<html><head></head><body><pre style="word-wrap: break-word; white-space: pre-wrap;">/**
 * Simple program demonstrating shared memory in POSIX systems.
 *
 * This is the producer process that writes to the shared memory region.
 *
 * Figure 3.17
 *
 * @author Silberschatz, Galvin, and Gagne
 * Operating System Concepts - Ninth Edition
 * Copyright John Wiley & Sons - 2013
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <sys/shm.h>
#include <sys/stat.h>
#include <sys/mman.h>

int main()
{
        const int SIZE = 4096;
        const char *name = "OS";
        const char *message0= "Studying ";
        const char *message1= "Operating Systems ";
        const char *message2= "Is Fun!";

        int shm_fd;
        void *ptr;

        /* create the shared memory segment */
        shm_fd = shm_open(name, O_CREAT | O_RDWR, 0666);

        /* configure the size of the shared memory segment */
        ftruncate(shm_fd,SIZE);

        /* now map the shared memory segment in the address space of the process */
        ptr = mmap(0,SIZE, PROT_READ | PROT_WRITE, MAP_SHARED, shm_fd, 0);
        if (ptr == MAP_FAILED) {
                printf("Map failed\n");
                return -1;
        }

        /**
         * Now write to the shared memory region.
         *
         * Note we must increment the value of ptr after each write.
         */
        sprintf(ptr,"%s",message0);
        ptr += strlen(message0);
        sprintf(ptr,"%s",message1);
        ptr += strlen(message1);
        sprintf(ptr,"%s",message2);
        ptr += strlen(message2);

        return 0;
}
</pre></body></html>