Experiment - 4

Abhilasha Jolly

20223008

1.Design a Distributed application using Socket. Application consists of a server which takes an integer value from the client, calculates factorial and returns the result to the client program

Code:

Server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>

long factorial(int num) {
        if (num <= 1) {
        return 1;
        }
        long fact = 1;
        for (int i = 2; i <= num; i++) {
        fact *= i;
        }
        return fact;
}

int main() {
        int server_fd, client_fd;
```

```
    struct sockaddr_in server_addr, client_addr;
    socklen_t client_addr_len = sizeof(client_addr);
    char buffer[1024];
    int num;

    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
    perror("Socket creation failed");
    exit(1);
    }

    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = INADDR_ANY;
    server_addr.sin_port = htons(8080);

    if (bind(server_fd, (struct sockaddr *)&server_addr, sizeof(server_addr)) == -1) {
    perror("Binding failed");
    close(server_fd);
    exit(1);
    }

    if (listen(server_fd, 3) == -1) {
    perror("Listen failed");
    close(server_fd);
    exit(1);
    }

    printf("Server listening on port 8080...\n");

    if ((client_fd = accept(server_fd, (struct sockaddr *)&client_addr, &client_addr_len)) ==
-1) {
    perror("Client connection failed");
    close(server_fd);
    exit(1);
    }

    recv(client_fd, buffer, sizeof(buffer), 0);
    num = atoi(buffer);

    long result = factorial(num);

    snprintf(buffer, sizeof(buffer), "%ld", result);
    send(client_fd, buffer, sizeof(buffer), 0);

    close(client_fd);
```

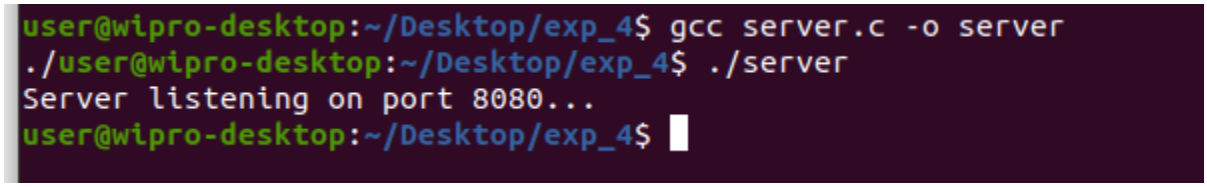
```
        close(server_fd);

        return 0;
}
```

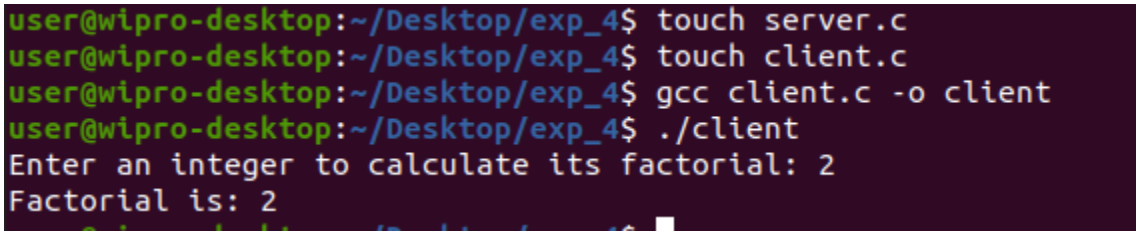
Client.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>

int main() {
        int sock;
        struct sockaddr_in server_addr;
        char buffer[1024];
        int num;

        if ((sock = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
        perror("Socket creation failed");
        exit(1);
        }

        server_addr.sin_family = AF_INET;
        server_addr.sin_port = htons(8080);
        server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

        if (connect(sock, (struct sockaddr *)&server_addr, sizeof(server_addr)) == -1) {
        perror("Connection failed");
        close(sock);
        exit(1);
        }

        printf("Enter an integer to calculate its factorial: ");
        scanf("%d", &num);

        snprintf(buffer, sizeof(buffer), "%d", num);
        send(sock, buffer, sizeof(buffer), 0);

        recv(sock, buffer, sizeof(buffer), 0);
        printf("Factorial is: %s\n", buffer);
```

```
    close(sock);

    return 0;
}
```

Output:





2.Find out the list of users who owns a file having maximum size in the current working directory using Map Reduce Program.

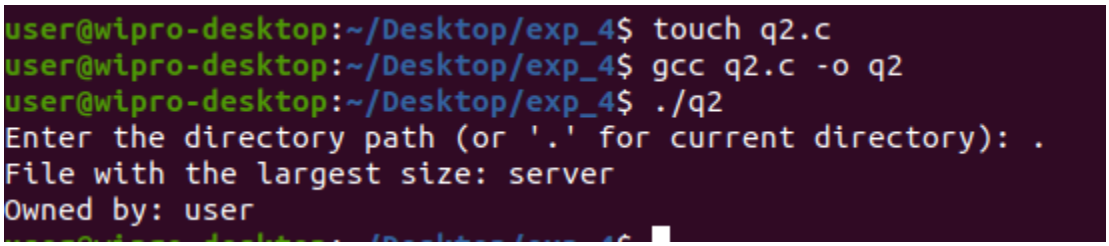
Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <dirent.h>
#include <sys/stat.h>
#include <pwd.h>
#include <string.h>
#include <unistd.h>

void get_max_file_user(char *path) {
    DIR *dir;
    struct dirent *entry;
    struct stat file_stat;
    char max_file[256];
    long max_size = 0;
    char max_user[256];

    if ((dir = opendir(path)) == NULL) {
```

```
        perror("opendir");
        exit(1);
        }

        while ((entry = readdir(dir)) != NULL) {
        if (entry->d_type == DT_DIR) {
        continue;
        }

        char file_path[512];
        snprintf(file_path, sizeof(file_path), "%s/%s", path, entry->d_name);

        if (stat(file_path, &file_stat) == -1) {
        perror("stat");
        continue;
        }

        struct passwd *pwd = getpwuid(file_stat.st_uid);
        if (pwd == NULL) {
        perror("getpwuid");
        continue;
        }

        if (file_stat.st_size > max_size) {
        max_size = file_stat.st_size;
        strncpy(max_file, entry->d_name, sizeof(max_file) - 1);
        strncpy(max_user, pwd->pw_name, sizeof(max_user) - 1);
        }
        }

        closedir(dir);

        printf("File with the largest size: %s\n", max_file);
        printf("Owned by: %s\n", max_user);
}

int main() {
        char path[256];
        printf("Enter the directory path (or '.' for current directory): ");
        scanf("%s", path);

        get_max_file_user(path);

        return 0;
```

}

Output:


user@wipro-desktop:~/Desktop/exp_4$ touch q2.c
user@wipro-desktop:~/Desktop/exp_4$ gcc q2.c -o q2
user@wipro-desktop:~/Desktop/exp_4$ ./q2
Enter the directory path (or '.' for current directory): .
File with the largest size: server
Owned by: user