broadcast\_server.c  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <unistd.h>  
#include <arpa/inet.h>  
  
#define BROADCAST\_IP "192.168.0.255"  
#define BROADCAST\_PORT 8888  
#define BUFFER\_SIZE 1024  
#define SLEEP\_INTERVAL 10  
  
int main() {  
 int sockfd;  
 struct sockaddr\_in broadcast\_addr;  
 int broadcast\_enable = 1;  
 char buffer[BUFFER\_SIZE];  
  
 // Create socket  
 if ((sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0) {  
 perror("Socket creation failed");  
 exit(EXIT\_FAILURE);  
 }  
  
 // Set socket options to enable broadcasting  
 if (setsockopt(sockfd, SOL\_SOCKET, SO\_BROADCAST, &broadcast\_enable, sizeof(broadcast\_enable)) < 0) {  
 perror("Error in setting broadcast option");  
 close(sockfd);  
 exit(EXIT\_FAILURE);  
 }  
  
 // Set up the broadcast address struct  
 memset(&broadcast\_addr, 0, sizeof(broadcast\_addr));  
 broadcast\_addr.sin\_family = AF\_INET;  
 broadcast\_addr.sin\_port = htons(BROADCAST\_PORT);  
 broadcast\_addr.sin\_addr.s\_addr = inet\_addr(BROADCAST\_IP);  
  
 while (1) {  
 // Set the message to "Hello"  
 snprintf(buffer, BUFFER\_SIZE, "Hello");  
  
 // Send broadcast message  
 if (sendto(sockfd, buffer, strlen(buffer), 0, (struct sockaddr\*)&broadcast\_addr, sizeof(broadcast\_addr)) < 0) {  
 perror("Broadcast message send failed");  
 close(sockfd);  
 exit(EXIT\_FAILURE);  
 }  
  
 printf("Broadcast message sent: %s\n", buffer);  
  
 // Sleep for the defined interval  
 sleep(SLEEP\_INTERVAL);  
 }  
  
 // Close the socket  
 close(sockfd);  
  
 return 0;  
}

udp\_client.c

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <unistd.h>  
#include <arpa/inet.h>  
  
#define BROADCAST\_PORT 8888  
#define BUFFER\_SIZE 1024  
  
int main() {  
 int sockfd;  
 struct sockaddr\_in listen\_addr;  
 char buffer[BUFFER\_SIZE];  
 ssize\_t num\_bytes\_received;  
 socklen\_t addr\_len = sizeof(listen\_addr);  
  
 // Create socket  
 if ((sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0) {  
 perror("Socket creation failed");  
 exit(EXIT\_FAILURE);  
 }  
  
 // Set up the listen address struct  
 memset(&listen\_addr, 0, sizeof(listen\_addr));  
 listen\_addr.sin\_family = AF\_INET;  
 listen\_addr.sin\_port = htons(BROADCAST\_PORT);  
 listen\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);  
  
 // Bind the socket to the port  
 if (bind(sockfd, (struct sockaddr\*)&listen\_addr, sizeof(listen\_addr)) < 0) {  
 perror("Bind failed");  
 close(sockfd);  
 exit(EXIT\_FAILURE);  
 }  
  
 printf("Listening for broadcast messages on port %d...\n", BROADCAST\_PORT);  
  
 while (1) {  
 // Receive broadcast message  
 num\_bytes\_received = recvfrom(sockfd, buffer, BUFFER\_SIZE - 1, 0, (struct sockaddr\*)&listen\_addr, &addr\_len);  
 if (num\_bytes\_received < 0) {  
 perror("Receive failed");  
 close(sockfd);  
 exit(EXIT\_FAILURE);  
 }  
  
 // Null-terminate the received data  
 buffer[num\_bytes\_received] = '\0';  
  
 // Print the received message  
 printf("Received broadcast message: %s\n", buffer);  
 }  
  
 // Close the socket  
 close(sockfd);  
  
 return 0;  
}