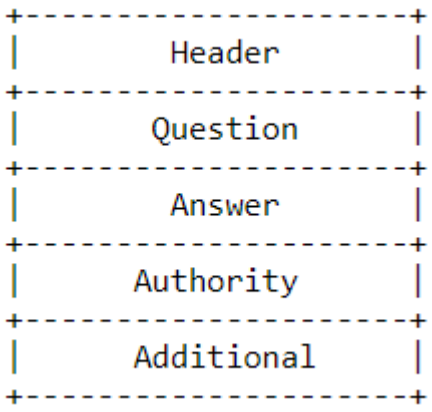
client.cpp | dns_protocol.h | server.cpp

```
91  int main(int argc, char **argv) {
92    uint16_t portno;
93    if (argc == 2) {
94      portno = (uint16_t)atoi(argv[1]);
95    }
96
97    else {
98      portno = 53;
99    }
100
101   bzero(&serv_addr, sizeof(serv_addr));
102   sockfd = socket(AF_INET, SOCK_DGRAM, 0);
103   serv_addr.sin_family = AF_INET;
104   serv_addr.sin_port = htons(portno);
105   serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
106
107   printf("--------------------start listenning--------------------\n");
108
109   if (bind(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) == -1) {
110     perror("bind failed\n");
111     exit(1);
112   }
113
114   addrlen = sizeof(serv_addr);
115
116   memset(&hints, 0, sizeof(struct addrinfo));
117   hints.ai_family = AF_INET;
118   hints.ai_socktype = SOCK_STREAM;
119   server();
120
121   freeaddrinfo(res);
122
123   return 0;
124 }
```

Server():

1. Creates message in a buffer, like:



2. Sent the whole message to a client, like:

sendto(sockfd, buffer, res_len, 0, (struct sockaddr *)&serv_addr, addrlen);

client.cpp | dns_protocol.h | server.cpp

```
86  // memcpy(server->h_addr, (char *) &serv_addr.sin_addr.s_addr, (size_t)
87  // server->h_length);
88
89  struct DNS_HEADER *dns = (struct DNS_HEADER *)buffer;
90  dns->rd = 1;
91  dns->id = htons(6);
92  dns->q_count = htons(1);
93  char *qname = (char *)(buffer + sizeof(struct DNS_HEADER));
94  int domain_len = convert_domain(domain_name, (char *)qname);
95
96  struct QUESTION *question = (struct QUESTION *)(qname + domain_len);
97  question->qtype = htons(1);
98  question->qclass = htons(1);
99
100 socklen_t addrlen = sizeof(serv_addr);
101 int query_len = sendto(sockfd, buffer, 12 + domain_len + 4, 0,
102                        (struct sockaddr *)&serv_addr, addrlen);
103 if (query_len < 0) {
104   perror("client send failed");
105   return 1;
106 }
107
```

Client():

1. Creates message in a buffer, like:

```
+--------------------+
|       Header       |
+--------------------+
|      Question      |
+--------------------+
|       Answer       |
+--------------------+
|     Authority      |
+--------------------+
|     Additional     |
+--------------------+
```

2. Sent the whole message to a server

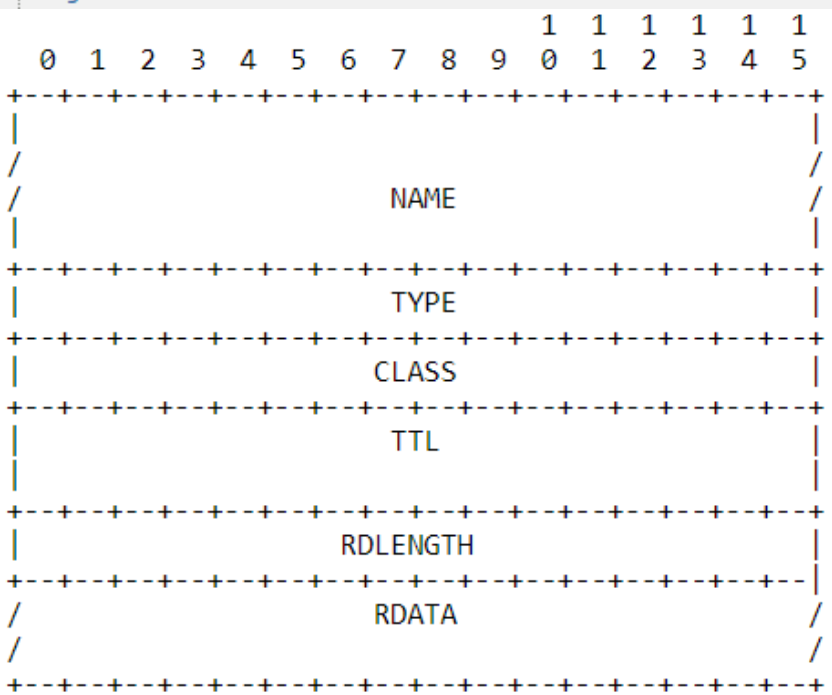
3. Reads the response message from server:

```
if (recvfrom(sockfd, buffer, 255, 0, (struct sockaddr *)&serv_addr,
        &addrlen) <= 0) {
return 1;  }
```

4. Gets the IP from the Resourse record:

```
struct R_DATA *r_data;
char *res = buffer + query_len;
```

```
    r_data = (struct R_DATA *)(res + strlen(qname) + 1);
    if (ntohs(r_data->type) == 1) {
      printf(
          "ip: %s\n",
          inet_ntop(AF_INET, res + strlen(qname) + 1 + sizeof(struct R_DATA),
                    str, sizeof(str)));
    }
```



www.yandex.ru
1 (A запись)
1 (IN, Интернет)
90
4
77.88.55.66