CS-342 Operating Systems

Project 3

Section 2

Atilla Alp Yavuz 22102191

Tolga Artun Koçak 22102132

NOTE: The project includes every functionality, however, we chose to prevent collisions when creating message queues and also retain an ability to overwrite previously deleted memory addreses. To accomplish this we used bitmaps and which did not fit in the fixed area. This means that while individual processes can interact with memory in the library, inter-process communication is lacking. To show functionality we will use code snippets and their outputs as evidence of their functioning. These code snippets will be single process but will convey the functioning of the system itself.

Connecting creating and destroying

```
mf_connect();
mf_create( mqname: "mq1", mqsize: 16);
mf_create( mqname: "mq2", mqsize: 16);
mf_create( mqname: "mq3", mqsize: 16);
mf_create( mqname: "mq4", mqsize: 16);
mf_remove( mqname: "mq2");
mf_create( mqname: "mq5", mqsize: 32);

mf_create( mqname: "mq6", mqsize: 16);
mf_disconnect();
return 0;
```

The code above is the source code of the following output. Note that the mq2 message queue is deleted after which a replacement of the correct size (mq6) is placed into its previous address. Also the lack of mf_init will represent that these functions are capable of communicating with a server. The outputs can be seen below. The codes for these tests will be sent along with the code. This is the App1.c function.

```
4096Connection succesfulBase 0x7dd3927bc000
Message queue created with name mq1 at 0x7dd3927bc050
Message queue ends at 0x7dd3927bc0d0
Allocated 128 blocks starting at 0 (bitmap idx 15, bit 7)
Base 0x7dd3927bc000
Message queue created with name mq2 at 0x7dd3927bc0d0
Message queue ends at 0x7dd3927bc150
Allocated 128 blocks starting at 128 (bitmap idx 31, bit 7)
Base 0x7dd3927bc000
Message queue created with name mq3 at 0x7dd3927bc150
Message queue ends at 0x7dd3927bc1d0
Allocated 128 blocks starting at 256 (bitmap idx 47, bit 7)
Base 0x7dd3927bc000
Message queue created with name mq4 at 0x7dd3927bc1d0
Message queue ends at 0x7dd3927bc250
Allocated 128 blocks starting at 384 (bitmap idx 63, bit 7)
Cleared
Base 0x7dd3927bc000
Message queue created with name mq5 at 0x7dd3927bc250
Message queue ends at 0x7dd3927bc350
Allocated 256 blocks starting at 512 (bitmap idx 95, bit 7)
Base 0x7dd3927bc000
Message queue created with name mq6 at 0x7dd3927bc0d0
Message queue ends at 0x7dd3927bc150
Allocated 128 blocks starting at 128 (bitmap idx 31, bit 7)
Disconnect succesfulartun@Linuxtest:~/Desktop/a/b$ S
```

Messages

Messages can be sent and received also along the messege queues, as you can observe from the example below. messages are FIFO as the first messages are output first as well. The MQ's are connected to via the mf_send and mf_recv functions. Even though mf_open and mf_close were implemented we were not able to convey that and instead chose to represent the send and recieve functions.

```
mf_connect();
mf_create( mqname: "mq1", mqsize: 16);
mf_create( mqname: "mq2", mqsize: 16);
mf_create( mqname: "mq3", mqsize: 16);

mf_remove( mqname: "mq2");
mf_create( mqname: "mq4" ,  mqsize: 16);


char *bufptr = (char *) malloc ( size: 8);
mf_send ( qid: 0,  bufptr: (void *) bufptr,  datalen: 4);
printf( format: "%s\n",(char*)bufptr);
char *bufptre = (char *) malloc ( size: 8);
mf_send ( qid: 0,  bufptr: (void *) bufptre,  datalen: 8);
char *bufptrew = (char *) malloc ( size: 8);
mf_send ( qid: 0,  bufptr: (void *) bufptrew,  datalen: 4);

char *bufptrewww = (char *) malloc ( size: 16);
mf_recv( qid: 0, bufptr: bufptrewww, bufsize: 32);
printf( format: "%s\n",(char*)bufptrewww);
```

The output can be seen below.

```
Message queue created with name mq1 at 0x73e05a99d050
Message queue ends at 0x73e05a99d0d0
Allocated 128 blocks starting at 0 (bitmap idx 15, bit 7)
Base 0x73e05a99d000
Message queue created with name mq2 at 0x73e05a99d0d0
Message queue ends at 0x73e05a99d150
Allocated 128 blocks starting at 128 (bitmap idx 31, bit 7)
Base 0x73e05a99d000
Message queue created with name mq3 at 0x73e05a99d150
Message queue ends at 0x73e05a99d1d0
Allocated 128 blocks starting at 256 (bitmap idx 47, bit 7)
The reference count is not zero
Base 0x73e05a99d000
Message queue created with name mq4 at 0x73e05a99d1d0
Message queue ends at 0x73e05a99d250
Allocated 128 blocks starting at 384 (bitmap idx 63, bit 7)
Available space: 256, Data length: 32
Message added at 0x73e05a99d050, new base_ptr is 0x73e05a99d090
Z◆s
Available space: 192, Data length: 64
Message added at 0x73e05a99d090, new base_ptr is 0x73e05a99d0f0
Available space: 128, Data length: 32
Message added at 0x73e05a99d0f0, new base_ptr is 0x73e05a99d130
Z◆s
```