

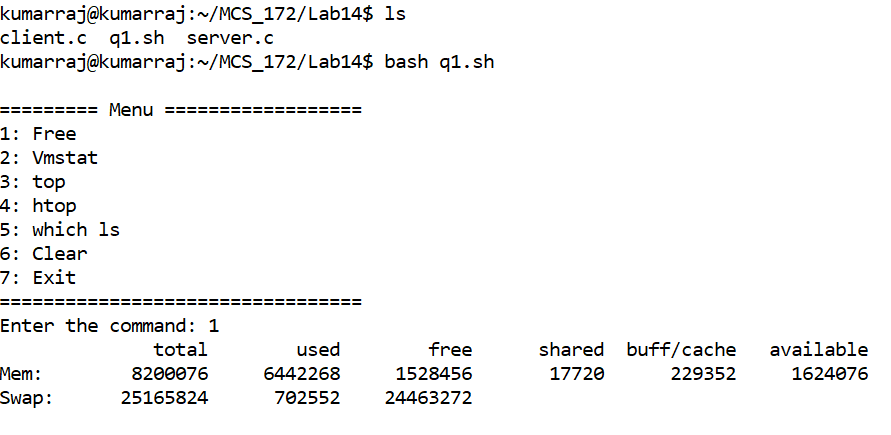
**Reg.No : 2047120**

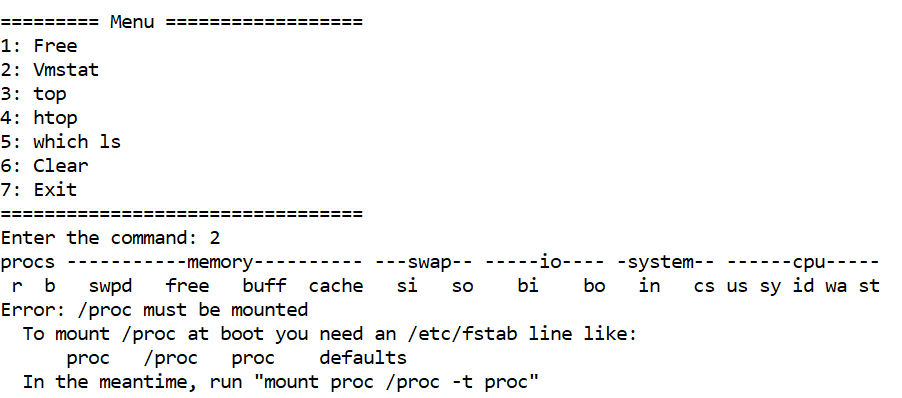
**Course : MCS 172 – Lab 14**

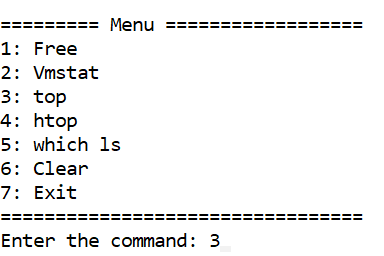
**: Rajkumar B L**

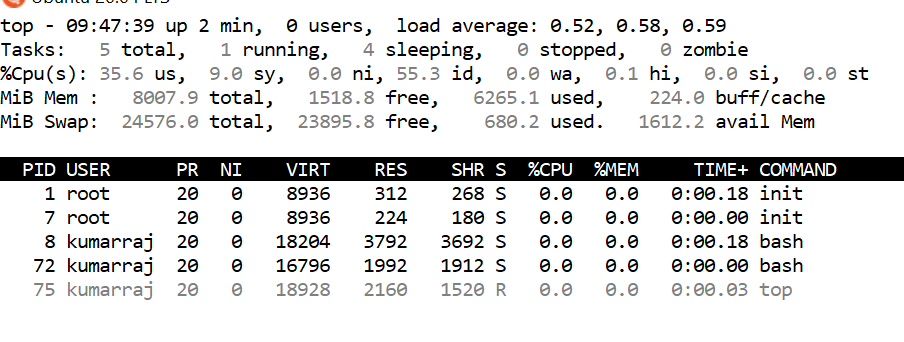
**Name**

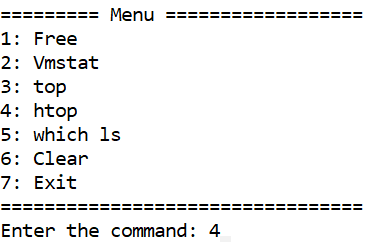
Question 01: Shell Commands

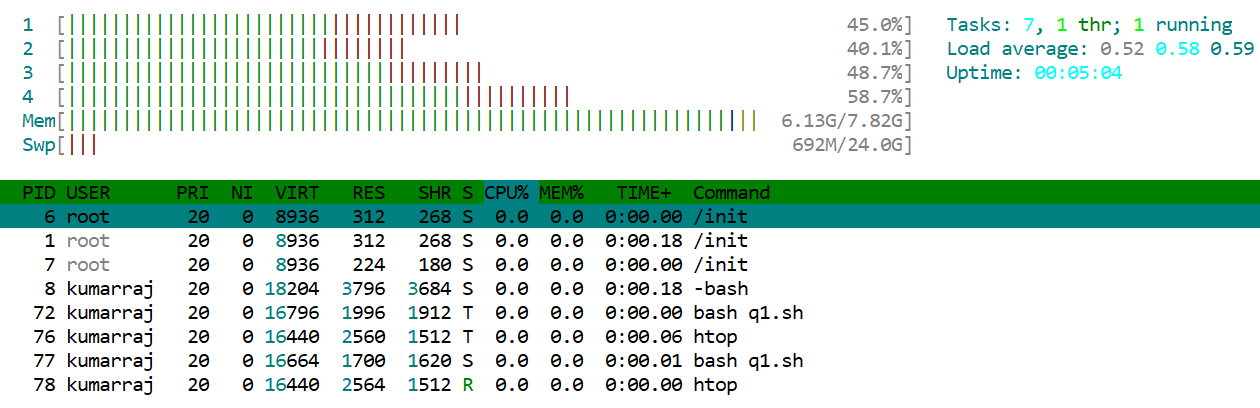


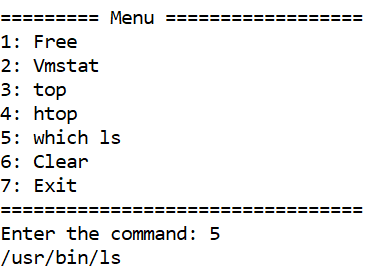


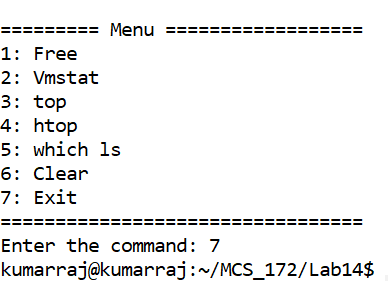




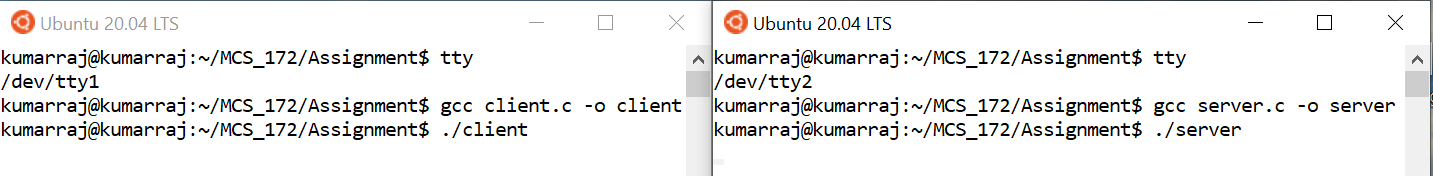


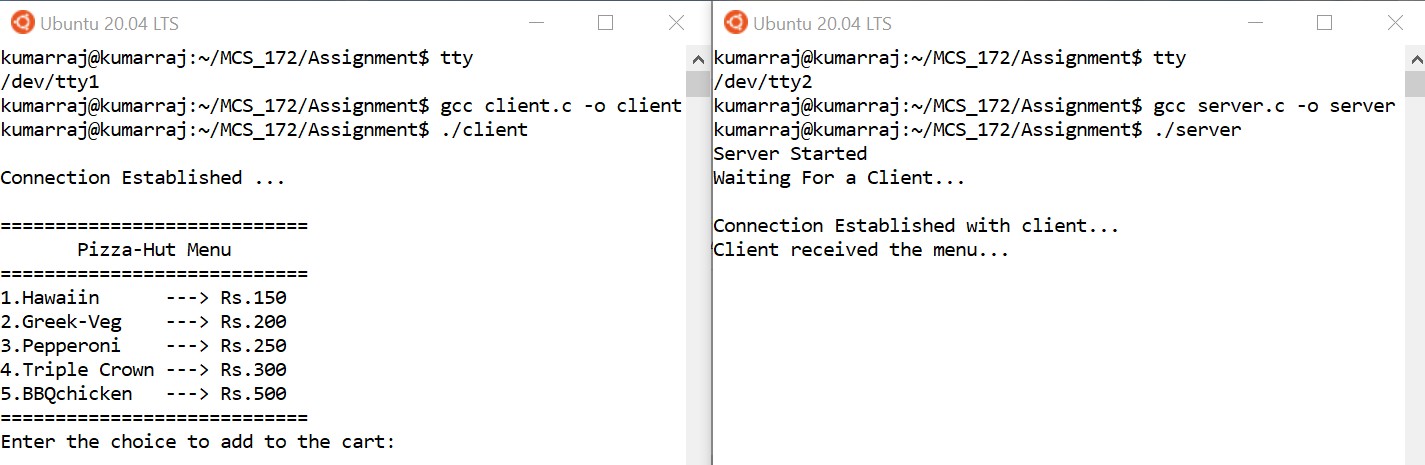
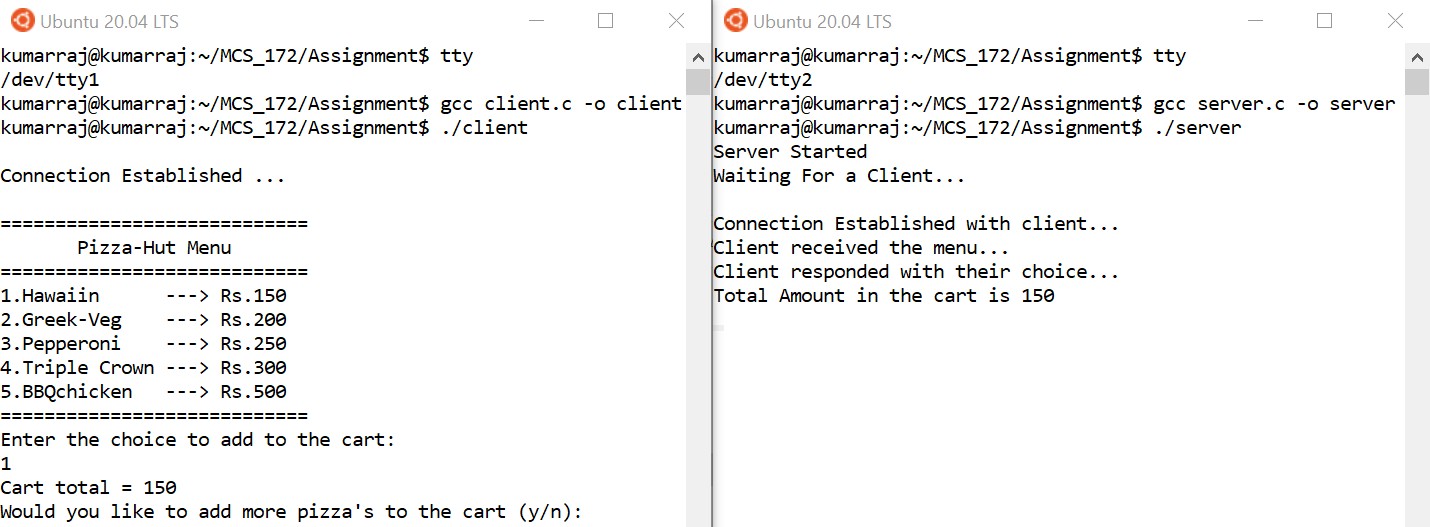


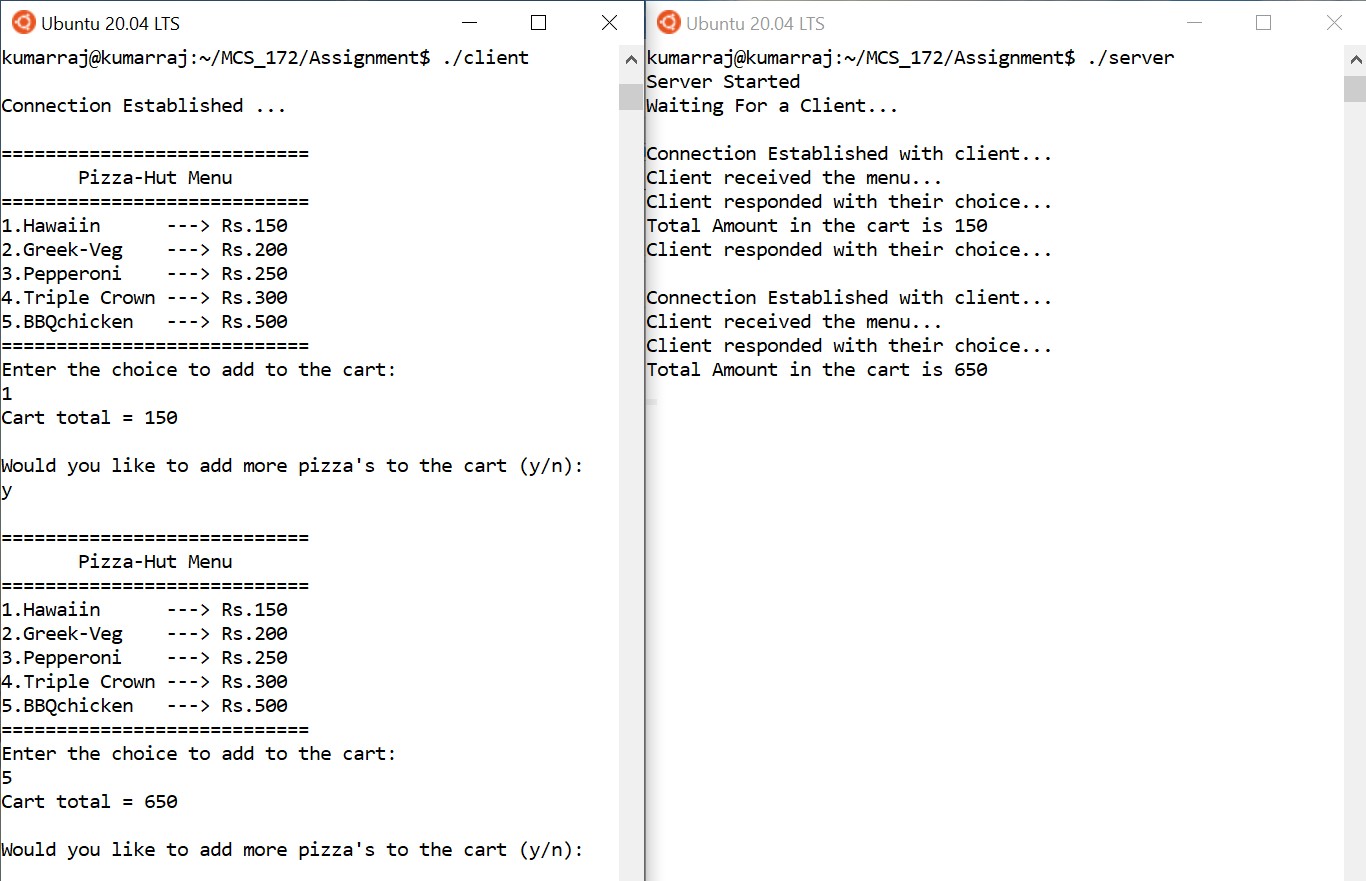




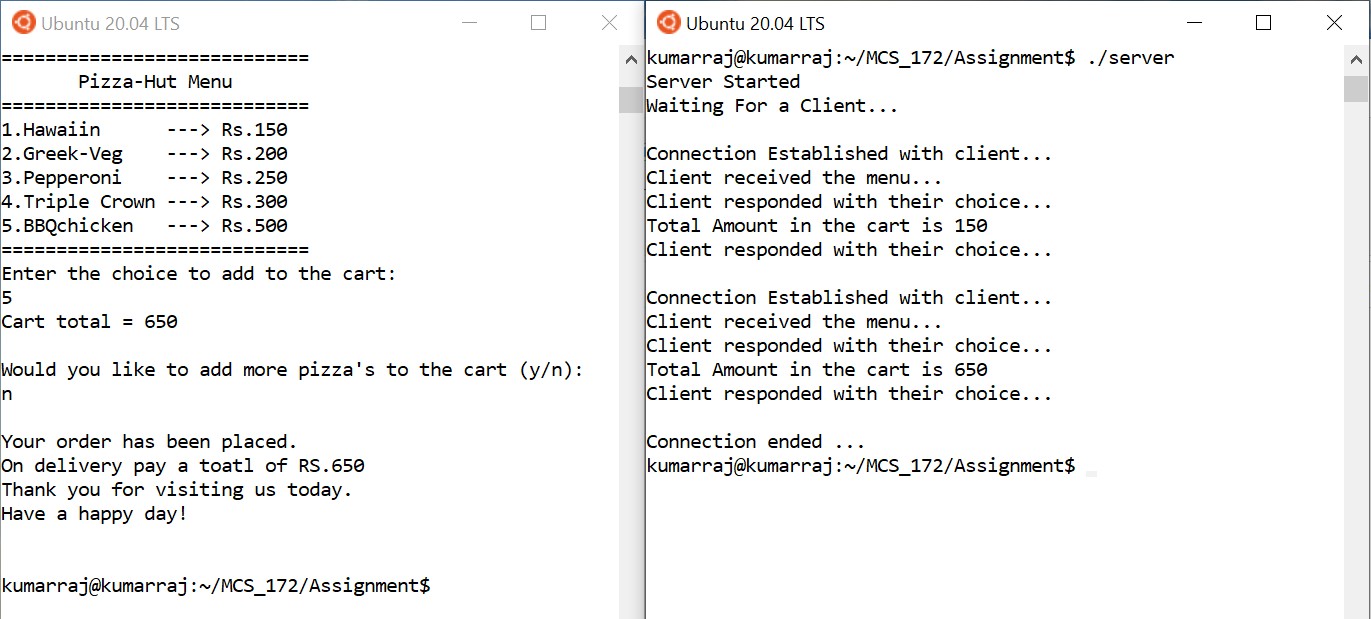
Question 02: Socket Inter-Process Communication







1. After adding desired pizzas to the cart, the client now responds to the server as no. The server accepts the client's response and replies with an endnote stating the client to be ready with the cart total at the time of delivery. With this, the connection comes to an end.



**Code for Client.c and Server.c is attached below: -**

1 /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* 1. \* MCS 172 - Assignment
  2. \* Filename : server.c
  3. \* Author : Rajkumar B L 5 \* Reg.No : 2047120

6 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

1. #include <unistd.h>
2. #include <stdio.h>
3. #include <sys/socket.h>
4. #include <stdlib.h>
5. #include <netinet/in.h>
6. #include <string.h>
7. #include <stdbool.h>
8. #define PORT 8080
9. int cart(); 16

17 int main() 18 {

1. int server\_fd, new\_socket, valread;
2. char info\_to\_customer[250];
3. struct sockaddr\_in address;
4. int opt = 1;
5. int addrlen = sizeof(address); 24 char buffer[1024] = {0};

25 bool s\_conct = true; 26

27 char menu[250] = "\n============================\n Pizza-Hut

Menu\n============================\n1.Hawaiin ---> Rs.150\n2.Greek-Veg ---> Rs.200\n3.Pepperoni ---> Rs.250\n4.Triple Crown ---> Rs.300\n5.BBQchicken ---> Rs.500\n============================";

28 int cart\_amount = 0; 29

1. // Creating socket file descriptor
2. if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0) 32 {
3. perror("socket failed");
4. exit(EXIT\_FAILURE); 35 }

36

1. // Forcefully attaching socket to the port
2. if (setsockopt(server\_fd, SOL\_SOCKET, SO\_REUSEADDR | SO\_REUSEPORT, &opt, sizeof(opt)))

39 {

1. perror("setsockopt");
2. exit(EXIT\_FAILURE); 42 }
3. address.sin\_family = AF\_INET;
4. address.sin\_addr.s\_addr = INADDR\_ANY;
5. address.sin\_port = htons(PORT); 46
6. // Forcefully attaching socket to the port
7. if (bind(server\_fd, (struct sockaddr \*)&address, sizeof(address)) < 0) 49 {
8. perror("bind failed");
9. exit(EXIT\_FAILURE); 52 }

53 if (listen(server\_fd, 3) < 0) 54 {

1. perror("listen");
2. exit(EXIT\_FAILURE);

57 }

58 if ((new\_socket = accept(server\_fd, (struct sockaddr \*)&address,(socklen\_t

\*)&addrlen)) < 0)

59 {

1. perror("accept");
2. exit(EXIT\_FAILURE); 62 }

63

1. printf("Server Started\n");
2. printf("Waiting For a Client...\n"); 66

67 while (s\_conct)

68 {

1. char ch;
2. //Sending Menu to the client
3. send(new\_socket, menu, strlen(menu), 0); //Send 1 Menu 72
4. valread = read(new\_socket, buffer, 1024); // Read 1 if the menu is received
5. printf("\nConnection Established with client... \n");
6. printf("%s\n", buffer);
7. memset(buffer, 0, strlen(buffer)); 77
8. //Ask the user to enter the choice
9. memset(info\_to\_customer, 0, strlen(info\_to\_customer));
10. strcpy(info\_to\_customer, "Enter the choice to add to the cart:");
11. send(new\_socket, info\_to\_customer, strlen(info\_to\_customer), 0); //Send 2 - Enter the choice
12. memset(info\_to\_customer, 0, strlen(info\_to\_customer)); 83
13. //Read the option selected by the client
14. valread = read(new\_socket, buffer, 1024); // Read 2
15. printf("Client responded with their choice...\n"); 87
16. char ct\_amt\_st[5];
17. cart\_amount = cart\_amount + cart(buffer);
18. printf("Total Amount in the cart is %d\n", cart\_amount);
19. memset(buffer, 0, sizeof(buffer));
20. //Sending current cart total amount to client
21. strcpy(info\_to\_customer, "Cart total = ");
22. snprintf(ct\_amt\_st, 5, "%d", cart\_amount);
23. strcat(info\_to\_customer, ct\_amt\_st);
24. send(new\_socket, info\_to\_customer, strlen(info\_to\_customer), 0); //Send 3 - Sending total cart value
25. memset(info\_to\_customer, 0, sizeof(info\_to\_customer)); 98

99 //Asking customer if they want to add more pizzas to the cart

1. strcpy(info\_to\_customer, "\nWould you like to add more pizza's to the cart (y/n):");
2. send(new\_socket, info\_to\_customer, strlen(info\_to\_customer), 0); //Send 4 - Asking to add more item

|  |  |
| --- | --- |
| 102 | memset(info\_to\_customer, 0, sizeof(info\_to\_customer)); |
| 103 |  |
| 104 | //Read the option selected by the client for more pizza |
| 105 | valread = read(new\_socket, buffer, 1024); // Read 3 |
| 106 | printf("Client responded with their choice...\n"); |
| 107 | //printf("%s", buffer); |
| 108 | ch=buffer[0]; |
| 109 | if (buffer[0] == 'y') |
| 110 | { |
| 111 | s\_conct = true; |

|  |  |  |
| --- | --- | --- |
| 112 |  | memset(buffer, 0, strlen(buffer)); |
| 113 |  | } |
| 114 |  | else |
| 115 |  | { |
| 116 |  | memset(info\_to\_customer, 0, sizeof(info\_to\_customer)); |
| 117 |  | strcpy(info\_to\_customer, "\nYour order has been placed.\nOn delivery pay |
|  | a toatl | of RS."); |
| 118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157 | snprintf(ct\_amt\_st, 5, "%d", cart\_amount); strcat(info\_to\_customer, ct\_amt\_st);  send(new\_socket, info\_to\_customer, strlen(info\_to\_customer), 0); //Send end 1 - Sending total cart value  memset(info\_to\_customer, 0, sizeof(info\_to\_customer));  s\_conct = false;  strcpy(info\_to\_customer, "\nThank you for visiting us today.\nHave a happy day!");  send(new\_socket, info\_to\_customer, strlen(info\_to\_customer), 0); //Send end 2 - sending end note  memset(buffer, 0, strlen(buffer));  printf("\nConnection ended ... \n");  }  }  return 0;  }  int cart(char choice[])  {  int cost=0;  switch (choice[0])  {  case '1':  cost=150; break;  case '2':  cost = 200; break;  case '3':  cost = 250; break;  case '4':  cost = 300; break;  case '5':  cost = 500; break;  }  return cost;  } | |

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7 | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   * MCS 172 - Assignment * Filename : client.c * Author : Rajkumar B L   \* Reg.No : 2047120  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ | |
| 8 | #include <stdio.h> |  |
| 9 | #include <sys/socket.h> |  |
| 10 | #include <arpa/inet.h> |  |
| 11 | #include <unistd.h> |  |
| 12 | #include <string.h> |  |
| 13 | #include <stdbool.h> |  |
| 14 | #define PORT 8080 |  |
| 15 |  |  |
| 16 | int main() |  |
| 17 | { |  |
| 18 | int sock = 0, valread; |  |
| 19 | char info\_to\_server[100]; |  |
| 20 | struct sockaddr\_in serv\_addr; |  |
| 21 | char info\_buff[1024] = {0}; |  |
| 22 |  |  |
| 23 | bool c\_conct=true; |  |
| 24 |  |  |
| 25 | if ((sock = socket(AF\_INET, SOCK\_STREAM, 0)) < 0) |  |
| 26 | { |  |
| 27 | printf("\n Socket creation error \n"); |  |
| 28 | return -1; |  |
| 29 | } |  |
| 30 |  |  |
| 31 | serv\_addr.sin\_family = AF\_INET; |  |
| 32 | serv\_addr.sin\_port = htons(PORT); |  |
| 33 |  |  |
| 34 | // Convert IPv4 and IPv6 addresses from text to binary form |  |
| 35 | if (inet\_pton(AF\_INET, "127.0.0.1", &serv\_addr.sin\_addr) <= 0) |  |
| 36 | { |  |
| 37 | printf("\nInvalid address/ Address not supported \n"); |  |
| 38 | return -1; |  |
| 39 | } |  |
| 40 |  |  |
| 41 | if (connect(sock, (struct sockaddr \*)&serv\_addr, sizeof(serv\_addr)) | < 0) |
| 42 | { |  |
| 43 | printf("\nConnection Failed \n"); |  |
| 44 | return -1; |  |
| 45 | } |  |
| 46 | else |  |
| 47 | { |  |
| 48 | printf("\nConnection Established ... \n"); |  |
| 49 | } |  |
| 50 |  |  |
| 51 | while (c\_conct) |  |
| 52 | { |  |
| 53 |  |  |
| 54 | char ch; |  |
| 55 | //Menu Received |  |
| 56 | valread = read(sock, info\_buff, 1024); //Read 1 |  |
| 57 | printf("%s\n", info\_buff); |  |
| 58 | memset(info\_buff, 0, sizeof(info\_buff)); |  |
| 60 | memset(info\_to\_server, 0, sizeof(info\_to\_server)); |  |

59

/ Send 1

|  |  |  |
| --- | --- | --- |
| 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88 | strcpy(info\_to\_server, "Client received the menu...");  send(sock, info\_to\_server, strlen(info\_to\_server), 0); / memset(info\_to\_server, 0, sizeof(info\_to\_server));  //Reading form server - Asking to enter the choice valread = read(sock, info\_buff, 1024); //Read 2  printf("%s\n", info\_buff);  memset(info\_buff, 0, sizeof(info\_buff));  //Sending the Choice to the server scanf("%s",info\_to\_server);  send(sock, info\_to\_server, strlen(info\_to\_server), 0); / client choice  memset(info\_to\_server, 0, sizeof(info\_to\_server));  //Reading the total cart value  valread = read(sock, info\_buff, 1024); //Read 3 printf("%s\n", info\_buff);  memset(info\_buff, 0, sizeof(info\_buff));  //Reading - if to add more pizza to cart  valread = read(sock, info\_buff, 1024); //Read 4 printf("%s\n", info\_buff);  memset(info\_buff, 0, sizeof(info\_buff));  //Sending the more pizza Choice to the server scanf("%s", info\_to\_server);  send(sock, info\_to\_server, strlen(info\_to\_server), 0); / client choice  //printf("%s", info\_to\_server); | |
| 89 |  | ch = info\_to\_server[0]; |
| 90 |  | if (info\_to\_server[0] == 'y') |
| 91 |  | { |
| 92 |  | c\_conct = true; |
| 93 |  | memset(info\_to\_server, 0, sizeof(info\_to\_server)); |
| 94 |  | } |
| 95 |  | else |
| 96 |  | { |
| 97 |  | c\_conct = false; |
| 98 |  | //Reading - end note |
| 99 |  | valread = read(sock, info\_buff, 1024); //Read end 1 |
| 100 |  | printf("%s\n", info\_buff); |
| 101 |  | memset(info\_buff, 0, sizeof(info\_buff)); |
| 102 |  |  |
| 103 |  | valread = read(sock, info\_buff, 1024); //Read end 2 |
| 104 |  | printf("%s\n\n", info\_buff); |
| 105 |  | memset(info\_buff, 0, sizeof(info\_buff)); |
| 106 |  | memset(info\_to\_server, 0, sizeof(info\_to\_server)); |
| 107 |  | } |
| 108 |  | } |
| 109 |  |  |
| 110 |  | return 0; |
| 111 | } |  |
| 112 |  |  |

/ Send 2 - sending

/ Send 3 - sending