OS LAB 6 Shared Memory

Aniruddha Amit Dutta

Roll - 58

180905488

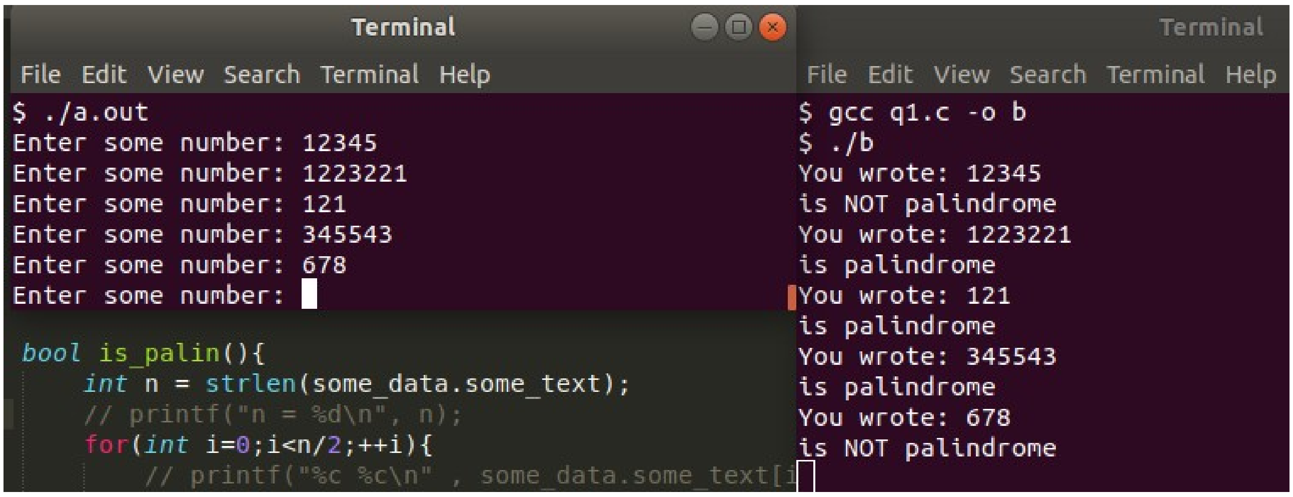
Q1. message queue

//check palindrome

```
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include<errno.h>
#include<unistd.h>
#include <sys/msg.h>
#include <stdbool.h>

struct my_msg_st {
	long int my_msg_type;
	char some_text[BUFSIZ];
};

struct my_msg_st some_data;

bool is_palin(){
	int n = strlen(some_data.some_text);
	// printf("n = %d\n", n);
	for(int i=0;i<n/2;++i){
		// printf("%c %c\n" , some_data.some_text[i],some_data.some_text[n-i-2]);
		if(some_data.some_text[i]!=some_data.some_text[n-i-2]){
			return false;
		}
	}
	return true;
}

int main()
{
	int running = 1;
	int msgid;

	long int msg_to_receive = 0;
	msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
	if (msgid == -1) {
		fprintf(stderr, "msgget failed with error: %d\n", errno);
		exit(EXIT_FAILURE);
	}
	while(running) {
		if (msgrcv(msgid, (void *)&some_data, BUFSIZ,msg_to_receive, 0) == -1) {
			fprintf(stderr, "msgrcv failed with error: %d\n", errno);
			exit(EXIT_FAILURE);
		}
		printf("You wrote: %s", some_data.some_text);
		if(is_palin()){
			printf("is palindrome\n");
		}else{
			printf("is NOT palindrome\n");
		}
		if (strncmp(some_data.some_text, "end", 3) == 0) {
			running = 0;
		}

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


// enter num

#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include<errno.h>
#include<unistd.h>
#include <sys/msg.h>

#define MAX_TEXT 512

struct my_msg_st {
	long int my_msg_type;
	char some_text[BUFSIZ];
};

int main()
{
	int running = 1;
	struct my_msg_st some_data;
	int msgid;
	char buffer[BUFSIZ];
	msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
	if (msgid == -1) {
		fprintf(stderr, "msgget failed with error: %d\n", errno);
		exit(EXIT_FAILURE);
	}
	while(running) {
```

```
                printf("Enter some number: ");
                fgets(buffer, BUFSIZ, stdin);
                some_data.my_msg_type = 1;
                strcpy(some_data.some_text, buffer);
                if (msgsnd(msgid, (void *)&some_data, MAX_TEXT, 0) == -1) {
                        fprintf(stderr, "msgsnd failed\n");
                        exit(EXIT_FAILURE);
                }
                if (strncmp(buffer, "end", 3) == 0) {
                running = 0;
                }
        }
        exit(EXIT_SUCCESS);
}
```

output ->



```
$ ./a.out
Enter some number: 12345
Enter some number: 1223221
Enter some number: 121
Enter some number: 345543
Enter some number: 678
Enter some number:
```

```
bool is_palin(){
    int n = strlen(some_data.some_text);
    // printf("n = %d\n", n);
    for(int i=0;i<n/2;++i){
        // printf("%c %c\n" , some_data.some_text[i
```

```
$ gcc q1.c -o b
$ ./b
You wrote: 12345
is NOT palindrome
You wrote: 1223221
is palindrome
You wrote: 121
is palindrome
You wrote: 345543
is palindrome
You wrote: 678
is NOT palindrome
```

Q2. Shared memeory

// shared memeory shm_com.h

```
#define TEXT_SZ 2048

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

// enter an alphabet

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
```

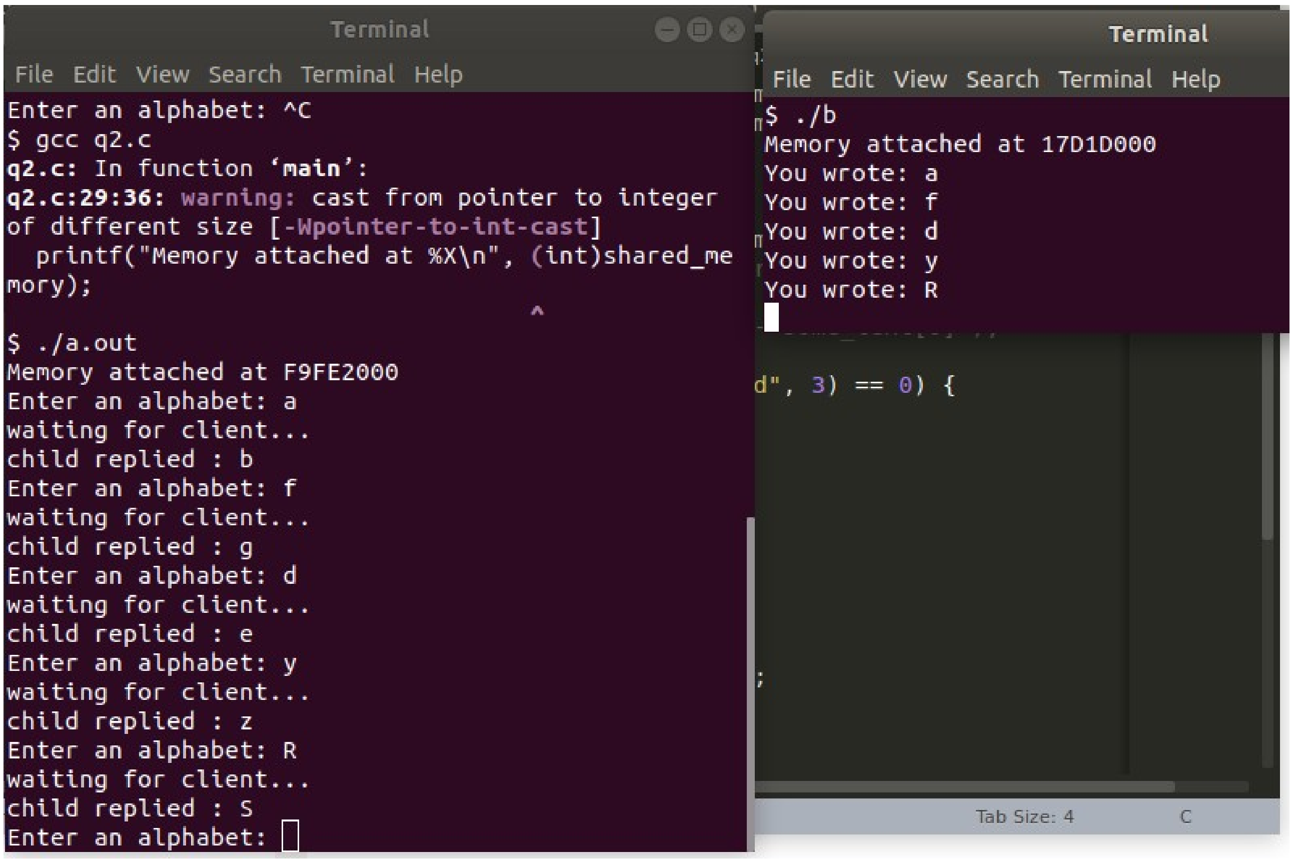
```
#include <sys/shm.h>
#include "shm_com.h"

int main()
{
	int running = 1,cnt=0;
	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", (int)shared_memory);
	shared_stuff = (struct shared_use_st *)shared_memory;

	while(running) {
		while(shared_stuff->written_by_you == 1) {
			sleep(4);
			printf("waiting for client...\n");
		}
		if(cnt){
			printf("child replied : ");
			printf("%c \n",shared_stuff->some_text[0]);
		}

		printf("Enter an alphabet: "); ++cnt;
		fgets(buffer, BUFSIZ, stdin);
		strncpy(shared_stuff->some_text, buffer, TEXT_SZ);
		shared_stuff->written_by_you = 1;
		if (strncmp(buffer, "end", 3) == 0) {
			running = 0;
		}
	}

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

// get next alphabet

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <sys/shm.h>
#include "shm_com.h"

int main()
{
	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);
	}
	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", (int)shared_memory);
	shared_stuff = (struct shared_use_st *)shared_memory;
	shared_stuff->written_by_you = 0;
	while(running) {
		if (shared_stuff->written_by_you) {
			printf("You wrote: %s", shared_stuff->some_text);
			sleep( rand() % 4 ); /* make the other process wait for us ! */
			shared_stuff->some_text[0] += 1;
			// printf(" changed = %c\n",shared_stuff->some_text[0] );
			shared_stuff->written_by_you = 0;
			if (strncmp(shared_stuff->some_text, "end", 3) == 0) {
				running = 0;
			}
		}
	}
	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);
	}
	exit(EXIT_SUCCESS);
```

}

output ->



```
Enter an alphabet: ^C
$ gcc q2.c
q2.c: In function ‘main’:
q2.c:29:36: warning: cast from pointer to integer
of different size [-Wpointer-to-int-cast]
  printf("Memory attached at %X\n", (int)shared_me
mory);
                                    ^
$ ./a.out
Memory attached at F9FE2000
Enter an alphabet: a
waiting for client...
child replied : b
Enter an alphabet: f
waiting for client...
child replied : g
Enter an alphabet: d
waiting for client...
child replied : e
Enter an alphabet: y
waiting for client...
child replied : z
Enter an alphabet: R
waiting for client...
child replied : S
Enter an alphabet:
```

```
$ ./b
Memory attached at 17D1D000
You wrote: a
You wrote: f
You wrote: d
You wrote: y
You wrote: R
```

Terminal
File Edit View Search Terminal Help
Terminal
File Edit View Search Terminal Help
d", 3) == 0) {
Tab Size: 4
C