WEEK ONE UPDATE

5G TESTBED PROJECT

NADIPINENI HEMANTH KUMAR

23rd May, 2020

READING ASSIGNMENT

1. Recap of data structures – C struct, Ring buffers, Arrays, Lists & Queues - Completed(.)
2. Dynamic Memory Allocation - Completed
3. Multi-threading programming - Completed
4. Stack memory & Heap memory - Completed
5. Ring Buffers - Completed

Questions:

1. Program to spawn a new process

```
#include <unistd.h>
#include <stdio.h>
int main(int argc, char *argv[]) {
 pid_t fork_pid = fork();
 if (fork_pid == 0) {
   printf("Hello from the child!\n");
 } else {
   printf("Hello from the parent!\n");
 } return 0;}
```

2. Program to spawn multiple threads

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>
void *myThread(void *vargp)
{
        sleep(1);
        printf("Printing Thread \n");
        return NULL;
}
int main()
{
        pthread_t thread_id;
        printf("Before Thread\n");
        pthread_create(&thread_id, NULL, myThread, NULL);
        pthread_join(thread_id, NULL);
        printf("After Thread\n");
        exit(0);
}
```

Advanced Question:

Allocate a huge memory block to store N blocks each of M bytes data. Take N and M as input parameters and define variables that keep track of the number of blocks of data inputted and each data block's size.

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

int main()
{
 int* ptr;
 int n, i;
 n = 5;
 printf("Enter number of elements: %d\n", n);
 ptr = (int*)malloc(n * sizeof(int));
 if (ptr == NULL) {
            printf("Memory not allocated.\n");
            exit(0);
 }
 else {
            printf("Memory successfully allocated using malloc.\n");
            for (i = 0; i < n; ++i) {
                        ptr[i] = i + 1;
            }
            printf("The elements of the array are: ");
            for (i = 0; i < n; ++i) {
                        printf("%d, ", ptr[i]);
            }
 }
 return 0;
}
```