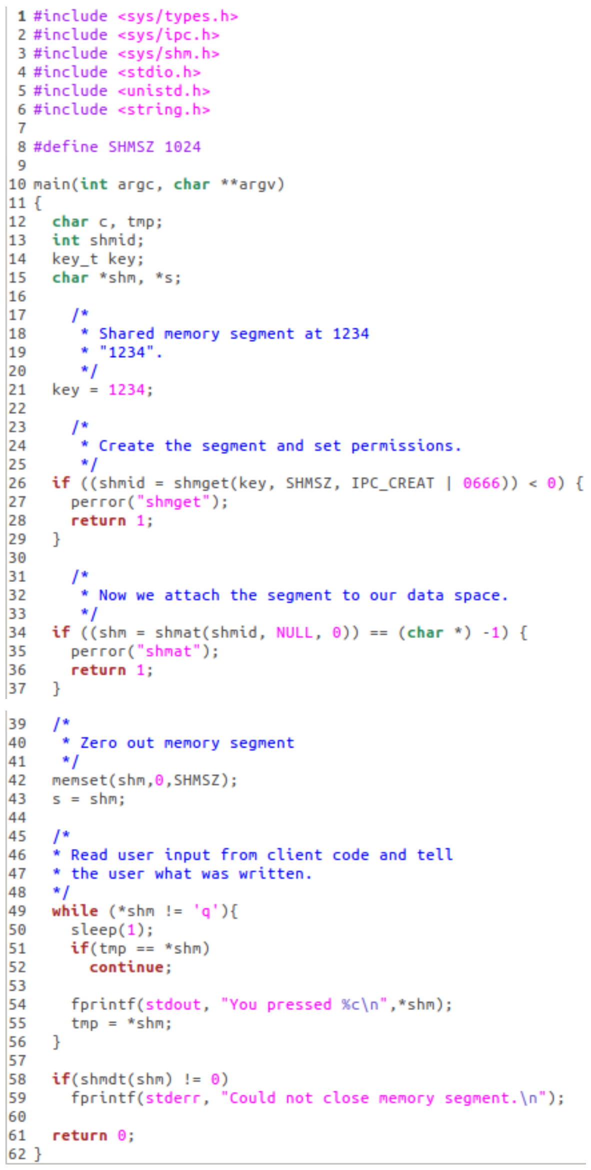
National University
of Computer & Emerging Sciences
Peshawar Campus

Problem Set:	Assignment 06	**Semester**:	Spring 2013
Points:	4		
Date Set:	April 16, 2013	**Due Date**:	April 24, 2013
Course:	CS206 Operating Systems	**Instructor**:	Nauman

Note: Code in the following is intentionally left low-quality. Please write the code yourself.

1. The following code deals with creating shared memory segments between two procesess. The code is well commented and complete. You only need to type it up, compile it and execute it.
2. There are two programs: `shm_server.c` and `shm_client.c`.
3. The server waits for changes to a shared memory location and prints the character written to it. Here's the code for the server:



```c
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>

#define SHMSZ 1024

main(int argc, char **argv)
{
  char c, tmp;
  int shmid;
  key_t key;
  char *shm, *s;

    /*
     * Shared memory segment at 1234
     * "1234".
     */
  key = 1234;

    /*
     * Create the segment and set permissions.
     */
  if ((shmid = shmget(key, SHMSZ, IPC_CREAT | 0666)) < 0) {
    perror("shmget");
    return 1;
  }

    /*
     * Now we attach the segment to our data space.
     */
  if ((shm = shmat(shmid, NULL, 0)) == (char *) -1) {
    perror("shmat");
    return 1;
  }
  /*
   * Zero out memory segment
   */
  memset(shm,0,SHMSZ);
  s = shm;

  /*
  * Read user input from client code and tell
  * the user what was written.
  */
  while (*shm != 'q'){
    sleep(1);
    if(tmp == *shm)
      continue;

    fprintf(stdout, "You pressed %c\n",*shm);
    tmp = *shm;
  }

  if(shmdt(shm) != 0)
    fprintf(stderr, "Could not close memory segment.\n");

  return 0;
}
```

4. Compile the program and execute it in a terminal. Then open another terminal for the client.

5. The client reads a character from the console and writes it to the shared memory. Here's the code:

```c
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>

#define SHMSZ     1024

main()
{
  int shmid;
  key_t key;
  char *shm, *s;

  /*
  * We need to get the segment named
  * "1234", created by the server.
  */
  key = 1234;

  /*
  * Locate the segment.
  */
  if ((shmid = shmget(key, SHMSZ, 0666)) < 0) {
    perror("shmget");
    return 1;
  }

  /*
  * Now we attach the segment to our data space.
  */
  if ((shm = shmat(shmid, NULL, 0)) == (char *) -1) {
    perror("shmat");
    return 1;
  }

  /*
  * Zero out memory segment
  */
  memset(shm,0,SHMSZ);
  s = shm;

  /*
  * Client writes user input character to memory
  * for server to read.
  */
  for(;;){
    char tmp = getchar();
    // Eat the enter key
    getchar();

    if(tmp == 'q'){
      *shm = 'q';
      break;
    }
    *shm = tmp;
  }

  if(shmdt(shm) != 0)
    fprintf(stderr, "Could not close memory segment.\n");

  return 0;
}
```

6. Compile the client program and execute it in the second terminal. Enter any character then hit enter. Observe the output in the server terminal.

7. Submit the source code and screenshots of execution.

Note: You might want to read the manual of `shmget` and related commands to understand fully what's going on with the code.