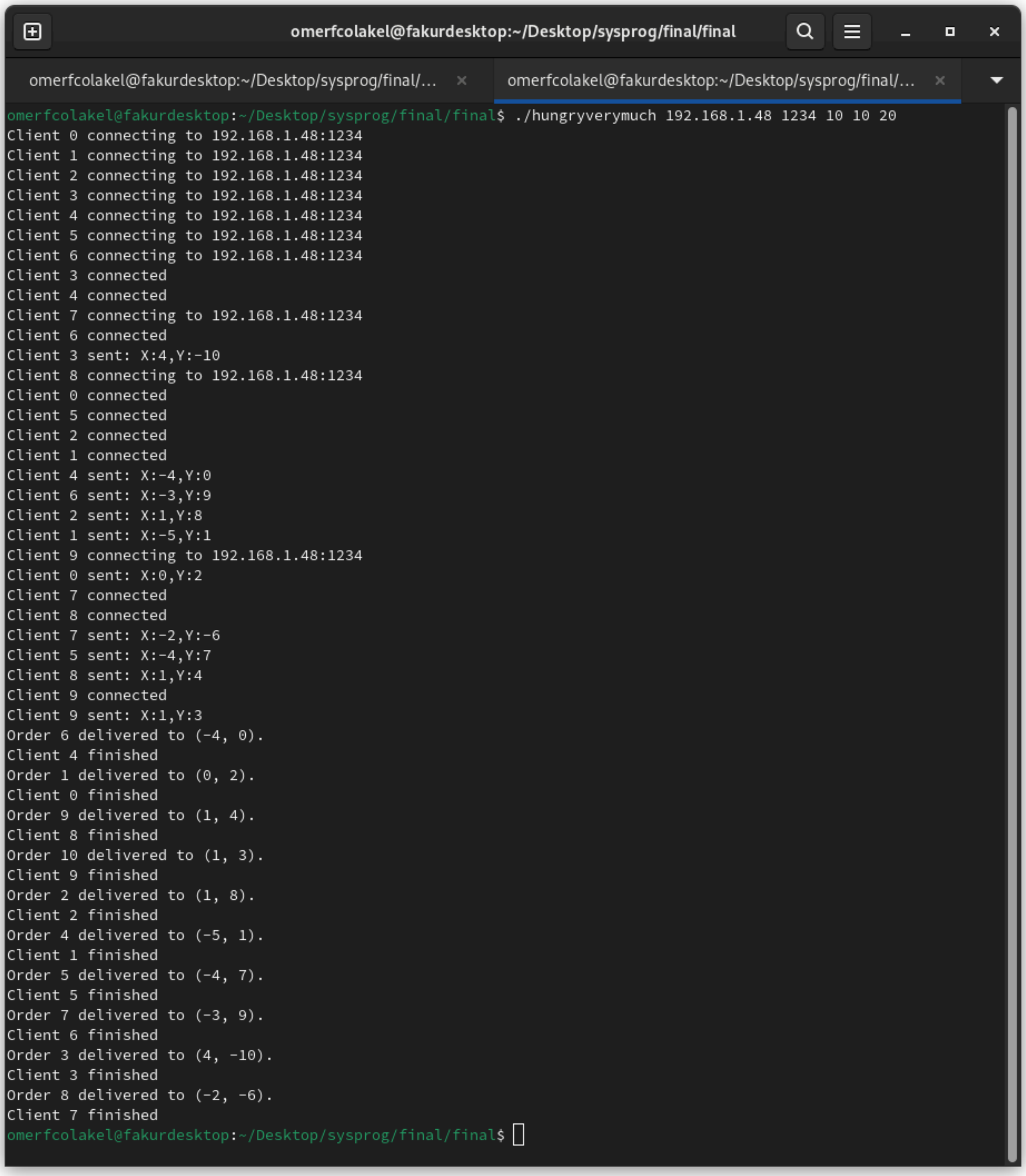
```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ valgrind ./pideshop 1234 5 6 1
==84297== Memcheck, a memory error detector
==84297== Copyright (C) 2002-2024, and GNU GPL'd, by Julian Seward et al.
==84297== Using Valgrind-3.23.0 and LibVEX; rerun with -h for copyright info
==84297== Command: ./pideshop 1234 5 6 1
==84297==
Server is running on IP: 192.168.1.48, Port: 1234
^C
Termination signal received: 2
Most delivered orders by a delivery thread: 0 by the 6th thread

Server terminated.
==84297==
==84297== HEAP SUMMARY:
==84297==     in use at exit: 8,820 bytes in 27 blocks
==84297==   total heap usage: 387 allocs, 360 frees, 203,691 bytes allocated
==84297==
==84297== LEAK SUMMARY:
==84297==    definitely lost: 0 bytes in 0 blocks
==84297==    indirectly lost: 0 bytes in 0 blocks
==84297==      possibly lost: 3,344 bytes in 11 blocks
==84297==    still reachable: 5,432 bytes in 15 blocks
==84297==         suppressed: 44 bytes in 1 blocks
==84297== Rerun with --leak-check=full to see details of leaked memory
==84297==
==84297== For lists of detected and suppressed errors, rerun with: -s
==84297== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ valgrind ./pideshop 1234 5 6 1
==85104== Memcheck, a memory error detector
==85104== Copyright (C) 2002-2024, and GNU GPL'd, by Julian Seward et al.
==85104== Using Valgrind-3.23.0 and LibVEX; rerun with -h for copyright info
==85104== Command: ./pideshop 1234 5 6 1
==85104==
Server is running on IP: 192.168.1.48, Port: 1234
```

4. Running the server and the client with 10 orders (Server - Client)


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final — valgrind ./pideshop 1234 5 6 1

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/... omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
Server is running on IP: 192.168.1.48, Port: 1234
Client connected
Received order 1: x=0, y=2
Client connected
Received order 2: x=1, y=8
Client connected
Received order 3: x=4, y=-10
Cook is preparing order 3. Cooking time: 4
Client connected
Cook is preparing order 2. Cooking time: 2
Client connected
Cook is preparing order 1. Cooking time: 3
Client connected
Received order 6: x=-4, y=0
Cook is preparing order 6. Cooking time: 1
Received order 4: x=-5, y=1
Client connected
Received order 5: x=-4, y=7
Client connected
Received order 8: x=-2, y=-6
Received order 7: x=-3, y=9
Cook is preparing order 5. Cooking time: 4
Client connected
Received order 9: x=1, y=4
Client connected
Received order 10: x=1, y=3
Cook is putting order 6 into the oven.
Order 6 is ready for delivery.
Cook is preparing order 9. Cooking time: 1
Delivery thread 1 is delivering order 6. Delivery time: 4
Cook is putting order 2 into the oven.
Cook is putting order 9 into the oven.
Order 9 is ready for delivery.
Cook is preparing order 7. Cooking time: 2
Delivery thread 2 is delivering order 9. Delivery time: 4
Cook is putting order 1 into the oven.
Order 2 is ready for delivery.
Cook is preparing order 10. Cooking time: 3
Delivery thread 3 is delivering order 2. Delivery time: 8
Cook is putting order 3 into the oven.
Order 1 is ready for delivery.
Cook is preparing order 8. Cooking time: 5
Delivery thread 4 is delivering order 1. Delivery time: 2
Cook is putting order 5 into the oven.
Cook is putting order 7 into the oven.
Order 7 is ready for delivery.
Cook is preparing order 4. Cooking time: 2
Delivery thread 5 is delivering order 7. Delivery time: 9
Delivery thread 1 delivered 1 orders.
Order 3 is ready for delivery.
Delivery thread 0 is delivering order 3. Delivery time: 10
Cook is putting order 10 into the oven.
Delivery thread 4 delivered 1 orders.
Order 5 is ready for delivery.
Delivery thread 1 is delivering order 5. Delivery time: 8
Delivery thread 2 delivered 1 orders.
Order 10 is ready for delivery.
Delivery thread 4 is delivering order 10. Delivery time: 3
Cook is putting order 4 into the oven.
Order 4 is ready for delivery.
Delivery thread 2 is delivering order 4. Delivery time: 5
Cook is putting order 8 into the oven.
Delivery thread 4 delivered 2 orders.
Order 8 is ready for delivery.
Delivery thread 4 is delivering order 8. Delivery time: 6
Delivery thread 3 delivered 1 orders.
Delivery thread 2 delivered 2 orders.
Delivery thread 1 delivered 2 orders.
Delivery thread 5 delivered 1 orders.
Delivery thread 0 delivered 1 orders.
Delivery thread 4 delivered 3 orders.
```



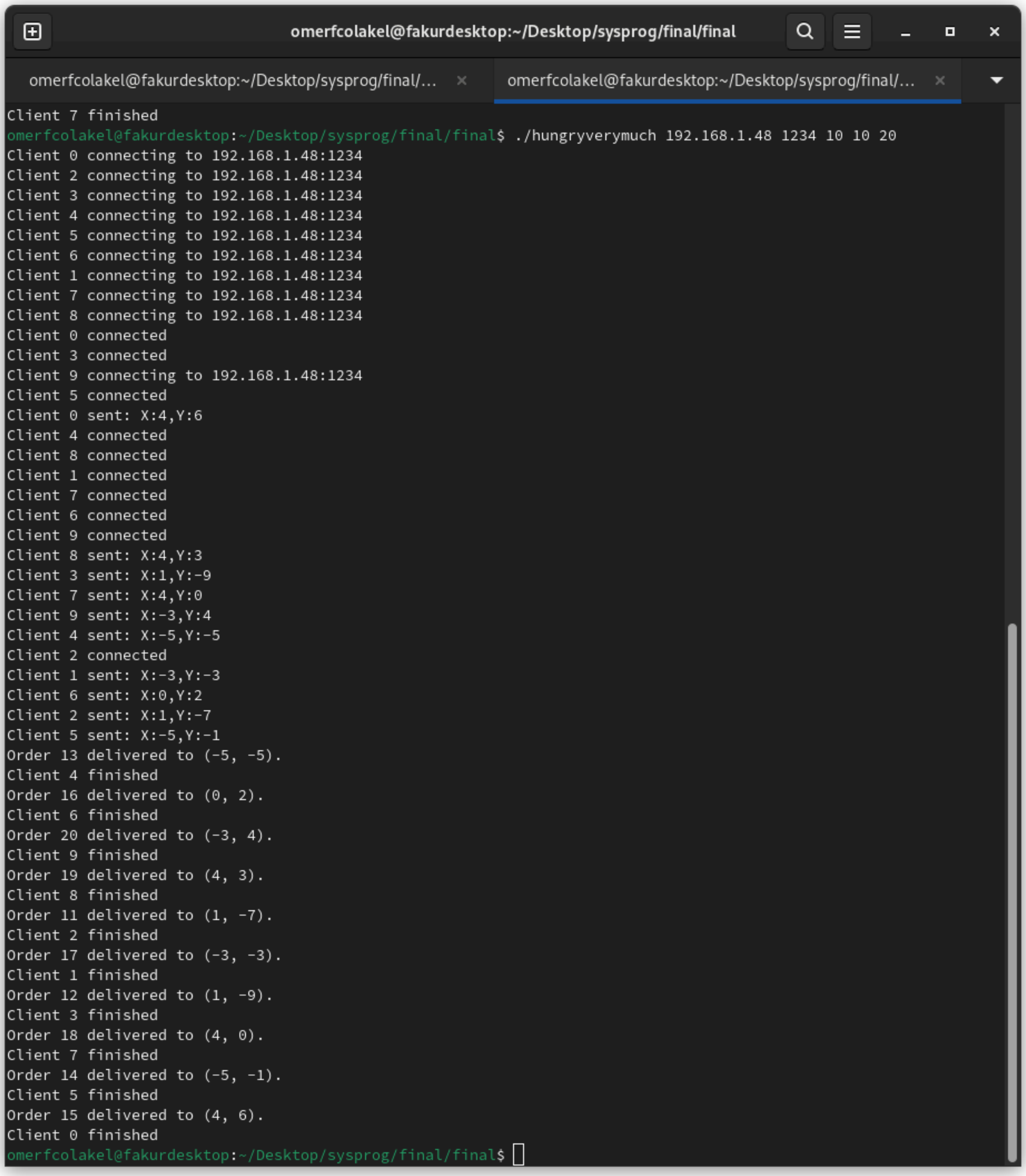
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

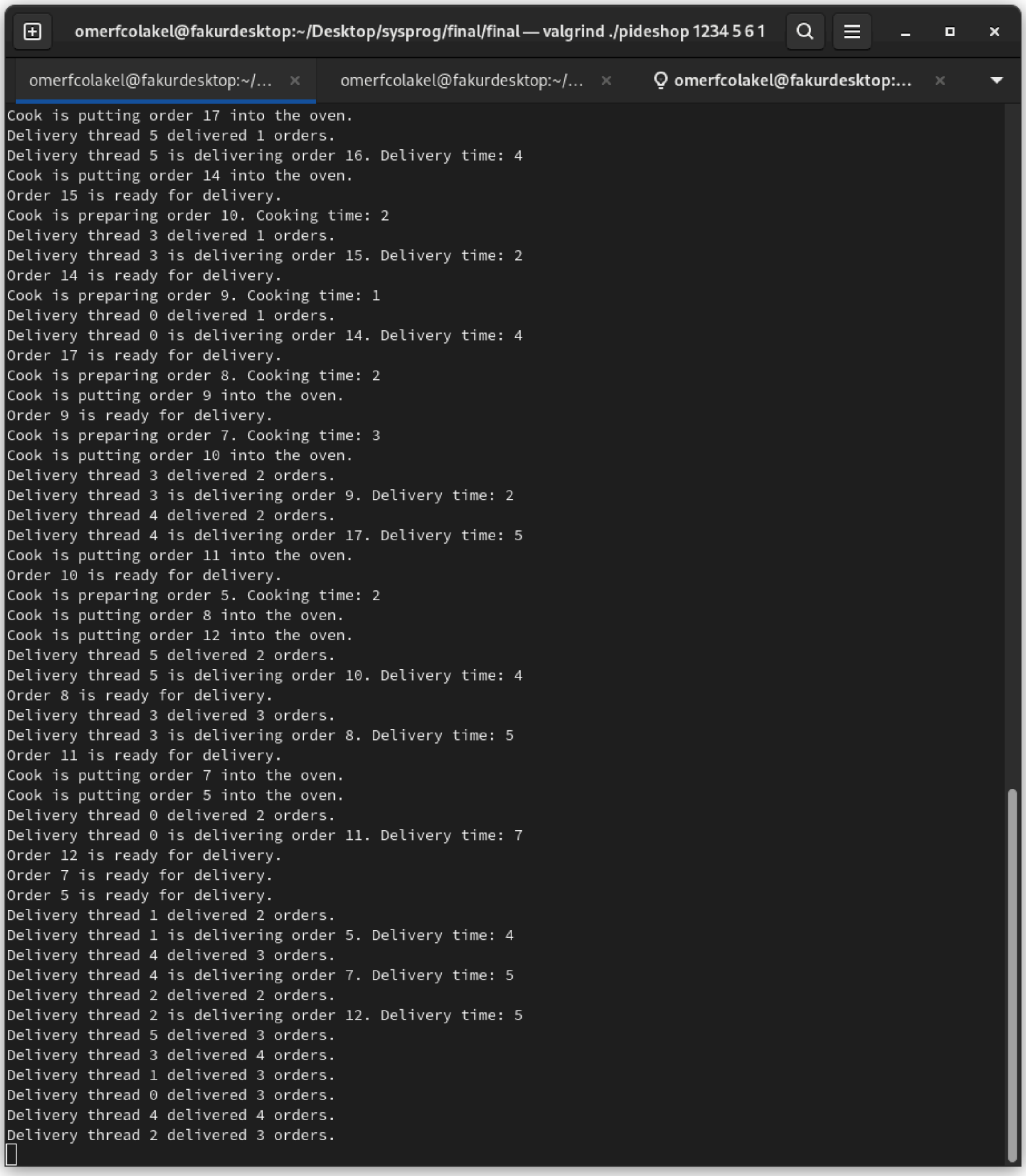
```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.168.1.48 1234 10 10 20
Client 0 connecting to 192.168.1.48:1234
Client 1 connecting to 192.168.1.48:1234
Client 2 connecting to 192.168.1.48:1234
Client 3 connecting to 192.168.1.48:1234
Client 4 connecting to 192.168.1.48:1234
Client 5 connecting to 192.168.1.48:1234
Client 6 connecting to 192.168.1.48:1234
Client 3 connected
Client 4 connected
Client 7 connecting to 192.168.1.48:1234
Client 6 connected
Client 3 sent: X:4,Y:-10
Client 8 connecting to 192.168.1.48:1234
Client 0 connected
Client 5 connected
Client 2 connected
Client 1 connected
Client 4 sent: X:-4,Y:0
Client 6 sent: X:-3,Y:9
Client 2 sent: X:1,Y:8
Client 1 sent: X:-5,Y:1
Client 9 connecting to 192.168.1.48:1234
Client 0 sent: X:0,Y:2
Client 7 connected
Client 8 connected
Client 7 sent: X:-2,Y:-6
Client 5 sent: X:-4,Y:7
Client 8 sent: X:1,Y:4
Client 9 connected
Client 9 sent: X:1,Y:3
Order 6 delivered to (-4, 0).
Client 4 finished
Order 1 delivered to (0, 2).
Client 0 finished
Order 9 delivered to (1, 4).
Client 8 finished
Order 10 delivered to (1, 3).
Client 9 finished
Order 2 delivered to (1, 8).
Client 2 finished
Order 4 delivered to (-5, 1).
Client 1 finished
Order 5 delivered to (-4, 7).
Client 5 finished
Order 7 delivered to (-3, 9).
Client 6 finished
Order 3 delivered to (4, -10).
Client 3 finished
Order 8 delivered to (-2, -6).
Client 7 finished
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

5. Running the server and 2 clients with 10 orders each back to back but not at the same time (I ran it after the test case 4)


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
Client 7 finished
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.168.1.48 1234 10 10 20
Client 0 connecting to 192.168.1.48:1234
Client 2 connecting to 192.168.1.48:1234
Client 3 connecting to 192.168.1.48:1234
Client 4 connecting to 192.168.1.48:1234
Client 5 connecting to 192.168.1.48:1234
Client 6 connecting to 192.168.1.48:1234
Client 1 connecting to 192.168.1.48:1234
Client 7 connecting to 192.168.1.48:1234
Client 8 connecting to 192.168.1.48:1234
Client 0 connected
Client 3 connected
Client 9 connecting to 192.168.1.48:1234
Client 5 connected
Client 0 sent: X:4,Y:6
Client 4 connected
Client 8 connected
Client 1 connected
Client 7 connected
Client 6 connected
Client 9 connected
Client 8 sent: X:4,Y:3
Client 3 sent: X:1,Y:-9
Client 7 sent: X:4,Y:0
Client 9 sent: X:-3,Y:4
Client 4 sent: X:-5,Y:-5
Client 2 connected
Client 1 sent: X:-3,Y:-3
Client 6 sent: X:0,Y:2
Client 2 sent: X:1,Y:-7
Client 5 sent: X:-5,Y:-1
Order 13 delivered to (-5, -5).
Client 4 finished
Order 16 delivered to (0, 2).
Client 6 finished
Order 20 delivered to (-3, 4).
Client 9 finished
Order 19 delivered to (4, 3).
Client 8 finished
Order 11 delivered to (1, -7).
Client 2 finished
Order 17 delivered to (-3, -3).
Client 1 finished
Order 12 delivered to (1, -9).
Client 3 finished
Order 18 delivered to (4, 0).
Client 7 finished
Order 14 delivered to (-5, -1).
Client 5 finished
Order 15 delivered to (4, 6).
Client 0 finished
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

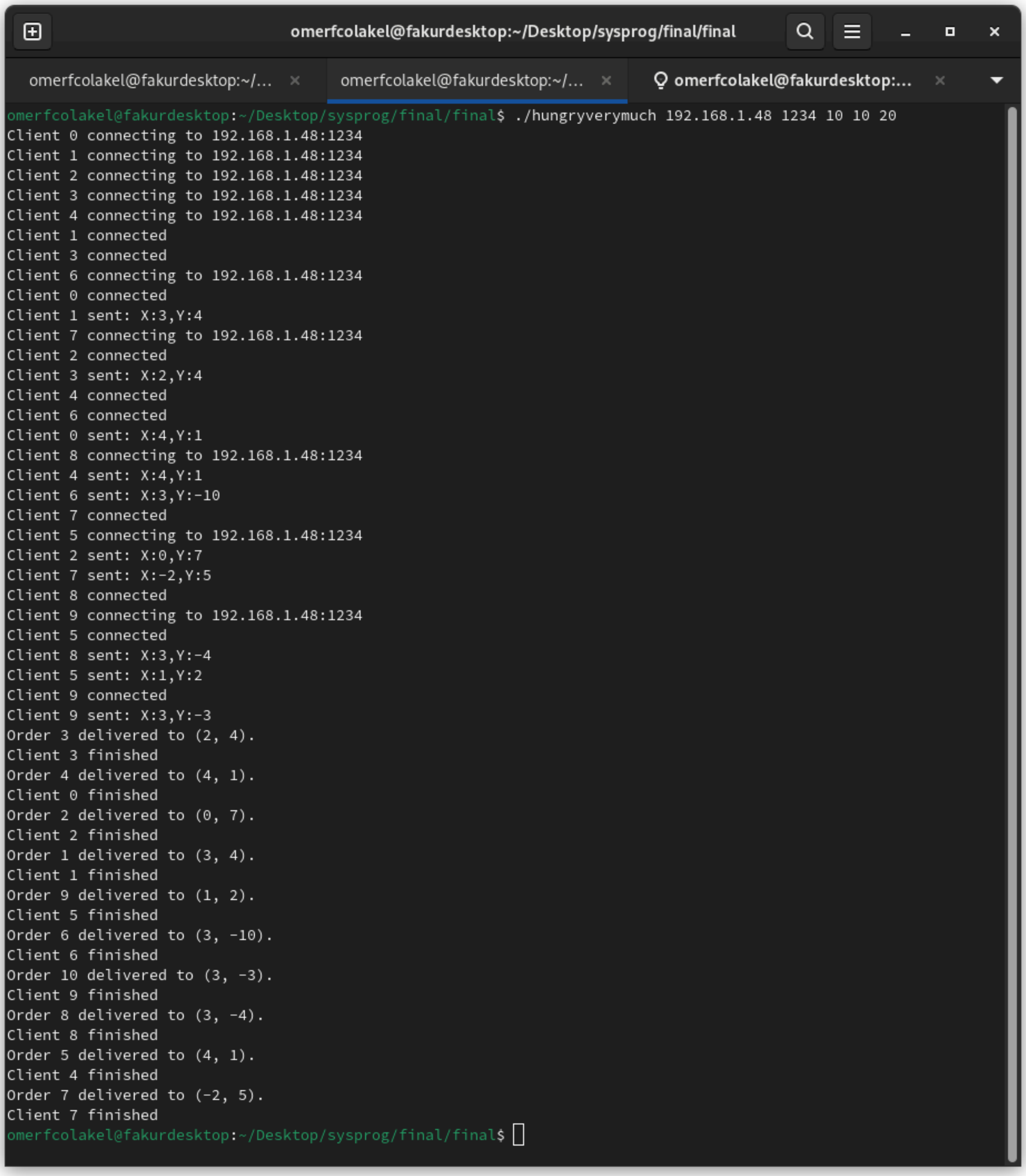
6. Running two clients at the same time (Server - Client1 - Client2)



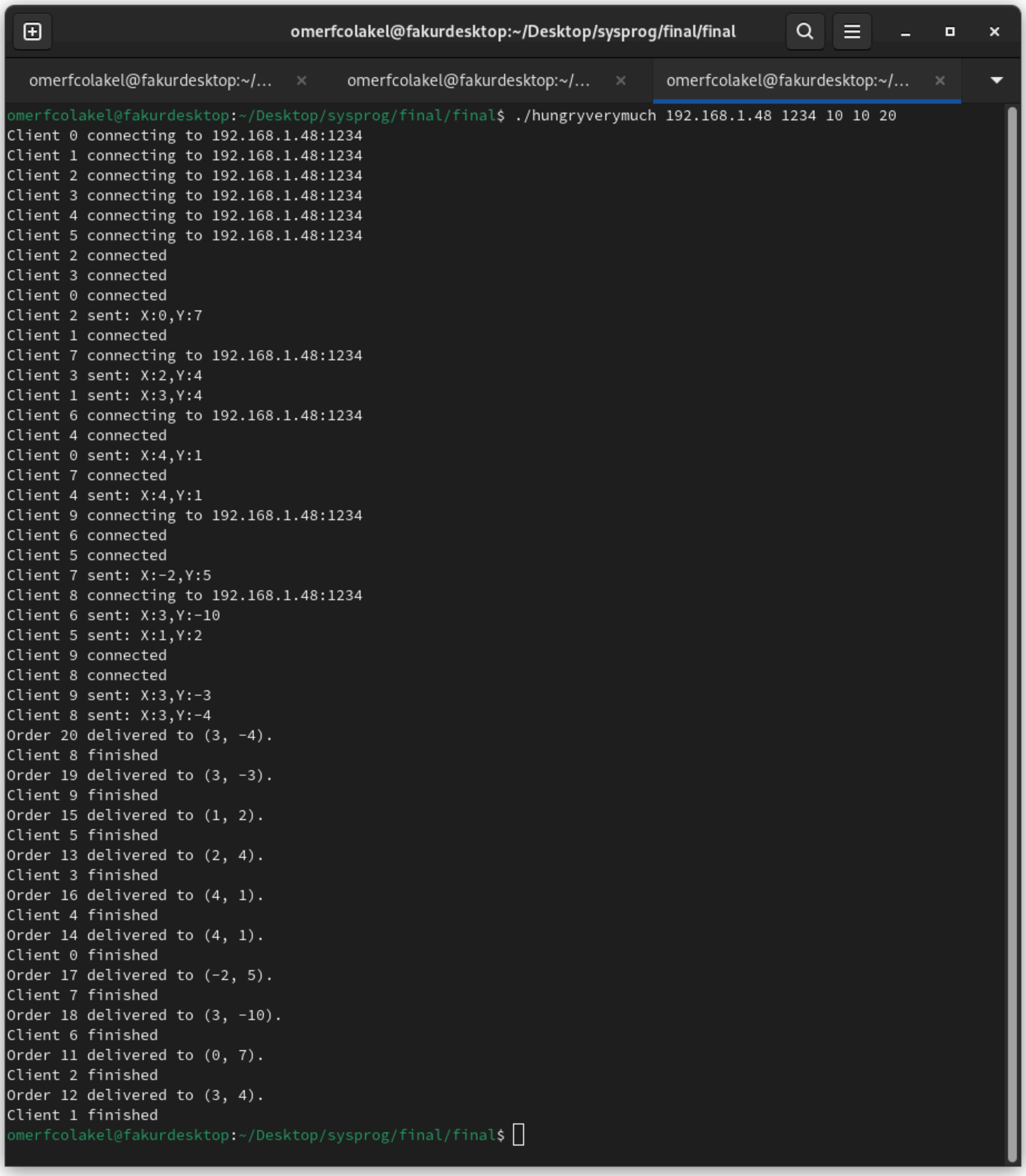
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final — valgrind ./pideshop 1234 5 6 1

omerfcolakel@fakurdesktop:~/... | omerfcolakel@fakurdesktop:~/... | omerfcolakel@fakurdesktop:...

```
Cook is putting order 17 into the oven.
Delivery thread 5 delivered 1 orders.
Delivery thread 5 is delivering order 16. Delivery time: 4
Cook is putting order 14 into the oven.
Order 15 is ready for delivery.
Cook is preparing order 10. Cooking time: 2
Delivery thread 3 delivered 1 orders.
Delivery thread 3 is delivering order 15. Delivery time: 2
Order 14 is ready for delivery.
Cook is preparing order 9. Cooking time: 1
Delivery thread 0 delivered 1 orders.
Delivery thread 0 is delivering order 14. Delivery time: 4
Order 17 is ready for delivery.
Cook is preparing order 8. Cooking time: 2
Cook is putting order 9 into the oven.
Order 9 is ready for delivery.
Cook is preparing order 7. Cooking time: 3
Cook is putting order 10 into the oven.
Delivery thread 3 delivered 2 orders.
Delivery thread 3 is delivering order 9. Delivery time: 2
Delivery thread 4 delivered 2 orders.
Delivery thread 4 is delivering order 17. Delivery time: 5
Cook is putting order 11 into the oven.
Order 10 is ready for delivery.
Cook is preparing order 5. Cooking time: 2
Cook is putting order 8 into the oven.
Cook is putting order 12 into the oven.
Delivery thread 5 delivered 2 orders.
Delivery thread 5 is delivering order 10. Delivery time: 4
Order 8 is ready for delivery.
Delivery thread 3 delivered 3 orders.
Delivery thread 3 is delivering order 8. Delivery time: 5
Order 11 is ready for delivery.
Cook is putting order 7 into the oven.
Cook is putting order 5 into the oven.
Delivery thread 0 delivered 2 orders.
Delivery thread 0 is delivering order 11. Delivery time: 7
Order 12 is ready for delivery.
Order 7 is ready for delivery.
Order 5 is ready for delivery.
Delivery thread 1 delivered 2 orders.
Delivery thread 1 is delivering order 5. Delivery time: 4
Delivery thread 4 delivered 3 orders.
Delivery thread 4 is delivering order 7. Delivery time: 5
Delivery thread 2 delivered 2 orders.
Delivery thread 2 is delivering order 12. Delivery time: 5
Delivery thread 5 delivered 3 orders.
Delivery thread 3 delivered 4 orders.
Delivery thread 1 delivered 3 orders.
Delivery thread 0 delivered 3 orders.
Delivery thread 4 delivered 4 orders.
Delivery thread 2 delivered 3 orders.
```

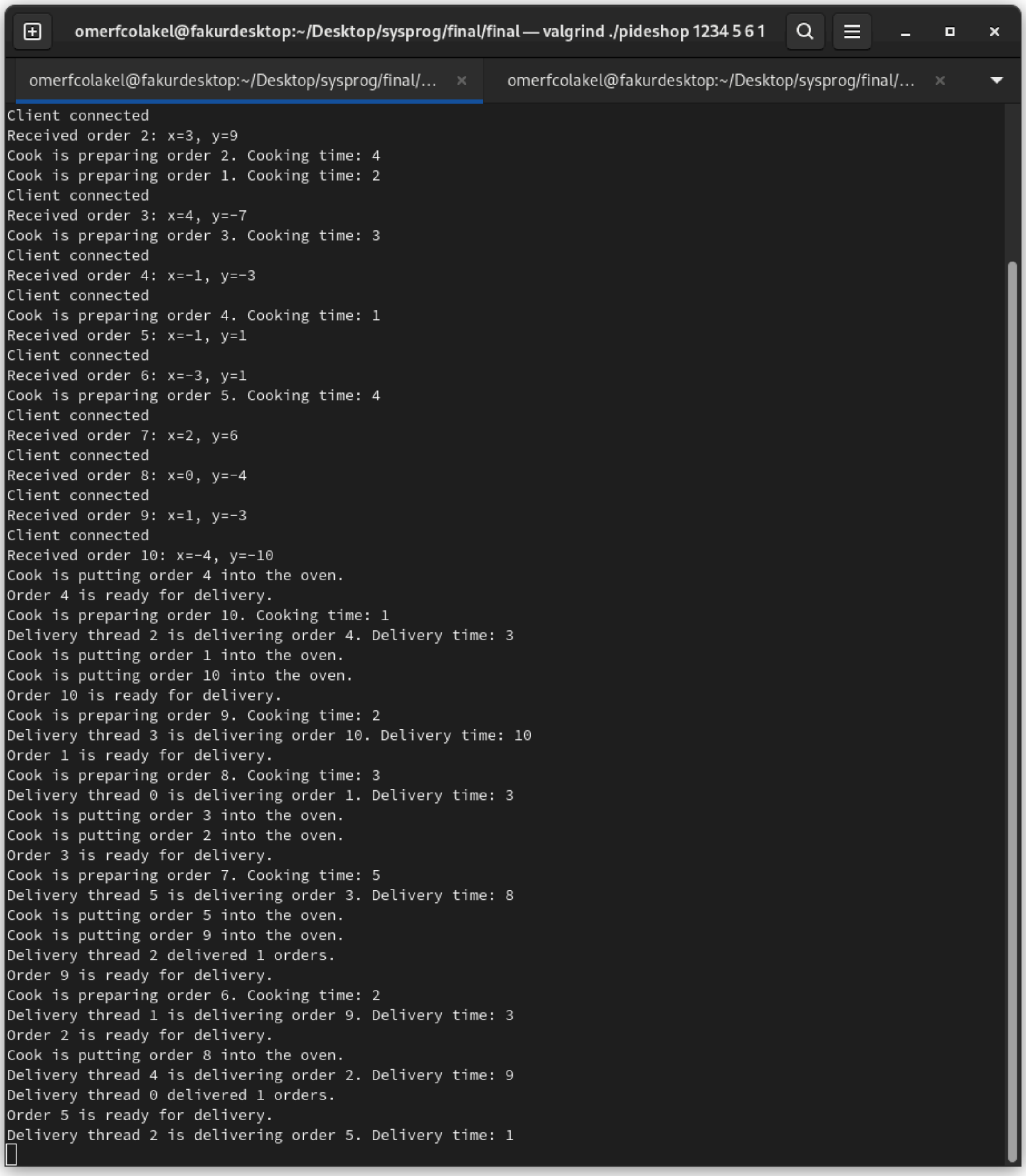

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final
omerfcolakel@fakurdesktop:~/...
omerfcolakel@fakurdesktop:~/...
omerfcolakel@fakurdesktop:...

```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.168.1.48 1234 10 10 20
Client 0 connecting to 192.168.1.48:1234
Client 1 connecting to 192.168.1.48:1234
Client 2 connecting to 192.168.1.48:1234
Client 3 connecting to 192.168.1.48:1234
Client 4 connecting to 192.168.1.48:1234
Client 1 connected
Client 3 connected
Client 6 connecting to 192.168.1.48:1234
Client 0 connected
Client 1 sent: X:3,Y:4
Client 7 connecting to 192.168.1.48:1234
Client 2 connected
Client 3 sent: X:2,Y:4
Client 4 connected
Client 6 connected
Client 0 sent: X:4,Y:1
Client 8 connecting to 192.168.1.48:1234
Client 4 sent: X:4,Y:1
Client 6 sent: X:3,Y:-10
Client 7 connected
Client 5 connecting to 192.168.1.48:1234
Client 2 sent: X:0,Y:7
Client 7 sent: X:-2,Y:5
Client 8 connected
Client 9 connecting to 192.168.1.48:1234
Client 5 connected
Client 8 sent: X:3,Y:-4
Client 5 sent: X:1,Y:2
Client 9 connected
Client 9 sent: X:3,Y:-3
Order 3 delivered to (2, 4).
Client 3 finished
Order 4 delivered to (4, 1).
Client 0 finished
Order 2 delivered to (0, 7).
Client 2 finished
Order 1 delivered to (3, 4).
Client 1 finished
Order 9 delivered to (1, 2).
Client 5 finished
Order 6 delivered to (3, -10).
Client 6 finished
Order 10 delivered to (3, -3).
Client 9 finished
Order 8 delivered to (3, -4).
Client 8 finished
Order 5 delivered to (4, 1).
Client 4 finished
Order 7 delivered to (-2, 5).
Client 7 finished
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

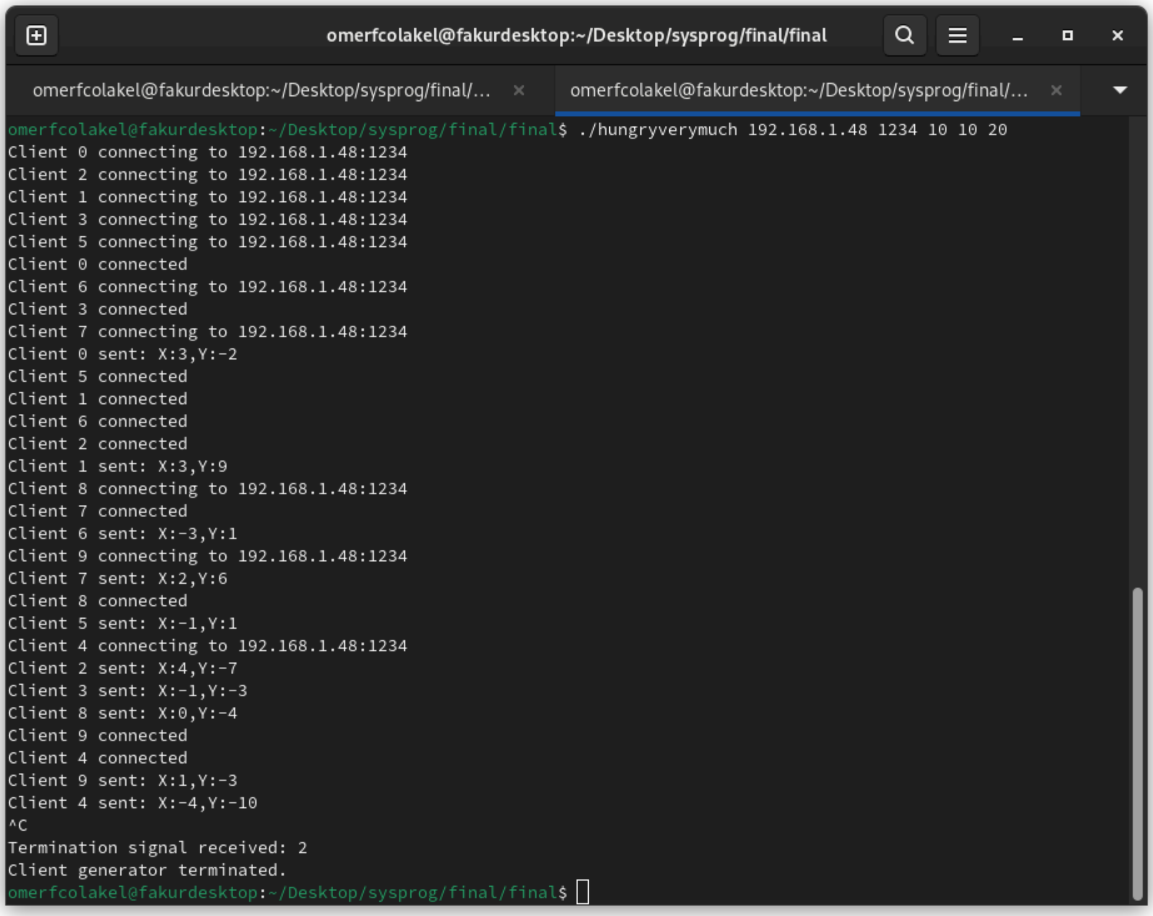

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final
omerfcolakel@fakurdesktop:~/...
omerfcolakel@fakurdesktop:~/...
omerfcolakel@fakurdesktop:~/...

```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.168.1.48 1234 10 10 20
Client 0 connecting to 192.168.1.48:1234
Client 1 connecting to 192.168.1.48:1234
Client 2 connecting to 192.168.1.48:1234
Client 3 connecting to 192.168.1.48:1234
Client 4 connecting to 192.168.1.48:1234
Client 5 connecting to 192.168.1.48:1234
Client 2 connected
Client 3 connected
Client 0 connected
Client 2 sent: X:0,Y:7
Client 1 connected
Client 7 connecting to 192.168.1.48:1234
Client 3 sent: X:2,Y:4
Client 1 sent: X:3,Y:4
Client 6 connecting to 192.168.1.48:1234
Client 4 connected
Client 0 sent: X:4,Y:1
Client 7 connected
Client 4 sent: X:4,Y:1
Client 9 connecting to 192.168.1.48:1234
Client 6 connected
Client 5 connected
Client 7 sent: X:-2,Y:5
Client 8 connecting to 192.168.1.48:1234
Client 6 sent: X:3,Y:-10
Client 5 sent: X:1,Y:2
Client 9 connected
Client 8 connected
Client 9 sent: X:3,Y:-3
Client 8 sent: X:3,Y:-4
Order 20 delivered to (3, -4).
Client 8 finished
Order 19 delivered to (3, -3).
Client 9 finished
Order 15 delivered to (1, 2).
Client 5 finished
Order 13 delivered to (2, 4).
Client 3 finished
Order 16 delivered to (4, 1).
Client 4 finished
Order 14 delivered to (4, 1).
Client 0 finished
Order 17 delivered to (-2, 5).
Client 7 finished
Order 18 delivered to (3, -10).
Client 6 finished
Order 11 delivered to (0, 7).
Client 2 finished
Order 12 delivered to (3, 4).
Client 1 finished
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

7. Run the server and client. Terminate the client (Server - Client) (As I said server doesn't understand that the client is terminated)


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final — valgrind ./pideshop 1234 5 6 1
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

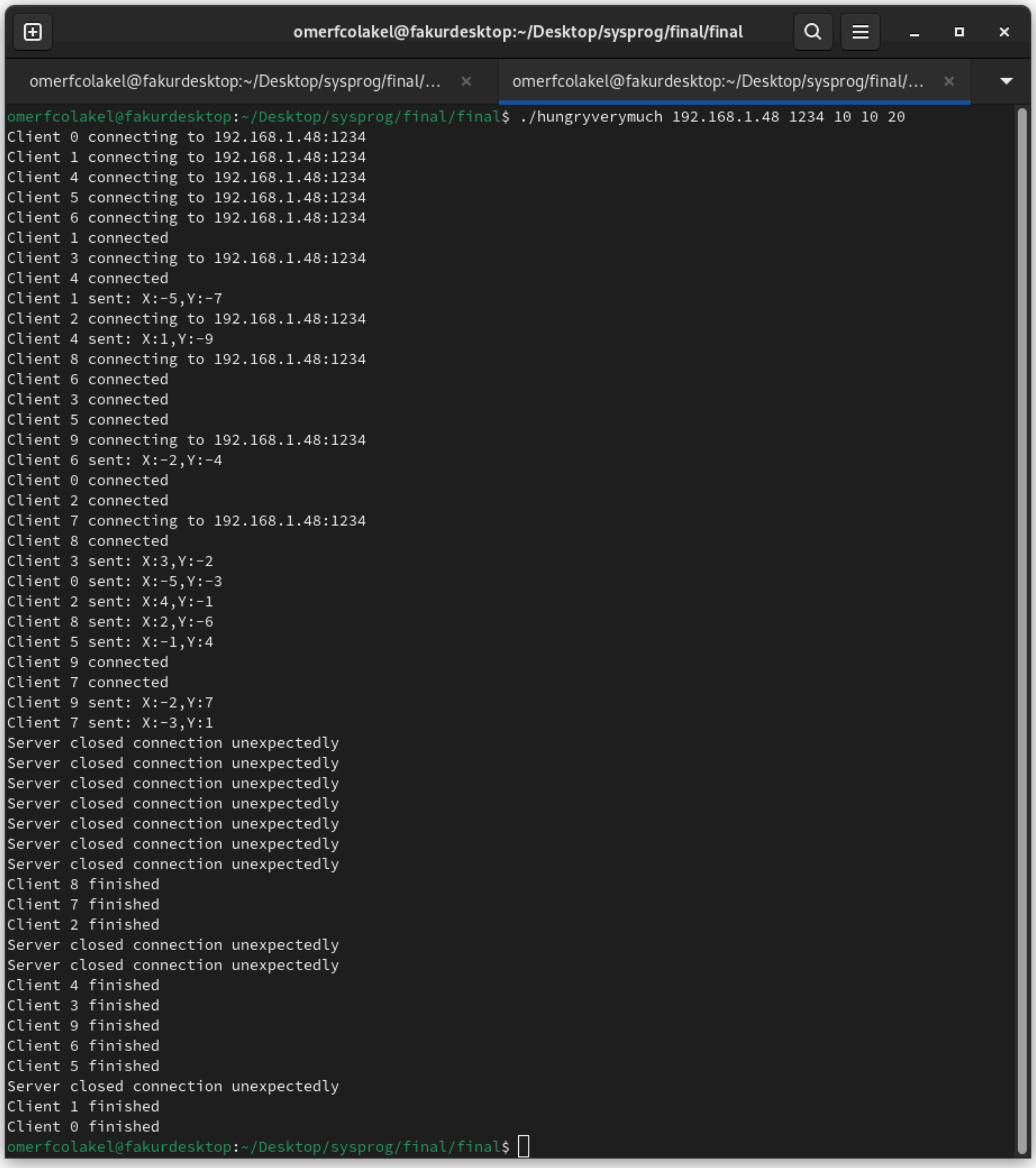
```
Client connected
Received order 2: x=3, y=9
Cook is preparing order 2. Cooking time: 4
Cook is preparing order 1. Cooking time: 2
Client connected
Received order 3: x=4, y=-7
Cook is preparing order 3. Cooking time: 3
Client connected
Received order 4: x=-1, y=-3
Client connected
Cook is preparing order 4. Cooking time: 1
Received order 5: x=-1, y=1
Client connected
Received order 6: x=-3, y=1
Cook is preparing order 5. Cooking time: 4
Client connected
Received order 7: x=2, y=6
Client connected
Received order 8: x=0, y=-4
Client connected
Received order 9: x=1, y=-3
Client connected
Received order 10: x=-4, y=-10
Cook is putting order 4 into the oven.
Order 4 is ready for delivery.
Cook is preparing order 10. Cooking time: 1
Delivery thread 2 is delivering order 4. Delivery time: 3
Cook is putting order 1 into the oven.
Cook is putting order 10 into the oven.
Order 10 is ready for delivery.
Cook is preparing order 9. Cooking time: 2
Delivery thread 3 is delivering order 10. Delivery time: 10
Order 1 is ready for delivery.
Cook is preparing order 8. Cooking time: 3
Delivery thread 0 is delivering order 1. Delivery time: 3
Cook is putting order 3 into the oven.
Cook is putting order 2 into the oven.
Order 3 is ready for delivery.
Cook is preparing order 7. Cooking time: 5
Delivery thread 5 is delivering order 3. Delivery time: 8
Cook is putting order 5 into the oven.
Cook is putting order 9 into the oven.
Delivery thread 2 delivered 1 orders.
Order 9 is ready for delivery.
Cook is preparing order 6. Cooking time: 2
Delivery thread 1 is delivering order 9. Delivery time: 3
Order 2 is ready for delivery.
Cook is putting order 8 into the oven.
Delivery thread 4 is delivering order 2. Delivery time: 9
Delivery thread 0 delivered 1 orders.
Order 5 is ready for delivery.
Delivery thread 2 is delivering order 5. Delivery time: 1
```



omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.168.1.48 1234 10 10 20
Client 0 connecting to 192.168.1.48:1234
Client 2 connecting to 192.168.1.48:1234
Client 1 connecting to 192.168.1.48:1234
Client 3 connecting to 192.168.1.48:1234
Client 5 connecting to 192.168.1.48:1234
Client 0 connected
Client 6 connecting to 192.168.1.48:1234
Client 3 connected
Client 7 connecting to 192.168.1.48:1234
Client 0 sent: X:3,Y:-2
Client 5 connected
Client 1 connected
Client 6 connected
Client 2 connected
Client 1 sent: X:3,Y:9
Client 8 connecting to 192.168.1.48:1234
Client 7 connected
Client 6 sent: X:-3,Y:1
Client 9 connecting to 192.168.1.48:1234
Client 7 sent: X:2,Y:6
Client 8 connected
Client 5 sent: X:-1,Y:1
Client 4 connecting to 192.168.1.48:1234
Client 2 sent: X:4,Y:-7
Client 3 sent: X:-1,Y:-3
Client 8 sent: X:0,Y:-4
Client 9 connected
Client 4 connected
Client 9 sent: X:1,Y:-3
Client 4 sent: X:-4,Y:-10
^C
Termination signal received: 2
Client generator terminated.
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

8. Run the server and the client. Terminate the server. (Client - Server)

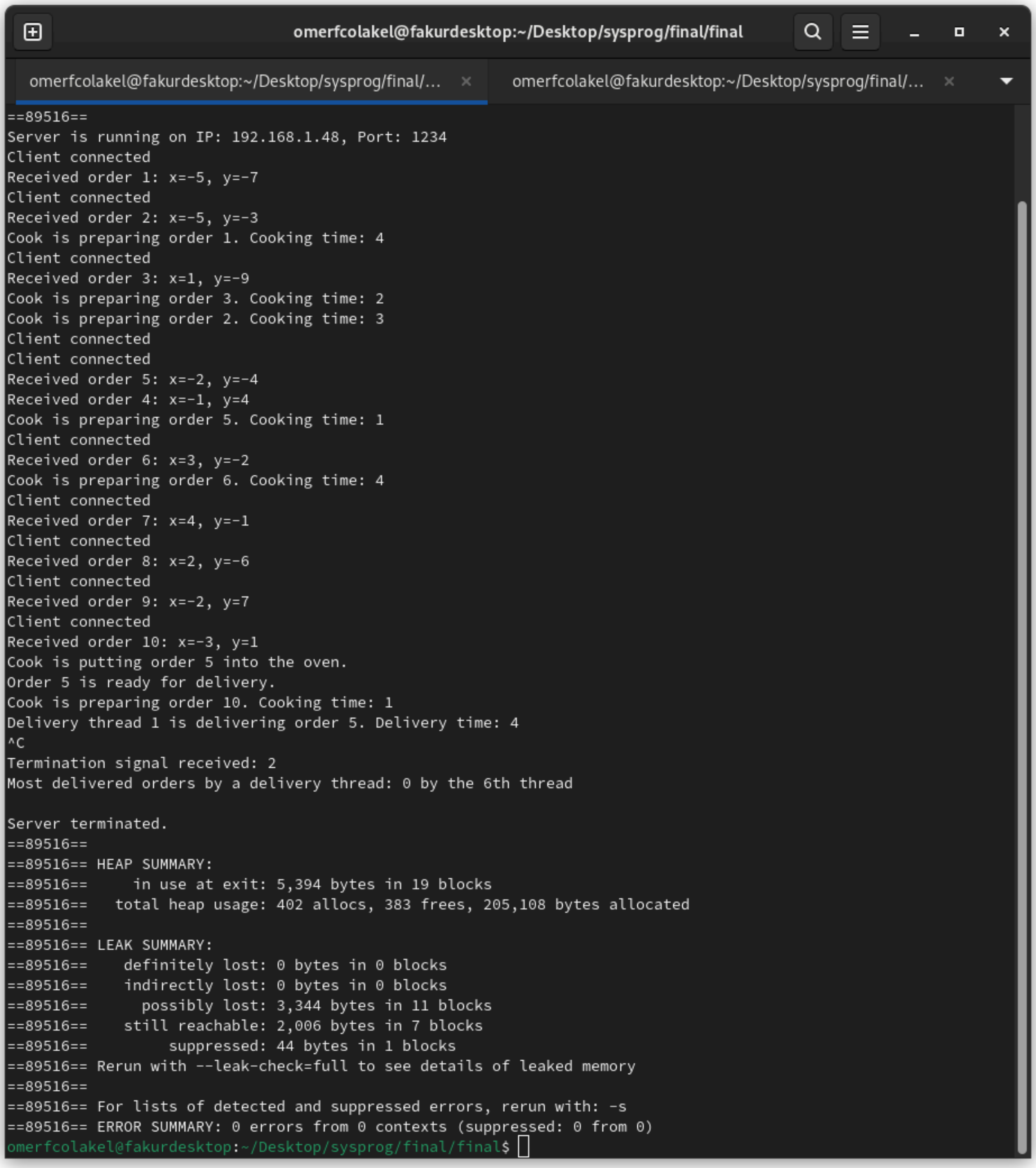


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

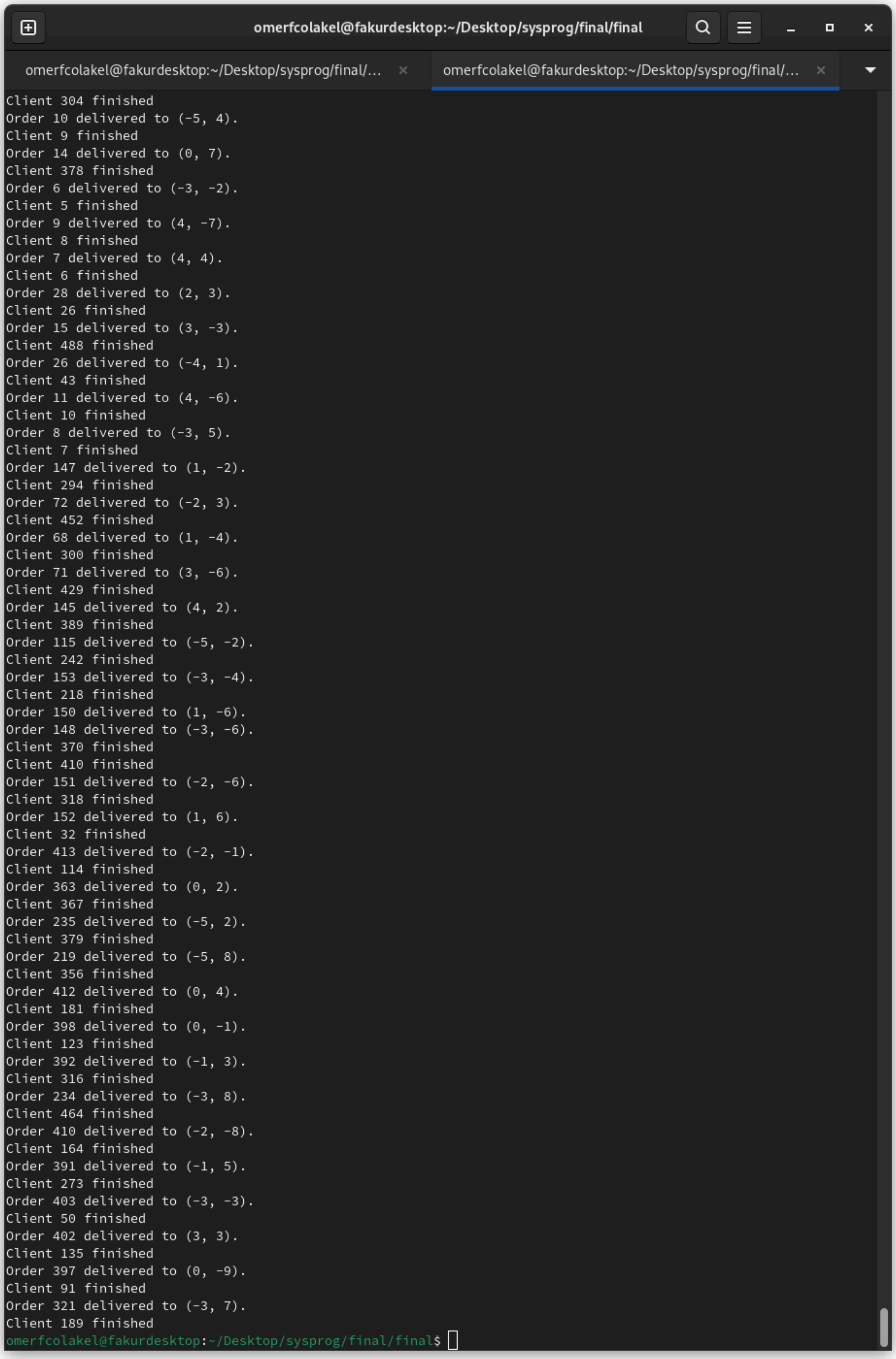
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ ./hungryverymuch 192.168.1.48 1234 10 10 20
Client 0 connecting to 192.168.1.48:1234
Client 1 connecting to 192.168.1.48:1234
Client 4 connecting to 192.168.1.48:1234
Client 5 connecting to 192.168.1.48:1234
Client 6 connecting to 192.168.1.48:1234
Client 1 connected
Client 3 connecting to 192.168.1.48:1234
Client 4 connected
Client 1 sent: X:-5,Y:-7
Client 2 connecting to 192.168.1.48:1234
Client 4 sent: X:1,Y:-9
Client 8 connecting to 192.168.1.48:1234
Client 6 connected
Client 3 connected
Client 5 connected
Client 9 connecting to 192.168.1.48:1234
Client 6 sent: X:-2,Y:-4
Client 0 connected
Client 2 connected
Client 7 connecting to 192.168.1.48:1234
Client 8 connected
Client 3 sent: X:3,Y:-2
Client 0 sent: X:-5,Y:-3
Client 2 sent: X:4,Y:-1
Client 8 sent: X:2,Y:-6
Client 5 sent: X:-1,Y:4
Client 9 connected
Client 7 connected
Client 9 sent: X:-2,Y:7
Client 7 sent: X:-3,Y:1
Server closed connection unexpectedly
Server closed connection unexpectedly
Server closed connection unexpectedly
Server closed connection unexpectedly
Server closed connection unexpectedly
Server closed connection unexpectedly
Server closed connection unexpectedly
Client 8 finished
Client 7 finished
Client 2 finished
Server closed connection unexpectedly
Server closed connection unexpectedly
Client 4 finished
Client 3 finished
Client 9 finished
Client 6 finished
Client 5 finished
Server closed connection unexpectedly
Client 1 finished
Client 0 finished
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

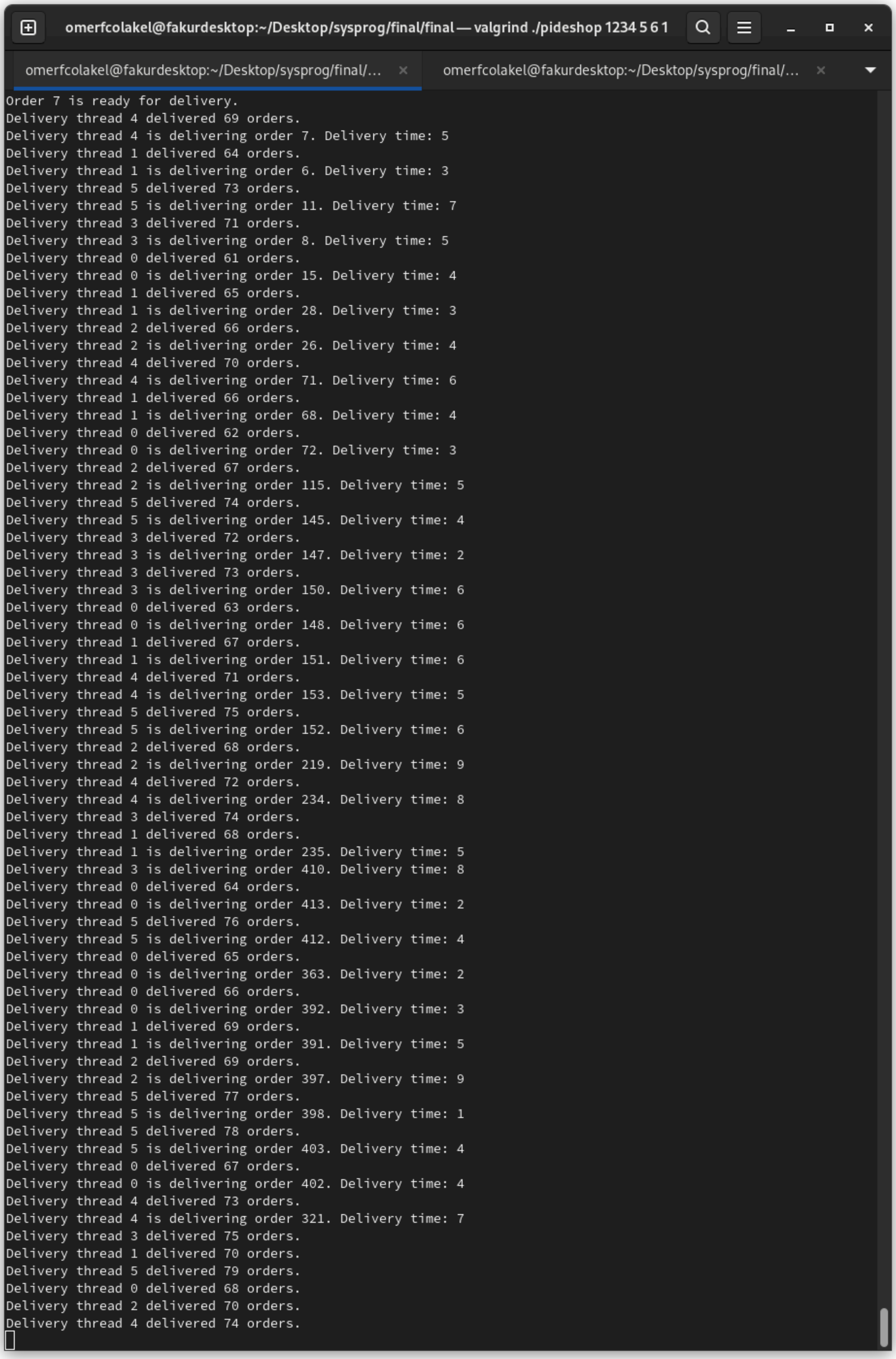

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
==89516==
Server is running on IP: 192.168.1.48, Port: 1234
Client connected
Received order 1: x=-5, y=-7
Client connected
Received order 2: x=-5, y=-3
Cook is preparing order 1. Cooking time: 4
Client connected
Received order 3: x=1, y=-9
Cook is preparing order 3. Cooking time: 2
Cook is preparing order 2. Cooking time: 3
Client connected
Client connected
Received order 5: x=-2, y=-4
Received order 4: x=-1, y=4
Cook is preparing order 5. Cooking time: 1
Client connected
Received order 6: x=3, y=-2
Cook is preparing order 6. Cooking time: 4
Client connected
Received order 7: x=4, y=-1
Client connected
Received order 8: x=2, y=-6
Client connected
Received order 9: x=-2, y=7
Client connected
Received order 10: x=-3, y=1
Cook is putting order 5 into the oven.
Order 5 is ready for delivery.
Cook is preparing order 10. Cooking time: 1
Delivery thread 1 is delivering order 5. Delivery time: 4
^C
Termination signal received: 2
Most delivered orders by a delivery thread: 0 by the 6th thread

Server terminated.
==89516==
==89516== HEAP SUMMARY:
==89516==     in use at exit: 5,394 bytes in 19 blocks
==89516==   total heap usage: 402 allocs, 383 frees, 205,108 bytes allocated
==89516==
==89516== LEAK SUMMARY:
==89516==    definitely lost: 0 bytes in 0 blocks
==89516==    indirectly lost: 0 bytes in 0 blocks
==89516==      possibly lost: 3,344 bytes in 11 blocks
==89516==    still reachable: 2,006 bytes in 7 blocks
==89516==         suppressed: 44 bytes in 1 blocks
==89516== Rerun with --leak-check=full to see details of leaked memory
==89516==
==89516== For lists of detected and suppressed errors, rerun with: -s
==89516== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

9. Client with 500 orders (Client - Server)


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
Client 304 finished
Order 10 delivered to (-5, 4).
Client 9 finished
Order 14 delivered to (0, 7).
Client 378 finished
Order 6 delivered to (-3, -2).
Client 5 finished
Order 9 delivered to (4, -7).
Client 8 finished
Order 7 delivered to (4, 4).
Client 6 finished
Order 28 delivered to (2, 3).
Client 26 finished
Order 15 delivered to (3, -3).
Client 488 finished
Order 26 delivered to (-4, 1).
Client 43 finished
Order 11 delivered to (4, -6).
Client 10 finished
Order 8 delivered to (-3, 5).
Client 7 finished
Order 147 delivered to (1, -2).
Client 294 finished
Order 72 delivered to (-2, 3).
Client 452 finished
Order 68 delivered to (1, -4).
Client 300 finished
Order 71 delivered to (3, -6).
Client 429 finished
Order 145 delivered to (4, 2).
Client 389 finished
Order 115 delivered to (-5, -2).
Client 242 finished
Order 153 delivered to (-3, -4).
Client 218 finished
Order 150 delivered to (1, -6).
Order 148 delivered to (-3, -6).
Client 370 finished
Client 410 finished
Order 151 delivered to (-2, -6).
Client 318 finished
Order 152 delivered to (1, 6).
Client 32 finished
Order 413 delivered to (-2, -1).
Client 114 finished
Order 363 delivered to (0, 2).
Client 367 finished
Order 235 delivered to (-5, 2).
Client 379 finished
Order 219 delivered to (-5, 8).
Client 356 finished
Order 412 delivered to (0, 4).
Client 181 finished
Order 398 delivered to (0, -1).
Client 123 finished
Order 392 delivered to (-1, 3).
Client 316 finished
Order 234 delivered to (-3, 8).
Client 464 finished
Order 410 delivered to (-2, -8).
Client 164 finished
Order 391 delivered to (-1, 5).
Client 273 finished
Order 403 delivered to (-3, -3).
Client 50 finished
Order 402 delivered to (3, 3).
Client 135 finished
Order 397 delivered to (0, -9).
Client 91 finished
Order 321 delivered to (-3, 7).
Client 189 finished
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ 
```



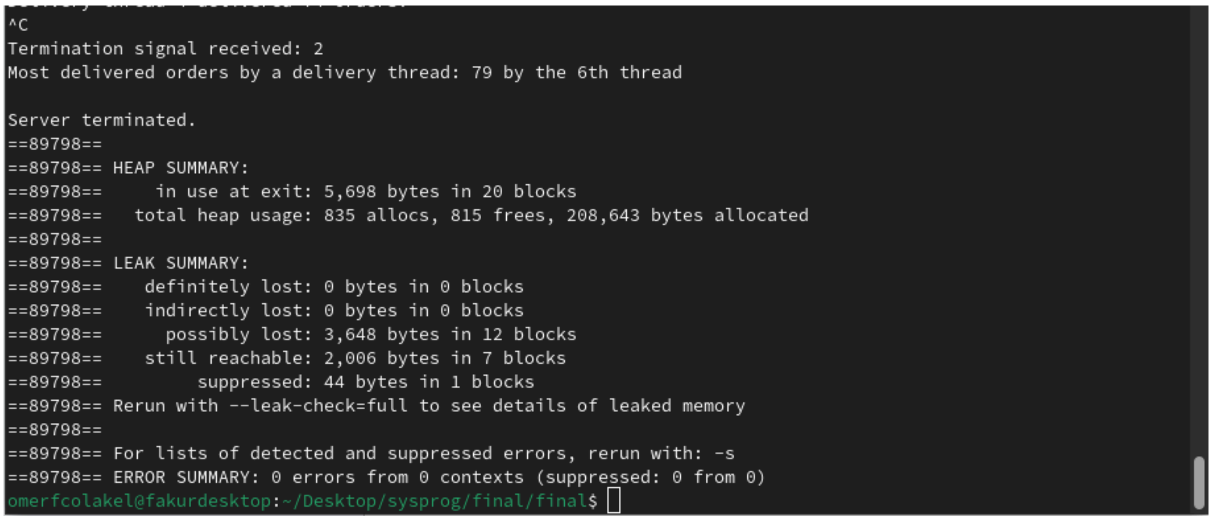
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final — valgrind ./pideshop 1234 5 6 1

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

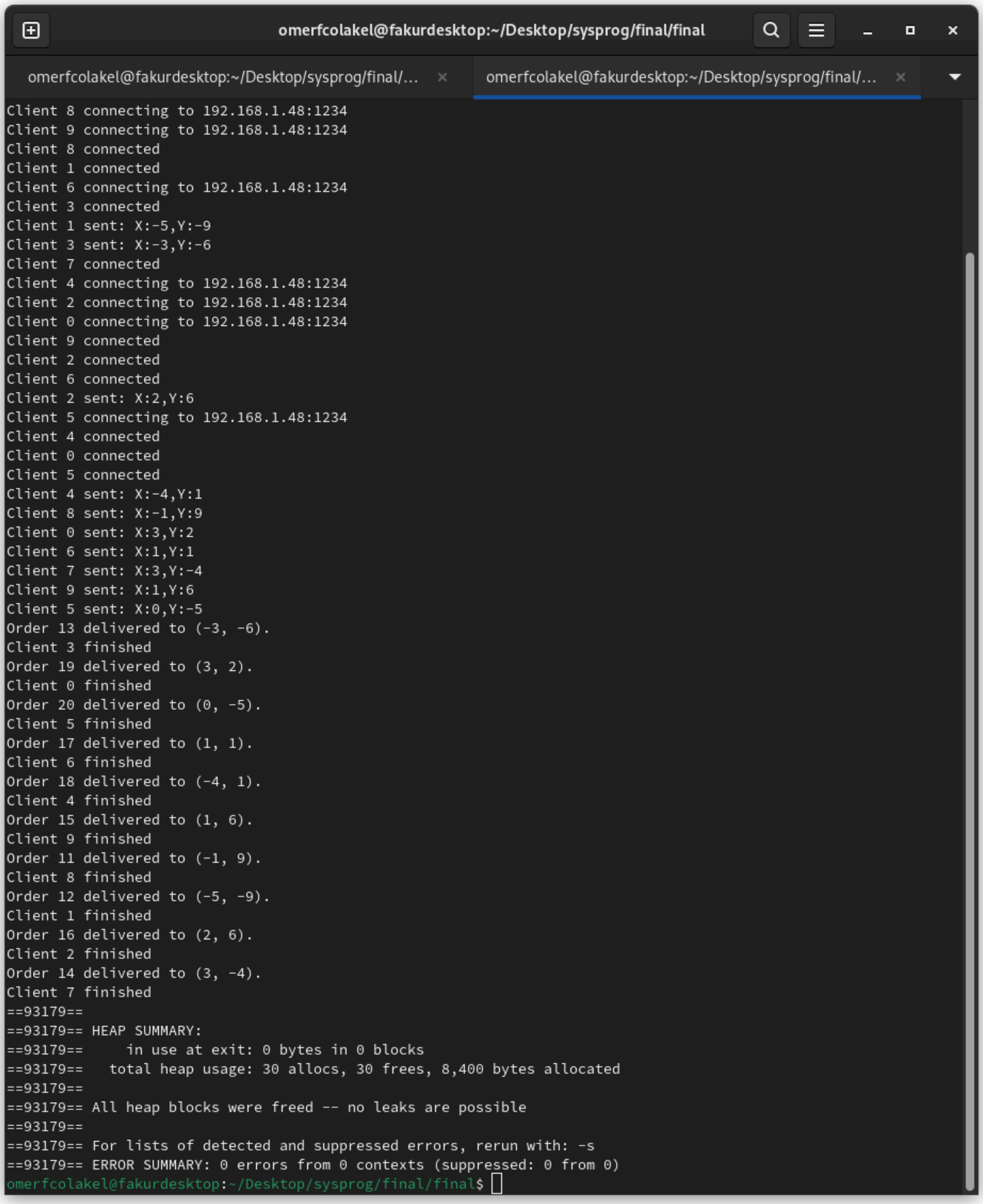
```
Order 7 is ready for delivery.
Delivery thread 4 delivered 69 orders.
Delivery thread 4 is delivering order 7. Delivery time: 5
Delivery thread 1 delivered 64 orders.
Delivery thread 1 is delivering order 6. Delivery time: 3
Delivery thread 5 delivered 73 orders.
Delivery thread 5 is delivering order 11. Delivery time: 7
Delivery thread 3 delivered 71 orders.
Delivery thread 3 is delivering order 8. Delivery time: 5
Delivery thread 0 delivered 61 orders.
Delivery thread 0 is delivering order 15. Delivery time: 4
Delivery thread 1 delivered 65 orders.
Delivery thread 1 is delivering order 28. Delivery time: 3
Delivery thread 2 delivered 66 orders.
Delivery thread 2 is delivering order 26. Delivery time: 4
Delivery thread 4 delivered 70 orders.
Delivery thread 4 is delivering order 71. Delivery time: 6
Delivery thread 1 delivered 66 orders.
Delivery thread 1 is delivering order 68. Delivery time: 4
Delivery thread 0 delivered 62 orders.
Delivery thread 0 is delivering order 72. Delivery time: 3
Delivery thread 2 delivered 67 orders.
Delivery thread 2 is delivering order 115. Delivery time: 5
Delivery thread 5 delivered 74 orders.
Delivery thread 5 is delivering order 145. Delivery time: 4
Delivery thread 3 delivered 72 orders.
Delivery thread 3 is delivering order 147. Delivery time: 2
Delivery thread 3 delivered 73 orders.
Delivery thread 3 is delivering order 150. Delivery time: 6
Delivery thread 0 delivered 63 orders.
Delivery thread 0 is delivering order 148. Delivery time: 6
Delivery thread 1 delivered 67 orders.
Delivery thread 1 is delivering order 151. Delivery time: 6
Delivery thread 4 delivered 71 orders.
Delivery thread 4 is delivering order 153. Delivery time: 5
Delivery thread 5 delivered 75 orders.
Delivery thread 5 is delivering order 152. Delivery time: 6
Delivery thread 2 delivered 68 orders.
Delivery thread 2 is delivering order 219. Delivery time: 9
Delivery thread 4 delivered 72 orders.
Delivery thread 4 is delivering order 234. Delivery time: 8
Delivery thread 3 delivered 74 orders.
Delivery thread 1 delivered 68 orders.
Delivery thread 1 is delivering order 235. Delivery time: 5
Delivery thread 3 is delivering order 410. Delivery time: 8
Delivery thread 0 delivered 64 orders.
Delivery thread 0 is delivering order 413. Delivery time: 2
Delivery thread 5 delivered 76 orders.
Delivery thread 5 is delivering order 412. Delivery time: 4
Delivery thread 0 delivered 65 orders.
Delivery thread 0 is delivering order 363. Delivery time: 2
Delivery thread 0 delivered 66 orders.
Delivery thread 0 is delivering order 392. Delivery time: 3
Delivery thread 1 delivered 69 orders.
Delivery thread 1 is delivering order 391. Delivery time: 5
Delivery thread 2 delivered 69 orders.
Delivery thread 2 is delivering order 397. Delivery time: 9
Delivery thread 5 delivered 77 orders.
Delivery thread 5 is delivering order 398. Delivery time: 1
Delivery thread 5 delivered 78 orders.
Delivery thread 5 is delivering order 403. Delivery time: 4
Delivery thread 0 delivered 67 orders.
Delivery thread 0 is delivering order 402. Delivery time: 4
Delivery thread 4 delivered 73 orders.
Delivery thread 4 is delivering order 321. Delivery time: 7
Delivery thread 3 delivered 75 orders.
Delivery thread 1 delivered 70 orders.
Delivery thread 5 delivered 79 orders.
Delivery thread 0 delivered 68 orders.
Delivery thread 2 delivered 70 orders.
Delivery thread 4 delivered 74 orders.
```

10. Valgrind results for the server



```
^C
Termination signal received: 2
Most delivered orders by a delivery thread: 79 by the 6th thread

Server terminated.
==89798==
==89798== HEAP SUMMARY:
==89798==     in use at exit: 5,698 bytes in 20 blocks
==89798==   total heap usage: 835 allocs, 815 frees, 208,643 bytes allocated
==89798==
==89798== LEAK SUMMARY:
==89798==    definitely lost: 0 bytes in 0 blocks
==89798==    indirectly lost: 0 bytes in 0 blocks
==89798==      possibly lost: 3,648 bytes in 12 blocks
==89798==    still reachable: 2,006 bytes in 7 blocks
==89798==         suppressed: 44 bytes in 1 blocks
==89798== Rerun with --leak-check=full to see details of leaked memory
==89798==
==89798== For lists of detected and suppressed errors, rerun with: -s
==89798== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

11. Valgrind result of a client ended by receiving all the orders

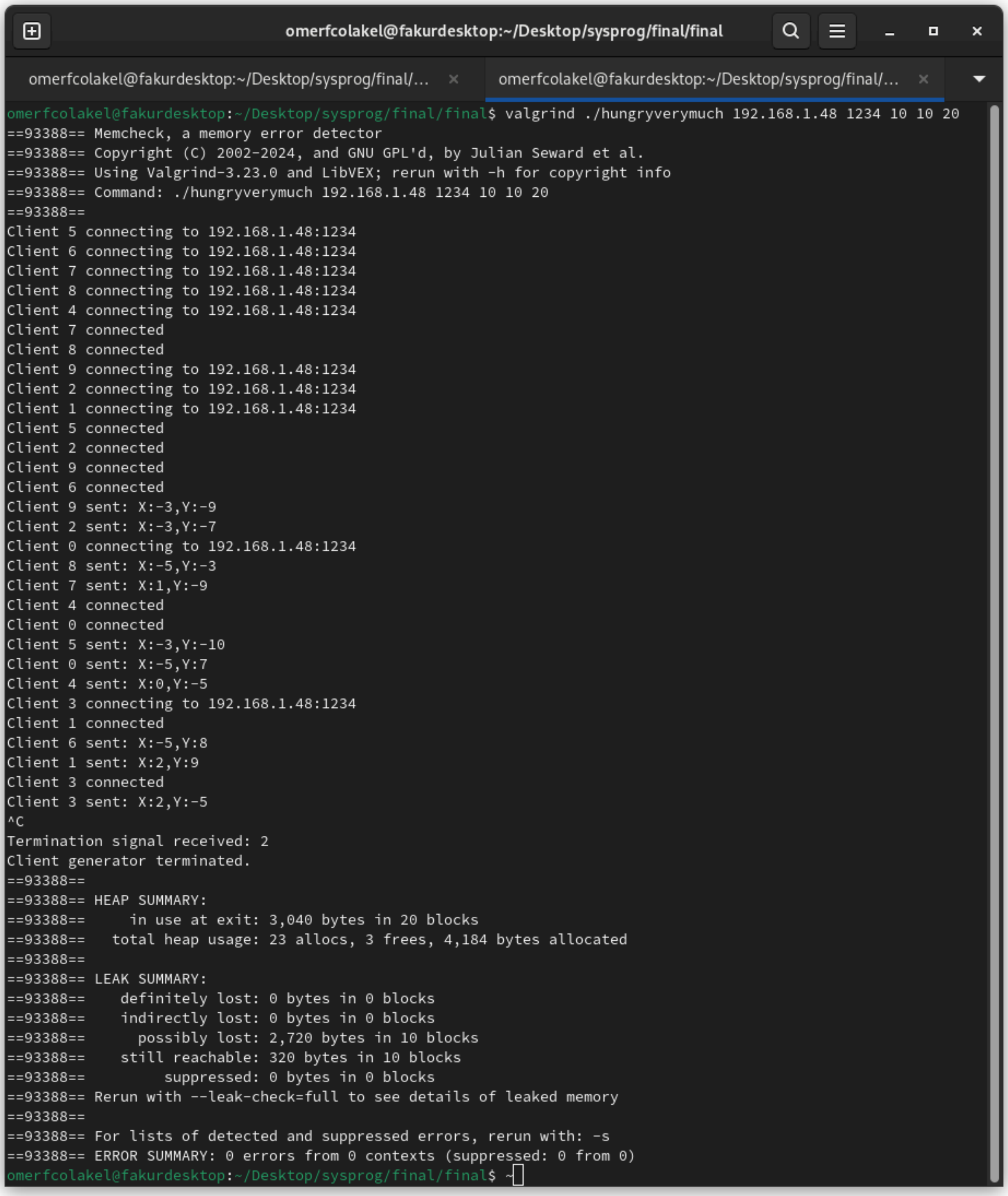


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/... omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/...

```
Client 8 connecting to 192.168.1.48:1234
Client 9 connecting to 192.168.1.48:1234
Client 8 connected
Client 1 connected
Client 6 connecting to 192.168.1.48:1234
Client 3 connected
Client 1 sent: X:-5,Y:-9
Client 3 sent: X:-3,Y:-6
Client 7 connected
Client 4 connecting to 192.168.1.48:1234
Client 2 connecting to 192.168.1.48:1234
Client 0 connecting to 192.168.1.48:1234
Client 9 connected
Client 2 connected
Client 6 connected
Client 2 sent: X:2,Y:6
Client 5 connecting to 192.168.1.48:1234
Client 4 connected
Client 0 connected
Client 5 connected
Client 4 sent: X:-4,Y:1
Client 8 sent: X:-1,Y:9
Client 0 sent: X:3,Y:2
Client 6 sent: X:1,Y:1
Client 7 sent: X:3,Y:-4
Client 9 sent: X:1,Y:6
Client 5 sent: X:0,Y:-5
Order 13 delivered to (-3, -6).
Client 3 finished
Order 19 delivered to (3, 2).
Client 0 finished
Order 20 delivered to (0, -5).
Client 5 finished
Order 17 delivered to (1, 1).
Client 6 finished
Order 18 delivered to (-4, 1).
Client 4 finished
Order 15 delivered to (1, 6).
Client 9 finished
Order 11 delivered to (-1, 9).
Client 8 finished
Order 12 delivered to (-5, -9).
Client 1 finished
Order 16 delivered to (2, 6).
Client 2 finished
Order 14 delivered to (3, -4).
Client 7 finished
==93179==
==93179== HEAP SUMMARY:
==93179==     in use at exit: 0 bytes in 0 blocks
==93179==   total heap usage: 30 allocs, 30 frees, 8,400 bytes allocated
==93179==
==93179== All heap blocks were freed -- no leaks are possible
==93179==
==93179== For lists of detected and suppressed errors, rerun with: -s
==93179== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$
```

12. Valgrind result of a client ended by signal interrupt


omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final

omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/... ×
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/... ×

```
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ valgrind ./hungryverymuch 192.168.1.48 1234 10 10 20
==93388== Memcheck, a memory error detector
==93388== Copyright (C) 2002-2024, and GNU GPL'd, by Julian Seward et al.
==93388== Using Valgrind-3.23.0 and LibVEX; rerun with -h for copyright info
==93388== Command: ./hungryverymuch 192.168.1.48 1234 10 10 20
==93388==
Client 5 connecting to 192.168.1.48:1234
Client 6 connecting to 192.168.1.48:1234
Client 7 connecting to 192.168.1.48:1234
Client 8 connecting to 192.168.1.48:1234
Client 4 connecting to 192.168.1.48:1234
Client 7 connected
Client 8 connected
Client 9 connecting to 192.168.1.48:1234
Client 2 connecting to 192.168.1.48:1234
Client 1 connecting to 192.168.1.48:1234
Client 5 connected
Client 2 connected
Client 9 connected
Client 6 connected
Client 9 sent: X:-3,Y:-9
Client 2 sent: X:-3,Y:-7
Client 0 connecting to 192.168.1.48:1234
Client 8 sent: X:-5,Y:-3
Client 7 sent: X:1,Y:-9
Client 4 connected
Client 0 connected
Client 5 sent: X:-3,Y:-10
Client 0 sent: X:-5,Y:7
Client 4 sent: X:0,Y:-5
Client 3 connecting to 192.168.1.48:1234
Client 1 connected
Client 6 sent: X:-5,Y:8
Client 1 sent: X:2,Y:9
Client 3 connected
Client 3 sent: X:2,Y:-5
^C
Termination signal received: 2
Client generator terminated.
==93388==
==93388== HEAP SUMMARY:
==93388==     in use at exit: 3,040 bytes in 20 blocks
==93388==   total heap usage: 23 allocs, 3 frees, 4,184 bytes allocated
==93388==
==93388== LEAK SUMMARY:
==93388==    definitely lost: 0 bytes in 0 blocks
==93388==    indirectly lost: 0 bytes in 0 blocks
==93388==      possibly lost: 2,720 bytes in 10 blocks
==93388==    still reachable: 320 bytes in 10 blocks
==93388==         suppressed: 0 bytes in 0 blocks
==93388== Rerun with --leak-check=full to see details of leaked memory
==93388==
==93388== For lists of detected and suppressed errors, rerun with: -s
==93388== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
omerfcolakel@fakurdesktop:~/Desktop/sysprog/final/final$ 
```