# CS60038: Assignment 2

## (Handling Syscalls using LKM)

This assignment is a continuation of Assignment 1. In this assignment, you now need to develop an LKM that supports various ioctl calls to set and retrieve various configurations and internal information respectively. The following table describes the ioctl commands and their task. Here, upon insertion of this LKM, it will create a file at the path **/proc/cs60038_a2_<roll no>**. This path will be world-readable and writable. A userspace program will interact with the LKM through this file. A userspace program can fire an ioctl system call any time it wants.

| Command | Description |
|---|---|
| PB2_SET_CAPACITY | This command initializes the LKM with the maximum deque capacity. If LKM is already initialized for the current process, this command resets the LKM and reinitializes again. It expects a pointer to a 32-bit integer. The value must be in the range between 1 and 100 (both inclusive). Invalid value or inaccessible pointer causes an error and sets the error number to EINVAL. On success, it returns 0.<br>No other operation can be performed (including read or write) before performing this system call. Every other system call on the LKM returns an error (-1) with code EACCES. |
| PB2_INSERT_RIGHT | This command inserts an integer into the right of the queue.. It takes a pointer to an integer as the input value. Invalid value causes an error in LKM and sets error number to EINVAL. On success, this command returns 0.<br>If the deque is full, it returns an error (-1) with code EACCES. |
| PB2_INSERT_LEFT | This command inserts an integer into the left of the queue.. It takes a pointer to an integer as the input value. Invalid value causes an error in LKM and sets error number to EINVAL. On success, this command returns 0.<br>If the deque is full, it returns an error (-1) with code EACCES. |
| PB2_GET_INFO | This command returns the information about the deque. PB2_GET_INFO command expects a pointer to a structure object of type **struct obj_info** as the value. On success, LKM returns 0 and fills the various fields of the pointer passed by the process. On error, LKM sets appropriate error no and returns -1. |
| PB2_POP_LEFT | This extracts the leftmost element from the deque. This function returns 0 on success and -1 on error. |
| PB2_POP_RIGHT | This extracts the rightmost element from the deque. This function returns 0 on success and -1 on error. |

The definition of ioctl commands are as follows:

```
#define PB2_SET_CAPACITY         _IOW(0x10, 0x31, int32_t*)
#define PB2_INSERT_RIGHT         _IOW(0x10, 0x32, int32_t*)
#define PB2_INSERT_LEFT          _IOR(0x10, 0x33, int32_t*)
#define PB2_GET_INFO             _IOR(0x10, 0x34, int32_t*)
#define PB2_EXTRACT_LEFT         _IOR(0x10, 0x35, int32_t*)
#define PB2_EXTRACT_RIGHT        _IOR(0x10, 0x36, int32_t*)
```

The definition of structures are as follows:

```
struct obj_info {
    int32_t deque_size; // current number of elements in deque
    int32_t capacity; // maximum capacity of the deque
};
```

A userspace process interacts with the LKM in the following manner only.

1. It will open the file (**/proc/cs60038_a2_<roll no>**) in read-write mode and obtain the file descriptor.
2. Userspace process calls the ioctl system call with command PB2_SET_CAPACITY and appropriate value to set the maximum capacity of the deque in the kernel.
3. Insert object using the ioctl system call with commands PB2_INSERT_LEFT and PB2_INSERT_RIGHT as many times as you want. However, there is no bound on the number of times the user program can call this insert operation. LKM verifies the arguments in the system call and inserts the input into the deque. In case the deque is full, an appropriate error is returned.
4. To perform read/ extract data from deque, the userspace program uses ioctl system call with command PB2_POP_LEFT or PB2_POP_RIGHT. The system call returns 0 in case of successful execution and populates the result**,** whose pointer is passed while making the call.
5. When all the operations are done, the userspace process closes the file and LKM releases all the resources allocated for the process.
6. In between the userspace program can call the ioctl system call with command PB2_GET_INFO to retrieve information and the current state of the deque.

LKM should be able to handle concurrency and separate data from multiple processes. Also, no userspace program should be able to open the file more than once simultaneously. However, the userspace process should be able to reset the LKM (for the said process) by reopening the file.

### Submission Details:

- Students have to submit the LKM Code and the corresponding Makefile.
- In every source code file write your Roll no in the comments at the top.
- Compress these files into a zip file and name the zip file: **ASGN2_<ROLL_NO>.zip.**