Distributed Systems (CS304)

Assignment - 7

U19CS012

Simulate RPC (Create any one procedure on remote machine and call it from local machine)

List of Programs for RPC

1.) String is **Palindrome or Not.**

[q1.x]

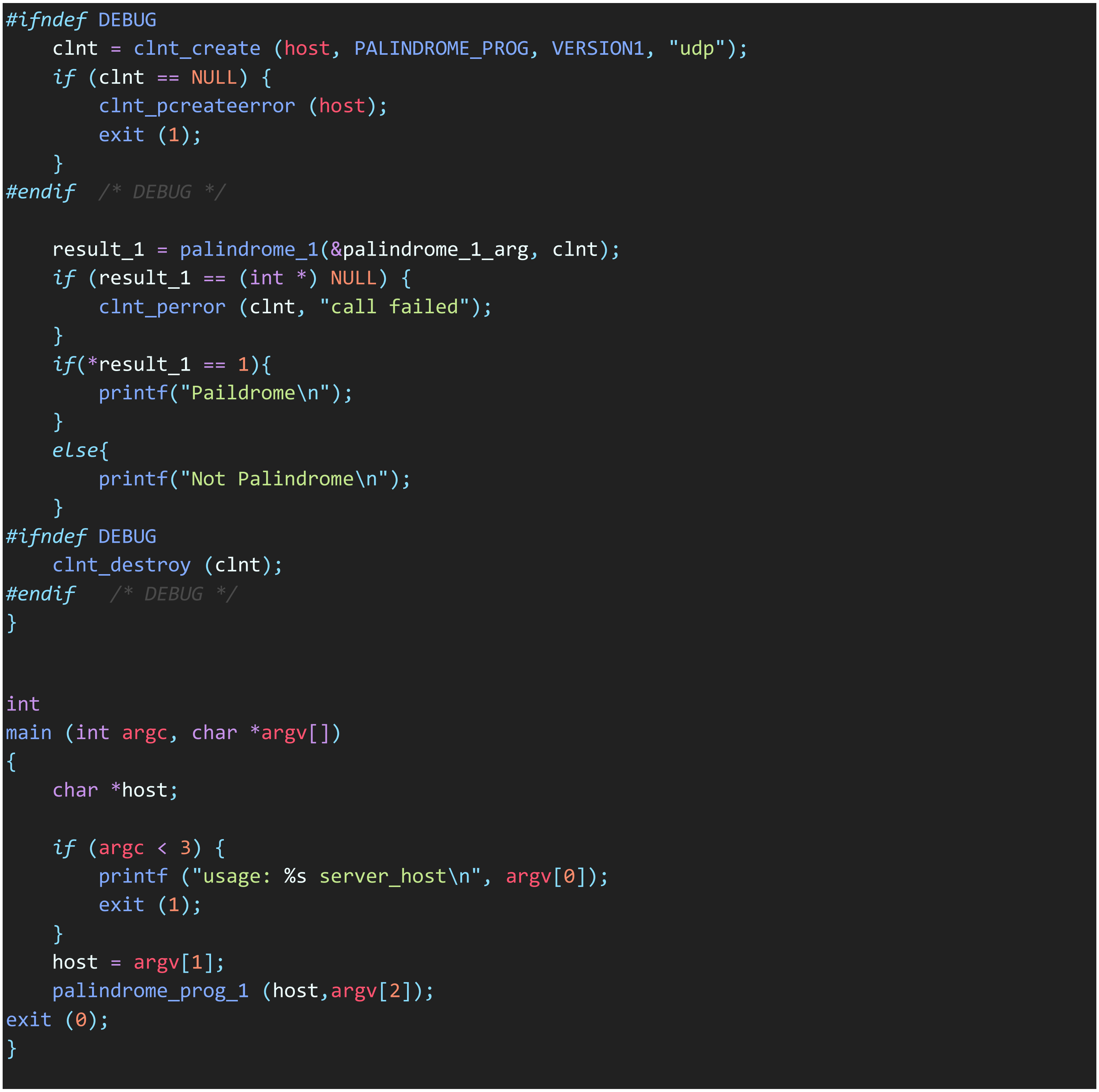
```
program PALINDROME_PROG{
    version VERSION1 {
        int palindrome(string s)=1;
    }=1;
}=0x4562877;
```

Run Command : `rpcgen -a -C q1.x`

- ✓ All required files will be created.
- ✓ The q1_client.c and q1_server.c files would be modified as following:

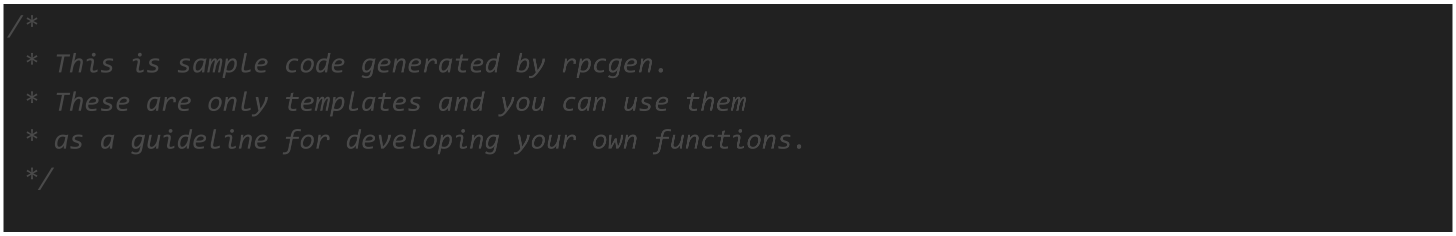
[q1_client.c]

```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q1.h"


void
palindrome_prog_1(char *host, char* str)
{
    CLIENT *clnt;
    int  *result_1;
    char * palindrome_1_arg = str;
```

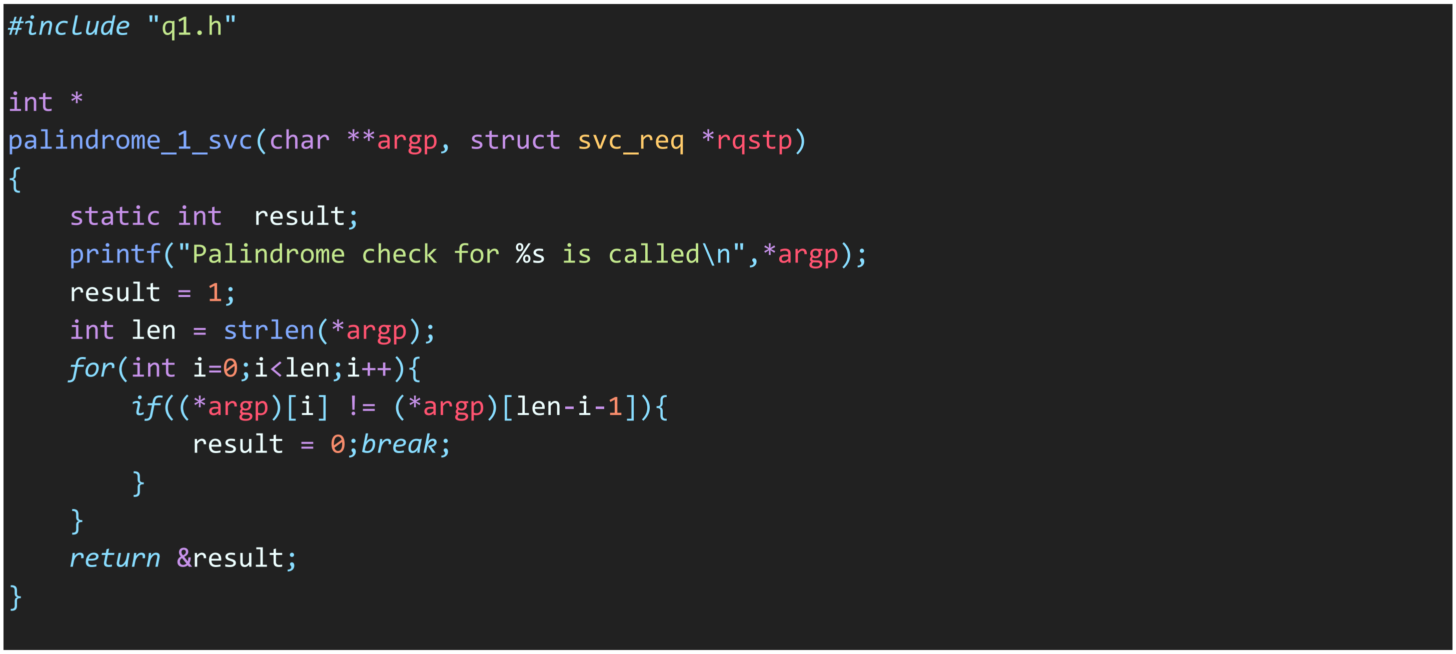


```
#ifndef DEBUG
    clnt = clnt_create (host, PALINDROME_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = palindrome_1(&palindrome_1_arg, clnt);
    if (result_1 == (int *) NULL) {
        clnt_perror (clnt, "call failed");
    }
    if(*result_1 == 1){
        printf("Paildrome\n");
    }
    else{
        printf("Not Palindrome\n");
    }
#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 3) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    palindrome_prog_1 (host,argv[2]);
exit (0);
}
```

[q1_server.c]



```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */
```

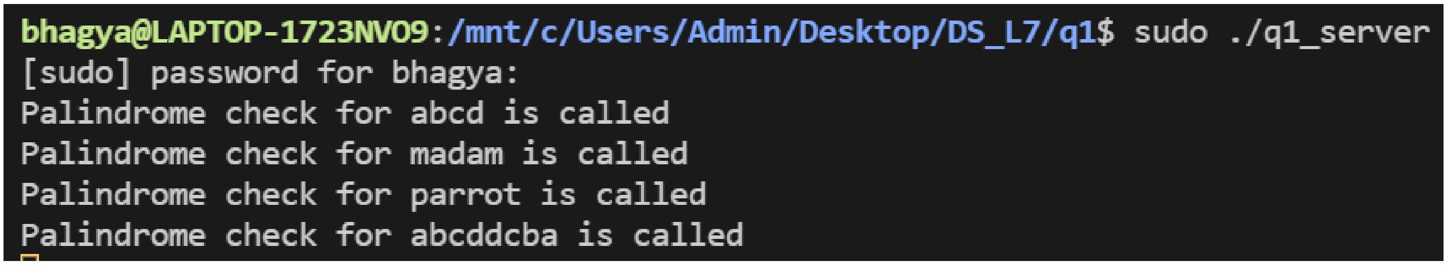


```
#include "q1.h"

int *
palindrome_1_svc(char **argp, struct svc_req *rqstp)
{
    static int  result;
    printf("Palindrome check for %s is called\n",*argp);
    result = 1;
    int len = strlen(*argp);
    for(int i=0;i<len;i++){
        if((*argp)[i] != (*argp)[len-i-1]){
            result = 0;break;
        }
    }
    return &result;
}
```

Run Command : `make -f Makefile.q1`

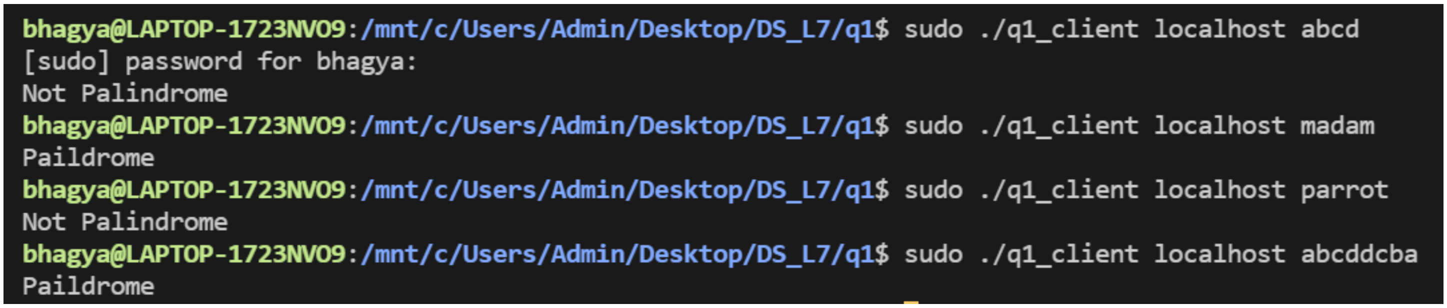
[Output]

Server:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q1$ sudo ./q1_server
[sudo] password for bhagya:
Palindrome check for abcd is called
Palindrome check for madam is called
Palindrome check for parrot is called
Palindrome check for abcddcba is called
```

Client:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q1$ sudo ./q1_client localhost abcd
[sudo] password for bhagya:
Not Palindrome
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q1$ sudo ./q1_client localhost madam
Paildrome
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q1$ sudo ./q1_client localhost parrot
Not Palindrome
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q1$ sudo ./q1_client localhost abcddcba
Paildrome
```

2.) Find out if a given year is a **Lear Year** or not.

[q2.x]

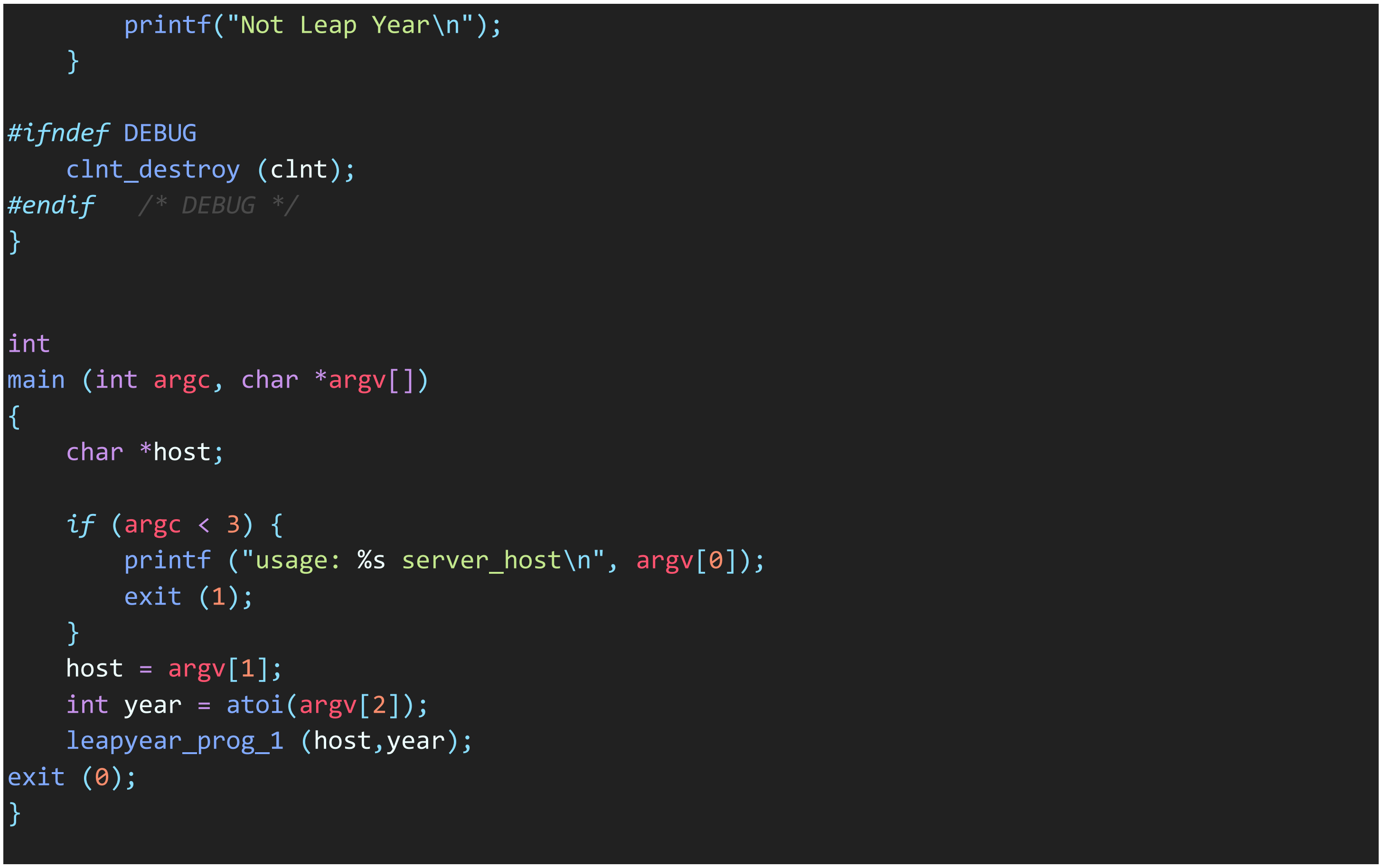
```
program LEAPYEAR_PROG{
    version VERSION1 {
        int leapyear(int year) = 1;
    } = 1;
} =  0x4562877;
```

Run Command : `rpcgen -a -C q2.x`

- ✓ All required files will be created.
- ✓ The q2_client.c and q2_server.c files would be modified as following:

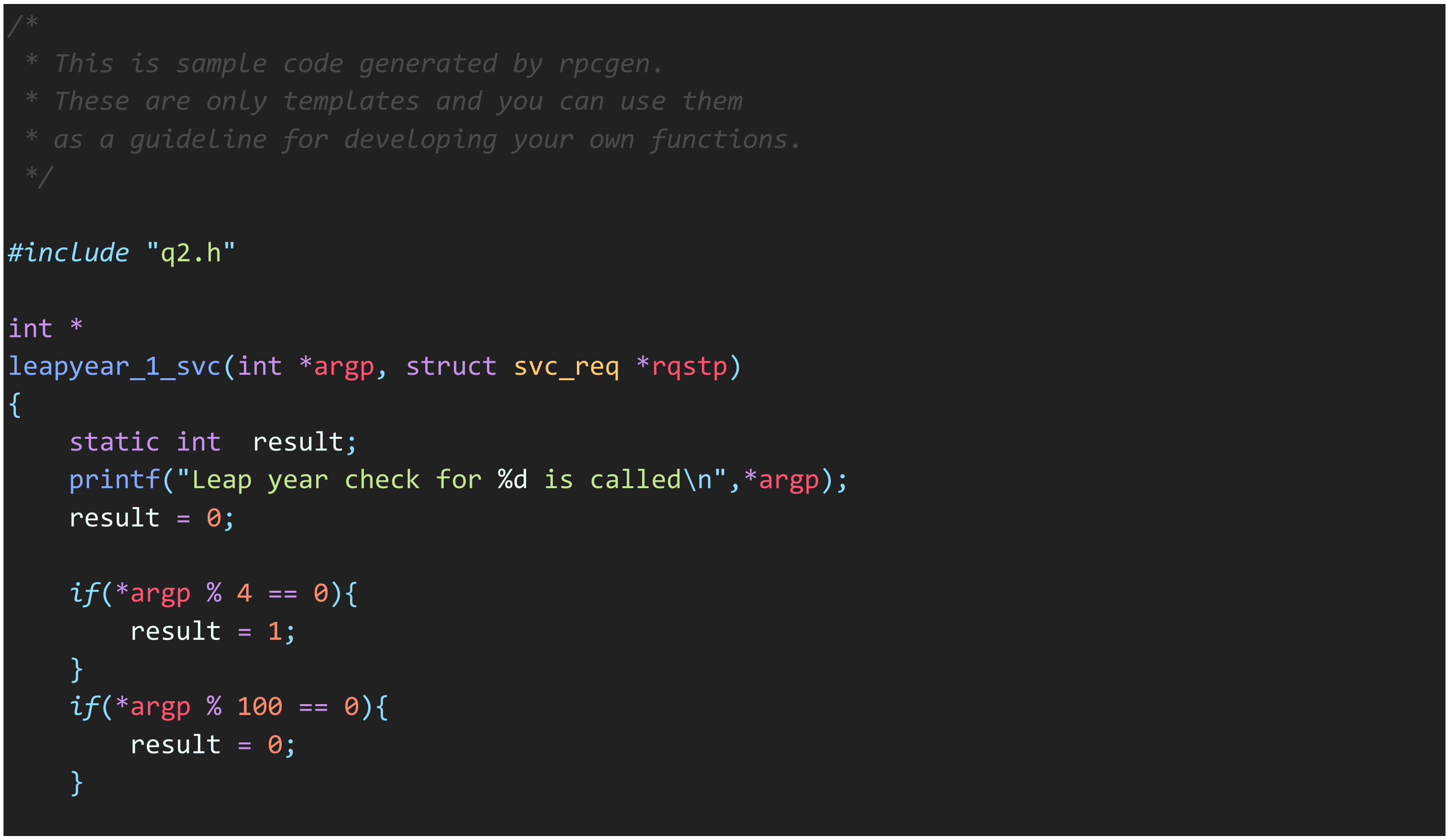
[q2_client.c]

```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q2.h"


void
leapyear_prog_1(char *host,int year)
{
    CLIENT *clnt;
    int  *result_1;
    int  leapyear_1_arg = year;

#ifndef DEBUG
    clnt = clnt_create (host, LEAPYEAR_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = leapyear_1(&leapyear_1_arg, clnt);
    if (result_1 == (int *) NULL) {
        clnt_perror (clnt, "call failed");
    }
    if(*result_1 == 1){
        printf("Leap Year\n");
    }else{
```

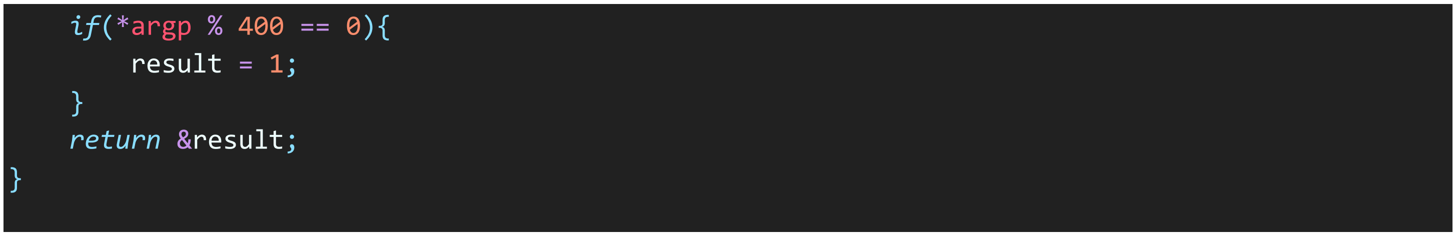


```c
        printf("Not Leap Year\n");
    }

#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 3) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    int year = atoi(argv[2]);
    leapyear_prog_1 (host,year);
exit (0);
}
```

[q2_server.c]



```c
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q2.h"

int *
leapyear_1_svc(int *argp, struct svc_req *rqstp)
{
    static int  result;
    printf("Leap year check for %d is called\n",*argp);
    result = 0;

    if(*argp % 4 == 0){
        result = 1;
    }
    if(*argp % 100 == 0){
        result = 0;
    }
```

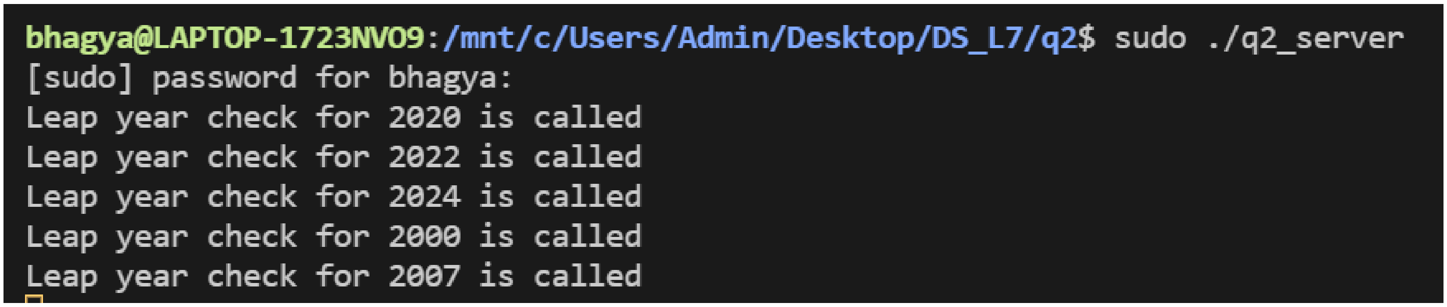


```
    if(*argp % 400 == 0){
        result = 1;
    }
    return &result;
}
```

Run Command : make -f Makefile.q2

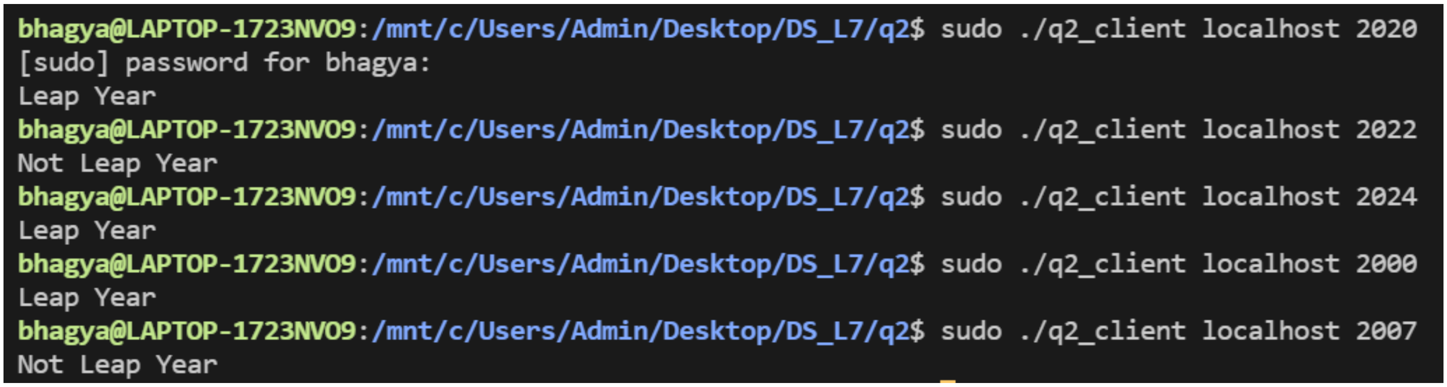
[Output]

Server:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q2$ sudo ./q2_server
[sudo] password for bhagya:
Leap year check for 2020 is called
Leap year check for 2022 is called
Leap year check for 2024 is called
Leap year check for 2000 is called
Leap year check for 2007 is called
```

Client:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q2$ sudo ./q2_client localhost 2020
[sudo] password for bhagya:
Leap Year
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q2$ sudo ./q2_client localhost 2022
Not Leap Year
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q2$ sudo ./q2_client localhost 2024
Leap Year
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q2$ sudo ./q2_client localhost 2000
Leap Year
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q2$ sudo ./q2_client localhost 2007
Not Leap Year
```

3.) Find out the **GCD** of a given number.

[q3.x]

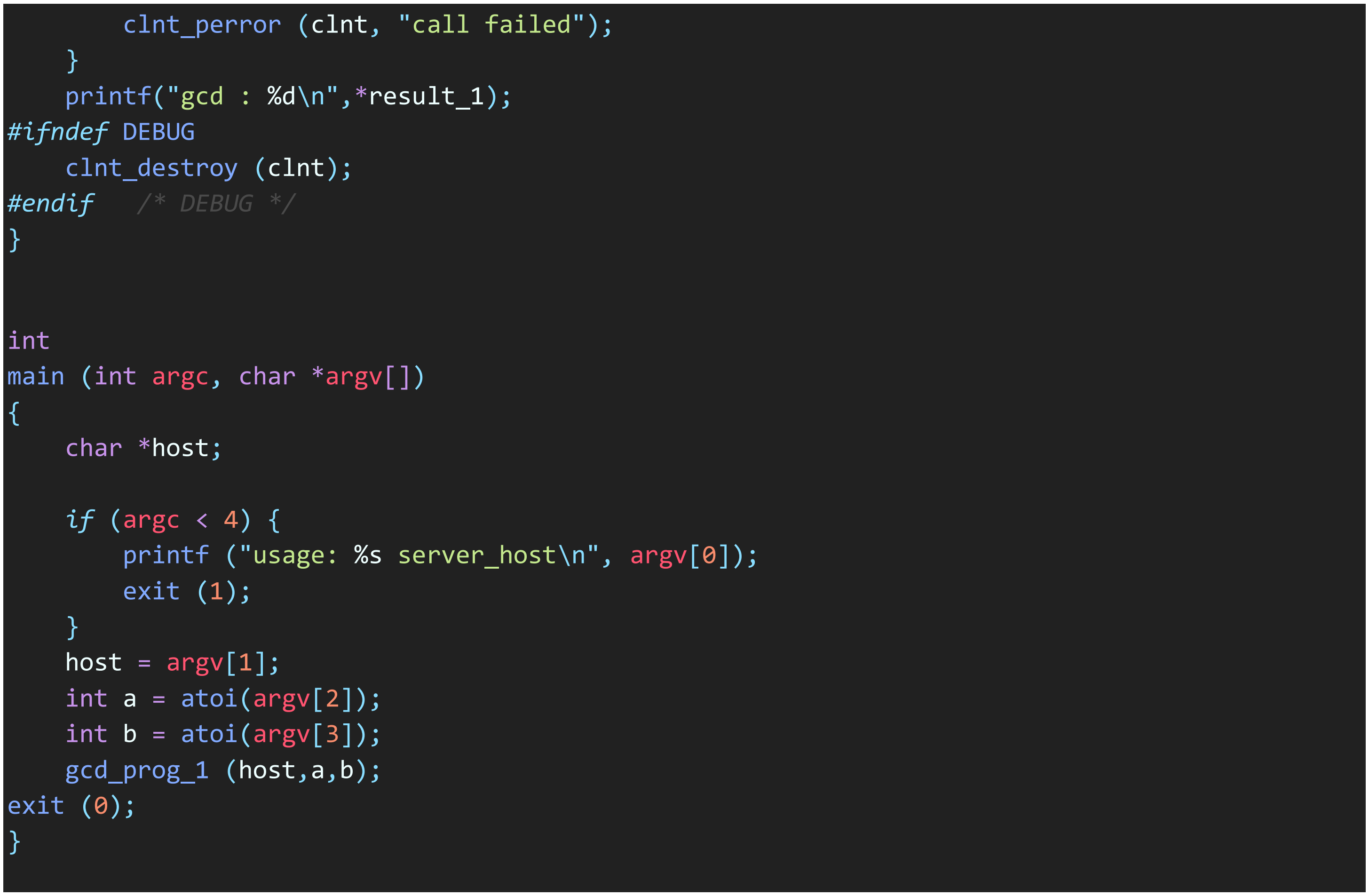
```
struct intpair{
    int a;
    int b;
};

program GCD_PROG{
    version VERSION1 {
        int gcd(intpair p) = 1;
    } = 1;
} =  0x4562877;
```

Run Command : `rpcgen -a -C q3.x`

- ✓ All required files will be created.
- ✓ The q3_client.c and q3_server.c files would be modified as following:

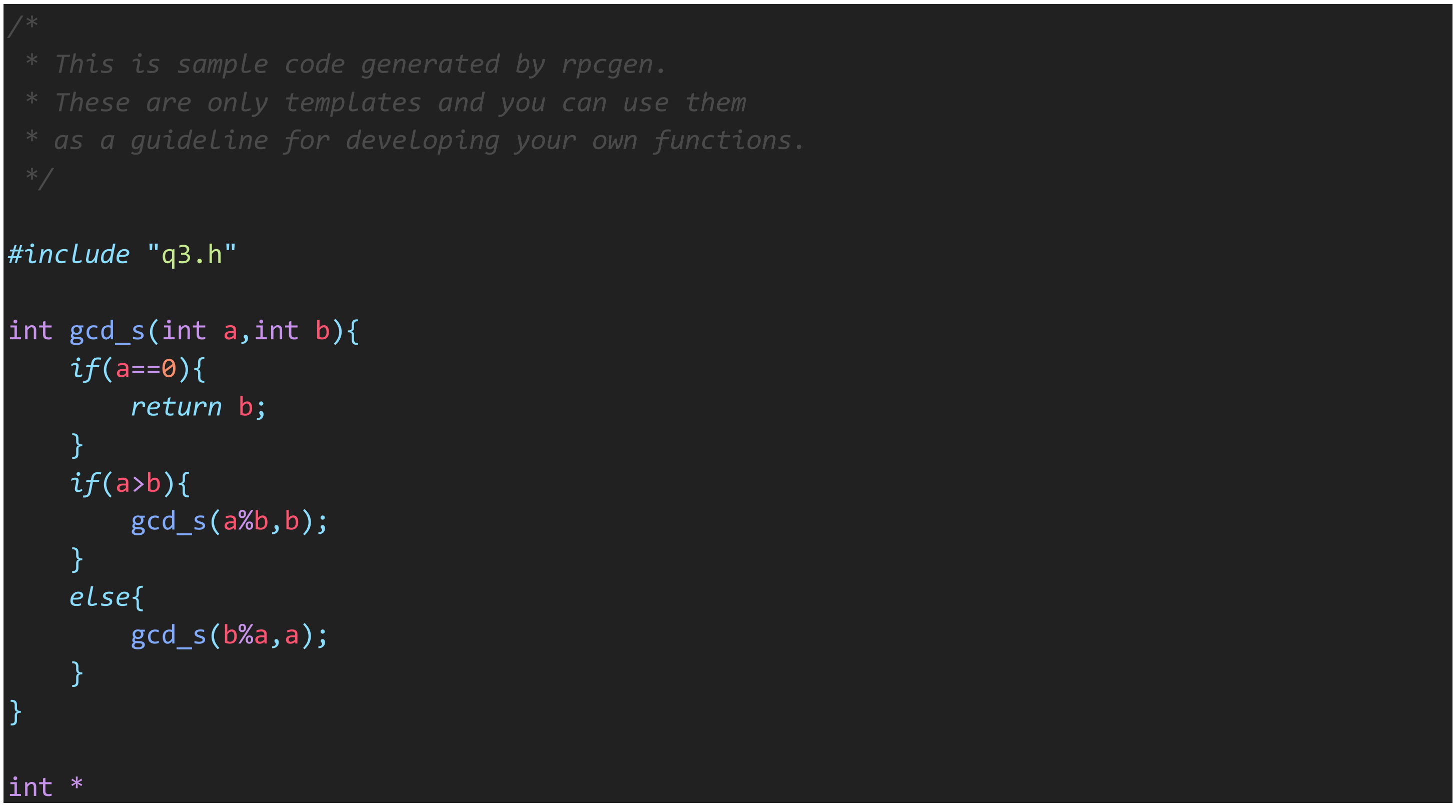
[q3_client.c]

```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q3.h"


void
gcd_prog_1(char *host,int a,int b)
{
    CLIENT *clnt;
    int  *result_1;
    intpair  gcd_1_arg = {a,b};

#ifndef DEBUG
    clnt = clnt_create (host, GCD_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = gcd_1(&gcd_1_arg, clnt);
    if (result_1 == (int *) NULL) {
```

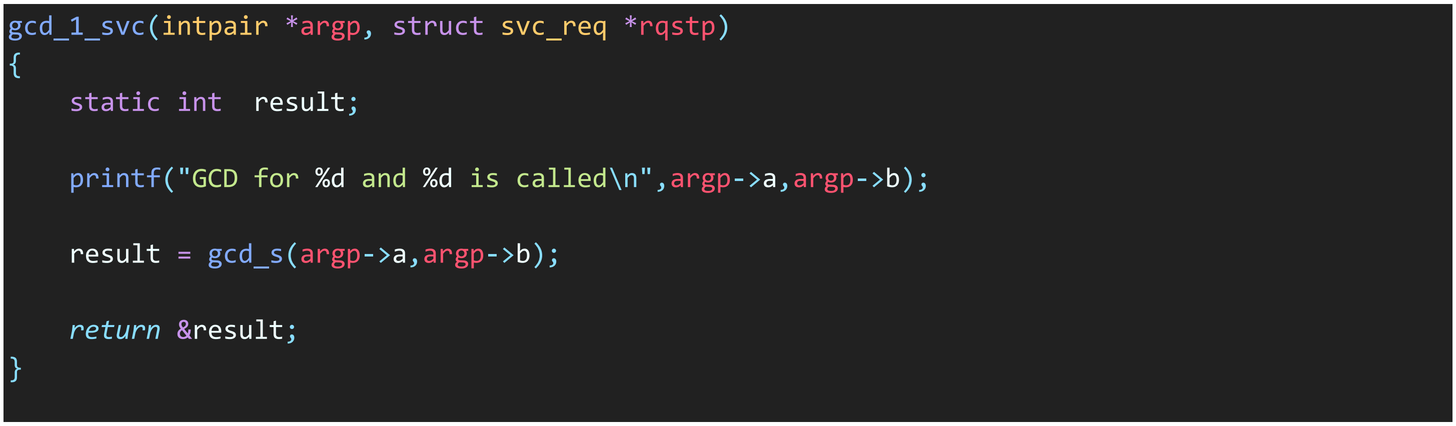


```
        clnt_perror (clnt, "call failed");
    }
    printf("gcd : %d\n",*result_1);
#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 4) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    int a = atoi(argv[2]);
    int b = atoi(argv[3]);
    gcd_prog_1 (host,a,b);
exit (0);
}
```

[q3_server.c]



```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q3.h"

int gcd_s(int a,int b){
    if(a==0){
        return b;
    }
    if(a>b){
        gcd_s(a%b,b);
    }
    else{
        gcd_s(b%a,a);
    }
}

int *
```

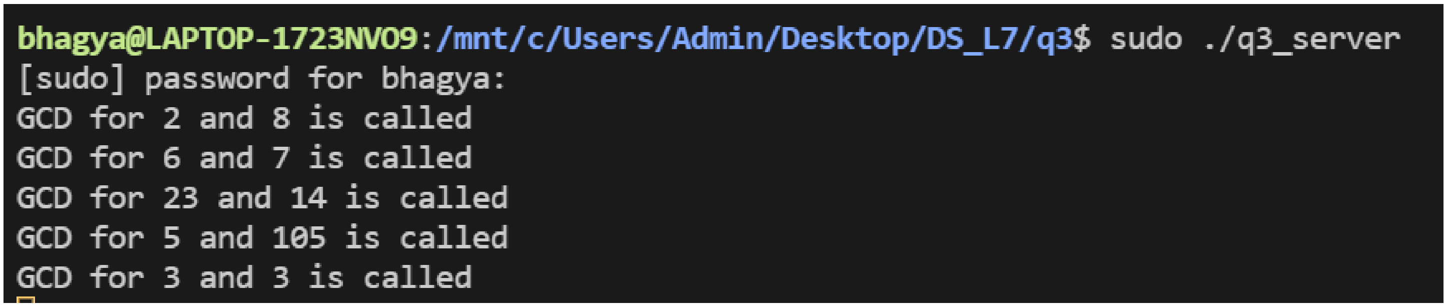


```
gcd_1_svc(intpair *argp, struct svc_req *rqstp)
{
    static int  result;

    printf("GCD for %d and %d is called\n",argp->a,argp->b);

    result = gcd_s(argp->a,argp->b);

    return &result;
}
```

Run Command : `make -f Makefile.q3`

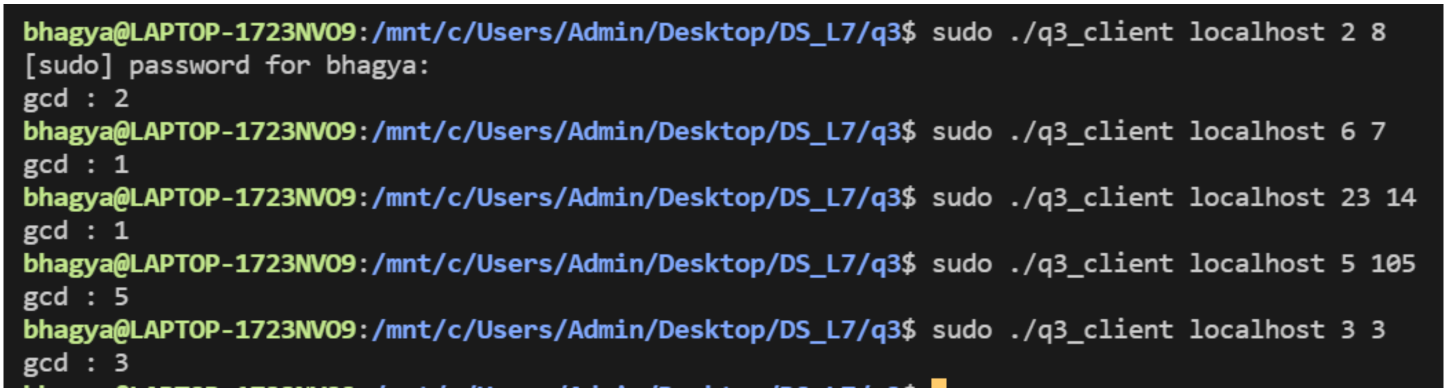
[**Output**]

Server:



```
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q3$ sudo ./q3_server
[sudo] password for bhagya:
GCD for 2 and 8 is called
GCD for 6 and 7 is called
GCD for 23 and 14 is called
GCD for 5 and 105 is called
GCD for 3 and 3 is called
```

Client:



```
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q3$ sudo ./q3_client localhost 2 8
[sudo] password for bhagya:
gcd : 2
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q3$ sudo ./q3_client localhost 6 7
gcd : 1
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q3$ sudo ./q3_client localhost 23 14
gcd : 1
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q3$ sudo ./q3_client localhost 5 105
gcd : 5
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q3$ sudo ./q3_client localhost 3 3
gcd : 3
```

4.) Find out the **Square root** of a given number.

[q4.x]

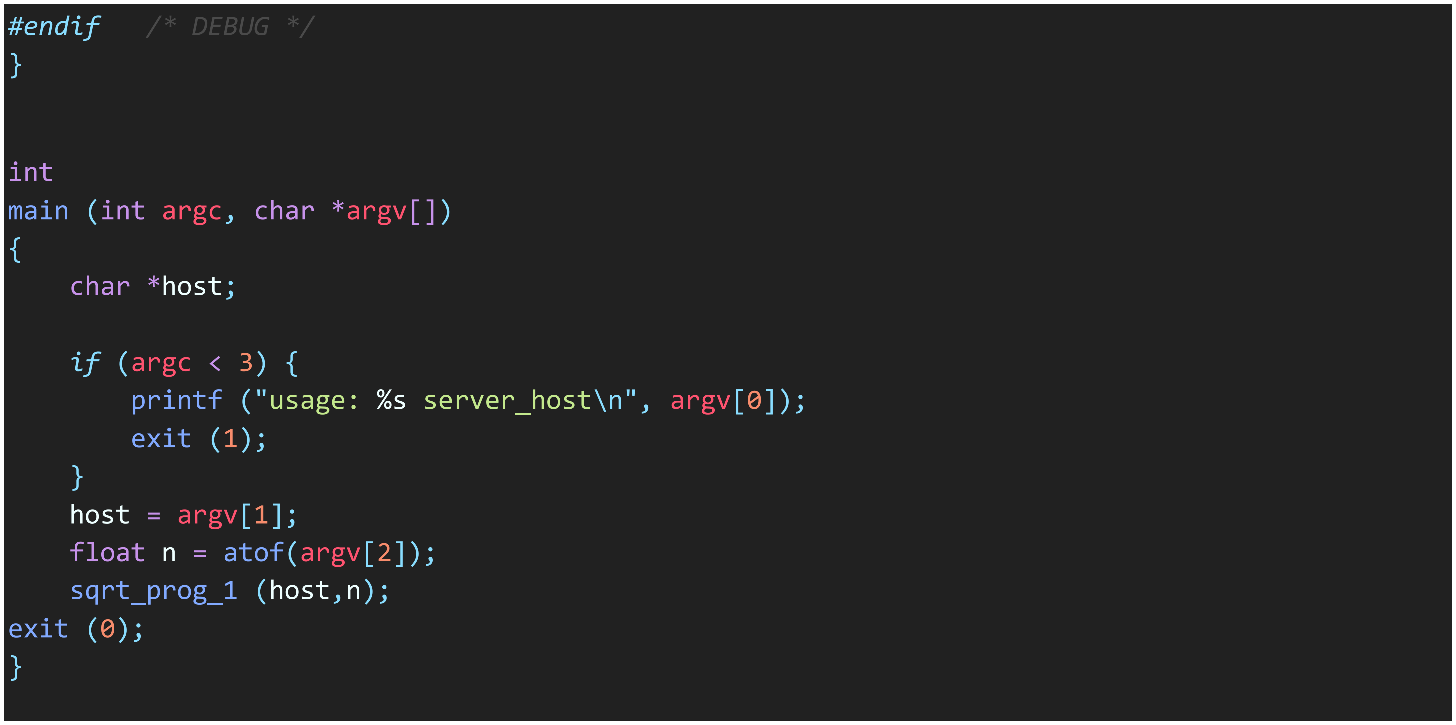
```
program SQRT_PROG{
    version VERSION1 {
        float sqrt(float n) = 1;
    } = 1;
} =  0x4562877;
```

Run Command : `rpcgen -a -C q4.x`

- ✓ All required files will be created.
- ✓ The q4_client.c and q4_server.c files would be modified as following:

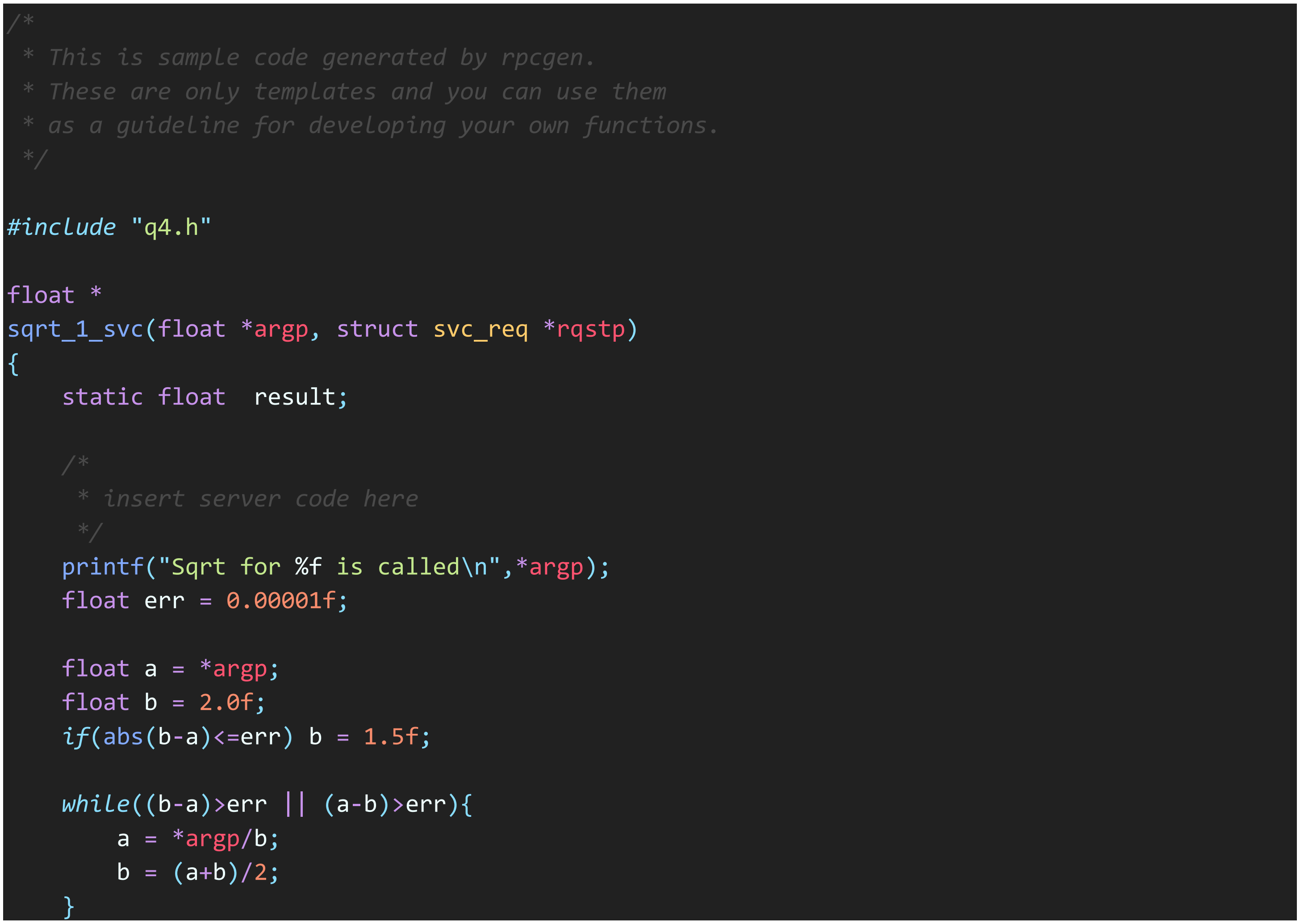
[q4_client.c]

```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q4.h"


void
sqrt_prog_1(char *host,float n)
{
    CLIENT *clnt;
    float  *result_1;
    float  sqrt_1_arg=n;

#ifndef DEBUG
    clnt = clnt_create (host, SQRT_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = sqrt_1(&sqrt_1_arg, clnt);
    if (result_1 == (float *) NULL) {
        clnt_perror (clnt, "call failed");
    }
    printf("sqrt : %f\n",*result_1);
#ifndef DEBUG
    clnt_destroy (clnt);
```



```
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 3) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    float n = atof(argv[2]);
    sqrt_prog_1 (host,n);
exit (0);
}
```

[q4_server.c]

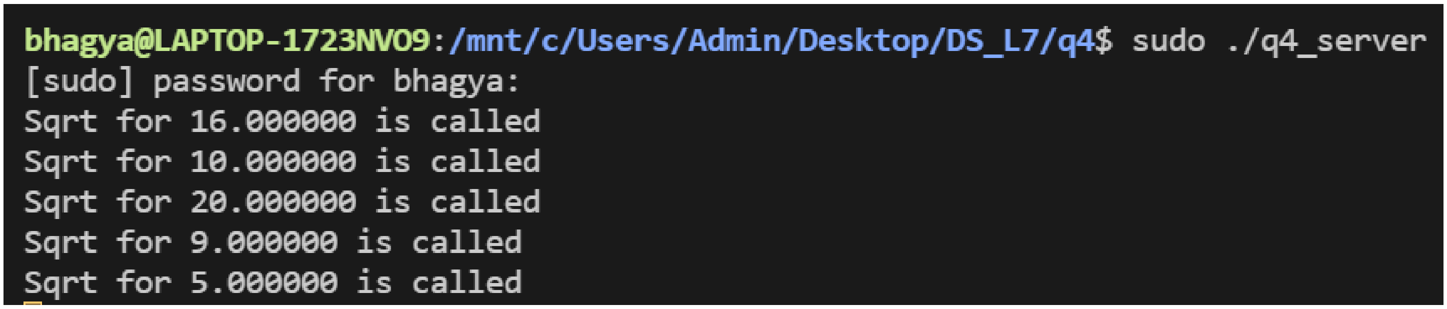


```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q4.h"

float *
sqrt_1_svc(float *argp, struct svc_req *rqstp)
{
    static float  result;

    /*
     * insert server code here
     */
    printf("Sqrt for %f is called\n",*argp);
    float err = 0.00001f;

    float a = *argp;
    float b = 2.0f;
    if(abs(b-a)<=err) b = 1.5f;

    while((b-a)>err || (a-b)>err){
        a = *argp/b;
        b = (a+b)/2;
    }
    result = a;
    return &result;
}
```

Run Command : `make -f Makefile.q4`

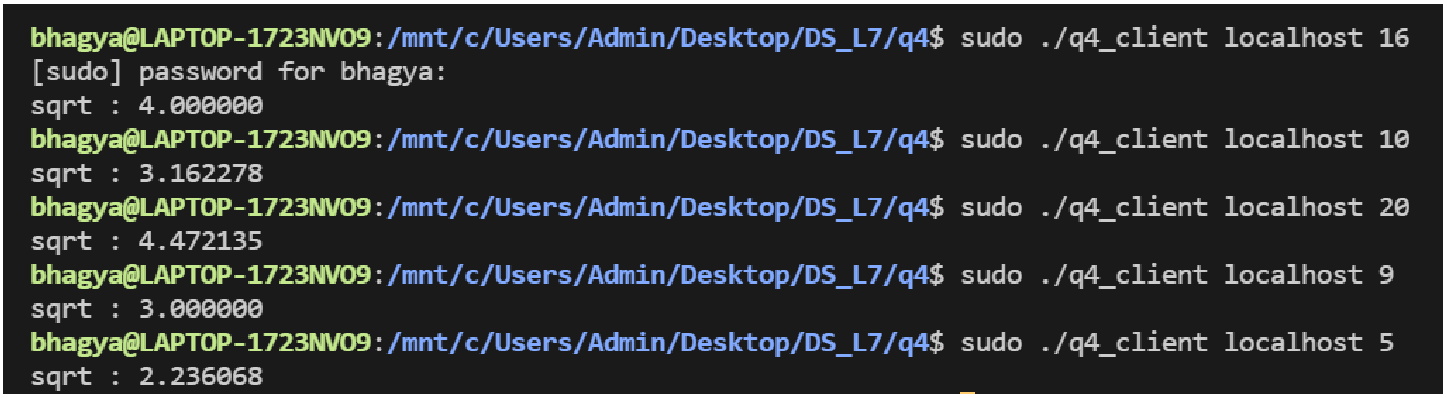
[Output]

Server:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q4$ sudo ./q4_server
[sudo] password for bhagya:
Sqrt for 16.000000 is called
Sqrt for 10.000000 is called
Sqrt for 20.000000 is called
Sqrt for 9.000000 is called
Sqrt for 5.000000 is called
```

Client:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q4$ sudo ./q4_client localhost 16
[sudo] password for bhagya:
sqrt : 4.000000
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q4$ sudo ./q4_client localhost 10
sqrt : 3.162278
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q4$ sudo ./q4_client localhost 20
sqrt : 4.472135
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q4$ sudo ./q4_client localhost 9
sqrt : 3.000000
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q4$ sudo ./q4_client localhost 5
sqrt : 2.236068
```

5.) **Swap two variables** without using the 3rd variable.

[q5.x]

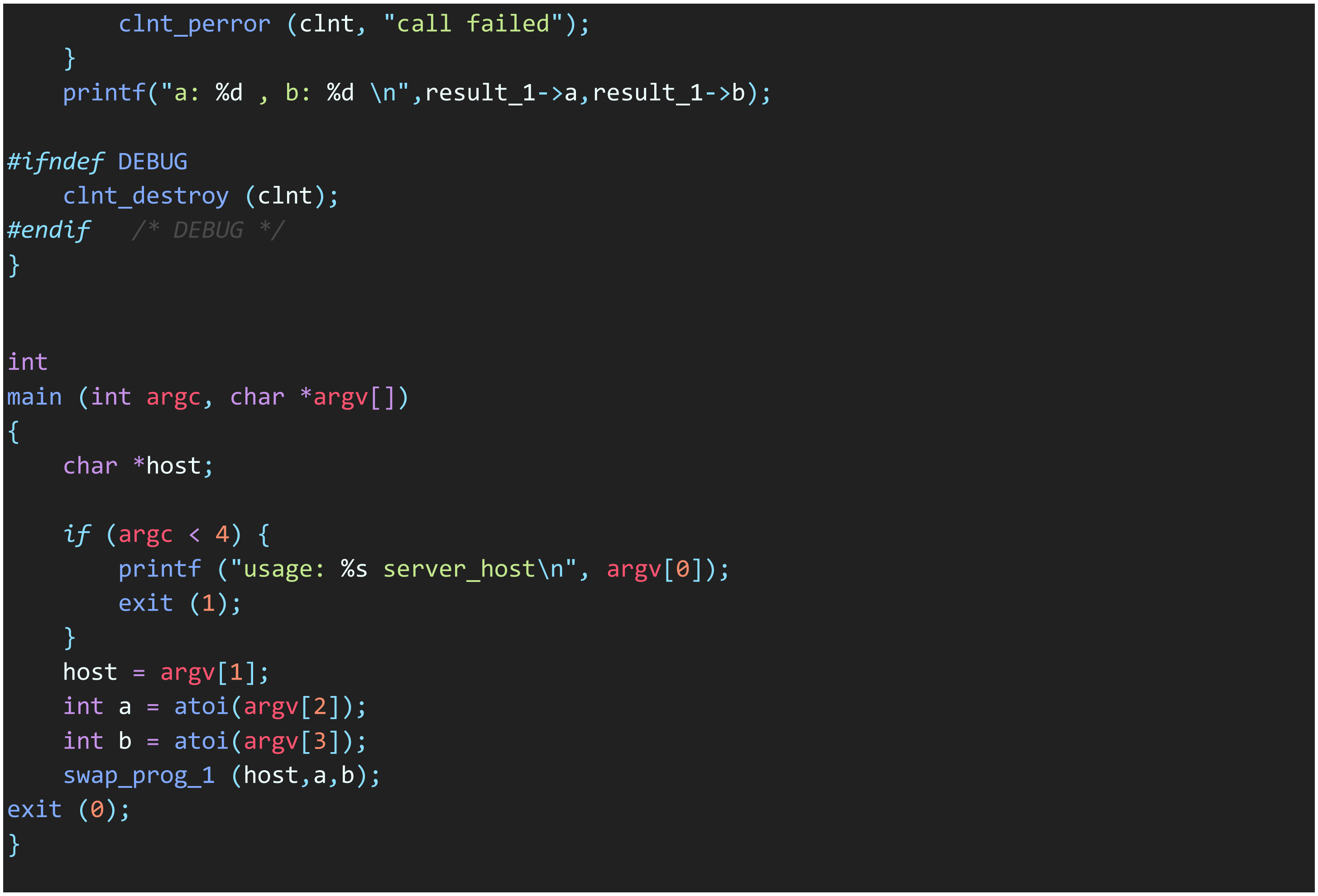
```
struct intpair{
    int a;
    int b;
};

program SWAP_PROG{
    version VERSION1 {
        intpair swap(intpair p) = 1;
    } = 1;
} =  0x4562877;
```

Run Command : `rpcgen -a -C q5.x`

- ✓ All required files will be created.
- ✓ The swap_client.c and swap_server.c files would be modified as following:

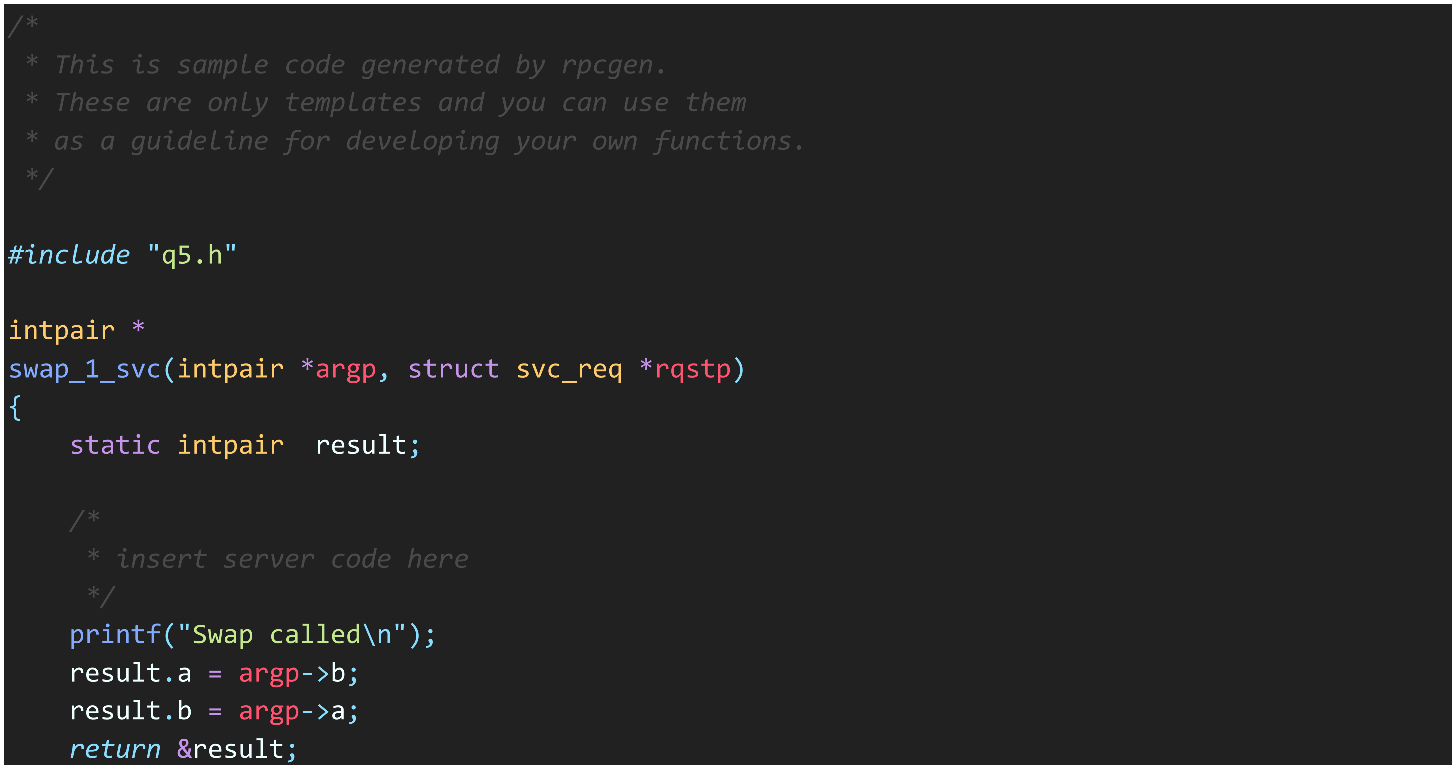
[q5_client.c]

```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q5.h"


void
swap_prog_1(char *host,int a,int b)
{
    CLIENT *clnt;
    intpair  *result_1;
    intpair  swap_1_arg = {a,b};

#ifndef DEBUG
    clnt = clnt_create (host, SWAP_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = swap_1(&swap_1_arg, clnt);
    if (result_1 == (intpair *) NULL) {
```



```
        clnt_perror (clnt, "call failed");
    }
    printf("a: %d , b: %d \n",result_1->a,result_1->b);

#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 4) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    int a = atoi(argv[2]);
    int b = atoi(argv[3]);
    swap_prog_1 (host,a,b);
exit (0);
}
```

[q5_server.c]

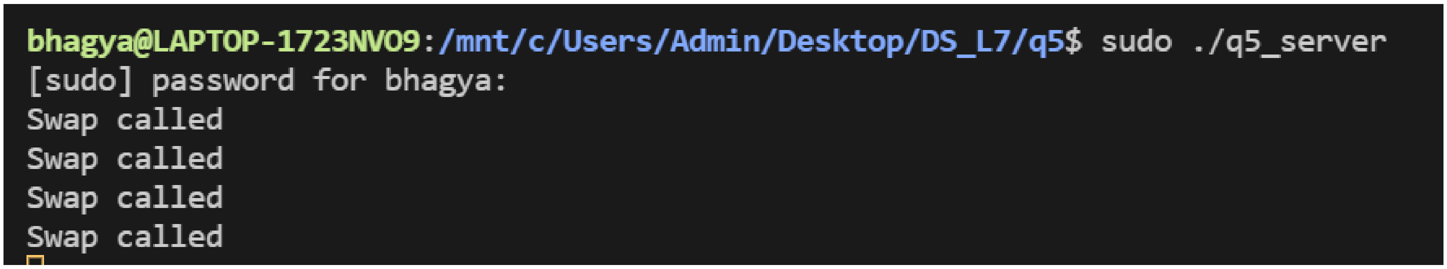


```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q5.h"

intpair *
swap_1_svc(intpair *argp, struct svc_req *rqstp)
{
    static intpair  result;

    /*
     * insert server code here
     */
    printf("Swap called\n");
    result.a = argp->b;
    result.b = argp->a;
    return &result;
}
```

Run Command : make -f Makefile.q5

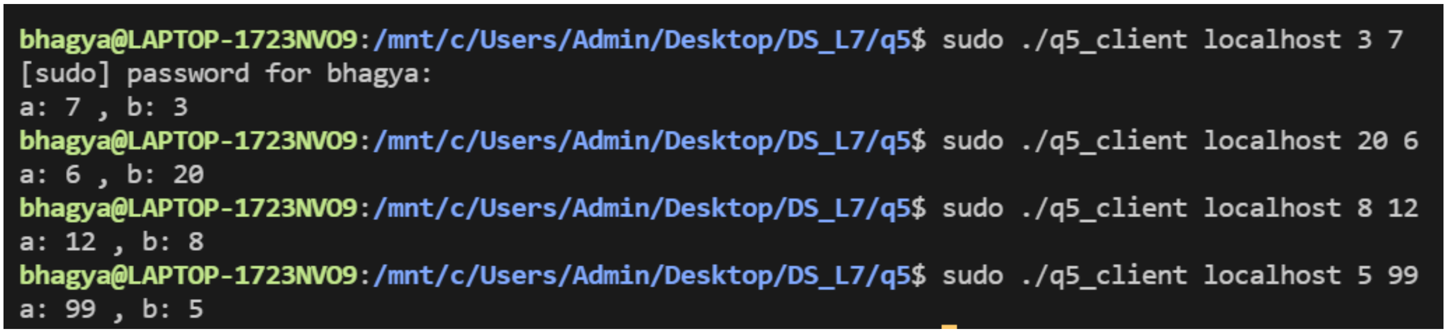
[Output]

Server:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q5$ sudo ./q5_server
[sudo] password for bhagya:
Swap called
Swap called
Swap called
Swap called
```

Client:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q5$ sudo ./q5_client localhost 3 7
[sudo] password for bhagya:
a: 7 , b: 3
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q5$ sudo ./q5_client localhost 20 6
a: 6 , b: 20
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q5$ sudo ./q5_client localhost 8 12
a: 12 , b: 8
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q5$ sudo ./q5_client localhost 5 99
a: 99 , b: 5
```

6.) Calculate **Maximum, Minimum, average** of given array.

[q6.x]

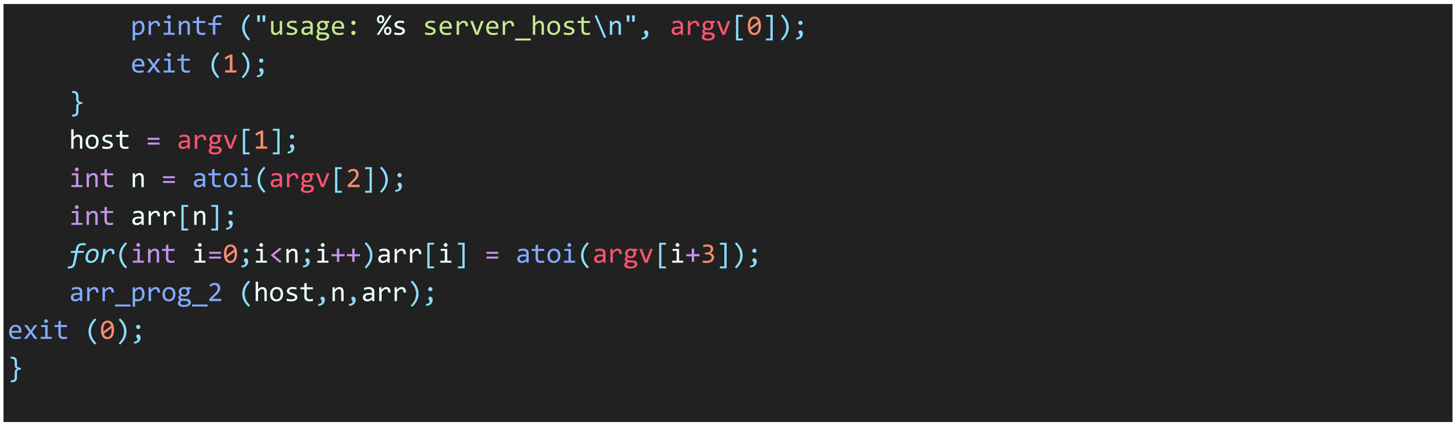
```
struct arraysize{
    int* arr;
    int size;
};

program ARR_PROG{
    version VERSION1 {
        arraysize arrop(arraysize arr) = 1;
    } = 2;
} =  0x20000005;
```

Run Command : `rpcgen -a -C q6.x`

- ✓ All required files will be created.
- ✓ The q6_client.c and q6_server.c files would be modified as following:

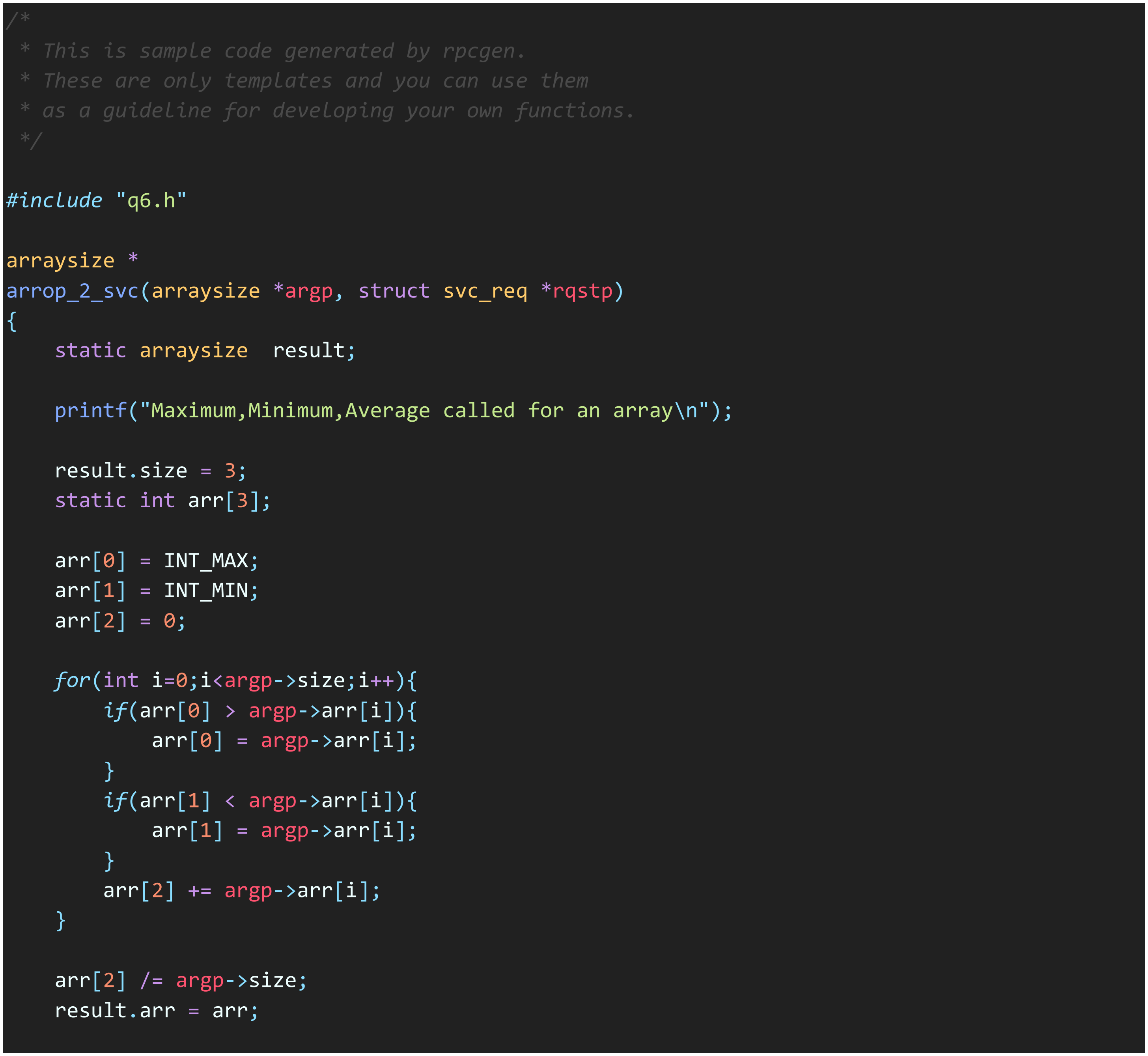
[q6_client.c]

```c
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q6.h"


void
arr_prog_2(char *host,int n,int* arr)
{
    CLIENT *clnt;
    arraysize  *result_1;
    arraysize  arrop_2_arg = {arr,n};

#ifndef DEBUG
    clnt = clnt_create (host, ARR_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = arrop_2(&arrop_2_arg, clnt);
    if (result_1 == (arraysize *) NULL) {
        clnt_perror (clnt, "call failed");
    }
    printf("Min:%d\nMax:%d\nAvg:%d\n",result_1->arr[0],result_1->arr[1],result_1->arr[2]);
#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 2) {
```



```
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    int n = atoi(argv[2]);
    int arr[n];
    for(int i=0;i<n;i++)arr[i] = atoi(argv[i+3]);
    arr_prog_2 (host,n,arr);
exit (0);
}
```

[q6_server.c]

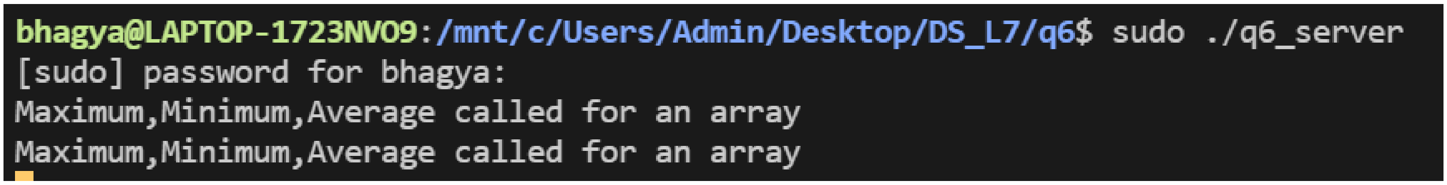


```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q6.h"

arraysize *
arrop_2_svc(arraysize *argp, struct svc_req *rqstp)
{
    static arraysize  result;

    printf("Maximum,Minimum,Average called for an array\n");

    result.size = 3;
    static int arr[3];

    arr[0] = INT_MAX;
    arr[1] = INT_MIN;
    arr[2] = 0;

    for(int i=0;i<argp->size;i++){
        if(arr[0] > argp->arr[i]){
            arr[0] = argp->arr[i];
        }
        if(arr[1] < argp->arr[i]){
            arr[1] = argp->arr[i];
        }
        arr[2] += argp->arr[i];
    }

    arr[2] /= argp->size;
    result.arr = arr;
    return &result;
}
```

Run Command : make -f Makefile.q6

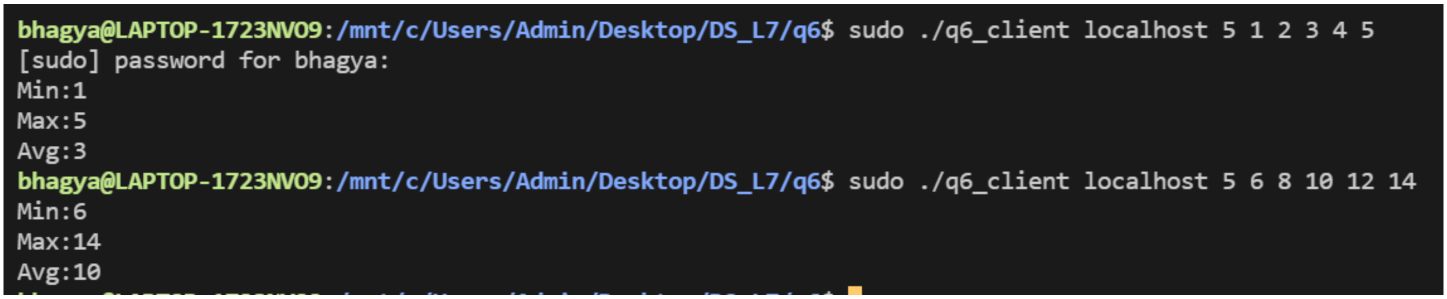
[Output]

Server:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q6$ sudo ./q6_server
[sudo] password for bhagya:
Maximum,Minimum,Average called for an array
Maximum,Minimum,Average called for an array
```

Client:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q6$ sudo ./q6_client localhost 5 1 2 3 4 5
[sudo] password for bhagya:
Min:1
Max:5
Avg:3
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q6$ sudo ./q6_client localhost 5 6 8 10 12 14
Min:6
Max:14
Avg:10
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q6$
```

7.) Compare the given two strings.

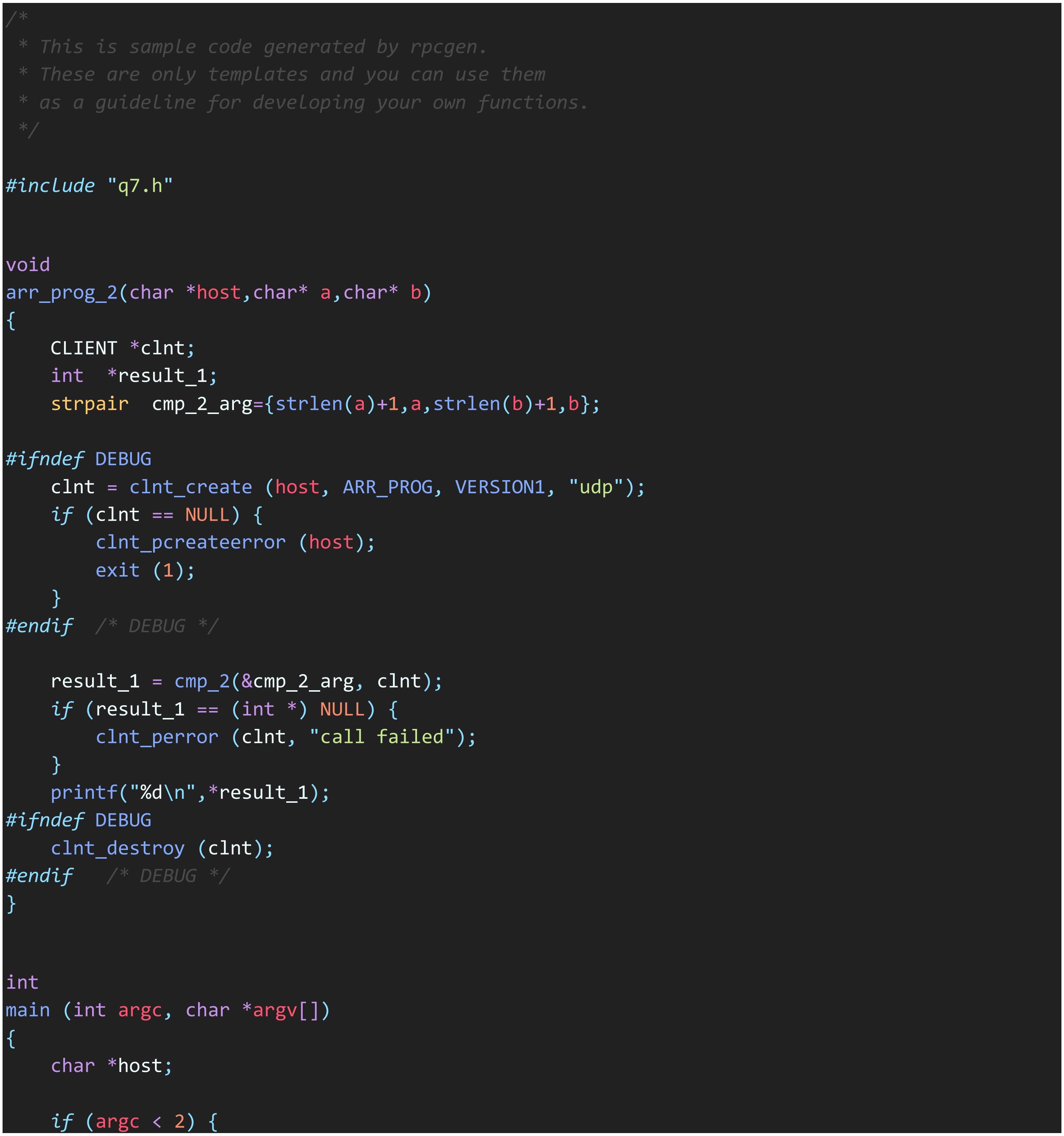
[q7.x]

```
struct strpair{
    int len1;
    char* a;
    int len2;
    char* b;
};

program ARR_PROG{
    version VERSION1 {
        int cmp(strpair p) = 1;
    } = 2;
} =  0x20000007;
```

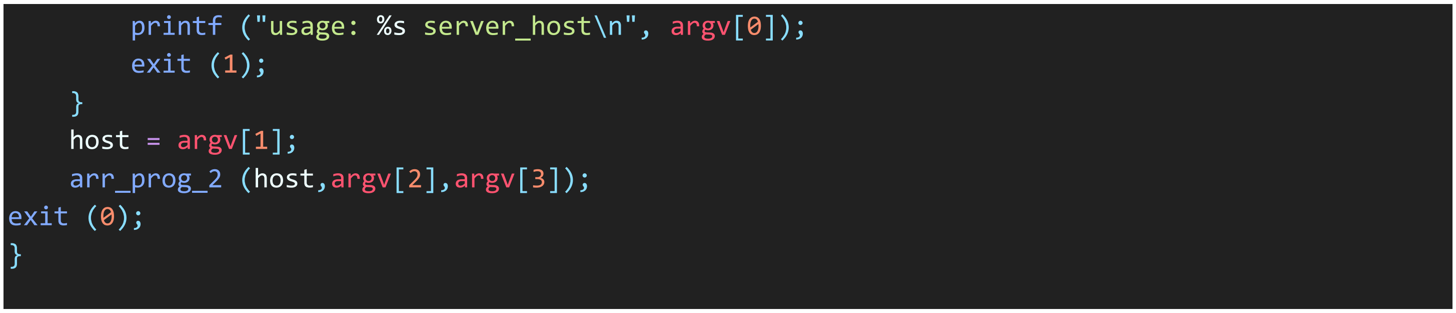
Run Command : rpcgen -a -C q7.x

- ✓ All required files will be created.
- ✓ The compare_client.c and compare_server.c files would be modified as following:

[q7_client.c]

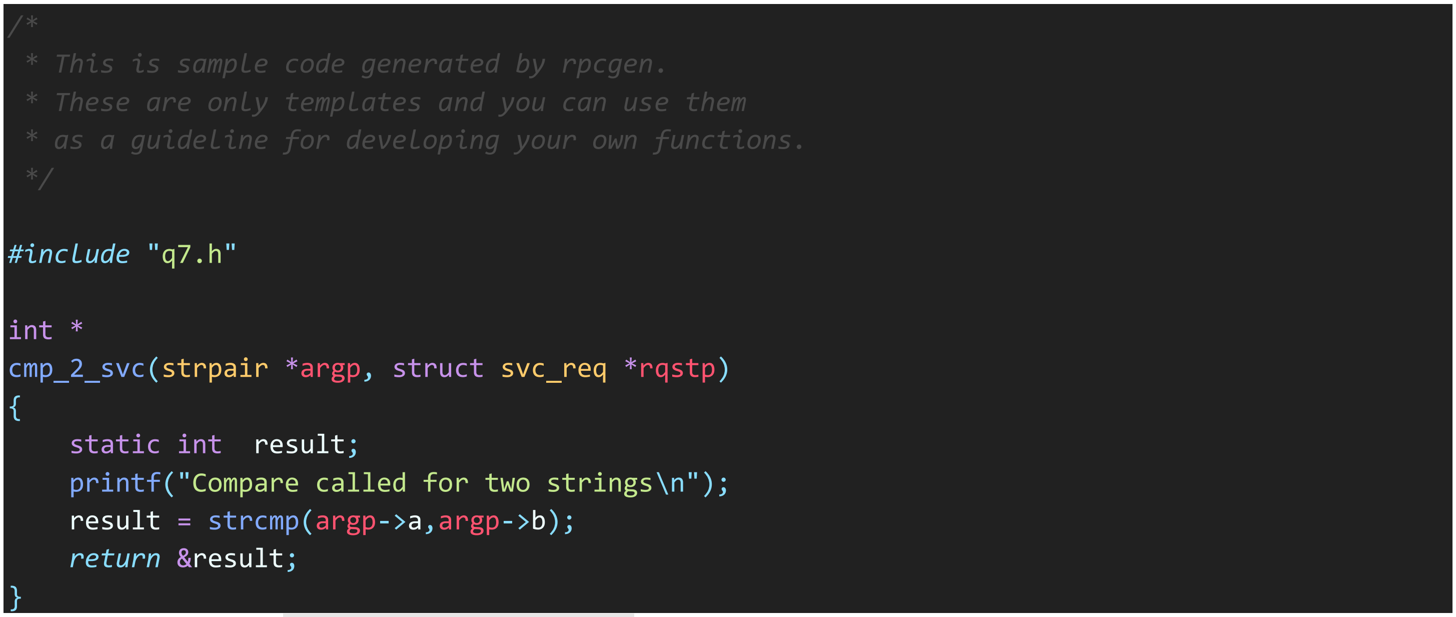


```c
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q7.h"


void
arr_prog_2(char *host,char* a,char* b)
{
    CLIENT *clnt;
    int  *result_1;
    strpair  cmp_2_arg={strlen(a)+1,a,strlen(b)+1,b};

#ifndef DEBUG
    clnt = clnt_create (host, ARR_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = cmp_2(&cmp_2_arg, clnt);
    if (result_1 == (int *) NULL) {
        clnt_perror (clnt, "call failed");
    }
    printf("%d\n",*result_1);
#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 2) {
```



```
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    arr_prog_2 (host,argv[2],argv[3]);
exit (0);
}
```

[q7_server.c]

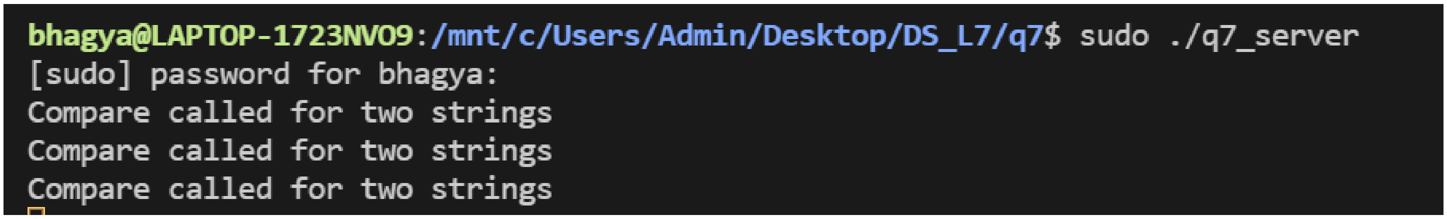


```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q7.h"

int *
cmp_2_svc(strpair *argp, struct svc_req *rqstp)
{
    static int  result;
    printf("Compare called for two strings\n");
    result = strcmp(argp->a,argp->b);
    return &result;
}
```

Run Command : `make -f Makefile.q7`

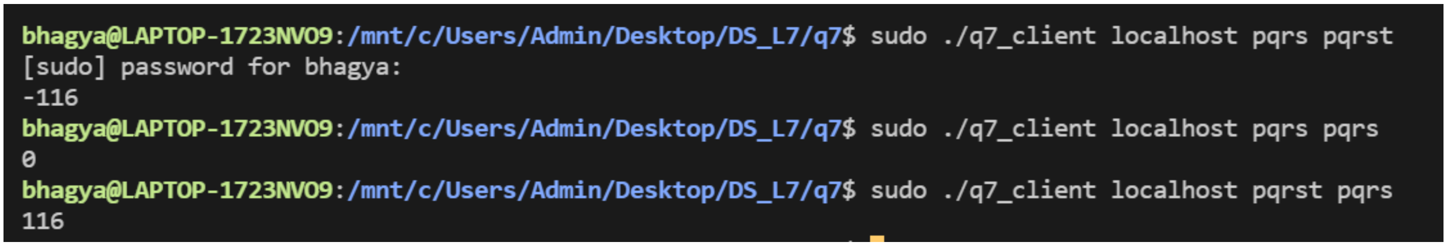
[Output]

Server:



```
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q7$ sudo ./q7_server
[sudo] password for bhagya:
Compare called for two strings
Compare called for two strings
Compare called for two strings
```

Client:



```
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q7$ sudo ./q7_client localhost pqrs pqrst
[sudo] password for bhagya:
-116
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q7$ sudo ./q7_client localhost pqrs pqrs
0
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q7$ sudo ./q7_client localhost pqrst pqrs
116
```

8.) Find out whether a given string is **substring or not**.

[q8.x]

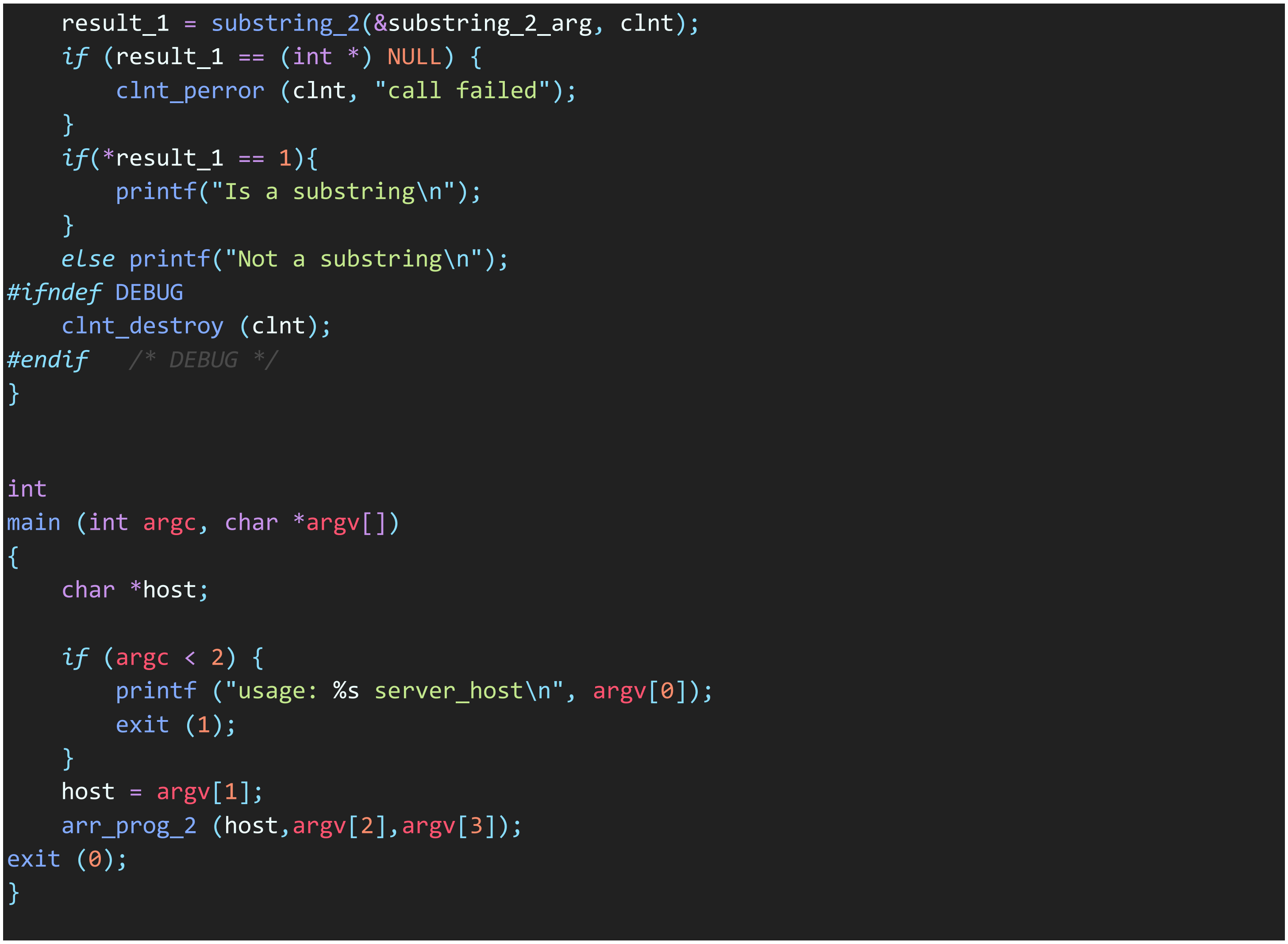
```
struct strpair{
    int len1;
    char* a;
    int len2;
    char* b;
};

program ARR_PROG{
    version VERSION1 {
        int substring(strpair p) = 1;
    } = 2;
} =  0x20000007;
```

Run Command : `rpcgen -a -C q8.x`

- ✓ All required files will be created.
- ✓ The q8_client.c and q8_server.c files would be modified as following:

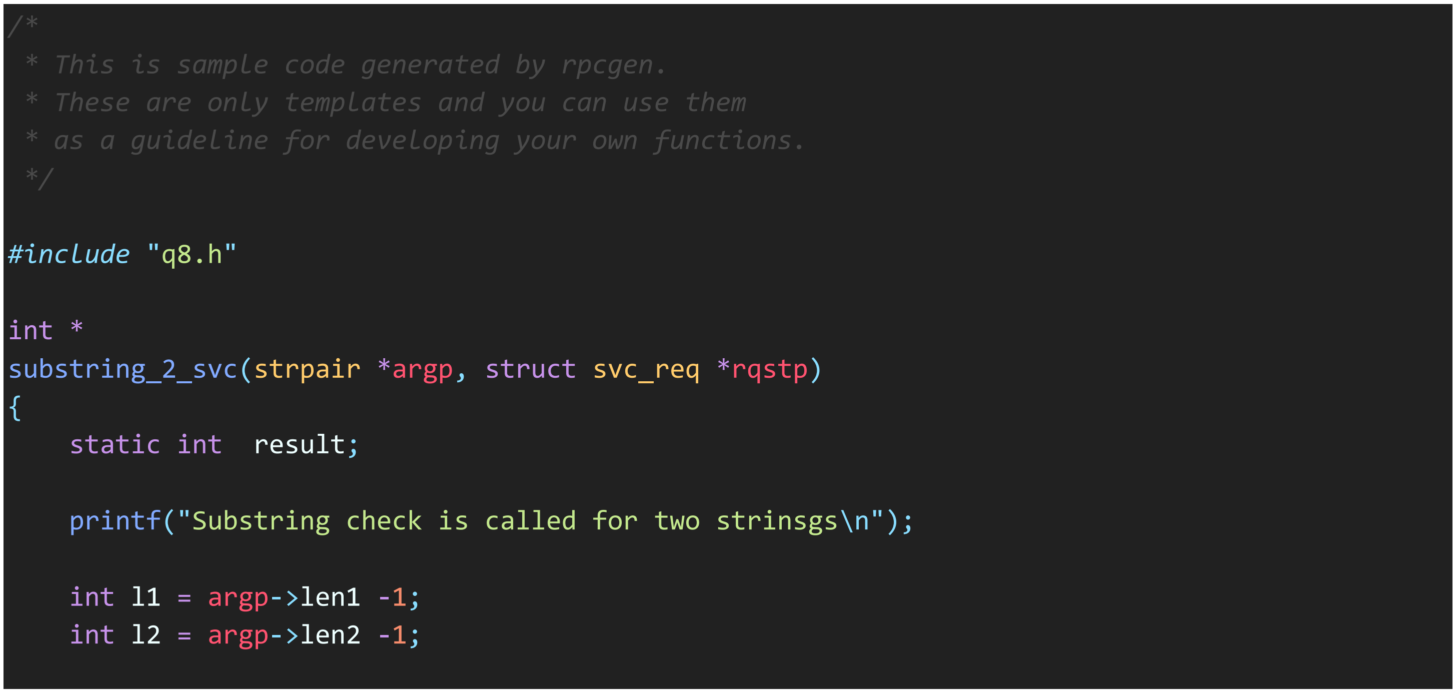
[q8_client.c]

```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q8.h"


void
arr_prog_2(char *host,char* a,char* b)
{
    CLIENT *clnt;
    int  *result_1;
    strpair  substring_2_arg = {strlen(a)+1,a,strlen(b)+1,b};

#ifndef DEBUG
    clnt = clnt_create (host, ARR_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */
```

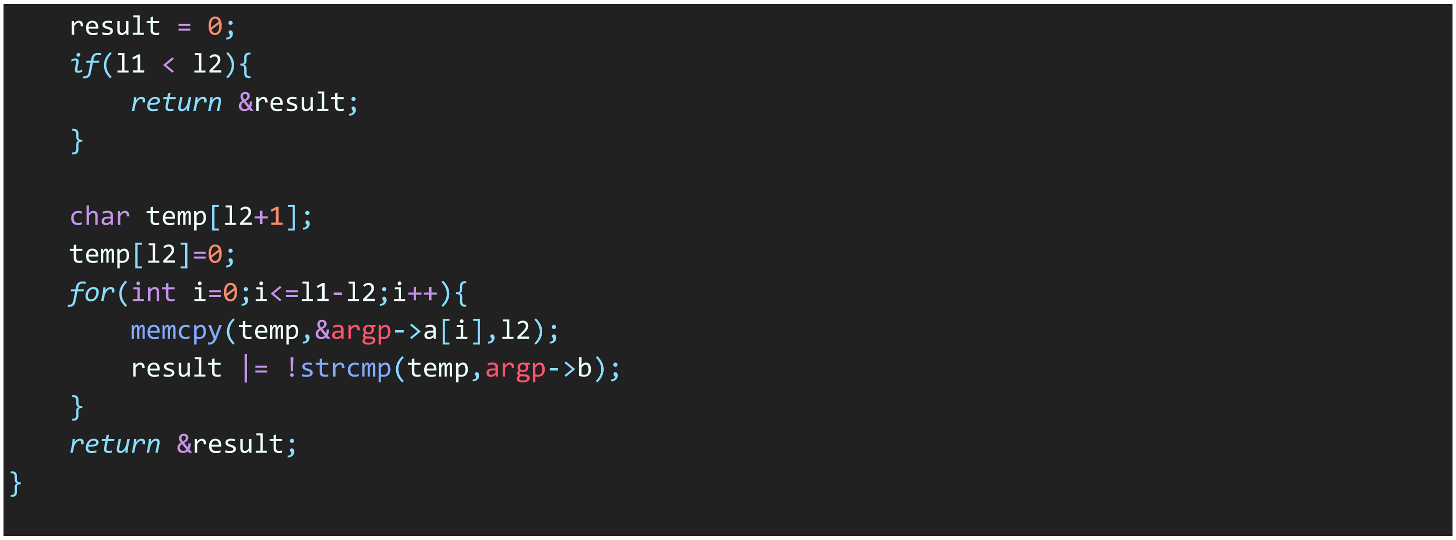


```
    result_1 = substring_2(&substring_2_arg, clnt);
    if (result_1 == (int *) NULL) {
        clnt_perror (clnt, "call failed");
    }
    if(*result_1 == 1){
        printf("Is a substring\n");
    }
    else printf("Not a substring\n");
#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 2) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    arr_prog_2 (host,argv[2],argv[3]);
exit (0);
}
```

[q8_server.c]



```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q8.h"

int *
substring_2_svc(strpair *argp, struct svc_req *rqstp)
{
    static int  result;

    printf("Substring check is called for two strinsgs\n");

    int l1 = argp->len1 -1;
    int l2 = argp->len2 -1;
```

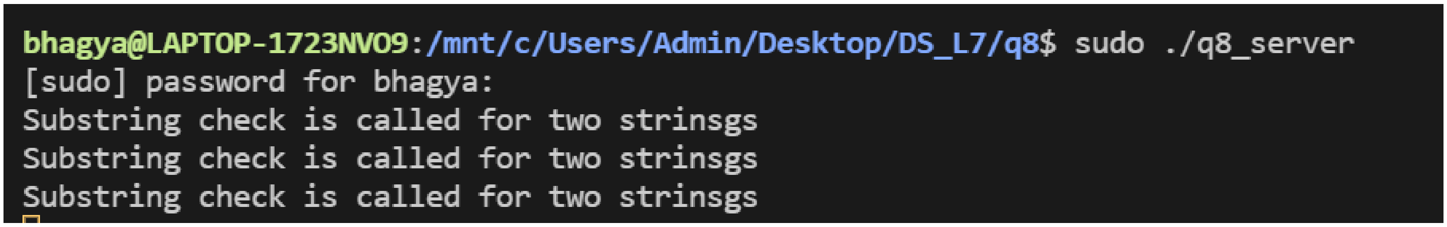


```
    result = 0;
    if(l1 < l2){
        return &result;
    }

    char temp[l2+1];
    temp[l2]=0;
    for(int i=0;i<=l1-l2;i++){
        memcpy(temp,&argp->a[i],l2);
        result |= !strcmp(temp,argp->b);
    }
    return &result;
}
```

Run Command : make -f Makefile.q8

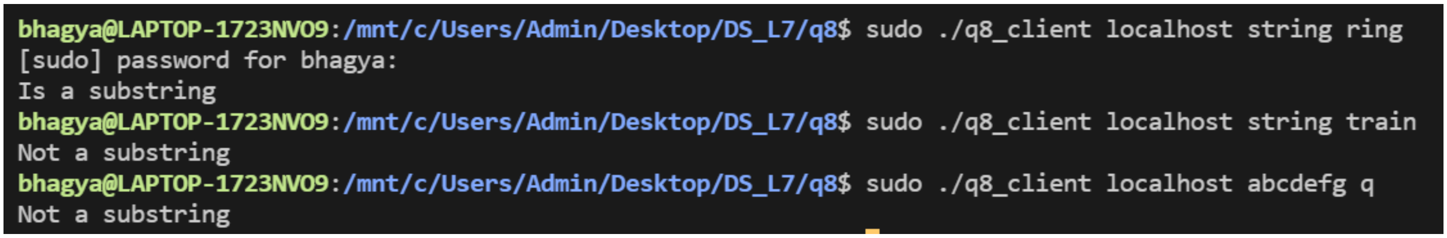
[Output]

Server:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q8$ sudo ./q8_server
[sudo] password for bhagya:
Substring check is called for two strinsgs
Substring check is called for two strinsgs
Substring check is called for two strinsgs
```

Client:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q8$ sudo ./q8_client localhost string ring
[sudo] password for bhagya:
Is a substring
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q8$ sudo ./q8_client localhost string train
Not a substring
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q8$ sudo ./q8_client localhost abcdefg q
Not a substring
```

9.) **Concatenate** the two strings.

[q9.x]

```
struct strpair{
    int len1;
    char* a;
    int len2;
    char* b;
};

program CONCAT_PROG{
    version VERSION1 {
        string concatenate(strpair p) = 1;
    } = 1;
} =  0x4562877;
```

Run Command : rpcgen -a -C q9.x

- ✓ All required files will be created.
- ✓ The q9_client.c and q9_server.c files would be modified as following:

[q9_client.c]

```
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q9.h"


void
concat_prog_1(char *host,char* a,char* b)
{
    CLIENT *clnt;
    char * *result_1;
    strpair  concatenate_1_arg = {strlen(a)+1,a,strlen(b)+1,b};

#ifndef DEBUG
    clnt = clnt_create (host, CONCAT_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */
```

```c
    result_1 = concatenate_1(&concatenate_1_arg, clnt);
    if (result_1 == (char **) NULL) {
        clnt_perror (clnt, "call failed");
    }
    printf("%s\n",*result_1);
#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 2) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    concat_prog_1 (host,argv[2],argv[3]);
exit (0);
}
```

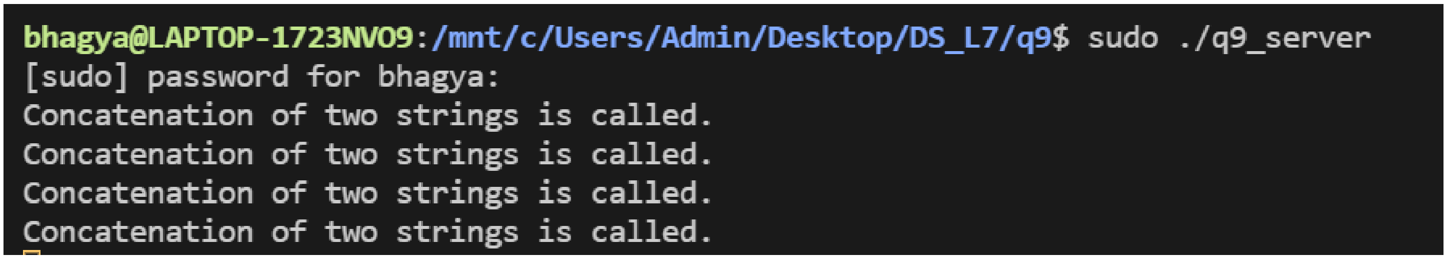
[q9_server.c]

```c
/*
 * This is sample code generated by rpcgen.
 * These are only templates and you can use them
 * as a guideline for developing your own functions.
 */

#include "q9.h"

char **
concatenate_1_svc(strpair *argp, struct svc_req *rqstp)
{
    static char * result;
    printf("Concatenation of two strings is called. \n");
    result = malloc(argp->len1 + argp->len2 +1);
    memcpy(result,argp->a,argp->len1);
    strcat(result,argp->b);
    return &result;
}
```

Run Command : `make -f Makefile.q9`

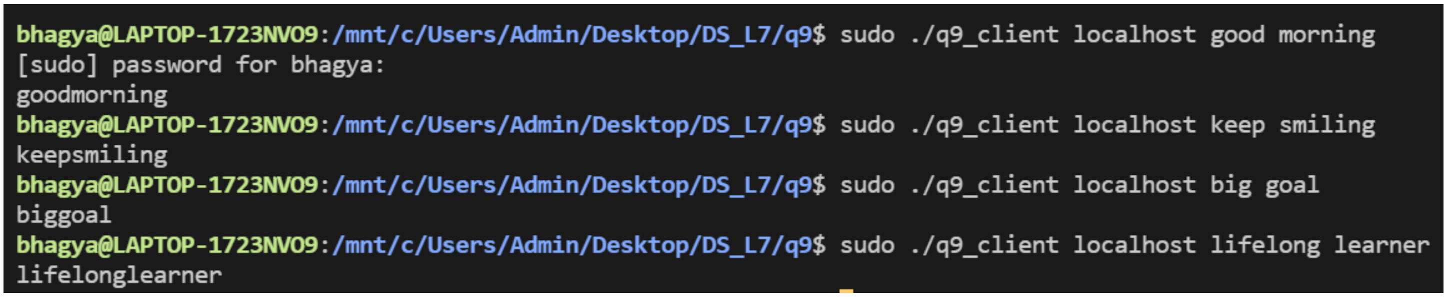
[**Output**]

Server:



```
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q9$ sudo ./q9_server
[sudo] password for bhagya:
Concatenation of two strings is called.
Concatenation of two strings is called.
Concatenation of two strings is called.
Concatenation of two strings is called.
```

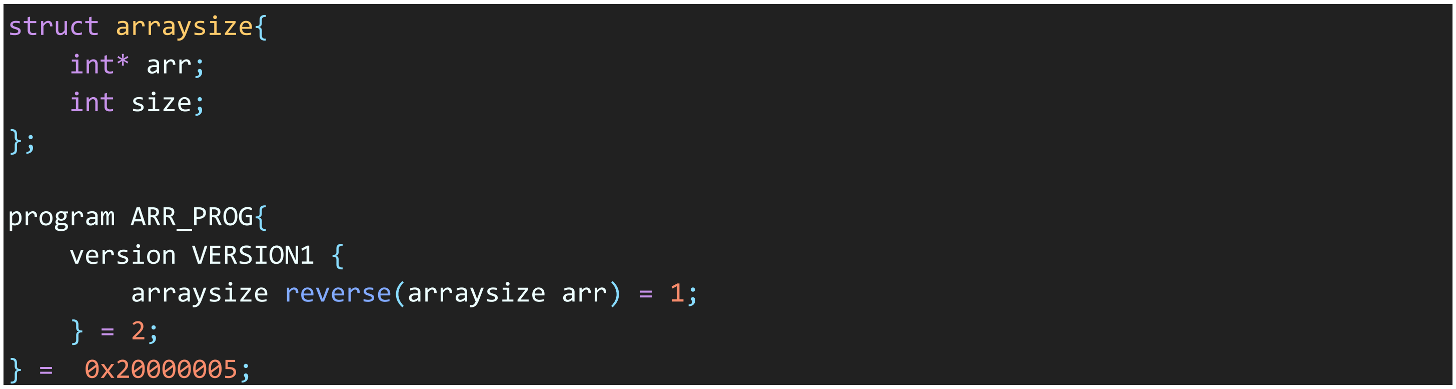
Client:



```
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q9$ sudo ./q9_client localhost good morning
[sudo] password for bhagya:
goodmorning
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q9$ sudo ./q9_client localhost keep smiling
keepsmiling
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q9$ sudo ./q9_client localhost big goal
biggoal
bhagya@LAPTOP-1723NV09:/mnt/c/Users/Admin/Desktop/DS_L7/q9$ sudo ./q9_client localhost lifelong learner
lifelonglearner
```

10.) **Reverse** the elements of an array.

[q10.x]



```
struct arraysize{
    int* arr;
    int size;
};

program ARR_PROG{
    version VERSION1 {
        arraysize reverse(arraysize arr) = 1;
    } = 2;
} =  0x20000005;
```

Run Command : `rpcgen -a -C q10.x`

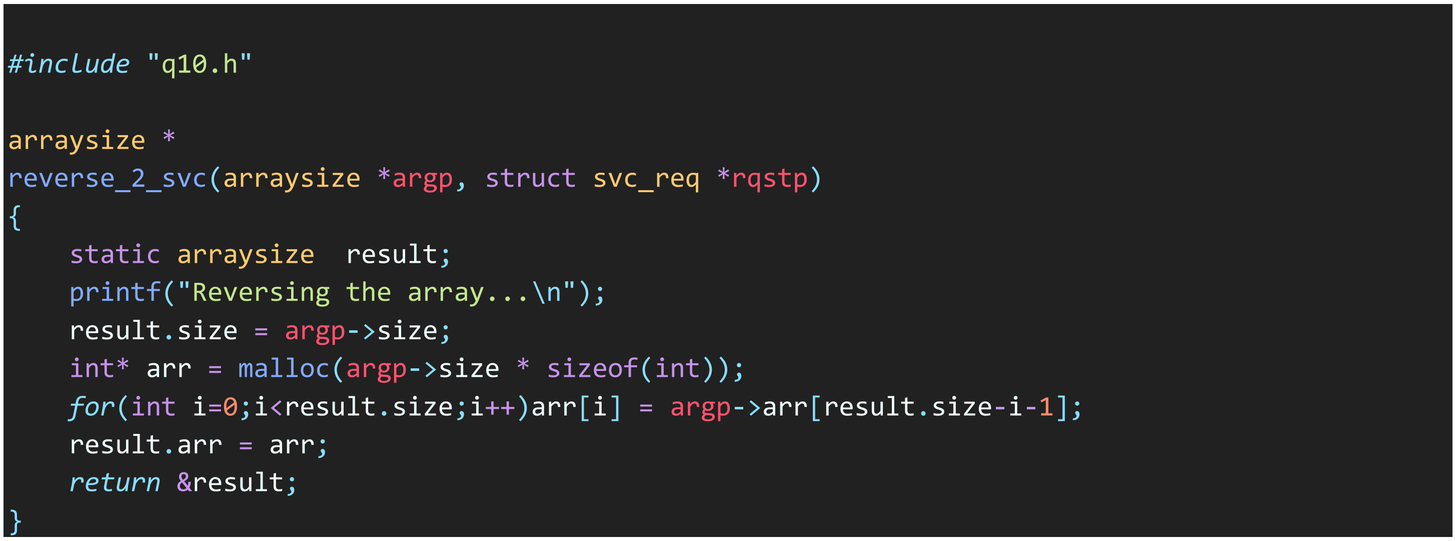
- ✓ All required files will be created.
- ✓ The q10_client.c and q10_server.c files would be modified as following:

[q10_client.c]

```c
#include "q10.h"
void
arr_prog_2(char *host,int n,int* arr)
{
    CLIENT *clnt;
    arraysize  *result_1;
    arraysize  reverse_2_arg = {arr,n};

#ifndef DEBUG
    clnt = clnt_create (host, ARR_PROG, VERSION1, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif  /* DEBUG */

    result_1 = reverse_2(&reverse_2_arg, clnt);
    if (result_1 == (arraysize *) NULL) {
        clnt_perror (clnt, "call failed");
    }
    for(int i=0;i<result_1->size;i++)printf("%d ",result_1->arr[i]);
    printf("\n");
#ifndef DEBUG
    clnt_destroy (clnt);
#endif   /* DEBUG */
}


int
main (int argc, char *argv[])
{
    char *host;

    if (argc < 2) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
    }
    host = argv[1];
    int n = atoi(argv[2]);
    int arr[n];
    for(int i=0;i<n;i++)arr[i] = atoi(argv[i+3]);
    arr_prog_2 (host,n,arr);
exit (0);
```

```
}
```

[q10_server.c]

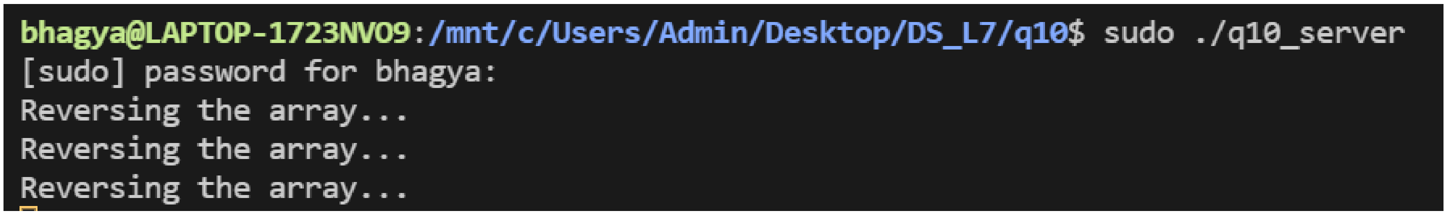


```
#include "q10.h"

arraysize *
reverse_2_svc(arraysize *argp, struct svc_req *rqstp)
{
    static arraysize  result;
    printf("Reversing the array...\n");
    result.size = argp->size;
    int* arr = malloc(argp->size * sizeof(int));
    for(int i=0;i<result.size;i++)arr[i] = argp->arr[result.size-i-1];
    result.arr = arr;
    return &result;
}
```

Run Command : make -f Makefile.q10

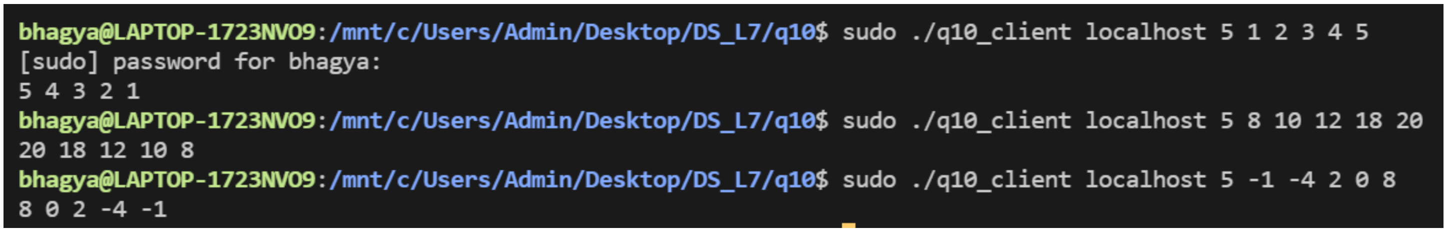
[Output]

Server:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q10$ sudo ./q10_server
[sudo] password for bhagya:
Reversing the array...
Reversing the array...
Reversing the array...
```

Client:



```
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q10$ sudo ./q10_client localhost 5 1 2 3 4 5
[sudo] password for bhagya:
5 4 3 2 1
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q10$ sudo ./q10_client localhost 5 8 10 12 18 20
20 18 12 10 8
bhagya@LAPTOP-1723NVO9:/mnt/c/Users/Admin/Desktop/DS_L7/q10$ sudo ./q10_client localhost 5 -1 -4 2 0 8
8 0 2 -4 -1
```

SUBMITTED BY: U19CS012

BHAGYA VINOD RANA