

# LAB ASSIGNMENT-1

## CSN-361 Computer Networks Laboratory

Submitted by - Prateek Mali  
Enrollment no. - 17114059

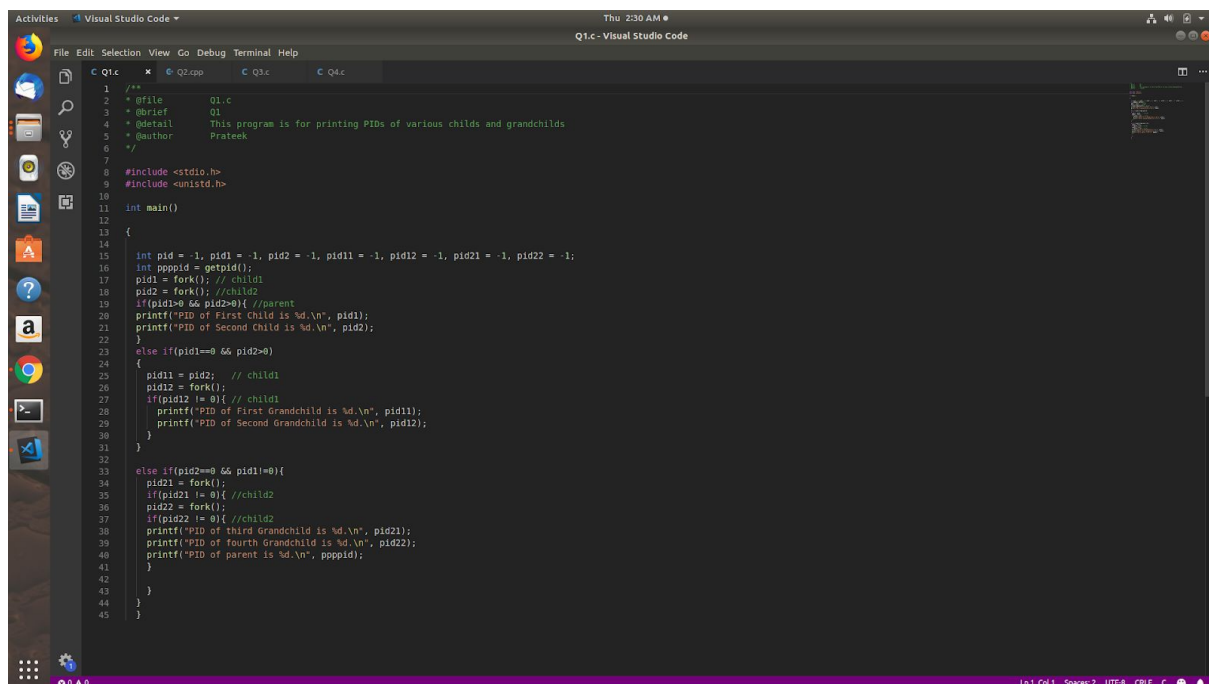
### **Problem Statements**

1. Write a C program in the UNIX system that creates two children and four grandchildren (two for each child). The program should then print the process-IDs of the two children, four grandchildren and the parent in this order.
2. Write a C++ program to print the MAC address of your computer.
3. Write your own version of ping program in C language.
4. Write a C program to find the host name from IP address.

## Implementations details (Data Structures and Algorithms)

### Q1

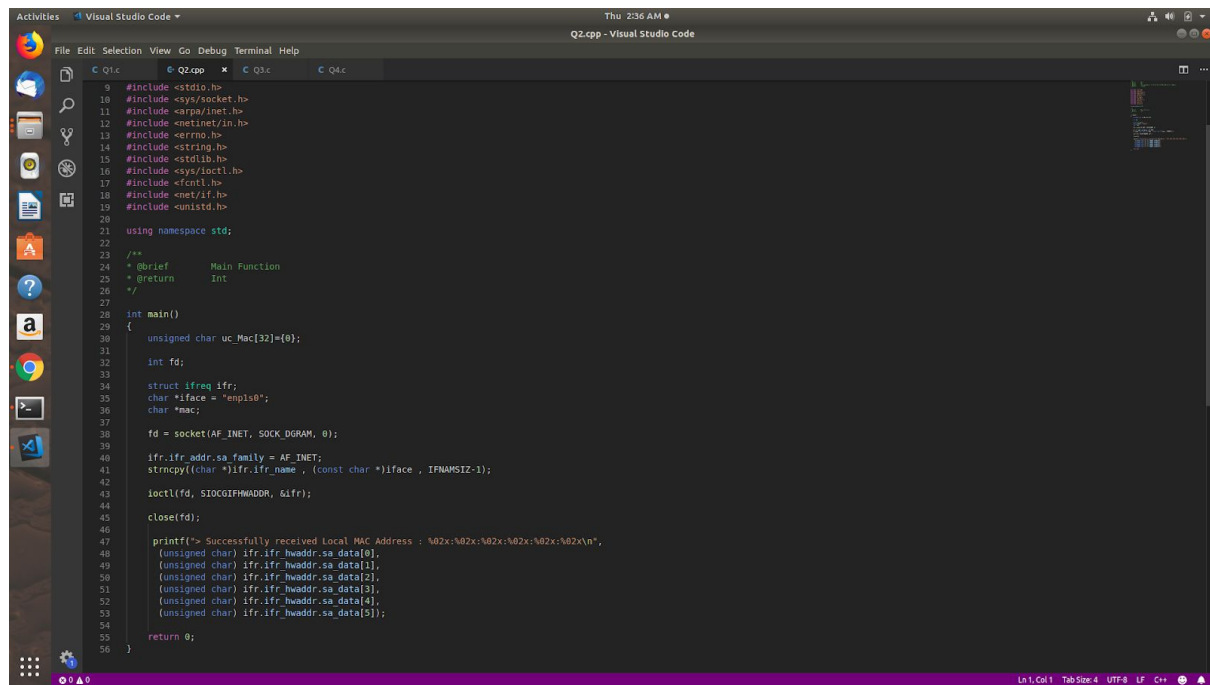
- Using fork we have created the childs and granchilds of the main process and have used the function getpid() to get the pid of parent.
- Libraries are used and no noticable data structure is used.



```
1  /**
2   * @file      Q1.c
3   * @brief      Q1
4   * @detail      This program is for printing PIDs of various childs and granchilds
5   * @author      Prateek
6   */
7
8  #include <stdio.h>
9  #include <unistd.h>
10
11  int main()
12  {
13      {
14          int pid = -1, pid1 = -1, pid2 = -1, pid11 = -1, pid12 = -1, pid21 = -1, pid22 = -1;
15          int pppid = getpid();
16          pid1 = fork(); // child1
17          pid2 = fork(); // child2
18          if(pid1 <= 0 && pid2 >= 0){ //parent
19              printf("PID of First Child is %d.\n", pid1);
20              printf("PID of Second Child is %d.\n", pid2);
21          }
22          else if(pid1 == 0 && pid2 > 0){
23              {
24                  pid11 = pid2; // child1
25                  pid12 = fork();
26                  if(pid12 != 0){ // child1
27                      printf("PID of First Grandchild is %d.\n", pid11);
28                      printf("PID of Second Grandchild is %d.\n", pid12);
29                  }
30              }
31          }
32          else if(pid2 == 0 && pid1 != 0){
33              pid21 = fork();
34              if(pid21 != 0){ //child2
35                  pid22 = fork();
36                  if(pid22 != 0){ //child2
37                      printf("PID of third Grandchild is %d.\n", pid21);
38                      printf("PID of fourth Grandchild is %d.\n", pid22);
39                      printf("PID of parent is %d.\n", pppid);
40                  }
41              }
42          }
43      }
44  }
```

### Q2

- Struct is the data structure which is used here.
- Interface id is defined.
- Socket is created.
- Mac Address is fetched in blocks but represented as a whole in the end.



```
1 #include <stdio.h>
2 #include <sys/socket.h>
3 #include <arpa/inet.h>
4 #include <netinet/in.h>
5 #include <errno.h>
6 #include <string.h>
7 #include <stdlib.h>
8 #include <sys/ioctl.h>
9 #include <fcntl.h>
10 #include <net/if.h>
11 #include <unistd.h>
12
13 using namespace std;
14
15 /**
16  * @brief Main Function
17  * @return Int
18  */
19
20 int main()
21 {
22     unsigned char uc_Mac[32]={0};
23
24     int fd;
25
26     struct ifreq ifr;
27     char *iface = "enp1s0";
28     char *mac;
29
30     fd = socket(AF_INET, SOCK_DGRAM, 0);
31
32     ifr.ifr_addr.sa_family = AF_INET;
33     strncpy((char *)ifr.ifr_name, (const char *)iface, IFNAMSIZ-1);
34
35     ioctl(fd, SIOCGIFHWADDR, &ifr);
36
37     close(fd);
38
39     printf("> Successfully received Local MAC Address : %02x:%02x:%02x:%02x:%02x:%02x\n",
40         (unsigned char) ifr.ifr_hwaddr.sa_data[0],
41         (unsigned char) ifr.ifr_hwaddr.sa_data[1],
42         (unsigned char) ifr.ifr_hwaddr.sa_data[2],
43         (unsigned char) ifr.ifr_hwaddr.sa_data[3],
44         (unsigned char) ifr.ifr_hwaddr.sa_data[4],
45         (unsigned char) ifr.ifr_hwaddr.sa_data[5]);
46
47     return 0;
48 }
```

### Q3

- Run `dsn_lookup()` by providing it the host address and retrieve the IP.
- Create socket.
- Exit if it fails to create socket.
- Create the ICMP struct header.
- Send Ping using `sendto` function.
- Reading and displaying of the response.
- **Struct is used as the main data structure here.**

### Code Snippets

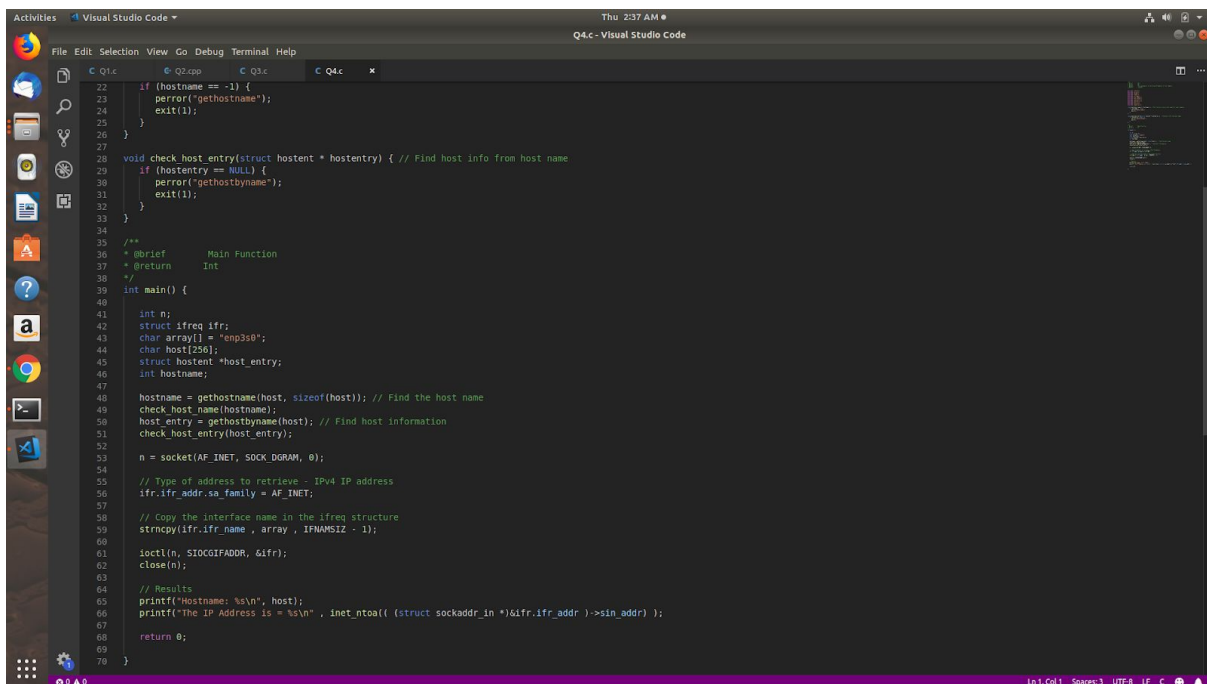
```
Activities Visual Studio Code Thu 2:48 AM Q3.c - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
C Q1.c Q2.c Q3.c Q4.c
9 #include <stdio.h>
10 #include <arpa/inet.h>
11 #include <sys/socket.h>
12 #include <fcntl.h>
13 #include <unistd.h>
14 #include <netdb.h>
15 #include <string.h>
16
17 /**
18  * @param address address of the website to ping
19  * @return IP address
20  */
21 char *dns_lookup(char *addr_host, struct sockaddr_in *addr_con)
22 {
23     printf("\nResolving DNS...\n");
24     struct hostent *host_entity;
25     char *ip=(char*)malloc(NI_MAXHOST*sizeof(char));
26     int i;
27     if ((host_entity = gethostbyname(addr_host)) == NULL)
28     {
29         // No IP found
30         return NULL;
31     }
32     //Filling up the address structure
33     strcpy(ip, inet_ntoa(*(struct in_addr *)
34         host_entity->h_addr));
35     (*addr_con).sin_family = host_entity->h_addrtype;
36     (*addr_con).sin_port = htons(0);
37     (*addr_con).sin_addr.s_addr = *(long*)host_entity->h_addr;
38     return ip;
39 }
40
41 /**
42  * @brief Main Function
43  * @return int
44  */
45 int main(int argc, char *argv[]) {
46     int sockfd;
47     char *ip_addr, *reverse_hostname;
48     struct sockaddr_in addr_con;
49     if (argc < 2)
50     {
51         printf("Usage: %s <address>\n", argv[0]);
52         return 0;
53     }
54     ip_addr = dns_lookup(argv[1], &addr_con);
55     // PING google.com (172.217.167.46) 56(84) bytes of data.
56     printf("\nPING '%s' IP: %s\n", argv[1], ip_addr);
57     // Creating Socket
58     int s = socket(PF_INET, SOCK_RAW, 1);
59     // Exit if the socket failed to be created
60     if (s <= 0)
61     {
62         perror("Socket Error");
63         exit(0);
64     }
65     // Create the ICMP Struct Header
66     typedef struct {
67         uint8_t type;
68         uint8_t code;
69         uint16_t chksum;
70         uint32_t data;
71     } icmp_hdr_t;
72     //Use the newly created struct to make a variable.
73     icmp_hdr_t pkt;
74     // Set the appropriate values to our struct, which is our ICMP header
75     pkt.type = 8;
76     pkt.code = 0;
77     pkt.chksum = 0xffff;
78     pkt.data = 0;
79     // Creating a IP Header from a struct that exists in another library
```

```
Activities Visual Studio Code Thu 2:49 AM Q3.c - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
C Q1.c Q2.c Q3.c Q4.c
56 struct sockaddr_in addr_con;
57
58 if (argc < 2)
59 {
60     printf("\nIncorrect Format %s <address>\n", argv[0]);
61     return 0;
62 }
63
64 ip_addr = dns_lookup(argv[1], &addr_con);
65 // PING google.com (172.217.167.46) 56(84) bytes of data.
66 printf("\nPING '%s' IP: %s\n", argv[1], ip_addr);
67 // Creating Socket
68 int s = socket(PF_INET, SOCK_RAW, 1);
69 // Exit if the socket failed to be created
70 if (s <= 0)
71 {
72     perror("Socket Error");
73     exit(0);
74 }
75 // Create the ICMP Struct Header
76 typedef struct {
77     uint8_t type;
78     uint8_t code;
79     uint16_t chksum;
80     uint32_t data;
81 } icmp_hdr_t;
82 //Use the newly created struct to make a variable.
83 icmp_hdr_t pkt;
84 // Set the appropriate values to our struct, which is our ICMP header
85 pkt.type = 8;
86 pkt.code = 0;
87 pkt.chksum = 0xffff;
88 pkt.data = 0;
89 // Creating a IP Header from a struct that exists in another library
```

```
Activities Visual Studio Code Thu 2:49 AM Q3.c - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
C Q1.c Q2.c Q3.c Q4.c
114 // Send our PING
115 int actionSendResult = sendto(s, &pkt, sizeof(pkt),
116     0, (struct sockaddr *)&addr_con, sizeof(addr_con));
117
118 // Exit the app if the option failed to be set
119 if (actionSendResult < 0)
120 {
121     perror("Ping Error");
122     exit(0);
123 }
124
125 // Prepare all the necessary variable to handle the response
126 unsigned int resAddressSize;
127 unsigned char res[30] = {};
128 struct sockaddr resAddress;
129
130 // Read the response from the remote host
131 int resresponse = recvfrom(s, res, sizeof(res), 0, &resAddress,
132     &resAddressSize);
133
134 //Display the response in its raw form (hex)
135 if (resresponse > 0)
136 {
137     printf("%d bytes from %s : %s\n", resresponse, ip_addr, argv[1]);
138     exit(0);
139 }
140 else
141 {
142     perror("Response Error");
143     exit(0);
144 }
145
146 return 0;
147 }
```

## Q4

- Struct is the data structure which is used here.
- Interface id is defined.
- Socket is created for the connection.
- One struct is used for storing the host information and the other one is used for storing the types of addresses to retrieve.



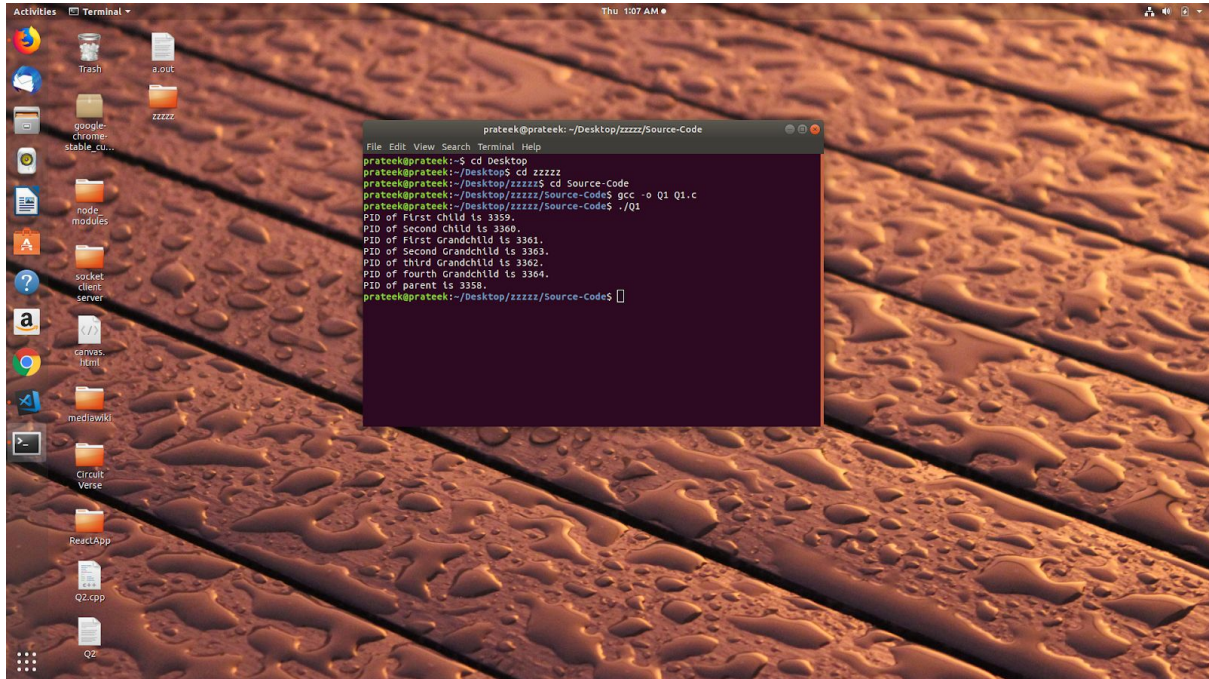
The screenshot shows a Visual Studio Code editor window with a C program. The code is as follows:

```
22 if (hostname == -1) {
23     perror("gethostname");
24     exit(1);
25 }
26
27
28 void check_host_entry(struct hostent * hostentry) { // Find host info from host name
29     if (hostentry == NULL) {
30         perror("gethostbyname");
31         exit(1);
32     }
33 }
34
35 /**
36  * @brief    Main Function
37  * @return   Int
38  */
39 int main() {
40
41     int n;
42     struct ifreq ifr;
43     char array[] = "enp3s0";
44     char host[256];
45     struct hostent *host_entry;
46     int hostname;
47
48     hostname = gethostname(host, sizeof(host)); // Find the host name
49     check_host_name(hostname);
50     host_entry = gethostbyname(host); // Find host information
51     check_host_entry(host_entry);
52
53     n = socket(AF_INET, SOCK_DGRAM, 0);
54
55     // Type of address to retrieve - IPv4 IP address
56     ifr.ifr_addr.sa_family = AF_INET;
57
58     // Copy the interface name in the ifreq structure
59     strncpy(ifr.ifr_name, array, IFNAMSIZ - 1);
60
61     ioctl(n, SIOCGIFADDR, &ifr);
62     close(n);
63
64     // Results
65     printf("Hostname: %s\n", host);
66     printf("The IP Address is %s\n", inet_ntoa((struct sockaddr_in *)&ifr.ifr_addr->sin_addr));
67
68     return 0;
69 }
70
```



# Snapshots of the running codes

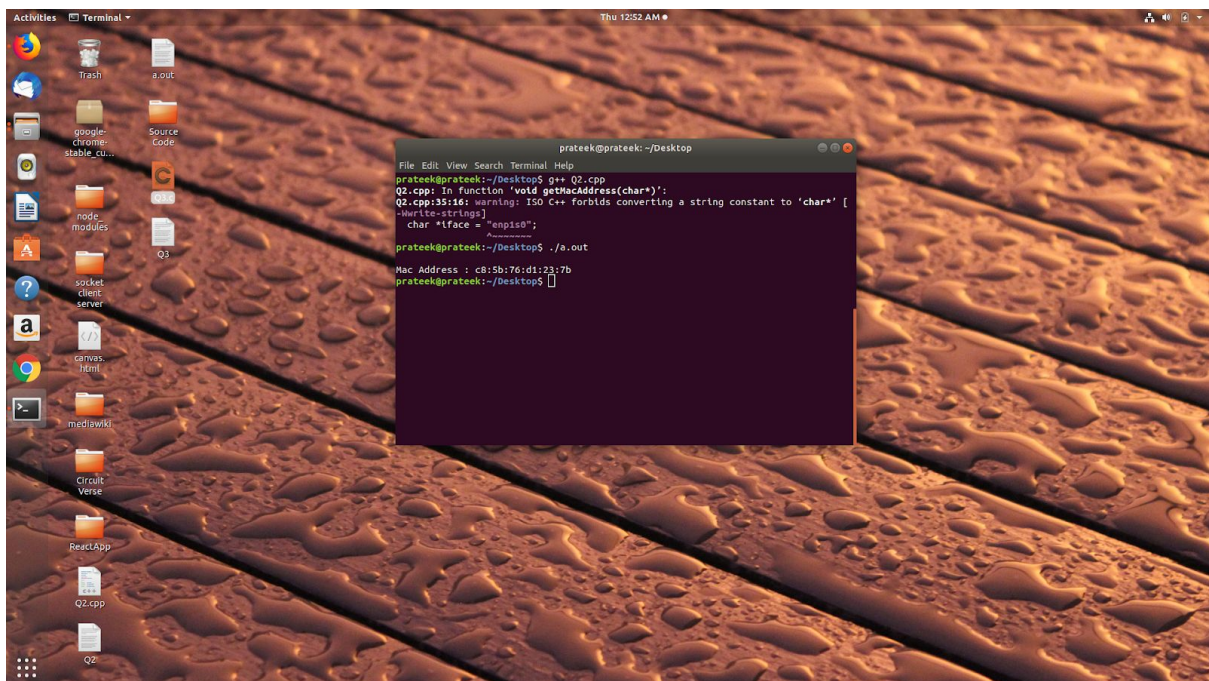
## Q1



A terminal window titled "prateek@prateek: ~/Desktop/zzzzz/Source-Code" is open on a desktop with a water droplet background. The terminal shows the following commands and output:

```
prateek@prateek:~/Desktop$ cd Desktop
prateek@prateek:~/Desktop$ cd zzzzz
prateek@prateek:~/Desktop/zzzzz$ cd Source-Code
prateek@prateek:~/Desktop/zzzzz/Source-Code$ gcc -o Q1 Q1.c
prateek@prateek:~/Desktop/zzzzz/Source-Code$ ./Q1
PID of First Child is 3359.
PID of Second Child is 3360.
PID of First Grandchild is 3361.
PID of Second Grandchild is 3363.
PID of Third Grandchild is 3362.
PID of Fourth Grandchild is 3364.
PID of parent is 3358.
prateek@prateek:~/Desktop/zzzzz/Source-Code$
```

## Q2

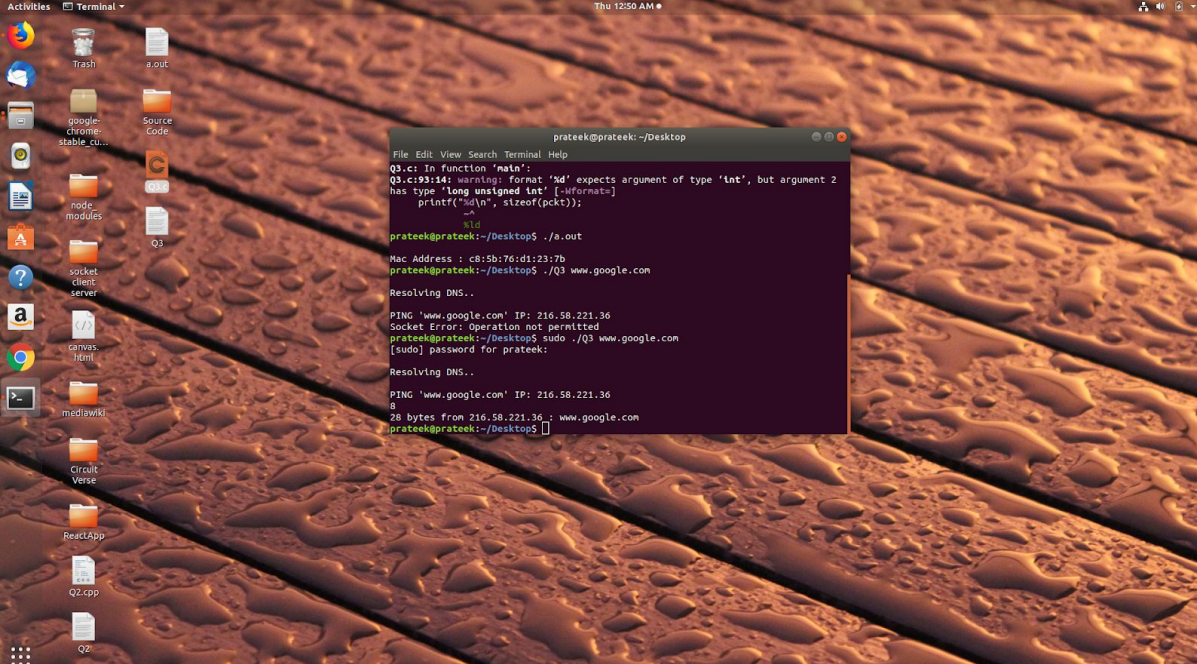


A terminal window titled "prateek@prateek: ~/Desktop" is open on the same desktop. The terminal shows the following commands and output:

```
prateek@prateek:~/Desktop$ g++ Q2.cpp
Q2.cpp: In function 'void getMacAddress(char*)':
Q2.cpp:35:16: warning: ISO C++ forbids converting a string constant to 'char*' [-Wwrite-strings]
    char *iface = "enp1s0";
prateek@prateek:~/Desktop$ ./a.out
Mac Address : c8:5b:76:d1:23:7b
prateek@prateek:~/Desktop$
```



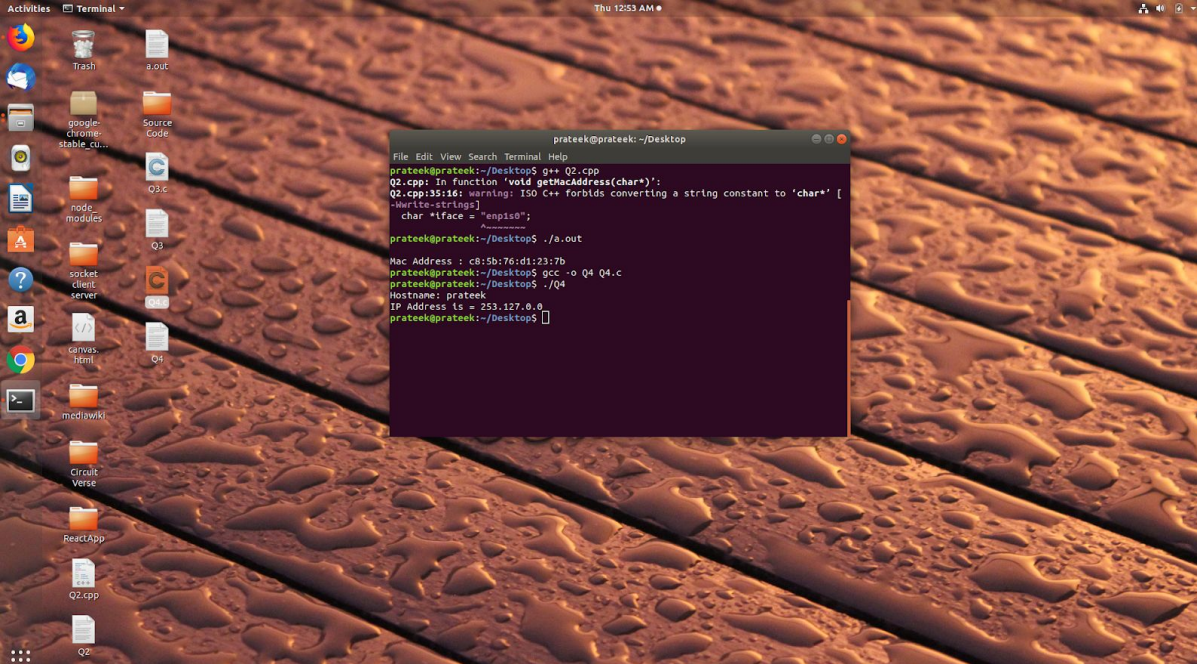
### Q3



The screenshot shows a Linux desktop with a wooden texture background. A terminal window is open, displaying the following output:

```
prateek@prateek: ~/Desktop
File Edit View Search Terminal Help
Q3.c: In function 'main':
Q3.c:13:14: warning: format '%d' expects argument of type 'int', but argument 2
      has type 'long unsigned int' [-Wformat=]
      printf("%d\n", sizeof(pkt));
              ^
              %ld
prateek@prateek:~/Desktop$ ./a.out
Mac Address : c8:5b:76:d1:23:7b
prateek@prateek:~/Desktop$ ./Q3 www.google.com
Resolving DNS..
PING 'www.google.com' IP: 216.58.221.36
Socket Error: Operation not permitted
prateek@prateek:~/Desktop$ sudo ./Q3 www.google.com
[sudo] password for prateek:
Resolving DNS..
PING 'www.google.com' IP: 216.58.221.36
6
28 bytes from 216.58.221.36 : www.google.com
prateek@prateek:~/Desktop$
```

### Q4



The screenshot shows a Linux desktop with a wooden texture background. A terminal window is open, displaying the following output:

```
prateek@prateek: ~/Desktop
File Edit View Search Terminal Help
prateek@prateek:~/Desktop$ g++ Q2.cpp
Q2.cpp: In function 'void getMacAddress(char*)':
Q2.cpp:35:16: warning: ISO C++ forbids converting a string constant to 'char*' [-Wwrite-strings]
      char "iface = 'enp1s0'";
              ^
prateek@prateek:~/Desktop$ ./a.out
Mac Address : c8:5b:76:d1:23:7b
prateek@prateek:~/Desktop$ gcc -o Q4 Q4.c
prateek@prateek:~/Desktop$ ./Q4
Hostnames: prateek
IP Address is = 253.127.0.0
prateek@prateek:~/Desktop$
```

