Avila, Sean Rendel S.

Queue

```
#include <iostream>

#include <queue>

#include <string>

#include <thread>

#include <chrono>

#include <atomic>


using namespace std;


class Person {

  public:

     string name;

     int ticketNum;

     static int ticketCount;


     Person(string name){
        this->name = name;

        this->ticketNum = ticketCount++;

      }

};


int Person::ticketCount = 1;


class Queue {

  private:

    queue<Person> line;

    atomic<bool> running;


  public:

    Queue() : running(true){}


    void enqueue(Person person){

      line.push(person);

      cout << person.name << " added to the queue with
Ticket #" << person.ticketNum << endl;
```

```
      size();
    }

    void autoDequeue(){
      while (running) {
        if (!isEmpty()) {
          this_thread::sleep_for(chrono::seconds(60));

          if (!isEmpty()) {
            Person frontPerson = line.front();
            cout << "\nAfter 1 minute...\n";
            cout << "Dequeue: " << frontPerson.name << " 
received a ticket (Ticket #" << frontPerson.ticketNum << 
")\n";
            line.pop();
            size();
            peek();
          }
        }
      }
    }

    void stopDequeueing() {
        running = false;
    }

    bool isEmpty() {
        return line.empty();
    }

    void size() {
        cout << "Queue size: " << line.size() << endl;
    }

    void peek() {
        if (!isEmpty()) {
            Person nextPerson = line.front();
            cout << "Next in line: " << nextPerson.name << "
(Ticket #" << nextPerson.ticketNum << ")\n";
```

```
        } else {
            cout << "No one is in line.\n";
        }
    }

    void position(const string& nameAndTicket) {
        queue<Person> tempQueue = line;
        int position = 1;
        while (!tempQueue.empty()) {
            Person person = tempQueue.front();
            if (person.name == nameAndTicket ||
to_string(person.ticketNum) == nameAndTicket) {
                cout << person.name << " is currently at position "
<< position << " in the queue.\n";


                return;
            }
            tempQueue.pop();
            position++;
        }
```

```
            cout << nameAndTicket << " is not in the queue.\n";
        }
};

int main() {
    Queue queue;
    int option;
    string name;

    cout << "Welcome to Olivia Rodrigo's Concert Ticketing
System!\n";

    thread dequeueThread(&Queue::autoDequeue, &queue);

    while (true) {
        cout << "\n1. Enqueue a person\n";
        cout << "2. Check your position in the queue\n";
        cout << "3. Exit\n";
        cout << "Choose an Option: ";
```

```
cin >> option;


if (option == 1){
   cout << "Enter the name: ";
   cin.ignore();
   getline(cin, name);
   Person person(name);
   queue.enqueue(person);
} else if (option == 2){
   cout << "Enter your name or ticket number: ";
   cin >> name;
   queue.position(name);
} else if (option == 3){
   cout << "Exiting...\n";
   queue.stopDequeueing();
   break;
} else {
   cout << "Invalid option! Please try again.\n";
}
```

```
    }

    dequeueThread.join();

    return 0;
}
```