Πανεπιστήμιου Δυτικής Αττικής
Σχολή Μηχανικών
Κατανεμημένα Συστήματα

Ονοματεπώνυμο: Γάγγας Ιωάννης
Αριθμός Μητρώου: 19390038
Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών
Ημερομηνία παράδωσης εργασίας: 02/05/2024

Η παρούσα εργασία στοχέυει στην δημιουργία ενός TCP Server ο οποίος χρησιμοποιεί το μοντέλο RPC, ώστε να ανταποκριθεί στα αιτήματα του χρήστη. Για την λειτουργία του προγράμματος θα χρειαστεί πρώτα μια διεπαφή με τον χρήστη δηλαδή ένα socket_client που θα παρέχει ένα διαδραστικό μενού και θα επικοινωνεί με τον server. Προτεραιώτητα έχει όμως η δημιουργία το απομακρυσμένο αρχείο που θα περιέχει τισ συναρτήσεις που επιλύουν τα ζητούμενα της εργασίας, δηλαδή τον υπολογισμό του μέσου όρου, την εύρεση μέγιστου και ελάχιστου και τον πολλαπλασιασμό πίνακα με αριθμό. Το αρχείο αυτό ειναι το **DSFunc.x**.

```
1 struct Y_arr{
2     int Y<25>;
3     int Y_size;
4 };
5
6 struct max_min{
7     int max;
8     int min;
9 };
10
11 struct a_mul_Y{
12     int Y<25>;
13     int Y_size;
14     float a;
15 };
16
17 struct aY{
18   float prod<25>;
19 };
20
21   program CONCURRENT_SER_PROG{
22     version CONCURRENT_SER_VER{
23         float average(Y_arr)=1;
24         max_min maxmin(Y_arr)=2;
25         aY product(a_mul_Y)=3;
26     }=1;
27   }=0x12345601;
```

Αφού δημιουργήθηκε το παραπάνω αρχείο η μηχανή RPC θα δημιουργήσει μόνη της τα απαιτούμενα αρχεία για την ορθή λειτουργία του προγράμματος με την παρακάτω εντολή.

```
rpc@rpc:~/ds$ rpcgen -C DSFunc.x
```

Στην συνέχεια τα RPC αρχεία πρέπει να γίνουν compile. Η διαδικασία αυτή επιτυγχάνεται με την παρκάτω εντολή.

```
rpc@rpc:~/ds$ make -f Makefile.DSFunc
```

Έπειτα πρεπει να δημιουργηθεί το αρχείο για το περιβάλλον διεπαφής το χρήστη που είναι το DS_Client.c, ενώ επιπλέον πρέπει να γίνει επεξεργασία των αυτόματα δημιουργημένων αρχείων DSFunc_server.c και DSFunc_client.c όπως παρουσίαζεται στις παρακάτω εικόνες.

DSFunc_server.c

```c
#include "DSFunc.h"

float *
average_1_svc(Y_arr *argp, struct svc_req *rqstp)
{
        static float  result;

        printf("Function has been called to calculate the AVG.\n");
        // Calculating the avg
        float sum=0;
        // With '->' we assign the value to 'Y_size'.
        for(int i=0; i<argp->Y_size;i++){
                sum+=argp->Y.Y_val[i];
        }
        result = sum/argp->Y_size;

        return &result;
}
```

```c
max_min *
maxmin_1_svc(Y_arr *argp, struct svc_req *rqstp)
{
        static max_min  result;

        printf("Function has been called to calculate the MAx and Min.\n");
        // Values to compare the array elements to get the maximum and minimum
        result.max=-999999;
        result.min=999999;
        // Using for to calculate max and min.
        for(int i=0;i<argp->Y_size;-i++){
                if(argp->Y.Y_val[i]>result.max){
                        result.max=argp->Y.Y_val[i];
                }
        }
        for(int i=0;i<argp->Y_size;i++){
                if(argp->Y.Y_val[i]<result.min){
                        result.min=argp->Y.Y_val[i];
                }
        }
        return &result;
}
```

```c
aY *
product_1_svc(a_mul_Y *argp, struct svc_req *rqstp)
{
        static aY  result;

        printf("Function has been called to calculate the a*Y[].\n");

        result.prod.prod_len=argp->Y_size;
        result.prod.prod_val=(float *)malloc(argp->Y_size*sizeof(float));

        for(int i=0;i<argp->Y_size;i++){
                result.prod.prod_val[i]=argp->a*argp->Y.Y_val[i];
        }

        return &result;
}
```

DSFunc_client.c

```
41                  average_1_arg.Y.Y_len=n;
42                  average_1_arg.Y_size=n;
43                  average_1_arg.Y.Y_val=(int *) malloc(n*sizeof(int));
44
45                  for(int i=0;i<n;i++){
46                  // Matching the variables.
47                          average_1_arg.Y.Y_val[i]=Y[i];
48                  }
 57                  maxmin_1_arg.Y.Y_len=n;
 58                  maxmin_1_arg.Y_size=n;
 59                  maxmin_1_arg.Y.Y_val=(int *)malloc(n*sizeof(int));
 60                  for(int i=0;i<n;i++){
 61                          maxmin_1_arg.Y.Y_val[i]= Y[i];
 62                  }
 63
70          else if (choice==3){
71          // Matching variables once again for a*Y[].
72                  product_1_arg.Y.Y_len=n;
73                  product_1_arg.Y_size=n;
74                  product_1_arg.Y.Y_val=(int *) malloc(n*sizeof(int));
75
76                  for(int i=0;i<n;i++){
77                          product_1_arg.Y.Y_val[i] = Y[i];
78                          product_1_arg.a=a;
79                  }
```

```
 95 int main (int argc, char *argv[])
 96 {
 97         char *host;
 98         int sockfd, newsockfd, portnum, cllen;
 99         int children =0;
100         struct sockaddr_in serverAddr, cli_addr;
101         pid_t procID;
102         if (argc < 3) {
103                 printf ("usage: %s server_host\n", argv[0]);
104                 exit (1);
105         }
106
107
108         sockfd = socket(AF_INET, SOCK_STREAM, 0);
109         if (sockfd < 0) {
110                 printf("[-]ERROR in connection.\n");
111                 exit(1);
112         }
113         //printf("[+]Client Socket created successfully.\n");
114
115         bzero((char *) &serverAddr, sizeof(serverAddr));
116         portnum = atoi(argv[2]);
117         serverAddr.sin_family = AF_INET;
118         serverAddr.sin_port = htons(portnum);
119         serverAddr.sin_addr.s_addr = INADDR_ANY;
120         // Establishing connection to the server.
121
122         if (bind(sockfd, (struct sockaddr *) &serverAddr, sizeof(serverAddr))<0){
123                 printf("[-]ERROR in binding.\n");
124                 exit(1);
125         }
126         listen(sockfd,5);
127         printf("[+]Connected to Server.\n");
128
129         for (;;){
130         printf("Waiting for connection...\n");
131         cllen = sizeof(cli_addr);
132         newsockfd = accept(sockfd, (struct sockaddr *) &cli_addr, &cllen);
133         if(newsockfd <0){
134                 printf("[-]ERROR in accepting.\n");
135                 exit(1);
136         }
```

Παρπάνω γίνεται η δημιουγία του Server.

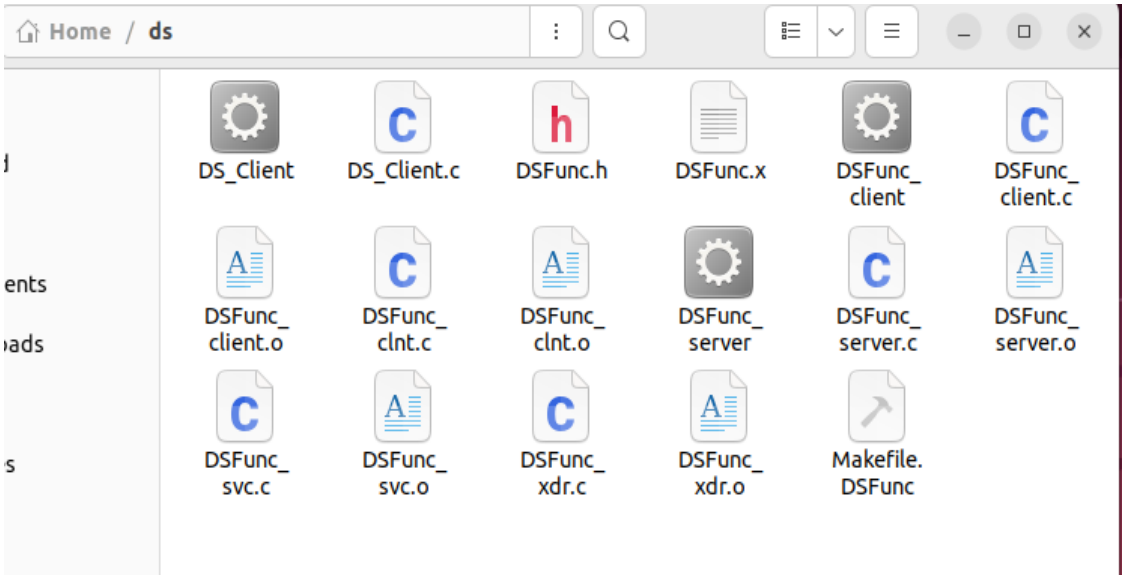
Παρακάτω παρουσιάζεται το αρχείο του socket client DS_Client.c. Ο κώδικας παρκάτω πρώτα ζητάει την θύρα σύνδεσης από τον χρήστη και ελέγχει αν έγινε ορθά η σύνδεση με τον διακομιστή. Αφού εδρεωθεί η σύνδεση με τον server εμφανίζεται στον χρήστη το μενού επιλογών. Ο κώδικας διαχειρίζεται τις λανθασμένες εισόδους με ένα μήνυμα λάθους και επανακτεύθυνση στο μενού, ενώ αν ο χρήστης δώσει την είσοδο “4” τότε η σύνδεση διαόπτεται και ο socket client κλείνει.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
        int flag=1, n, choice, *Y, result_2[2];
        float a,result_1, *result_3;
        int sockfd,portnum;
        struct sockaddr_in serverAddr;
        struct hostent *server;

        if (argc < 3) {
                printf("usage %s hostname port\n", argv[0]);
                exit(0);
        }

        portnum = atoi(argv[2]);
        sockfd = socket(AF_INET, SOCK_STREAM, 0);
        if (sockfd < 0) {
                printf("[-]ERROR in connection.\n");
                exit(1);
        }

        server = gethostbyname(argv[1]);
         if (server == NULL) {
                printf("ERROR, no such host\n");
                exit(0);
        }
        printf("[+]Client Socket created successfully.\n");

        bzero((char *) &serverAddr, sizeof(serverAddr));
        serverAddr.sin_family = AF_INET;
        serverAddr.sin_port = htons(portnum);
        bcopy((char *)server->h_addr,(char *)&serverAddr.sin_addr.s_addr,server->h_length);
        // Establishing connection to the server.
        if (connect(sockfd, (struct sockaddr *)&serverAddr, sizeof(serverAddr)) < 0){
                printf("[-]ERROR in binding.\n");
                exit(1);
        }
        printf("[+]Connected to Server.\n");


        do {
                //menu
```

```
50              printf("\n\nEnter choice: \n1. Calculate average of Y.\n2. Calculate maximum and minimum of Y.\n3. Calculate a*Y[].\n4. Exit.\n");
51
52              // We get rid of the rubbish.
53              fflush(stdout);
54              printf("\n\nPlease enter choice: ");
55              scanf("%d", &choice);
56              send(sockfd, &choice, sizeof(int), 0);
57
58              if (choice == 1){
59                      printf("Give size of Y[]: ");
60                      scanf("%d", &n);
61                      send(sockfd, &n, sizeof(int), 0);
62                      Y = (int *) malloc(n*sizeof(int));
63
64                      for(int i=0;i<n;i++){
65                              printf("Y[%d] = ", i+1);
66                              scanf("%d", &Y[i]);
67
68                      }
69                      send(sockfd, Y, n*sizeof(int), 0);
70                      recv(sockfd, &result_1, sizeof(float), 0);
71                      printf("\n\nAverage of Y[]: %.2f\n\n\n\n",result_1);
72
73              }
74
75              else if (choice == 2){
76
77                      printf("\n\n\nGive size of Y[]: ");
78                      scanf("%d", &n);
79                      send(sockfd, &n, sizeof(int), 0);
80
81                      Y = (int *) malloc(n*sizeof(int));
82
83                      for(int i=0;i<n;i++){
84                              printf("Y[%d] = ", i+1);
85                              scanf("%d", &Y[i]);
86
87                      }
88                      send(sockfd, Y, n*sizeof(int), 0);
89
90                      recv(sockfd, result_2, 2*sizeof(int), 0);
91                      printf("\n\nMax: %d\nMin: %d.\n\n", result_2[0], result_2[1]);
92
93              }
94              else if (choice == 3) {
95
96                      printf("Give size of Y[]: ");
97                      scanf("%d", &n);
98                      send(sockfd, &n, sizeof(int), 0);
```

```
 99
100                         Y = (int *) malloc(n*sizeof(int));
101
102                         for(int i=0; i<n; i++){
103                                 printf("Y[%d] = ", i+1);
104                                 scanf("%d", &Y[i]);
105
106                         }
107                 send(sockfd, Y, n*sizeof(int), 0);
108
109                 printf("Give floating number a: ");
110                 scanf("%f", &a);
111                 send(sockfd, &a, sizeof(float), 0);
112
113                 result_3 = (float *) malloc(n*sizeof(float));
114                 recv(sockfd, result_3, n*sizeof(float), 0);
115                 printf("\n\n");
116
117                 for(int i=0; i<n; i++){
118                         printf("a*Y[%d] = %.2f\n\n\n\n", i+1, result_3[i]);
119                 }
120     }
121     else if (choice == 4){
122       flag = 0;
123         }
124     else{
125       printf("Wrong input, please try again\n");
126         }
127
128
129   } while (flag);
130   close(sockfd);
131
132
133 return 0;
134
135 }
```

Ενδεικτικά στην παρακάτω εικόνα ο φάκελος με όλα τα απαιτούμενα αρχεία.



Τώρα είναι όλα έτοιμα για τα ενδεικτικά τρεξήματα. Πρώτα πρέπει να γίνει εκκίνηση του server με την εντολή ./DSFunc_server μετά του client host ./DSFunc_client και τέλος του αρχείου διεπαφής ./DS_Client, σε διαφορετικά terminals το καθένα. Για τα αρχεία DS_Client και DSFunc_client χρειάζεται να δωθεί το ip adress (δηλαδή localhost) και το port number (για παράδειγμα 6678).

Ενδεικτικά τρεξήματα

```
rpc@rpc:~/ds$ ./DSFunc_server
```

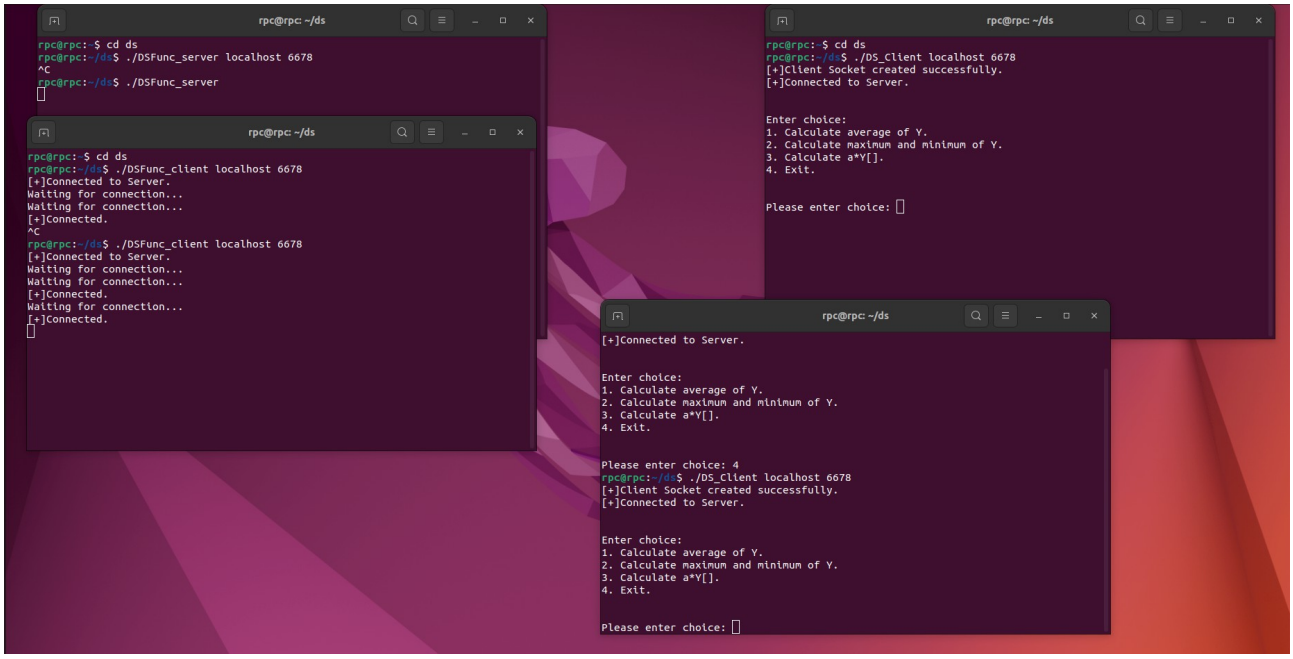
```
rpc@rpc: ~/ds
rpc@rpc:~$ cd ds
rpc@rpc:~/ds$ ./DSFunc_client localhost 6678
[+]Connected to Server.
Waiting for connection...
```

```
rpc@rpc:~/ds$ ./DS_Client localhost 6678
[+]Client Socket created successfully.
[+]Connected to Server.


Enter choice:
1. Calculate average of Y.
2. Calculate maximum and minimum of Y.
3. Calculate a*Y[].
4. Exit.


Please enter choice:
```

Μετά την σύνδεση του DS_Client στο terminal του client host εμφανίζεται η παρκάτω επιβεβαίωση σύνδεσης.

```
rpc@rpc:~/ds$ ./DSFunc_client localhost 6678
[+]Connected to Server.
Waiting for connection...
Waiting for connection...
[+]Connected.
```

Ενδεικτικά σύνδεση δύο χρηστών.



Δοκιμή της πρώτης λειτουργίας.

```
Please enter choice: 1
Give size of Y[]: 4
Y[1] = 8
Y[2] = 2
Y[3] = 6
Y[4] = 4

Average of Y[]: 5.00
```

```
Please enter choice: 1
Give size of Y[]: 4
Y[1] = 8
Y[2] = 2
Y[3] = 6
Y[4] = 4

Average of Y[]: 5.00

Enter choice:
1. Calculate average of Y.
2. Calculate maximum and minimum of Y.
3. Calculate a*Y[].
4. Exit.

Please enter choice: 1
Give size of Y[]: 10
Y[1] = 43
Y[2] = 4325
Y[3] = 70981
Y[4] = 4902435
Y[5] = 86
Y[6] = 0
Y[7] = 56738
Y[8] = 94090
Y[9] = 9996
Y[10] = 100000

Average of Y[]: 523869.41
```

Δοκιμή δεύτερης λειτουργίας.

```
Please enter choice: 2


Give size of Y[]: 4
Y[1] = 7
Y[2] = 5
Y[3] = 1
Y[4] = 2


Max: 7
Min: 1.
```

```
Give size of Y[]: 5
Y[1] = 99
Y[2] = 705
Y[3] = 0
Y[4] = -66
Y[5] = -3


Max: 705
Min: -66.



Enter choice:
1. Calculate average of Y.
2. Calculate maximum and minimum of Y.
3. Calculate a*Y[].
4. Exit.


Please enter choice:
```

Δοκιμή τρίτης λειτουργίας.

```
Please enter choice: 3
Give size of Y[]: 4
Y[1] = 5
Y[2] = 2
Y[3] = 8
Y[4] = 1
Give floating number a: 3


a*Y[1] = 15.00



a*Y[2] = 6.00



a*Y[3] = 24.00



a*Y[4] = 3.00
```

```
Please enter choice: 3
Give size of Y[]: 2
Y[1] = -88
Y[2] = 88
Give floating number a: 2


a*Y[1] = -176.00



a*Y[2] = 176.00
```

Δοκιμή εξόδου.

```
Enter choice:
1. Calculate average of Y.
2. Calculate maximum and minimum of Y.
3. Calculate a*Y[].
4. Exit.


Please enter choice: 4
rpc@rpc:~/ds$
```

Τέλος, επιβεβαίωση κλήσης συναρτήσεων από το απομακρισμένο αρχείο.

```
rpc@rpc:~/ds$ ./DSFunc_server
Function has been called to calculate the AVG.
Function has been called to calculate the AVG.
Function has been called to calculate the MAx and Min.
Function has been called to calculate the MAx and Min.
Function has been called to calculate the a*Y[].
Function has been called to calculate the a*Y[].
```

Προβλήματα

Τα προβλήματα εμφανίστηκαν κυρίως κατά την αυτόματη δημιουργία των rpc αρχείων. Με μερικά updates στην μηχανή rpc και κάποιες μικροαλλαγές στα αρχεία που είχαν δημιουργηθεί τα προβλήματα επιλύθηκαν. Με την εκτέλεση της εντολής Makefile εμφανιζόντουσαν warnings, παρόλαυτα δεν επηρρέασαν την ορθή λειτουργία του προγράμματος στην τελική.
Παράδειγμα μυνήματος προβλήματος:
DSFunc.h:9:10: fatal error: rpc/rpc.h: No such file or directory 9 | #include <rpc/rpc.h>

Πηγές:
https://stackoverflow.com/
https://docs.oracle.com/cd/E18752_01/html/816-1435/rpcproto-24229.html
https://eclass.uniwa.gr/modules/document/?course=CS157