README

In this project I implemented a producer / consumer problem. The program can read in three paramenters: main function sleep time, producer number and consumer number. It will create a number of producer and consumer threads based on the input. Then the producers will start to produce random number continuously, each producer can produce a product every 0-3 seconds. The products are stored in buffer, which is a circular array. The consumers will retrieve the product out of buffer(also one product per 0-3 seconds). The manipulation of producer and consumer on buffer is synchronized by using semaphores, to make sure the mutual exclusive. The main function will sleep for a certain period of time based on input and then start the termination process. The termination can also triggered by pressing Ctrl+C.

When the program start terminating(either by Ctrl+C or by main function running out of sleep time), the flag stop was set as 1. The producers will test the flag and if it is 1, it will terminate. After termination of producers, it will joined in main program by using thread_join function. In order to stop consumers, -1 was loaded into the buffer as a signal, the number of -1 is the same as the number of consumers. When receiving -1, the consumers will stop.

So when termination complete, **all of the semaphores, flags, are at their original state. The buffer will be empty.** The product the producers produced will be equal to the products the consumers consumed. So it will be a synchronous termination.

This folder contain 5 files:
1. buffer.h: which is the .h file containing semaphores and declaration of functions for buffer class.
2. buffer.c: which is the implementation of buffer which used a double ended queue.
3. main.c: which implements a producer/consumer problem
4. makefile: which can help compile the file and generate main.x
5. documentation.pdf: the documentation of project

To run the program:
Linux environment is needed, conpile the code and run the main.x with three parameters: main function sleep time, producer number and consumer number.
Example :

```
./main.x 100 6 5
```

Example output:

```
producer produced 1649614975
consumer consumed 1649614975
producer produced 505432479
producer produced 1910950088
consumer consumed 1910950088
consumer consumed 505432479
producer produced 1090349761
consumer consumed 1090349761
producer produced 841311189
producer produced 747344401
consumer consumed 747344401
consumer consumed 841311189
```

```
producer produced 1901630384
producer produced 1139401607
producer produced 1129134617
producer produced 679334807
consumer consumed 1901630384
^Cconsumer consumed 1139401607
consumer consumed 1129134617
consumer consumed 679334807
```

From the output above, we can see that the number of “producer produced...” is the same as the number of “consumer consumed...”, which means the buffer is empty when termination complete. The product produced early is also consumed early, since the buffer is “first in first out”. The start of termination is happened at line 18(^C symbol). When the termination start, the consumer are still working for a short period of time to make the buffer empty, thus print out the last three lines.