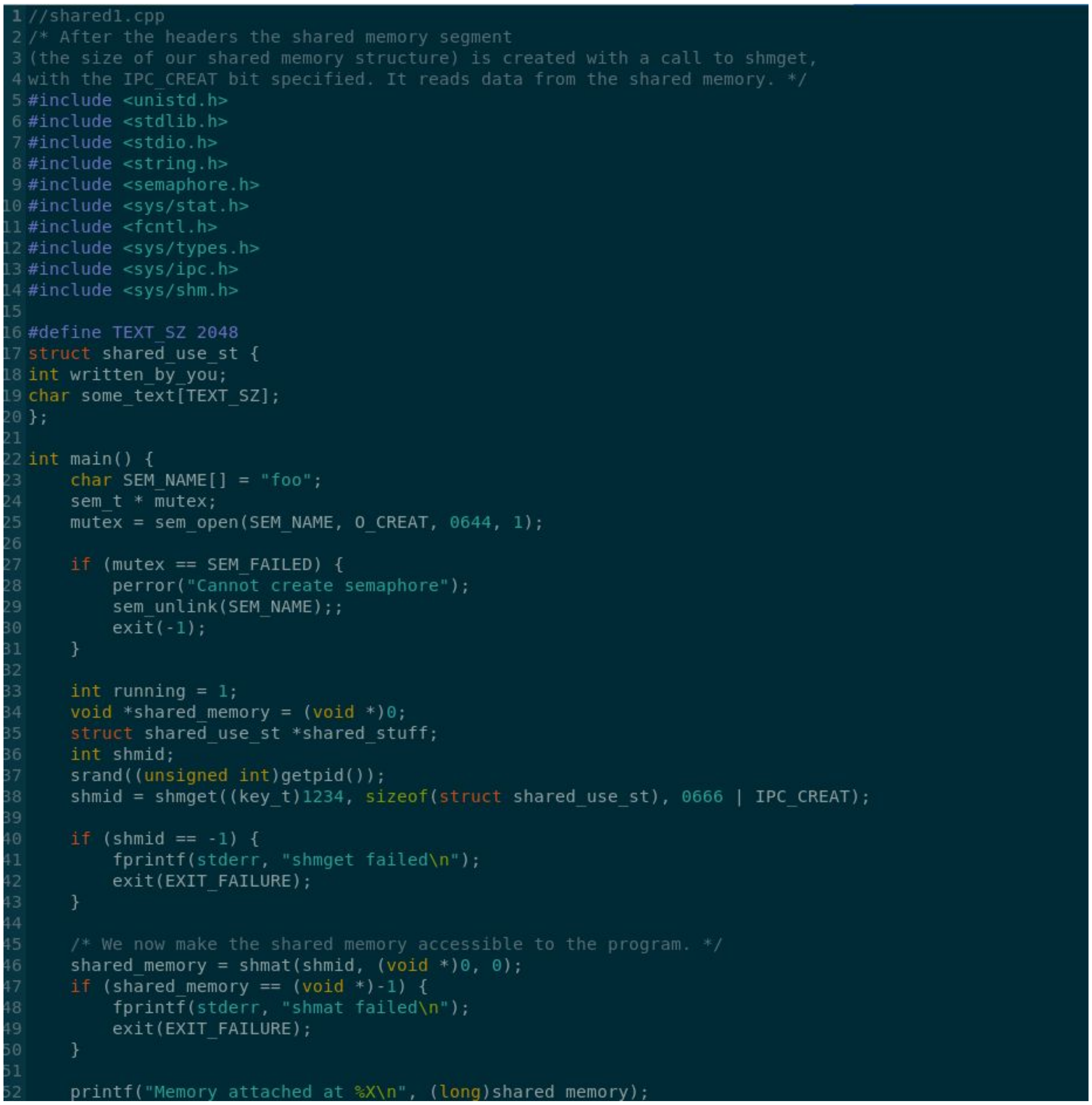
Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20

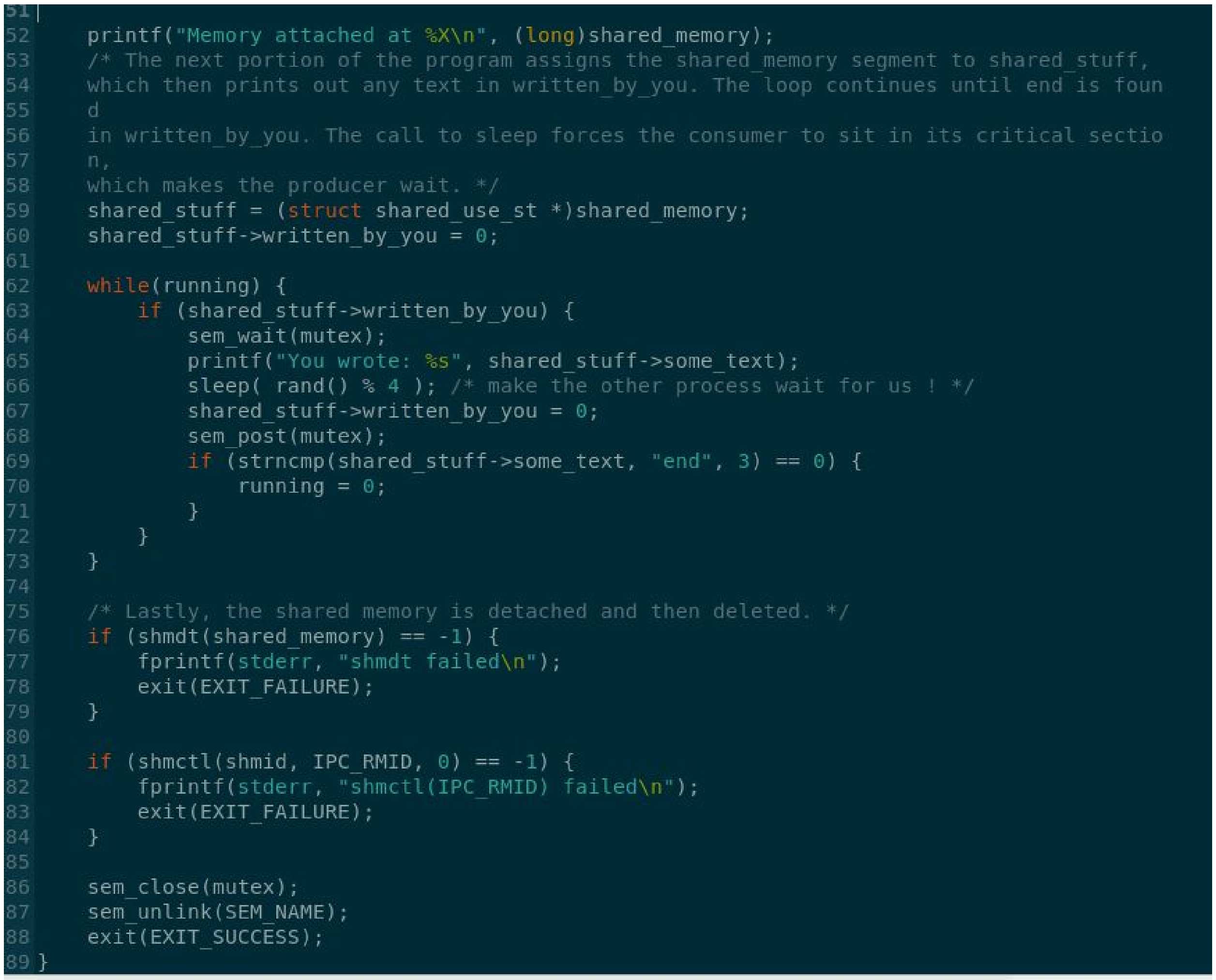
1. Shared Memory

Code:
Shared1



```
//shared1.cpp
/* After the headers the shared memory segment
(the size of our shared memory structure) is created with a call to shmget,
with the IPC_CREAT bit specified. It reads data from the shared memory. */
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <semaphore.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>

#define TEXT_SZ 2048
struct shared_use_st {
int written_by_you;
char some_text[TEXT_SZ];
};

int main() {
    char SEM_NAME[] = "foo";
    sem_t * mutex;
    mutex = sem_open(SEM_NAME, O_CREAT, 0644, 1);

    if (mutex == SEM_FAILED) {
        perror("Cannot create semaphore");
        sem_unlink(SEM_NAME);;
        exit(-1);
    }

    int running = 1;
    void *shared_memory = (void *)0;
    struct shared_use_st *shared_stuff;
    int shmid;
    srand((unsigned int)getpid());
    shmid = shmget((key_t)1234, sizeof(struct shared_use_st), 0666 | IPC_CREAT);

    if (shmid == -1) {
        fprintf(stderr, "shmget failed\n");
        exit(EXIT_FAILURE);
    }

    /* We now make the shared memory accessible to the program. */
    shared_memory = shmat(shmid, (void *)0, 0);
    if (shared_memory == (void *)-1) {
        fprintf(stderr, "shmat failed\n");
        exit(EXIT_FAILURE);
    }

    printf("Memory attached at %X\n", (long)shared_memory);
```

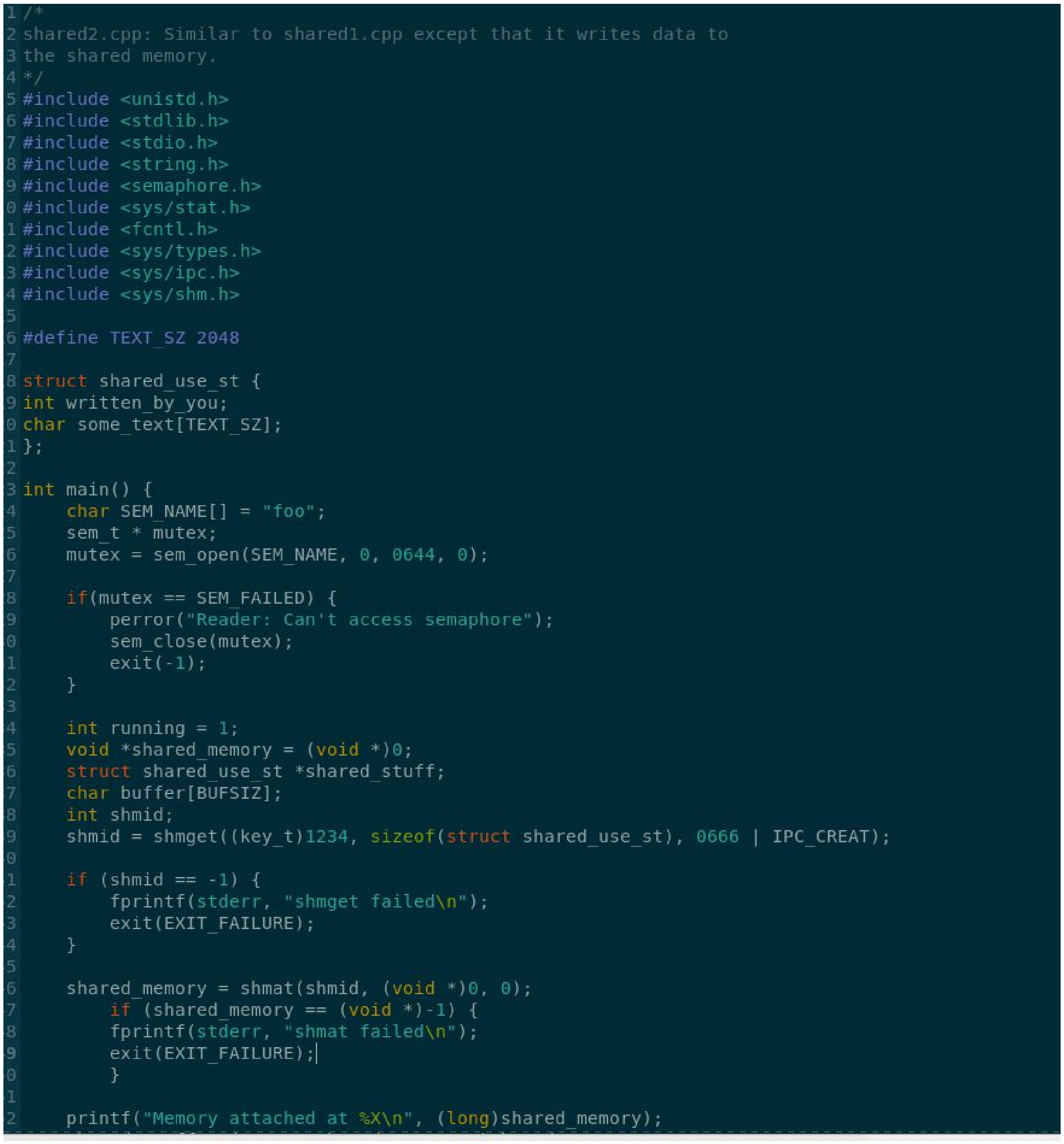
Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20



```

    printf("Memory attached at %X\n", (long)shared_memory);
    /* The next portion of the program assigns the shared_memory segment to shared_stuff,
    which then prints out any text in written_by_you. The loop continues until end is foun
    d
    in written_by_you. The call to sleep forces the consumer to sit in its critical sectio
    n,
    which makes the producer wait. */
    shared_stuff = (struct shared_use_st *)shared_memory;
    shared_stuff->written_by_you = 0;

    while(running) {
        if (shared_stuff->written_by_you) {
            sem_wait(mutex);
            printf("You wrote: %s", shared_stuff->some_text);
            sleep( rand() % 4 ); /* make the other process wait for us ! */
            shared_stuff->written_by_you = 0;
            sem_post(mutex);
            if (strncmp(shared_stuff->some_text, "end", 3) == 0) {
                running = 0;
            }
        }
    }

    /* Lastly, the shared memory is detached and then deleted. */
    if (shmdt(shared_memory) == -1) {
        fprintf(stderr, "shmdt failed\n");
        exit(EXIT_FAILURE);
    }

    if (shmctl(shmid, IPC_RMID, 0) == -1) {
        fprintf(stderr, "shmctl(IPC_RMID) failed\n");
        exit(EXIT_FAILURE);
    }

    sem_close(mutex);
    sem_unlink(SEM_NAME);
    exit(EXIT_SUCCESS);
}
```

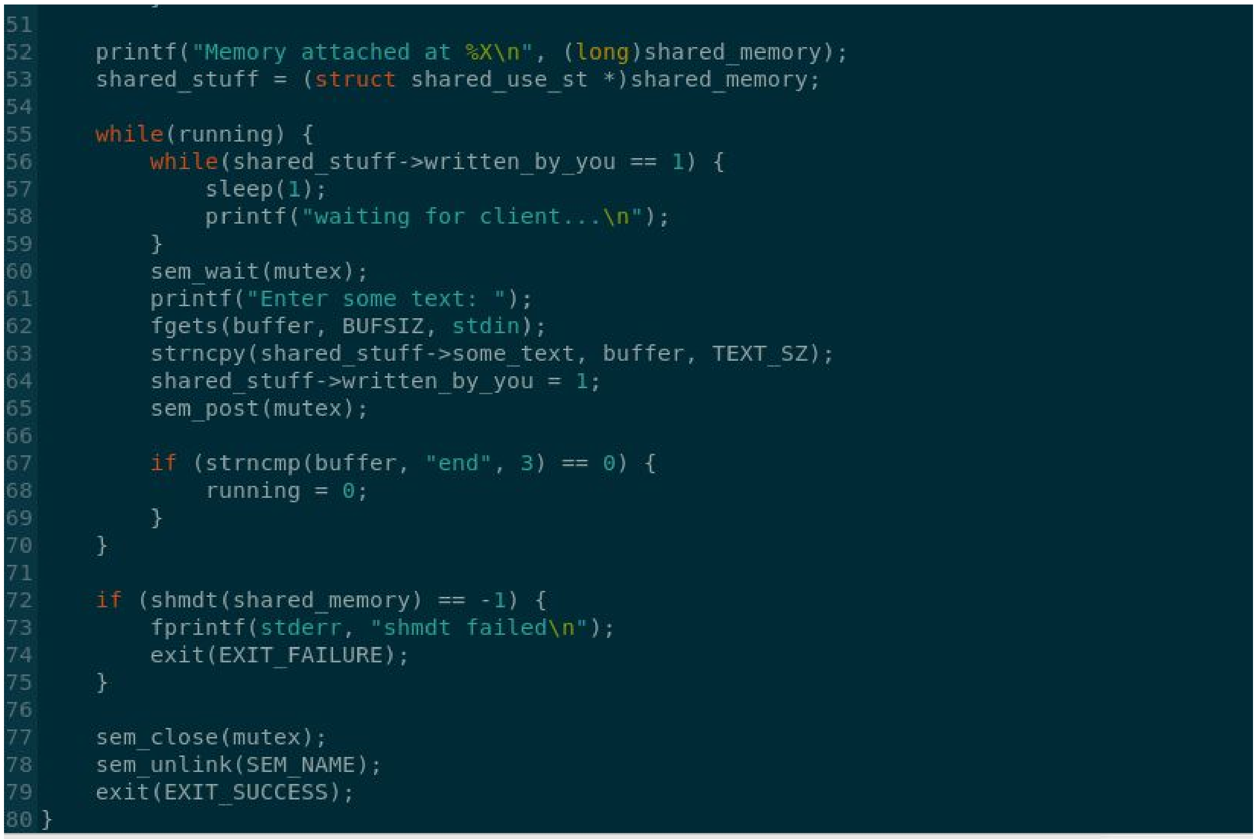
Shared2:

Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20



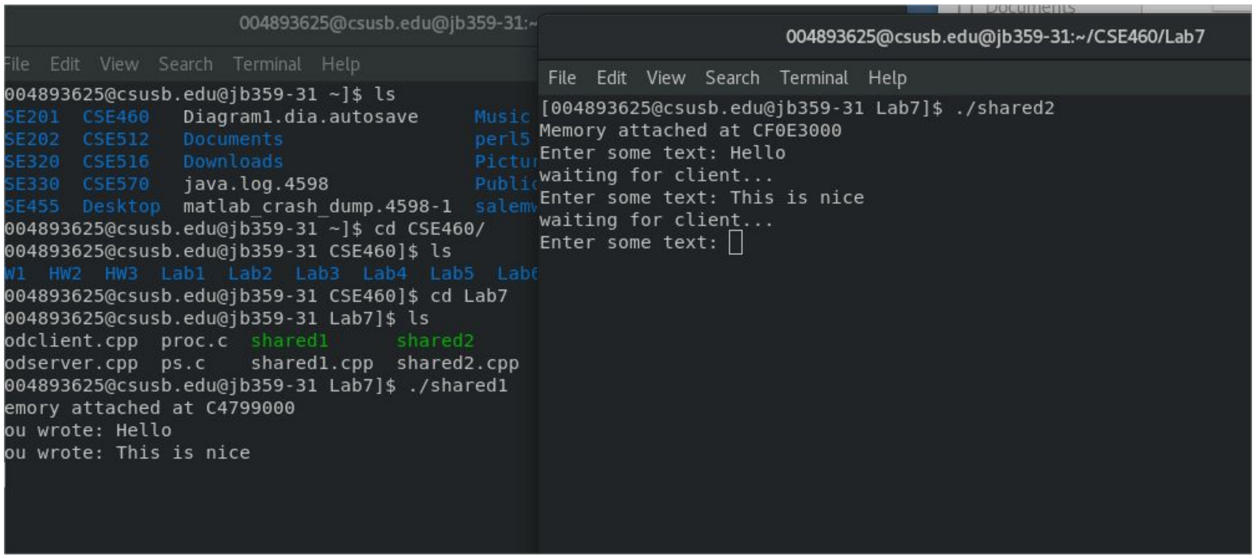
```
/*
shared2.cpp: Similar to shared1.cpp except that it writes data to
the shared memory.
*/
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <semaphore.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>

#define TEXT_SZ 2048

struct shared_use_st {
int written_by_you;
char some_text[TEXT_SZ];
};

int main() {
    char SEM_NAME[] = "foo";
    sem_t * mutex;
    mutex = sem_open(SEM_NAME, 0, 0644, 0);

    if(mutex == SEM_FAILED) {
        perror("Reader: Can't access semaphore");
        sem_close(mutex);
        exit(-1);
    }

    int running = 1;
    void *shared_memory = (void *)0;
    struct shared_use_st *shared_stuff;
    char buffer[BUFSIZ];
    int shmid;
    shmid = shmget((key_t)1234, sizeof(struct shared_use_st), 0666 | IPC_CREAT);

    if (shmid == -1) {
        fprintf(stderr, "shmget failed\n");
        exit(EXIT_FAILURE);
    }

    shared_memory = shmat(shmid, (void *)0, 0);
        if (shared_memory == (void *)-1) {
        fprintf(stderr, "shmat failed\n");
        exit(EXIT_FAILURE);
        }

    printf("Memory attached at %X\n", (long)shared_memory);
```

Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20



```
    printf("Memory attached at %X\n", (long)shared_memory);
    shared_stuff = (struct shared_use_st *)shared_memory;

    while(running) {
        while(shared_stuff->written_by_you == 1) {
            sleep(1);
            printf("waiting for client...\n");
        }
        sem_wait(mutex);
        printf("Enter some text: ");
        fgets(buffer, BUFSIZ, stdin);
        strncpy(shared_stuff->some_text, buffer, TEXT_SZ);
        shared_stuff->written_by_you = 1;
        sem_post(mutex);

        if (strncmp(buffer, "end", 3) == 0) {
            running = 0;
        }
    }

    if (shmdt(shared_memory) == -1) {
        fprintf(stderr, "shmdt failed\n");
        exit(EXIT_FAILURE);
    }

    sem_close(mutex);
    sem_unlink(SEM_NAME);
    exit(EXIT_SUCCESS);
}
```

Output:



```
004893625@csusb.edu@jb359-31 ~]$ ls
SE201  CSE460   Diagram1.dia.autosave     Music
SE202  CSE512   Documents                 perl5
SE320  CSE516   Downloads                 Pictu
SE330  CSE570   java.log.4598             Publi
SE455  Desktop  matlab_crash_dump.4598-1  salem
004893625@csusb.edu@jb359-31 ~]$ cd CSE460/
004893625@csusb.edu@jb359-31 CSE460]$ ls
W1  HW2  HW3  Lab1  Lab2  Lab3  Lab4  Lab5  Lab6
004893625@csusb.edu@jb359-31 CSE460]$ cd Lab7
004893625@csusb.edu@jb359-31 Lab7]$ ls
odclient.cpp  proc.c  shared1      shared2
odserver.cpp  ps.c    shared1.cpp  shared2.cpp
004893625@csusb.edu@jb359-31 Lab7]$ ./shared1
emory attached at C4799000
ou wrote: Hello
ou wrote: This is nice
```

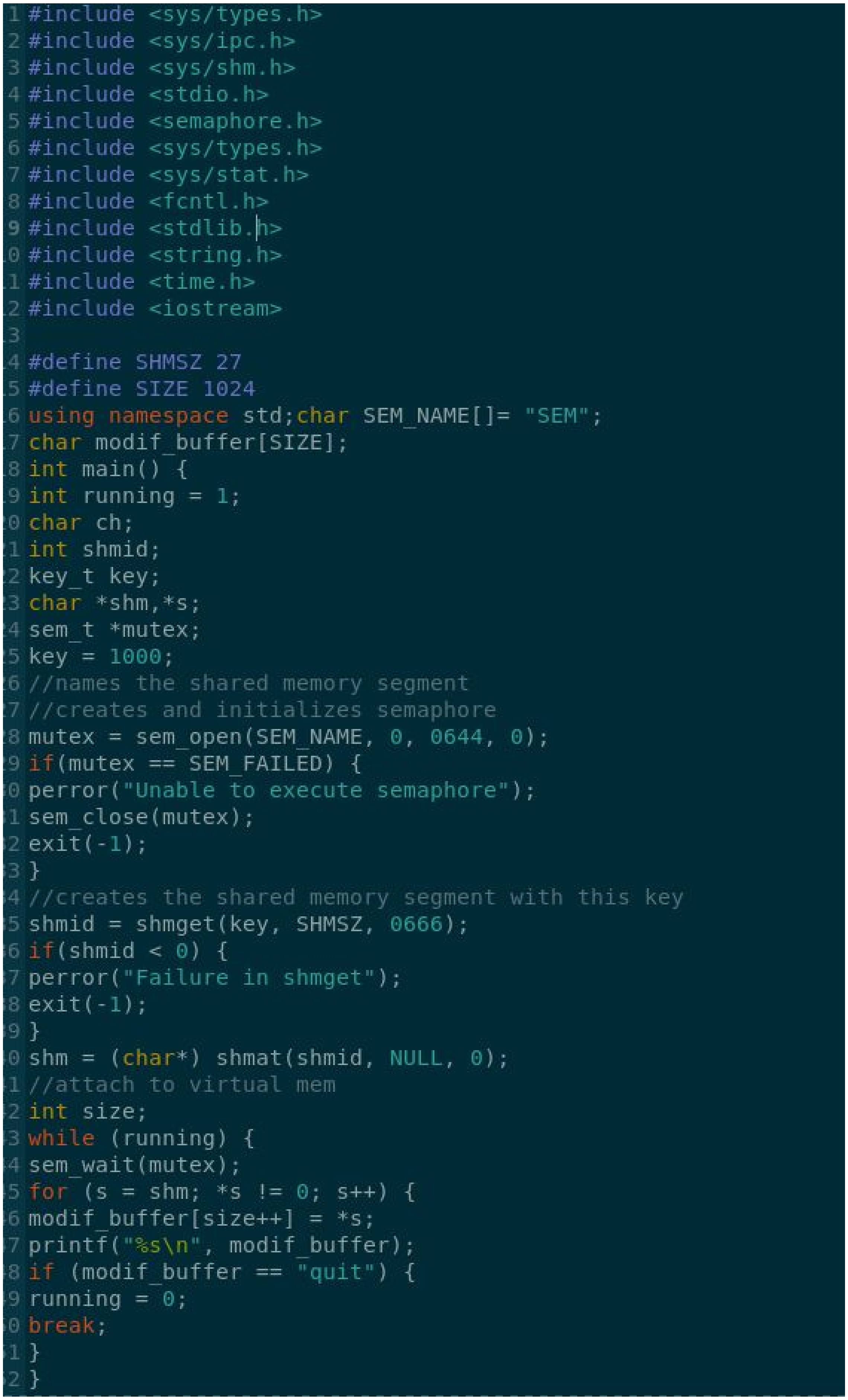
```
[004893625@csusb.edu@jb359-31 Lab7]$ ./shared2
Memory attached at CF0E3000
Enter some text: Hello
waiting for client...
Enter some text: This is nice
waiting for client...
Enter some text:
```

2. POSIX Semaphores

Code:

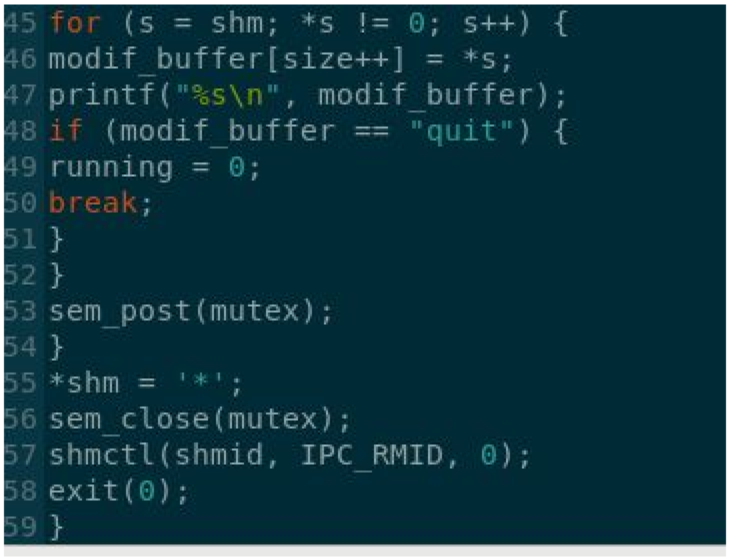
Modserver

Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20



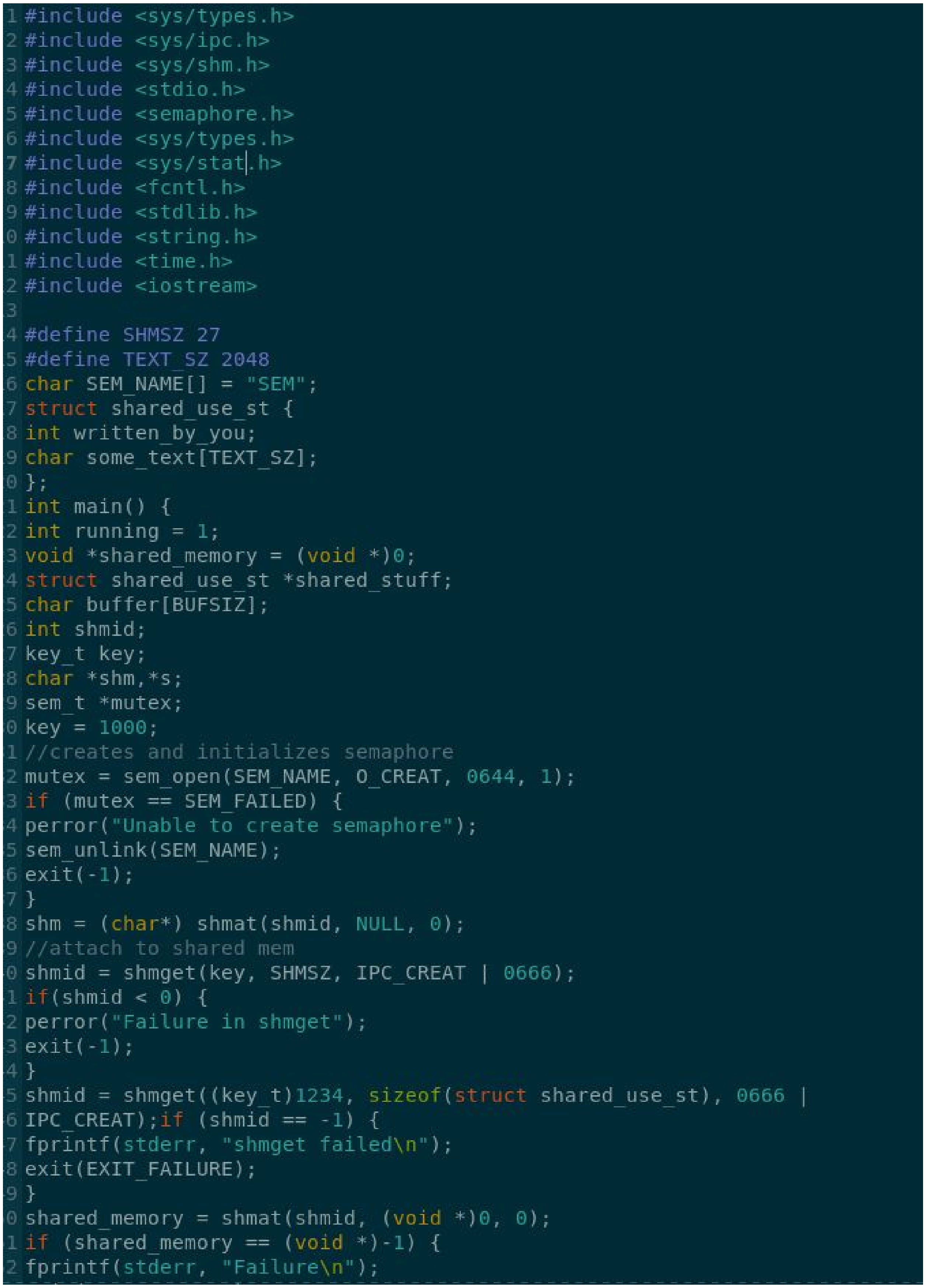
```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
#include <semaphore.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <iostream>

#define SHMSZ 27
#define SIZE 1024
using namespace std;char SEM_NAME[]= "SEM";
char modif_buffer[SIZE];
int main() {
int running = 1;
char ch;
int shmid;
key_t key;
char *shm,*s;
sem_t *mutex;
key = 1000;
//names the shared memory segment
//creates and initializes semaphore
mutex = sem_open(SEM_NAME, 0, 0644, 0);
if(mutex == SEM_FAILED) {
perror("Unable to execute semaphore");
sem_close(mutex);
exit(-1);
}
//creates the shared memory segment with this key
shmid = shmget(key, SHMSZ, 0666);
if(shmid < 0) {
perror("Failure in shmget");
exit(-1);
}
shm = (char*) shmat(shmid, NULL, 0);
//attach to virtual mem
int size;
while (running) {
sem_wait(mutex);
for (s = shm; *s != 0; s++) {
modif_buffer[size++] = *s;
printf("%s\n", modif_buffer);
if (modif_buffer == "quit") {
running = 0;
break;
}
}
```

Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20



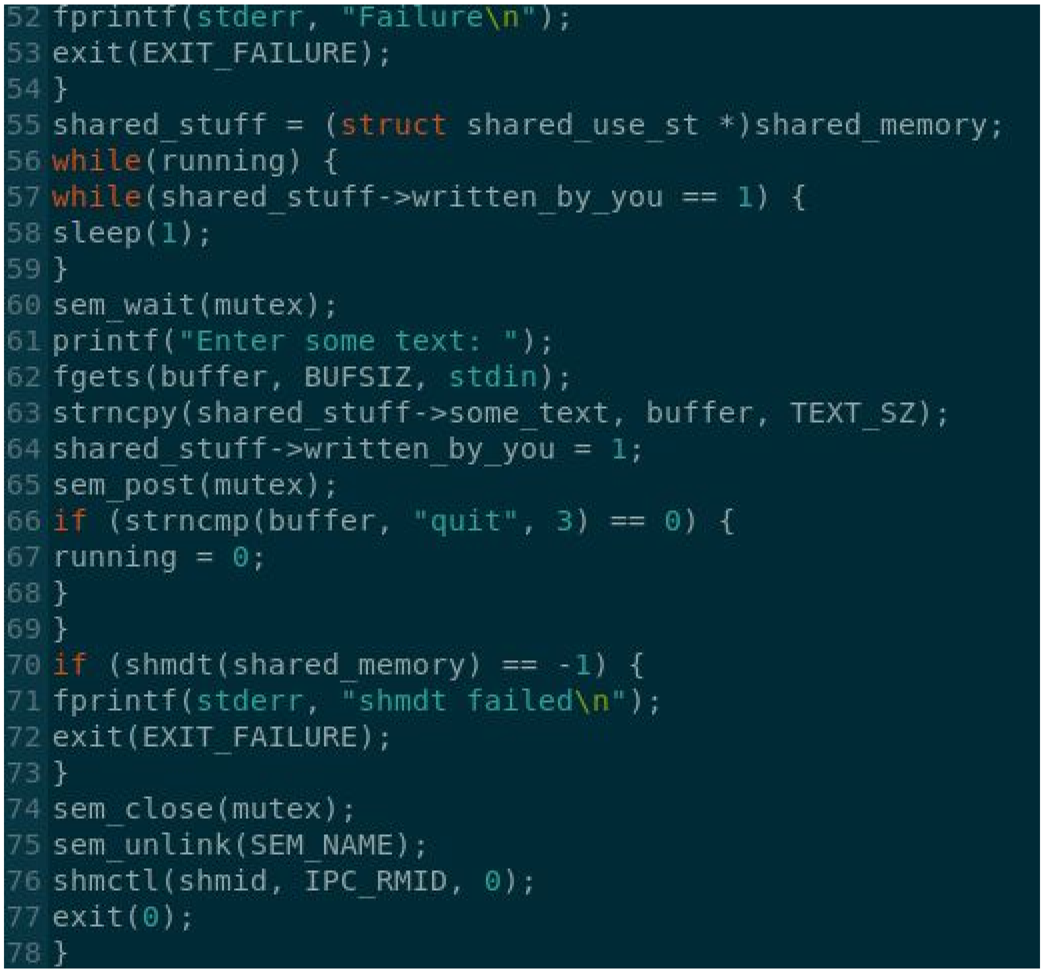
```
for (s = shm; *s != 0; s++) {
modif_buffer[size++] = *s;
printf("%s\n", modif_buffer);
if (modif_buffer == "quit") {
running = 0;
break;
}
}
sem_post(mutex);
}
*shm = '*';
sem_close(mutex);
shmctl(shmid, IPC_RMID, 0);
exit(0);
}
```

Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20



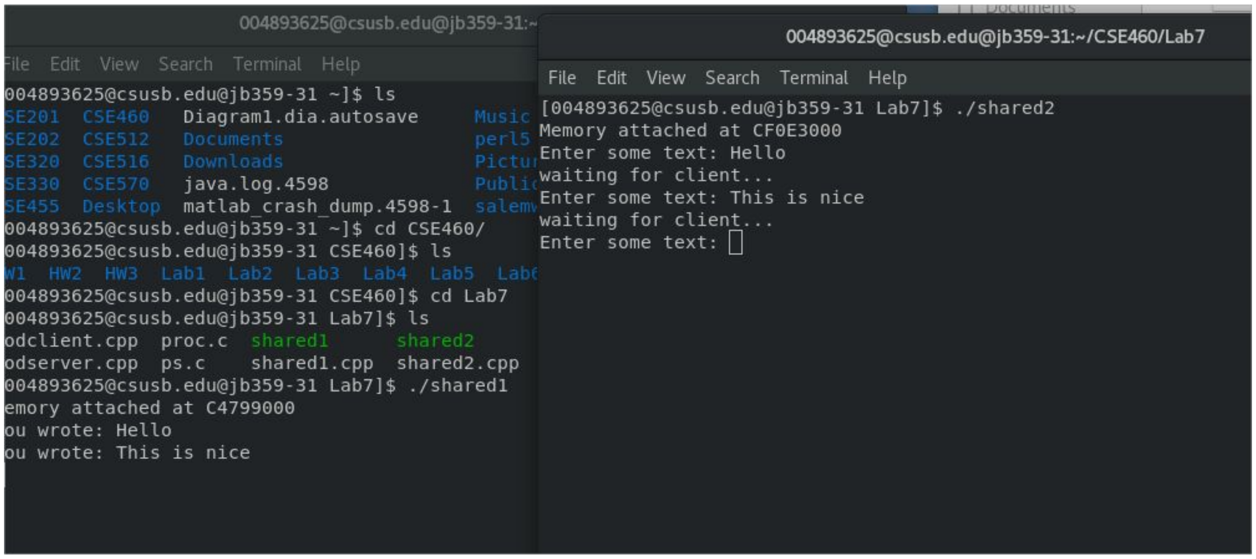
```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
#include <semaphore.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <iostream>

#define SHMSZ 27
#define TEXT_SZ 2048
char SEM_NAME[] = "SEM";
struct shared_use_st {
int written_by_you;
char some_text[TEXT_SZ];
};
int main() {
int running = 1;
void *shared_memory = (void *)0;
struct shared_use_st *shared_stuff;
char buffer[BUFSIZ];
int shmid;
key_t key;
char *shm,*s;
sem_t *mutex;
key = 1000;
//creates and initializes semaphore
mutex = sem_open(SEM_NAME, O_CREAT, 0644, 1);
if (mutex == SEM_FAILED) {
perror("Unable to create semaphore");
sem_unlink(SEM_NAME);
exit(-1);
}
shm = (char*) shmat(shmid, NULL, 0);
//attach to shared mem
shmid = shmget(key, SHMSZ, IPC_CREAT | 0666);
if(shmid < 0) {
perror("Failure in shmget");
exit(-1);
}
shmid = shmget((key_t)1234, sizeof(struct shared_use_st), 0666 |
IPC_CREAT);if (shmid == -1) {
fprintf(stderr, "shmget failed\n");
exit(EXIT_FAILURE);
}
shared_memory = shmat(shmid, (void *)0, 0);
if (shared_memory == (void *)-1) {
fprintf(stderr, "Failure\n");
```

Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20



```c
fprintf(stderr, "Failure\n");
exit(EXIT_FAILURE);
}
shared_stuff = (struct shared_use_st *)shared_memory;
while(running) {
while(shared_stuff->written_by_you == 1) {
sleep(1);
}
sem_wait(mutex);
printf("Enter some text: ");
fgets(buffer, BUFSIZ, stdin);
strncpy(shared_stuff->some_text, buffer, TEXT_SZ);
shared_stuff->written_by_you = 1;
sem_post(mutex);
if (strncmp(buffer, "quit", 3) == 0) {
running = 0;
}
}
if (shmdt(shared_memory) == -1) {
fprintf(stderr, "shmdt failed\n");
exit(EXIT_FAILURE);
}
sem_close(mutex);
sem_unlink(SEM_NAME);
shmctl(shmid, IPC_RMID, 0);
exit(0);
}
```

Output:

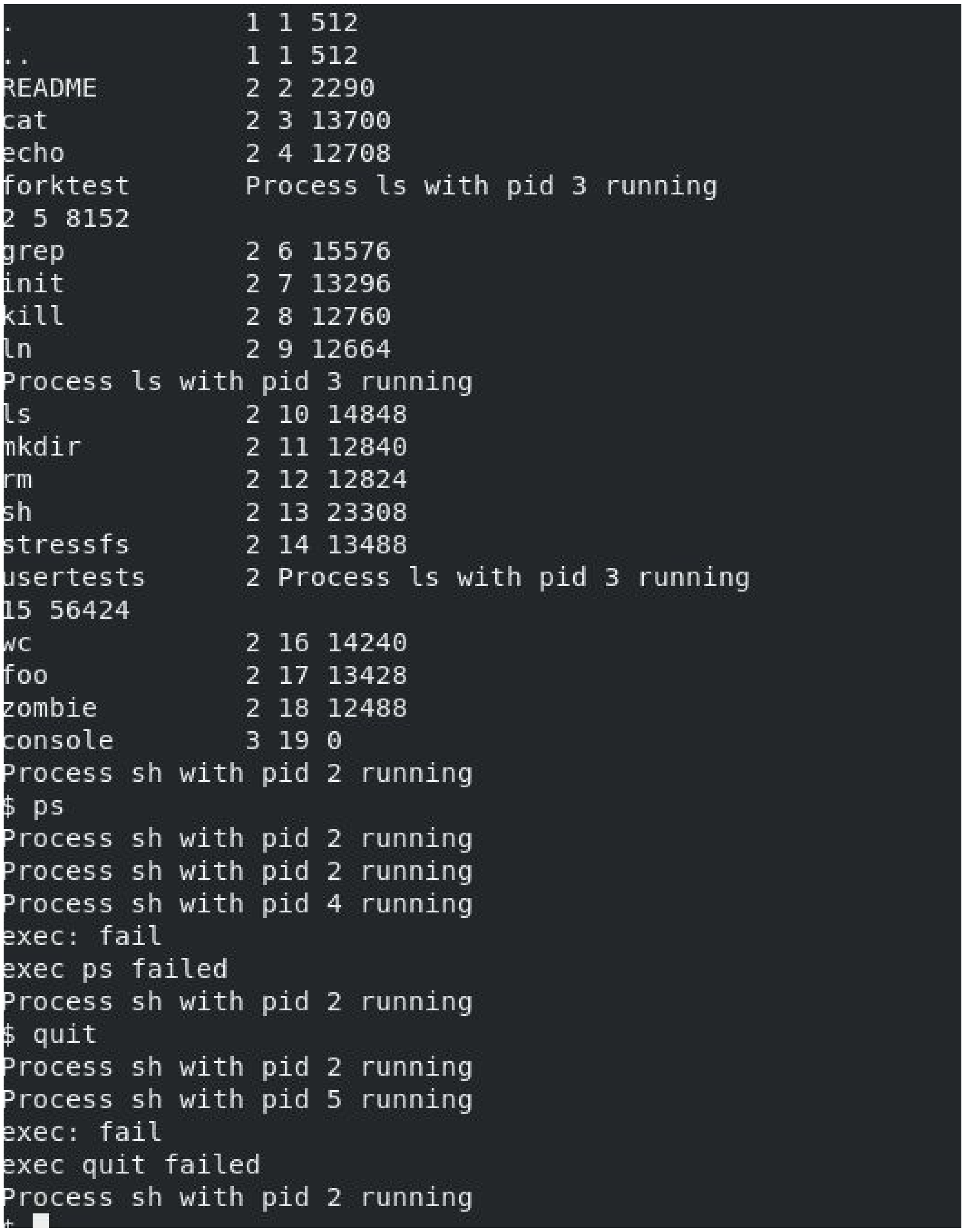


```
004893625@csusb.edu@jb359-31 ~]$ ls
SE201  CSE460   Diagram1.dia.autosave     Music
SE202  CSE512   Documents                 perl5
SE320  CSE516   Downloads                 Pictur
SE330  CSE570   java.log.4598             Public
SE455  Desktop  matlab_crash_dump.4598-1  salemw
004893625@csusb.edu@jb359-31 ~]$ cd CSE460/
004893625@csusb.edu@jb359-31 CSE460]$ ls
W1  HW2  HW3  Lab1  Lab2  Lab3  Lab4  Lab5  Lab6
004893625@csusb.edu@jb359-31 CSE460]$ cd Lab7
004893625@csusb.edu@jb359-31 Lab7]$ ls
odclient.cpp  proc.c  shared1      shared2
odserver.cpp  ps.c    shared1.cpp  shared2.cpp
004893625@csusb.edu@jb359-31 Lab7]$ ./shared1
emory attached at C4799000
ou wrote: Hello
ou wrote: This is nice
```

```
[004893625@csusb.edu@jb359-31 Lab7]$ ./shared2
Memory attached at CF0E3000
Enter some text: Hello
waiting for client...
Enter some text: This is nice
waiting for client...
Enter some text: 
```

Waled Salem and Saker Awad
CSE 460 LAB 7
SCORE: 20/20

3. XV6 System Calls



```
.               1 1 512
..              1 1 512
README          2 2 2290
cat             2 3 13700
echo            2 4 12708
forktest        Process ls with pid 3 running
2 5 8152
grep            2 6 15576
init            2 7 13296
kill            2 8 12760
ln              2 9 12664
Process ls with pid 3 running
ls              2 10 14848
mkdir           2 11 12840
rm              2 12 12824
sh              2 13 23308
stressfs        2 14 13488
usertests       2 Process ls with pid 3 running
15 56424
wc              2 16 14240
foo             2 17 13428
zombie          2 18 12488
console         3 19 0
Process sh with pid 2 running
$ ps
Process sh with pid 2 running
Process sh with pid 2 running
Process sh with pid 4 running
exec: fail
exec ps failed
Process sh with pid 2 running
$ quit
Process sh with pid 2 running
Process sh with pid 5 running
exec: fail
exec quit failed
Process sh with pid 2 running
$
```

Everything i completed for this assignment therefore the fulll score we expect is a **20/20 on this lab assignment.**