CS3330
Programming Project 3

In this programming project, you will write a C++ program that simulates a shooting queue at the shooting range. Assume that each shooter has x number of bullets to shoot. Each shooter takes only one shot at a time (consumes only one bullet in each turn). They will continually shoot until all bullets are consumed. Once he/she takes one shot, they must go to the end of the line and wait for the next turn to shoot again.

At the beginning, your program must read information from the input file, "bulletCount.txt". The information includes shooters' names and the number of bullets that each shooter has at the beginning of shooting. The format of the file as follows;

```
Amanda 13
Ruth 10
Kennan 19
Kelsie 16
Patrick 11
Anastasia 11
Samson 9
Jacqueline 23
Driscoll 9
Callie 23
Amela 16
Jana 17
Cameron 10
Lester 16
Adam 5
Francis 23
Courtney 23
Jillian 9
Gil 15
Shay 8
Rylee 21
Myles 7
Illana 11
Ria 11
Ira 13
```

As seen in the figure above, each line includes a shooter's name followed by his/her bullet count. For example, Amanda has 13 bullets in her gun and will have to shoot 13 times to consume all the bullets that she has. Kennan has 19 bullets and will have to shoot 19 times to consume all the bullets that he has. Every time they shoot, bullet count will be decreased by one. This means that each shooter will shoot different number of times until the bullet count becomes zero.

Your program will have a class called "Shooter". This class will have three attributes and several member functions. For the details of the class, please see the table below.

Shooter Class		
Type	**Attribute**	**Description**
String	name	First name of the shooter
int	bulletCount	Remaining bullet count
int	initialBulletCount	Initial bullet count
Return Type	**Method**	**Description**
(constructor)	Shooter (String name, int bullet)	Construct and initialize the object
void	shooting()	Simulate the shooting and decreases bullet count by one.
String	getName()	Returns the name of the shooter
int	getQuantity()	Returns the remaining bullet count
void	display()	Displays shooter's name and remaining bullet count on the screen

You may add additional functions as necessary. Once Shooter class is ready, your program will create a queue named "shootingRange". You may use Queue from C++ STL (Standard Template Library) whose description can be found at "https://www.geeksforgeeks.org/queue-cpp-stl/". Please note that function names of the STL queue class are not matching with the names of queue we looked in the class. Please use the STL function names accordingly by reading the description of the STL. You may challenge yourself to implement your own Queue class with associated functions (enqueue(), dequeue(), front(), empty(), etc.), which is encouraged.

Once started, the program will read the shooter's information from the file, "bulletCount.txt", create shooter objects for each shooter and insert the objects created into the shootingRange queue. As the shooters' objects are inserted in the queue, your program will display the status information in the following format:

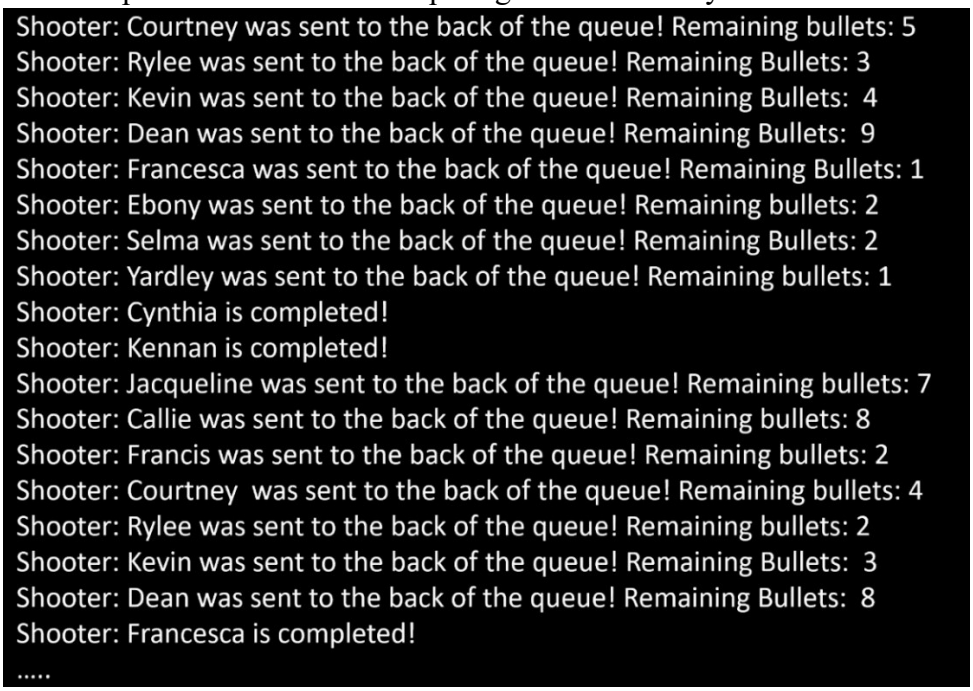
```
Shooter: Amanda added to the queue. Remaining Bullets: 13
Shooter: Ruth added to the queue. Remaining Bullets: 10
Shooter: Kennan added to the queue. Remaining Bullets: 19
Shooter: Kelsie added to the queue. Remaining Bullets: 11
Shooter: Anastasia added to the queue. Remaining Bullets: 11
Shooter: Samson added to the queue. Remaining Bullets: 9
Shooter: Jacqueline added to the queue. Remaining Bullets: 23
Shooter: Driscoll added to the queue. Remaining Bullets: 9
Shooter: Callie added to the queue. Remaining Bullets: 23
Shooter: Amelia added to the queue. Remaining Bullets: 16
Shooter: Jane added to the queue. Remaining Bullets: 17
Shooter: Cameron added to the queue. Remaining Bullets: 10
Shooter: Lester added to the queue. Remaining Bullets: 16
Shooter: Adam added to the queue. Remaining Bullets: 5
Shooter: Francis added to the queue. Remaining Bullets: 23
```

Then the shooting will start. The shooting will continue until every shooter consumes all bullets. For this, you may use while loop as follows:

```
While (shooter queue is not empty)
{
        Take the shooter at the head of the queue using front() and pop()
        Execute the shooting for that shooter
        If(the shooter's bullet count equals to zero)
        {
                Print "Shooter with Name=X is completed"
        }
        Else
        {
                Print "Shooter with Name=X needs to shoot Y more bullets"
                Send the shooter to the back of the queue using push()
                Print "Shooter Name=X is sent back to the queue. Remaining bullets: Y"
        }
```

The while loop continues until there is no one in the queue with remaining bullets. In the while loop (while the queue is not empty), you will dequeue() (pop() in the STL queue version) a shooter from the queue and let he/she shoot using the shooting() in Shooter class. This is very simple simulation where the bullet count of the shooter is decreased by 1. If the remaining bullet count is greater than zero after shooting, the shooter instance is sent back to the end of the queue using enqueue() (push() in the STL queue version) function. If the remaining bullet count is zero, the "Shooting Complete" message is displayed on the screen and the shooter doesn't have to be sent back to the queue again. In this way, in every iteration of the while loop, the shooter is dequeued from the queue, shoot and is added back to the queue (if bullets are still remaining).

A sample screen output of few execution steps is given below for your reference.



Submission Requirements

- You will submit your source code with ".cpp" extension. If you have multiple files, please archive/compress it into one file in a proper form (e.g. ".rar", ".zip"…).
- You must submit the program that compiles and runs without any error to receive full points.
- Don't forget to put comments in your source code so that the reader can understand your program easily.
- On the top of each source code file, please don't forget to put your Name.