# Homework 3 – Thread Programming

1) The following code has the problem of Thread synchronization.

```
// header files

pthread_t tid[2];
int process_count;

void* execute(void* arg)
{
	unsigned long i = 0;
	process_count += 1;
	printf ("\n Process %d has started\n", process_count);
	for (i = 0; i < (0xFFFFFFFF); i++)
		;
	printf ("\n Process %d has finished\n", process_count);
	return NULL;
}

int main(void)
{
	int i = 0;
	int error;

	while (i < 2) {
	   error = pthread_create(&(tid[i]), NULL, &execute, NULL);
	   if (error != 0)
	      printf("\nThread can't be created:[%s]", strerror(error));
		i++;
	}

	pthread_join(tid[0], NULL);
	pthread_join(tid[1], NULL);

	return 0;
}
```

- Update the code by **using mutex** to overcome the problem of thread synchronization.
- Execute the code and place the code & output screenshot in the answer file.

2) Producer / Consumer Problem

- The producer and consumer share a fixed-size buffer used as a queue.
- The producer's job is to generate data and put it in the buffer only when the buffer is not full. If the buffer is full, then the producer shouldn't be allowed to put any data into the buffer.
- The consumer's job is to consume the data from this buffer, one at a time and only when the buffer is not empty. If the buffer is empty, then the consumer shouldn't be allowed to take any data from the buffer.
- The producer and consumer should not access the buffer at the same time.

Write a c program for the above problem. It should be mutually exclusive and no deadlock.
Hint: You can refer the week 9 lab material. Understand the problem and code accordingly.
Place the code execution & output screenshot in the answer file.

Submission:

1) Name the answer file like below
   HW3_StudentID_Name
2) Deadline – 11 PM , November 7