```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <time.h>
#include <sys/time.h>

#define BUFFER_SIZE 512
#define PORT 8888

void die(char* error_message){
    perror(error_message);
    exit(1);
}

int main(){
    float NUM_TESTS = 0.0;
    float NUM_WINS = 0.0;
    struct sockaddr_in server_address, client_address;
    int server_socket;
    int client_address_length = sizeof(client_address);
    int received_length;
    char buffer[BUFFER_SIZE];
    char* days_of_the_week[7] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};
    const char match_message[] = "Match";
    const char unmatch_message[] = "Unmatch";
    time_t t;
    srand((unsigned) time(&t));
    int guess;

    if((server_socket = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1){
        die("socket()");
    }
    memset((char*) &server_address, 0, sizeof(server_address));
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(PORT);
    server_address.sin_addr.s_addr = htonl(INADDR_ANY);

    if(bind(server_socket, (struct sockaddr*) &server_address, sizeof(server_address)) == -1){
        die("bind()");
    }
    while(1){
        printf("Waiting for data...\n");
        fflush(stdout);
        memset(buffer, 0, BUFFER_SIZE);
        if((received_length = recvfrom(server_socket, buffer, BUFFER_SIZE, 0, (struct sockaddr*) &client_address, &client_address_length)) == -1){
            die("recvfrom()");
        }
        buffer[received_length - 1] = '\0';
        if(strcmp(buffer, "over") == 0){
            printf("Done receiving from %s:%d\n", inet_ntoa(client_address.sin_addr), ntohs(client_address.sin_port));
            if(sendto(server_socket, "over", strlen("over"), 0, (struct sockaddr*) &client_address, client_address_length) == -1){
                die("send_to()");
            }
            continue;
        }
        if(strcmp(buffer, "all over") == 0){
            printf("Done receiving\n");
            if(sendto(server_socket, "Completed successfully", strlen("Completed successfully"), 0, (struct sockaddr*) &client_address, client_address_length) == -1){
                die("send_to()");
```

```
63              }
64              break;
65          }
66          NUM_TESTS += 1;
67          guess = rand()%6;
68          printf("Receiving from %s:%d\n", inet_ntoa(client_address.sin_addr),
    ntohs(client_address.sin_port));
69          printf("Server\tClient\tResult\n");
70
71          printf("%s\t", days_of_the_week[guess]);
72          printf("%s\t", buffer);
73          if(strcmp(buffer, days_of_the_week[guess]) == 0){
74              printf("Match!\n");
75              NUM_WINS += 1;
76              if(sendto(server_socket, match_message, strlen(match_message), 0,
    (struct sockaddr*) &client_address, client_address_length) == -1){
77                  die("send_to()");
78              }
79          }
80          else{
81              printf("Differ!.\n");
82              if(sendto(server_socket, unmatch_message, strlen(unmatch_message),
    0, (struct sockaddr*) &client_address, client_address_length) == -1){
83                  die("send_to()");
84              }
85          }
86      }
87      float percentage = NUM_WINS/NUM_TESTS*100.0;
88      printf("Match percentage = %.2f\n", percentage);
89  }
```