

2020203002 박상천

```
psc@ubuntu:~$ ls
2020203002 Desktop execve main.out Music Public sigaction stack.out wait.c
alarm.c Documents execve.c main.sh myfifo queue.cpp sigaction.c string.cpp
alarm.c Downloads execvp malloc.c MyStudent.cpp queue.out signal string.out
array1.cpp dup2.c execvp.c malloc.out MyStudent.hpp realloc1.c signal.c StudentList.dat
array1.out dup2.out fork.c mecro1 pause realloc1.out sigprocmask Templates
array.cpp execl.c fork.c mecro1.c pause.c realloc2.c sigprocmask.c vector.cpp
array.out execl.c list.cpp mecro2 Pictures realloc2.out sigset_t vector.out
client.c execl list.out mecro2.c pipe.c server.c sigset_t.c Videos
client.out execl.c main.cpp nkfifo.out pipe.out server.out stack.cpp wait
psc@ubuntu:~$ cat stack.cpp
#include <iostream>
#include <stack>
#include <vector>

using namespace std;

int main(int argc, char const *argv[]) {
    vector<int> myVector(2, 100);
    stack<int> myStack1;
    stack<int, vector<int>> myStack2(myVector);

    cout << "Size of myStack1: " << myStack1.size() << endl;
    if (myStack1.empty() == 1) {
        cout << "myStack1 is empty" << endl;
    }

    cout << "Size of myStack2: " << myStack2.size() << endl;
    cout << "Top if myStack2: " << myStack2.top() << endl;
    cout << endl;
    myStack1.push(1);
    myStack1.push(2);
    myStack1.push(3);
    cout << "---After push ---" << endl;
    cout << "Size of myStack1: " << myStack1.size() << endl;
    cout << "Top of myStack1: " << myStack1.top() << endl;
    cout << endl;

    myStack1.pop();
    myStack2.pop();

    cout << "--- After pop ---" << endl;
    cout << "Size of myStack1: " << myStack1.size() << endl;
    cout << "Top of myStack1: " << myStack1.top() << endl;
    cout << "Size of myStack2: " << myStack2.size() << endl;
    cout << "Top of myStack2: " << myStack2.top() << endl;

    return 0;
}

psc@ubuntu:~$ ./stack.out
Size of myStack1: 0
myStack1 is empty
Size of myStack2: 2
Top if myStack2: 100

---After push ---
Size of myStack1: 3
Top of myStack1: 3

--- After pop ---
Size of myStack1: 2
Top of myStack1: 2
Size of myStack2: 1
Top of myStack2: 100
psc@ubuntu:~$
```

```
psc@ubuntu: ~  
psc@ubuntu:~$ cat queue.cpp  
#include <iostream>  
#include <list>  
#include <queue>  
  
using namespace std;  
  
int main(int argc, char const *argv[]) {  
    list<int> myList(2,100);  
    queue<int> myQueue1;  
    queue<int, list<int>> myQueue2(myList);  
  
    cout << "Size of myQueue1 : " << myQueue1.size() << endl;  
    if(myQueue1.empty() == 1){  
        cout << "myQueue is empty" << endl << endl;  
    }  
  
    cout << "Size of myQueue2 : " << myQueue2.size() << endl;  
    cout << "Front of myQueue2 : " << myQueue2.front() << endl;  
    cout << "Back of myQueue2 : " << myQueue2.back() << endl;  
    cout << endl;  
  
    myQueue1.push(1);  
    myQueue1.push(2);  
    myQueue1.push(3);  
    myQueue2.push(200);  
  
    cout << "--- After push ---" << endl;  
    cout << "Size of myQueue1 : " << myQueue1.size() << endl;  
    cout << "Front of myQueue1 : " << myQueue1.front() << endl;  
    cout << "Back of myQueue1 : " << myQueue1.back() << endl << endl;  
    cout << "Size of myQueue2 : " << myQueue2.size() << endl;  
    cout << "Front of myQueue2 : " << myQueue2.front() << endl;  
    cout << "Back of myQueue2 : " << myQueue2.back() << endl;  
    cout << endl;  
  
    myQueue1.pop();  
    myQueue2.pop();  
    myQueue2.pop();  
  
    cout << "---After pop ---" << endl;  
    cout << "Size of myQueue1 : " << myQueue1.size() << endl;  
    cout << "Front of myQueue1 : " << myQueue1.front() << endl;  
    cout << "Back of myQueue1 : " << myQueue1.back() << endl << endl;  
    cout << "Size of myQueue2 : " << myQueue2.size() << endl;  
    cout << "Front of myQueue2 : " << myQueue2.front() << endl;  
    cout << "Back of myQueue2 : " << myQueue2.back() << endl;  
  
    return 0;  
}  
psc@ubuntu:~$ ./queue.out  
Size of myQueue1 : 0  
myQueue is empty  
  
Size of myQueue2 : 2  
Front of myQueue2 : 100  
Back of myQueue2 : 100  
  
--- After push ---  
Size of myQueue1 : 3  
Front of myQueue1 : 1  
Back of myQueue1 : 3  
  
Size of myQueue2 : 3  
Front of myQueue2 : 100  
Back of myQueue2 : 200  
  
---After pop ---  
Size of myQueue1 : 2  
Front of myQueue1 : 2  
Back of myQueue1 : 3  
  
Size of myQueue2 : 1  
Front of myQueue2 : 200  
Back of myQueue2 : 200  
psc@ubuntu:~$
```

```
psc@ubuntu: ~  
psc@ubuntu:~$ ls  
2020203002 Desktop execve main.out mkfifo.out pipe.out server.out stack.cpp wait.c  
alarm.c Downloads execve.c main.sh Music Public sigaction string.cpp  
alarm.c Downloads execvp malloc.c myfifo queue.cpp sigaction.c string.out  
array1.cpp dup2.c execvp.c malloc.out MyStudent.cpp queue.out signal StudentList.dat  
array1.out dup2.out fork mecro1 MyStudent.hpp realloc1.c signal.c Templates  
array.cpp execl.c fork.c mecro1.c pause realloc1.out sigprocmask vector.cpp  
array.out execl.c list.cpp mecro2 pause.c realloc2.c sigprocmask.c vector.out  
client.c execv list.out mecro2.c Pictures realloc2.out sigset_t Videos  
client.out execv.c main.cpp mkfifo.c pipe.c server.c sigset_t.c wait  
psc@ubuntu:~$ cat pipe.c  
#include <stdio.h>  
#include <stdlib.h>  
#include <sys/wait.h>  
#include <time.h>  
#include <unistd.h>  
  
int main(int argc, char const *argv[]) {  
    int num = 0;  
  
    pid_t pid;  
    int pipefd[2] = {0, };  
  
    srand(time(NULL));  
  
    if(pipe(pipefd) == -1){  
        perror("pipe() error!");  
        return -1;  
    }  
  
    pid = fork();  
    if(pid == -1){  
        perror("fork() error!");  
    }else if(pid == 0){  
        close(pipefd[0]);  
        num = rand() % 10;  
        write(pipefd[1], (int*)&num, sizeof(int));  
  
        close(pipefd[1]);  
    }  
    else {  
        close(pipefd[1]);  
  
        while(read(pipefd[0], (int *)&num, sizeof(int)) < 0) {}  
        printf("Num: %d\n", num);  
  
        close(pipefd[0]);  
  
        wait(&pid);  
    }  
  
    return 0;  
}  
psc@ubuntu:~$ ./pipe.out  
Num: 0  
psc@ubuntu:~$
```

```
psc@ubuntu: ~  
psc@ubuntu:~$ ls  
2020203002 Desktop execve main.out mkfifo.out pipe.out server.out stack.cpp wait.c  
alarm Documents execve.c main.sh Music Public sigaction string.cpp  
alarm.c Downloads execvp malloc.c myfifo queue.cpp sigaction.c string.out  
array1.cpp dup2.c execvp.c malloc.out MyStudent.cpp queue.out signal StudentList.dat  
array1.out dup2.out fork mecro1 MyStudent.hpp realloc.c signal.c Templates  
array.cpp execl.c fork.c mecro1.c pause realloc1.out sigprocmask vector.cpp  
array.out execl.c list.cpp mecro2.c pause.c realloc2.c sigprocmask.c vector.out  
client.c execv list.out mecro2.c Pictures realloc2.out sigset_t Videos  
client.out execv.c main.cpp mkfifo.c pipe.c server.c sigset_t.c wait  
psc@ubuntu:~$ cat server.c  
#include <fcntl.h>  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
#include <unistd.h>  
  
#define PERMS 0666  
#define MAX_BUF_SIZE 32 + 1  
  
int main(int argc, char const *argv[]) {  
    int fd = 0;  
    char buf[MAX_BUF_SIZE] = { 0x00, };  
  
    if (argc != 2) {  
        printf("Usage: %s [pathname]\n", argv[0]);  
        return -1;  
    }  
  
    mkfifo(argv[1], PERMS);  
    fd = open(argv[1], O_RDONLY);  
  
    while (1) {  
        if (read(fd, (char *)buf, MAX_BUF_SIZE) > 0) {  
            puts(buf);  
            memset(buf, 0x00, MAX_BUF_SIZE);  
        }  
    }  
  
    return 0;  
}  
  
psc@ubuntu:~$ cat client.c  
#include <fcntl.h>  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
#include <time.h>  
#include <unistd.h>  
  
#define PERMS 0666  
  
int main(int argc, char const *argv[]) {  
    int num = 0;  
  
    int fd = 0;  
  
    if (argc != 2) {  
        printf("Usage: %s [pathname]\n", argv[0]);  
        return -1;  
    }  
  
    srand(time(NULL));  
  
    fd = open(argv[1], O_WRONLY);  
    dup2(fd, STDOUT_FILENO);  
  
    while (1) {  
        num = rand() % 10;  
        printf("[%d] %d", getpid(), num);  
        fflush(stdout);  
    }  
}
```

```
psc@ubuntu: ~  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
#include <unistd.h>  
  
#define PERMS 0666  
#define MAX_BUF_SIZE 32 + 1  
  
int main(int argc, char const *argv[]) {  
    int fd = 0;  
    char buf[MAX_BUF_SIZE] = { 0x00, };  
  
    if (argc != 2) {  
        printf("Usage: %s [pathname]\n", argv[0]);  
        return -1;  
    }  
  
    mkfifo(argv[1], PERMS);  
    fd = open(argv[1], O_RDONLY);  
  
    while (1) {  
        if (read(fd, (char *)buf, MAX_BUF_SIZE) > 0) {  
            puts(buf);  
            memset(buf, 0x00, MAX_BUF_SIZE);  
        }  
    }  
  
    return 0;  
}  
  
psc@ubuntu:~$ cat client.c  
#include <fcntl.h>  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
#include <time.h>  
#include <unistd.h>  
  
#define PERMS 0666  
  
int main(int argc, char const *argv[]) {  
    int num = 0;  
  
    int fd = 0;  
  
    if (argc != 2) {  
        printf("Usage: %s [pathname]\n", argv[0]);  
        return -1;  
    }  
  
    srand(time(NULL));  
  
    fd = open(argv[1], O_WRONLY);  
    dup2(fd, STDOUT_FILENO);  
  
    while (1) {  
        num = rand() % 10;  
        printf("[%d] %d", getpid(), num);  
        fflush(stdout);  
  
        if (num == 5) {  
            break;  
        } else {  
            sleep(1);  
        }  
    }  
  
    close(fd);  
  
    return 0;  
}  
psc@ubuntu:~$
```



```
psc@ubuntu: ~  
psc@ubuntu:~$ ls  
2020203002 Desktop execve main.out mkfifo.out pipe.out server.out stack.cpp wait.c  
alarm.c Documents execve.c main.sh Music Public sigaction sigaction.c string.cpp  
array1.out Downloads execvp malloc.c myfifo queue.cpp sigaction.c string.out  
array1.out dup2.out fork macro1 MyStudent.cpp queue.out signal.c StudentList.dat  
array.cpp execl fork.c macro1.c pause realloc1.c signal.c Templates  
array.out execl.c list.cpp macro2 pause.c realloc1.out sigprocmask vector.cpp  
client.c execv list.out macro2.c pause.c realloc2.c sigprocmask.c vector.out  
client.out execv.c main.cpp mkfifo.c pipe.c server.c sigset_t Videos  
psc@ubuntu:~$ cat dup2.c  
#include <stdio.h>  
#include <stdlib.h>  
#include <sys/wait.h>  
#include <unistd.h>  
  
int main(int argc, char const *argv[]) {  
    pid_t pid;  
    int pipefd[2] = {0,};  
  
    if(pipe(pipefd) == -1){  
        perror("pipe() error!");  
        return -1;  
    }  
  
    pid = fork();  
    if(pid == -1) {  
        perror("fork() error!");  
    }  
    else if (pid == 0){  
        close(pipefd[0]);  
  
        dup2(pipefd[1], STDOUT_FILENO);  
        execl("/bin/ls", "ls", "-a", "-l", NULL);  
    }  
    else{  
        close(pipefd[1]);  
  
        wait(&pid);  
  
        dup2(pipefd[0], STDIN_FILENO);  
        execl("/usr/bin/sort", "sort", "-r", NULL);  
    }  
  
    return 0;  
}  
psc@ubuntu:~$ ./dup2.out  
total 920  
-rwxrwxr-x 1 psc psc 53144 Nov 17 16:50 queue.out  
-rwxrwxr-x 1 psc psc 39320 Nov 11 07:19 list.out  
-rwxrwxr-x 1 psc psc 38152 Nov 11 07:51 main.out  
-rwxrwxr-x 1 psc psc 32952 Nov 3 18:40 vector.out  
-rwxrwxr-x 1 psc psc 20216 Nov 3 18:55 string.out  
-rwxrwxr-x 1 psc psc 18776 Nov 3 18:03 array.out  
-rwxrwxr-x 1 psc psc 18560 Nov 3 18:13 array1.out  
-rwxrwxr-x 1 psc psc 17200 Sep 29 18:49 sigprocmask  
-rwxrwxr-x 1 psc psc 17160 Nov 17 17:11 pipe.out  
-rwxrwxr-x 1 psc psc 17112 Sep 29 17:45 signal  
-rwxrwxr-x 1 psc psc 17112 Nov 17 17:39 client.out  
-rwxrwxr-x 1 psc psc 17048 Sep 24 16:57 wait  
-rwxrwxr-x 1 psc psc 17000 Nov 17 17:22 dup2.out  
-rwxrwxr-x 1 psc psc 16968 Sep 24 17:27 macro2  
-rwxrwxr-x 1 psc psc 16968 Sep 24 17:26 macro1  
-rwxrwxr-x 1 psc psc 16968 Nov 17 17:37 server.out  
-rwxrwxr-x 1 psc psc 16952 Sep 29 18:05 pause  
-rwxrwxr-x 1 psc psc 16920 Sep 26 22:13 execve  
-rwxrwxr-x 1 psc psc 16920 Sep 25 04:11 execvp  
-rwxrwxr-x 1 psc psc 16920 Sep 24 17:41 execv  
-rwxrwxr-x 1 psc psc 16920 Sep 24 17:34 execl  
-rwxrwxr-x 1 psc psc 16912 Sep 29 18:12 alarm  
-rwxrwxr-x 1 psc psc 16912 Sep 24 12:25 fork  
-rwxrwxr-x 1 psc psc 16896 Sep 29 18:28 sigset_t  
-rwxrwxr-x 1 psc psc 16880 Sep 29 19:04 sigaction  
-rwxrwxr-x 1 psc psc 16832 Nov 11 06:17 realloc2.out  
-rwxrwxr-x 1 psc psc 16832 Nov 11 06:12 realloc1.out  
-rwxrwxr-x 1 psc psc 16832 Nov 11 05:52 malloc.out
```

```
psc@ubuntu: ~  
-rwxrwxr-x 1 psc psc 16968 Sep 24 17:26 mecro1  
-rwxrwxr-x 1 psc psc 16968 Nov 17 17:37 server.out  
-rwxrwxr-x 1 psc psc 16952 Sep 29 18:05 pause  
-rwxrwxr-x 1 psc psc 16920 Sep 26 22:13 execve  
-rwxrwxr-x 1 psc psc 16920 Sep 25 04:11 execvp  
-rwxrwxr-x 1 psc psc 16920 Sep 24 17:41 execv  
-rwxrwxr-x 1 psc psc 16920 Sep 24 17:34 execl  
-rwxrwxr-x 1 psc psc 16912 Sep 29 18:12 alarm  
-rwxrwxr-x 1 psc psc 16912 Sep 24 12:25 fork  
-rwxrwxr-x 1 psc psc 16896 Sep 29 18:28 sigset_t  
-rwxrwxr-x 1 psc psc 16880 Sep 29 19:04 sigaction  
-rwxrwxr-x 1 psc psc 16832 Nov 11 06:17 realloc2.out  
-rwxrwxr-x 1 psc psc 16832 Nov 11 06:12 realloc1.out  
-rwxrwxr-x 1 psc psc 16832 Nov 11 05:52 malloc.out  
-rwxrwxr-x 1 psc psc 16744 Nov 17 17:29 mkfifo.out  
-rw-rw-r-- 1 psc psc 903 Nov 3 18:14 array1.cpp  
-rw-rw-r-- 1 psc psc 894 Nov 17 16:59 stack.cpp  
-rw-rw-r-- 1 psc psc 806 Nov 3 17:47 array.cpp  
-rw-rw-r-- 1 psc psc 796 Sep 24 16:57 wait.c  
-rw-rw-r-- 1 psc psc 752 Nov 17 17:10 pipe.c  
-rw-rw-r-- 1 psc psc 716 Nov 11 07:51 MyStudent.cpp  
-rw-rw-r-- 1 psc psc 685 Nov 17 17:39 client.c  
-rw-rw-r-- 1 psc psc 658 Sep 29 17:44 signal.c  
-rw-rw-r-- 1 psc psc 649 Nov 17 17:22 dup2.c  
-rw-rw-r-- 1 psc psc 624 Nov 17 17:37 server.c  
-rw-rw-r-- 1 psc psc 615 Sep 29 18:04 pause.c  
-rw-rw-r-- 1 psc psc 597 Nov 3 18:55 string.cpp  
-rw-rw-r-- 1 psc psc 593 Sep 24 17:16 mecro1.c  
-rw-rw-r-- 1 psc psc 591 Sep 24 17:27 mecro2.c  
-rw-rw-r-- 1 psc psc 564 Sep 29 18:49 sigprocmask.c  
-rw-rw-r-- 1 psc psc 559 Sep 24 12:25 fork.c  
-rw-rw-r-- 1 psc psc 557 Nov 11 05:52 malloc.c  
-rw-rw-r-- 1 psc psc 513 Sep 26 22:13 execve.c  
-rw-rw-r-- 1 psc psc 495 Nov 11 07:34 MyStudent.hpp  
-rw-rw-r-- 1 psc psc 475 Nov 11 06:17 realloc2.c  
-rw-rw-r-- 1 psc psc 448 Sep 24 17:41 execv.c  
-rw-rw-r-- 1 psc psc 445 Sep 25 04:11 execvp.c  
-rw-rw-r-- 1 psc psc 445 Sep 24 17:34 execl.c  
-rw-rw-r-- 1 psc psc 433 Sep 29 19:04 sigaction.c  
-rw-rw-r-- 1 psc psc 382 Nov 11 06:12 realloc1.c  
-rw-rw-r-- 1 psc psc 356 Sep 29 18:12 alarm.c  
-rw-rw-r-- 1 psc psc 286 Sep 29 18:28 sigset_t.c  
-rw-rw-r-- 1 psc psc 265 Nov 17 17:28 mkfifo.c  
-rw-rw-r-- 1 psc psc 1775 May 24 10:23 2020203002  
-rw-rw-r-- 1 psc psc 1606 Nov 17 16:50 queue.cpp  
-rw-rw-r-- 1 psc psc 1469 Nov 11 07:47 main.cpp  
-rw-rw-r-- 1 psc psc 1090 Nov 11 07:19 list.cpp  
-rw-rw-r-- 1 psc psc 1071 Nov 3 18:40 vector.cpp  
-rw-rw-r-- 1 psc psc 1007 May 24 09:44 .vimrc  
-rw-r--r-- 1 root root 5278 May 24 18:53 main.sh  
-rw-r--r-- 1 psc psc 96 Nov 11 07:56 StudentList.dat  
-rw-r--r-- 1 psc psc 807 Apr 20 2021 .profile  
-rw-r--r-- 1 psc psc 3771 Apr 20 2021 .bashrc  
-rw-r--r-- 1 psc psc 220 Apr 20 2021 .bash_logout  
-rw-r--r-- 1 psc psc 0 Apr 20 2021 .sudo_as_admin_successful  
-rw----- 1 psc psc 5830 Nov 17 17:40 .bash_history  
-rw----- 1 psc psc 13898 Nov 17 17:39 .viminfo  
prw-rw-r-- 1 psc psc 0 Nov 17 17:29 myfifo  
drwxr-xr-x 3 root root 4096 Apr 20 2021 ..  
drwxr-xr-x 3 psc psc 4096 Apr 20 2021 .local  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Videos  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Templates  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Public  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Pictures  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Music  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Downloads  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Documents  
drwxr-xr-x 2 psc psc 4096 Apr 20 2021 Desktop  
drwxr-xr-x 17 psc psc 4096 Nov 17 17:39 .  
drwx----- 6 psc psc 4096 May 24 10:25 .thunderbird  
drwx----- 4 psc psc 4096 May 24 10:25 .mozilla  
drwx----- 3 psc psc 4096 Apr 20 2021 .gnupg  
drwx----- 2 psc psc 4096 Apr 20 2021 .ssh  
drwx----- 13 psc psc 4096 Nov 3 17:29 .config  
drwx----- 13 psc psc 4096 May 24 10:31 .cache  
psc@ubuntu:~$
```

```
psc@ubuntu: ~  
psc@ubuntu:~$ ls  
2020203002 Desktop execve main.out mkfifo.out pipe.out server.out stack.cpp wait.c  
alarm Documents execve.c main.sh Music Public sigaction string.cpp  
alarm.c Downloads execvp malloc.c myfifo queue.cpp sigaction.c string.out  
array1.cpp dup2.c execvp.c malloc.out MyStudent.cpp queue.out signal StudentList.dat  
array1.out dup2.out fork mecro1 MyStudent.hpp realloc1.c signal.c Templates  
array.cpp execl fork.c mecro1.c pause realloc1.out sigprocmask vector.cpp  
array.out execl.c list.cpp mecro2 pause.c realloc2.c sigprocmask.c vector.out  
client.c execv list.out mecro2.c Pictures realloc2.out sigset_t Videos  
client.out execv.c main.cpp mkfifo.c pipe.c server.c sigset_t.c wait  
psc@ubuntu:~$ cat mkfifo.c  
#include <stdio.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
  
#define PERMS 0666  
  
int main(int argc, char const *argv[]){  
    if(argc != 2){  
        printf("Usage: %s [pathname]\n",argv[0]);  
        return -1;  
    }  
  
    mkfifo(argv[1], PERMS);  
  
    return 0;  
}  
psc@ubuntu:~$
```



```

pssc@ubuntu:~$ ls
2020203002  Desktop  execve  main.out  Music  Public  sigaction  stack.out  wait.c
alarm       Documents  execve.c  main.sh  myfifo  queue.cpp  sigaction.c  string.cpp
alarm.c     Downloads  execvp    malloc.c  MyStudent.cpp  queue.out  signal      string.out
array1.cpp  dup2.c    execvp.c  malloc.out  MyStudent.hpp  realloc1.c  signal.c    StudentList.dat
array1.out  dup2.out  fork      mecro1.c  pause      realloc1.out  sigprocmask  Templates
array.cpp   execl     fork.c    mecro1.c  pause.c    realloc2.c  sigprocmask.c  vector.cpp
array.out   execl.c   list.cpp  mecro2.c  Pictures   realloc2.out  sigset_t      vector.out
client.c    execv     list.out  mecro2.c  pipe.c     server.c    sigset_t.c    Videos
client.out  execv.c   main.cpp  mkfifo.out  pipe.out   server.out  stack.cpp     wait

pssc@ubuntu:~$ cat client.c
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <time.h>
#include <unistd.h>

#define PERMS 0666

int main(int argc, char const *argv[]) {
    int num = 0;

    int fd = 0;

    if (argc != 2) {
        printf("Usage: %s [pathname]\n", argv[0]);
        return -1;
    }

    srand(time(NULL));

    fd = open(argv[1], O_WRONLY);
    dup2(fd, STDOUT_FILENO);

    while (1) {
        num = rand() % 10;
        printf("[%d] %d", getpid(), num);
        fflush(stdout);

        if (num == 5) {
            break;
        } else {
            sleep(1);
        }
    }

    close(fd);

    return 0;
}

pssc@ubuntu:~$ cat server.c
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

#define PERMS 0666
#define MAX_BUF_SIZE 32 + 1

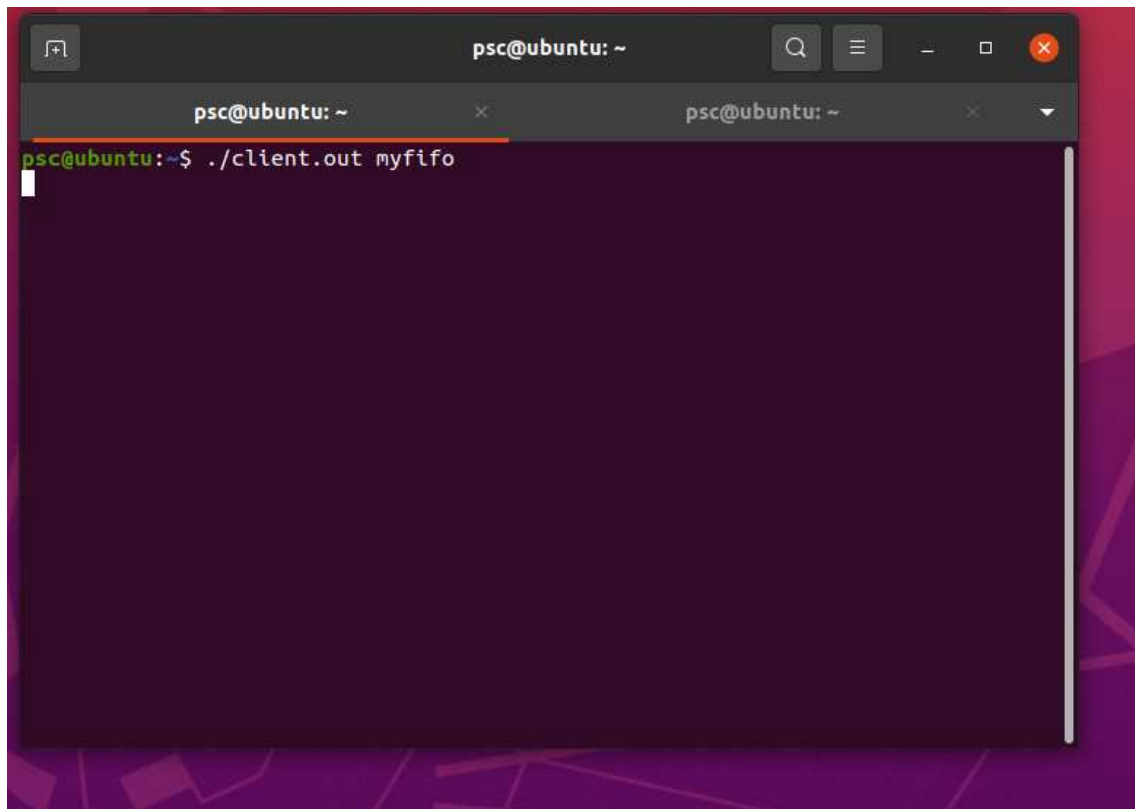
int main(int argc, char const *argv[]) {
    int fd = 0;
    char buf[MAX_BUF_SIZE] = { 0x00, };

    if (argc != 2) {
        printf("Usage: %s [pathname]\n", argv[0]);
        return -1;
    }

    mkfifo(argv[1], PERMS);
    fd = open(argv[1], O_RDONLY);

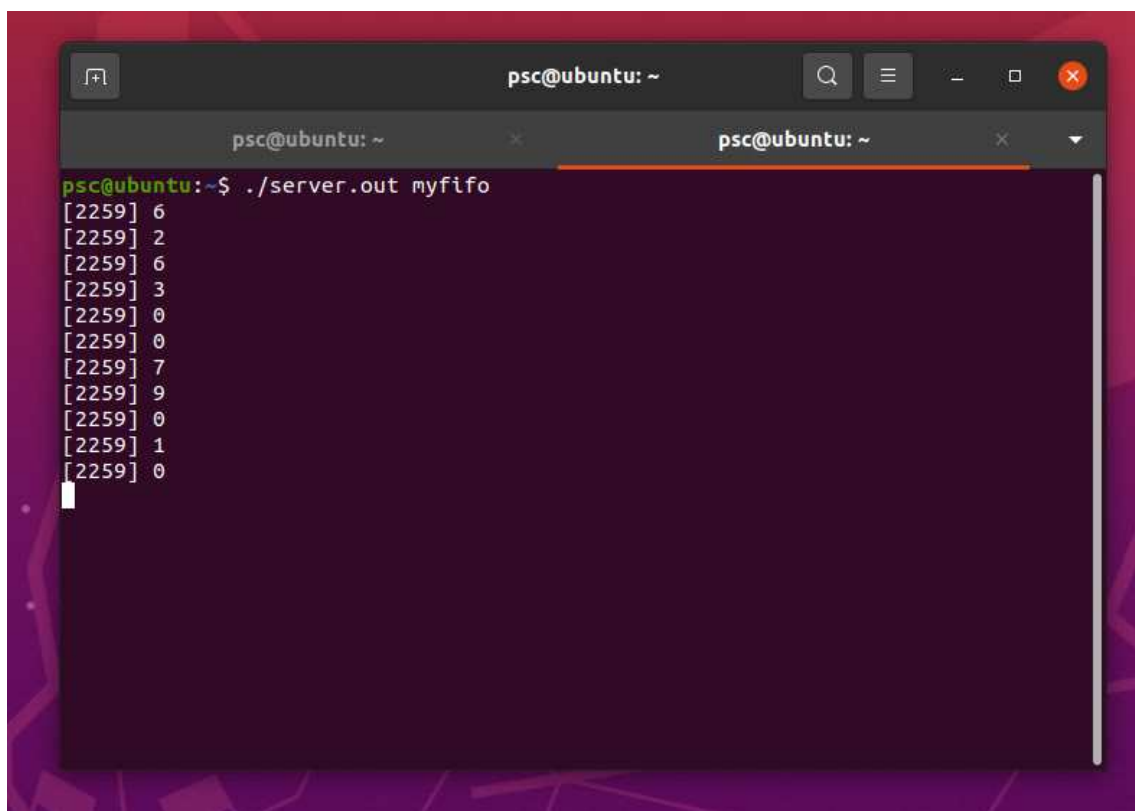
```

```
pasc@ubuntu: ~  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
#include <time.h>  
#include <unistd.h>  
  
#define PERMS 0666  
  
int main(int argc, char const *argv[]) {  
    int num = 0;  
  
    int fd = 0;  
  
    if (argc != 2) {  
        printf("Usage: %s [pathname]\n", argv[0]);  
        return -1;  
    }  
  
    srand(time(NULL));  
  
    fd = open(argv[1], O_WRONLY);  
    dup2(fd, STDOUT_FILENO);  
  
    while (1) {  
        num = rand() % 10;  
        printf("[%d] %d", getpid(), num);  
        fflush(stdout);  
  
        if (num == 5) {  
            break;  
        } else {  
            sleep(1);  
        }  
    }  
  
    close(fd);  
  
    return 0;  
}  
pasc@ubuntu:~$ cat server.c  
#include <fcntl.h>  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
#include <unistd.h>  
  
#define PERMS 0666  
#define MAX_BUF_SIZE 32 + 1  
  
int main(int argc, char const *argv[]) {  
    int fd = 0;  
    char buf[MAX_BUF_SIZE] = { 0x00, };  
  
    if (argc != 2) {  
        printf("Usage: %s [pathname]\n", argv[0]);  
        return -1;  
    }  
  
    mkfifo(argv[1], PERMS);  
    fd = open(argv[1], O_RDONLY);  
  
    while (1) {  
        if (read(fd, (char *)buf, MAX_BUF_SIZE) > 0) {  
            puts(buf);  
            memset(buf, 0x00, MAX_BUF_SIZE);  
        }  
    }  
  
    return 0;  
}  
pasc@ubuntu:~$
```



A terminal window titled "psc@ubuntu: ~" with a search icon, a menu icon, and window control buttons. The terminal shows the command `psc@ubuntu:~$./client.out myfifo` being entered. The cursor is at the end of the line.

```
psc@ubuntu:~$ ./client.out myfifo
```



A terminal window titled "psc@ubuntu: ~" with a search icon, a menu icon, and window control buttons. The terminal shows the command `psc@ubuntu:~$./server.out myfifo` being entered. The output consists of 12 lines, each starting with "[2259]".

```
psc@ubuntu:~$ ./server.out myfifo
[2259] 6
[2259] 2
[2259] 6
[2259] 3
[2259] 0
[2259] 0
[2259] 7
[2259] 9
[2259] 0
[2259] 1
[2259] 0
```