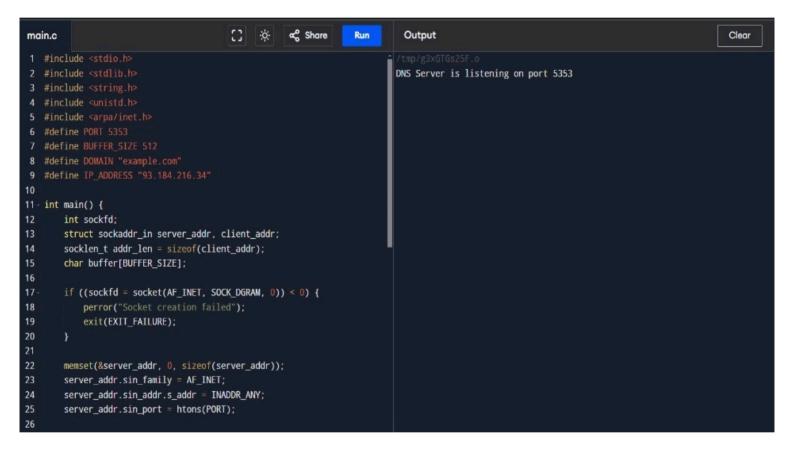
```
C3 ← α Share
                                                                                                Output
main.c
                                                                                     Run
1 #include <stdio.h>
                                                                                            ^ /tmp/BhtAnbCHAD.o
2 #include <stdlib.h>
                                                                                              Date and Time Server is listening on port 8080
3 #include <string.h>
4 #include <unistd.h>
5 #include <sys/socket.h>
 6 #include <netinet/in.h>
7 #include <time.h>
9 #define PORT 8080
10 #define BUFFER SIZE 256
11
12 - int main() {
13
       int server fd, new socket;
14
       struct sockaddr_in address;
15
      int opt = 1;
16
       int addrlen = sizeof(address);
17
       char buffer[BUFFER_SIZE];
18
       if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0) {
19+
           perror("Socket failed");
20
21
           exit(EXIT_FAILURE);
22
       }
23
24 -
       if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR, &opt, sizeof(opt))) {
25
           perror("Setsockopt failed");
26
           exit(EXIT_FAILURE);
27
28
29
       address.sin_family = AF_INET;
       address.sin_addr.s_addr = INADDR_ANY; // Listen on all interfaces
30
31
       address.sin_port = htons(PORT); // Convert port to network byte order
```



```
main.c
                                              () ×
                                                           ∝ Share
                                                                         Run
                                                                                     Output
                                                                                                                                                             Clear
                                                                                   Sent query for: google.com
   #define PORT 5353
#define DOMAIN "google.com"
#define BUFFER_SIZE 512
10
12 -
    int main() {
13
        int sockfd;
        struct sockaddr_in server_addr;
14
15
        char buffer[BUFFER_SIZE];
        if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {</pre>
18
19
20
21
             perror("Socket creation failed");
             exit(EXIT_FAILURE);
        memset(&server_addr, 0, sizeof(server_addr));
        server_addr.sin_family = AF_INET;
        server_addr.sin_port = htons(PORT);
        inet_pton(AF_INET, SERVER_IP, &server_addr.sin_addr);
```

```
main.c
                                         [] ×
                                                     ≪ Share
                                                                  Run
                                                                                                                                              Clear
                                                                            Output
        qimo->qciass - ntons(1), // ciass in
                                                                           Usage: /tmp/NVEDkntiaC.o <hostname>
        if (sendto(sock, (char *)buf, sizeof(struct DNS_HEADER) +
            (strlen((const char *)qname) + 1) + sizeof(struct QUESTION
                                                                           === Code Exited With Errors ===
            ), 0, (struct sockaddr *)&dest, sizeof(dest)) < 0) {
            perror("Query sending failed");
119
            int i = sizeof dest;
            if (recvfrom(sock, (char *)buf, 65536, 0, (struct sockaddr
                *)&dest, (socklen_t *)&i) < 0) {
                dns = (struct DNS_HEADER *) buf;
                unsigned char *reader = &buf[sizeof(struct DNS_HEADER)
                    + (strlen((const char *)qname) + 1) + sizeof(struct
                    QUESTION)];
                printf("Resolved IP Addresses:\n");
129
130
                for (i = 0; i < ntohs(dns->ans_count); i++) {
                    struct RES_RECORD answer;
                    answer.name = reader;
```