OS Lab 6

Reg No: 180905416
Name: Paawan Kohli

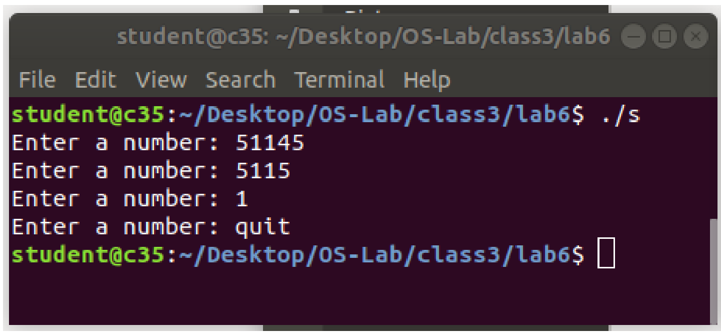
Q1. Process A wants to send a number to Process B. Once recieved, Process B has to check whether the number is palindrome or not. Write a C program to implement this interprocess communication using a message queue.

sender.c:

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

#define MAX_TEXT 512

struct message {
	long int msgType;
	char msgData[BUFSIZ];
};

int main() {
	struct message m1;
	char buffer[BUFSIZ];

	int msgid = msgget((key_t)1234, 0666 | IPC_CREAT);

	if (msgid == -1) {
		printf("msgget failed with error: %d\n", errno);
		exit(EXIT_FAILURE);
	}

	int run = 1;

	while (run) {

		printf("Enter a number:\t");
		fgets(buffer, BUFSIZ, stdin);

		m1.msgType = 1;
		strcpy(m1.msgData, buffer);

		if (msgsnd(msgid, (void *)&m1, MAX_TEXT, 0) == -1) {
			printf("msgsnd failed\n");
			exit(EXIT_FAILURE);
		}

		if (strncmp(buffer, "quit", 3) == 0) {
			run = 0;
		}
	}

	exit(EXIT_SUCCESS);
}
```

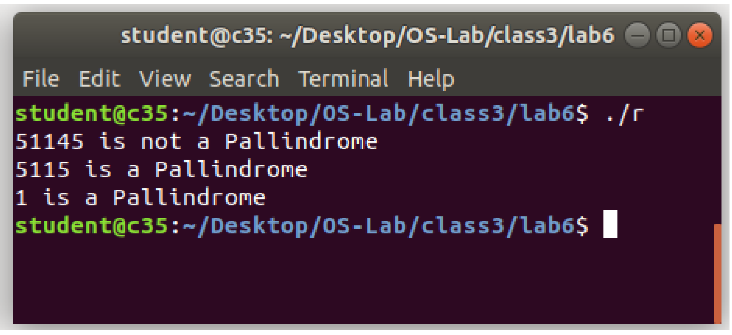
reciever.c:

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

struct mesage {
     long int msgType;
     char msgData[BUFSIZ];
};

int main() {
     struct mesage m1;
     long int msg_to_recieve = 0;

     int msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
     if (msgid == -1) {
          printf("msgget failed with error: %d\n", errno);
          exit(EXIT_FAILURE);
     }

     int run = 1;
     while (run) {
          if (msgrcv(msgid, (void*)&m1, BUFSIZ, msg_to_recieve, 0) == -1) {
               printf("msgrcv failed with error: %d\n", errno);
               exit(EXIT_FAILURE);
          }

          if (strncmp(m1.msgData, "quit", 3) == 0) {
               run = 0; break;
          }

          // check for palindrome
          char rev[BUFSIZ];
          int i;

          for (i = 0; m1.msgData[i] != '\n'; i++);

          for (int j = 0; j < i; j++)
               rev[j] = m1.msgData[i - j - 1];

          rev[i] = '\0';
          m1.msgData[i] = '\0';

          if (strcmp(m1.msgData, rev) == 0) {
               printf("%s is a Pallindrome\n", m1.msgData);
          } else {
               printf("%s is not a Pallindrome\n", m1.msgData);
          }
     }

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

     exit(EXIT_SUCCESS);
}
```



student@c35: ~/Desktop/OS-Lab/class3/lab6
File Edit View Search Terminal Help

```
student@c35:~/Desktop/OS-Lab/class3/lab6$ ./s
Enter a number: 51145
Enter a number: 5115
Enter a number: 1
Enter a number: quit
student@c35:~/Desktop/OS-Lab/class3/lab6$
```



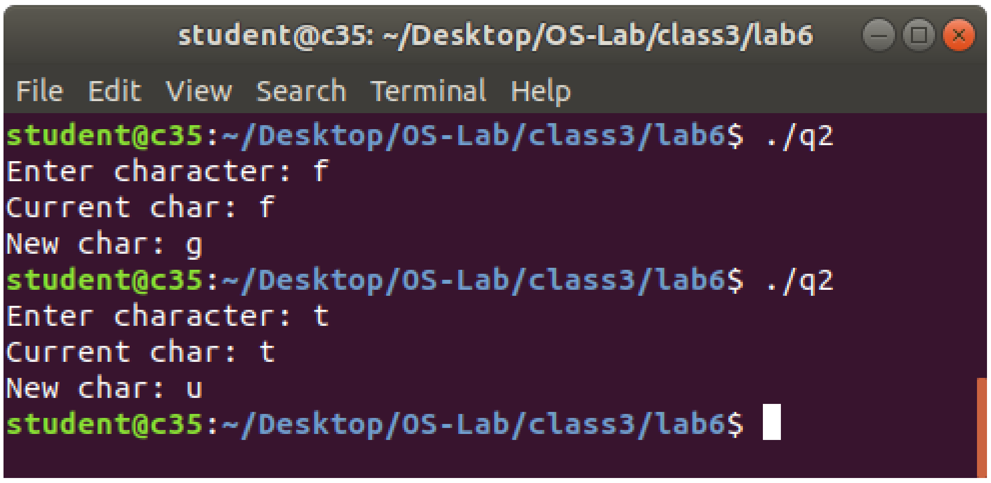
student@c35: ~/Desktop/OS-Lab/class3/lab6
File Edit View Search Terminal Help

```
student@c35:~/Desktop/OS-Lab/class3/lab6$ ./r
51145 is not a Pallindrome
5115 is a Pallindrome
1 is a Pallindrome
student@c35:~/Desktop/OS-Lab/class3/lab6$
```

Q2. Implement a parent process, which sends an english alphabet to a child process using shared memory. The child process responds with the next english alphabet to the parent. The parent displays the reply from the child.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/wait.h>

struct shared_use {
      int status;
      char c;
      char cnext;
};

int main() {
      void *shared_memory = (void *)0;
      struct shared_use *commonptr;
      commonptr->status = 0;

      pid_t pid = fork();

      if (pid == -1) {
            printf("Fork failed!!\n");
            exit(EXIT_FAILURE);
      }

      // child
      if (pid == 0) {
            int shmid = shmget((key_t)1234, sizeof(struct shared_use), 0666 |
                                IPC_CREAT);

            if (shmid == -1) {
                  printf("shmget failed!!\n");
                  exit(EXIT_FAILURE);
            }

            shared_memory = shmat(shmid, (void *)0, 0);

            if (shared_memory == (void *) - 1) {
                  printf("shmat failed\n");
                  exit(EXIT_FAILURE);
            }
```

```
        commonptr = (struct shared_use *)shared_memory;

        // wait for the parent to write to shared memory
        while(commonptr->status == 0);

        // child does work
        commonptr->cnext = commonptr->c + 1;

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

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

        exit(EXIT_SUCCESS);
    }

    // parent
    else {
        int shmid = shmget((key_t)1234, sizeof(struct shared_use), 0666 |
                        IPC_CREAT);

        if (shmid == -1) {
            printf("shmget failed!!\n");
            exit(EXIT_FAILURE);
        }

        shared_memory = shmat(shmid, (void *)0, 0);

        if (shared_memory == (void *) - 1) {
            printf("shmat failed\n");
            exit(EXIT_FAILURE);
        }

        commonptr = (struct shared_use *)shared_memory;

        char ch;
        printf("Enter character: ");
        scanf("%c", &ch);

        commonptr->c = ch;

        // indicates that parent has written to shared memory
        commonptr->status = 1;

        printf("Current char: %c\n", commonptr->c);

        // wait for the child to finsh its work
        wait(NULL);
        printf("New char: %c\n", commonptr->cnext);

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

        exit(EXIT_SUCCESS);
    }
}
```


student@c35: ~/Desktop/OS-Lab/class3/lab6
File Edit View Search Terminal Help

```
student@c35:~/Desktop/OS-Lab/class3/lab6$ ./q2
Enter character: f
Current char: f
New char: g
student@c35:~/Desktop/OS-Lab/class3/lab6$ ./q2
Enter character: t
Current char: t
New char: u
student@c35:~/Desktop/OS-Lab/class3/lab6$ 
```