Module 4 Assessments

1. **Explain the connection procedure followed in client server communication**
2. **What is the use of bind() function in socket programming ?**
3. **What is Datagram Socket ?**
4. **Write a server/client model socket program to exchange hello message between them.**
5. **Write a TCP server-client program to check if a given string is Palindrome**

   Input: level

   Output: Palindrome

   Input: Assessment

   Output: Not a Palindrome
6. **Write an example to demonstrate UDP server-client program**

Answers:

1. **Connection Procedure in Client-Server Communication:**
   - Client initiates a connection request to the server.
   - Server listens for incoming connection requests on a specific port.
   - When the server receives a connection request, it accepts the connection.
   - Once the connection is established, both the client and server can send and receive data through the connection.
   - The connection can be closed by either the client or the server when communication is complete.
2. **Use of bind() function in socket programming:**
   - The bind() function is used in socket programming to associate a socket with a specific address and port number on the local machine.
   - For server applications, bind() is used to specify the address and port on which the server will listen for incoming connections.
   - For client applications, bind() is typically not needed as the operating system will automatically assign an available port when the client initiates a connection.
   - After binding, the socket can be used to send or receive data on the specified address and port.

3. **Datagram Socket:**
    - A Datagram Socket is a type of socket used in network programming to enable communication between processes running on different hosts.
    - Unlike stream sockets, which provide reliable, connection-oriented communication (e.g., TCP), datagram sockets provide unreliable, connectionless communication (e.g., UDP).
    - Datagram sockets are used for applications where low overhead and real-time communication are more important than reliability, such as multimedia streaming, online gaming, and real-time data transmission.
4. **Server-Client Model Socket Program to Exchange "Hello" Message:**
   Server Code:

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

#define PORT 8080

int main() {
   int server_fd, new_socket;
   struct sockaddr_in address;
   int addrlen = sizeof(address);
   char buffer[1024] = {0};
   char *hello = "Hello from server";

   // Create socket file descriptor
   if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0) {
      perror("socket failed");
      exit(EXIT_FAILURE);
   }

   address.sin_family = AF_INET;
   address.sin_addr.s_addr = INADDR_ANY;
   address.sin_port = htons(PORT);

   // Bind socket to specified port
   if (bind(server_fd, (struct sockaddr *)&address, sizeof(address)) < 0) {
      perror("bind failed");
```

```
        exit(EXIT_FAILURE);
    }

    // Listen for incoming connections
    if (listen(server_fd, 3) < 0) {
        perror("listen");
        exit(EXIT_FAILURE);
    }

    // Accept incoming connection
    if ((new_socket = accept(server_fd, (struct sockaddr *)&address,
(socklen_t*)&addrlen)) < 0) {
        perror("accept");
        exit(EXIT_FAILURE);
    }

    // Send "Hello" message to client
    send(new_socket, hello, strlen(hello), 0);
    printf("Hello message sent to client\n");

    // Close socket
    close(new_socket);
    close(server_fd);
    return 0;
}
```

Client Code:

```
// TCP Client

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#include <sys/socket.h>


#define PORT 8080
```

```
int main() {
    int sock = 0;
    struct sockaddr_in serv_addr;
    char buffer[1024] = {0};
    char *hello = "Hello from client";

    // Create socket file descriptor
    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
        printf("\n Socket creation error \n");
        return -1;
    }

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);

    // Convert IPv4 and IPv6 addresses from text to binary form
    if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr) <= 0) {
        printf("\nInvalid address/ Address not supported \n");
        return -1;
    }

    // Connect to server
    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0) {
        printf("\nConnection Failed \n");
        return -1;
    }

    // Receive "Hello" message from server
```

```
    read(sock, buffer, 1024);
    printf("%s\n",buffer );

    // Close socket
    close(sock);
    return 0;
}
```

5. TCP Server-Client Program to Check if a String is Palindrome:

```
// TCP Server
// (Same server code as provided above)

// TCP Client
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/socket.h>

#define PORT 8080

int isPalindrome(char* str) {
    int len = strlen(str);
    for (int i = 0; i < len/2; i++) {
        if (str[i] != str[len-i-1]) {
            return
```