Name : Prem Sah
PRN : 123B1B234
Div : D

Assignment no. 9

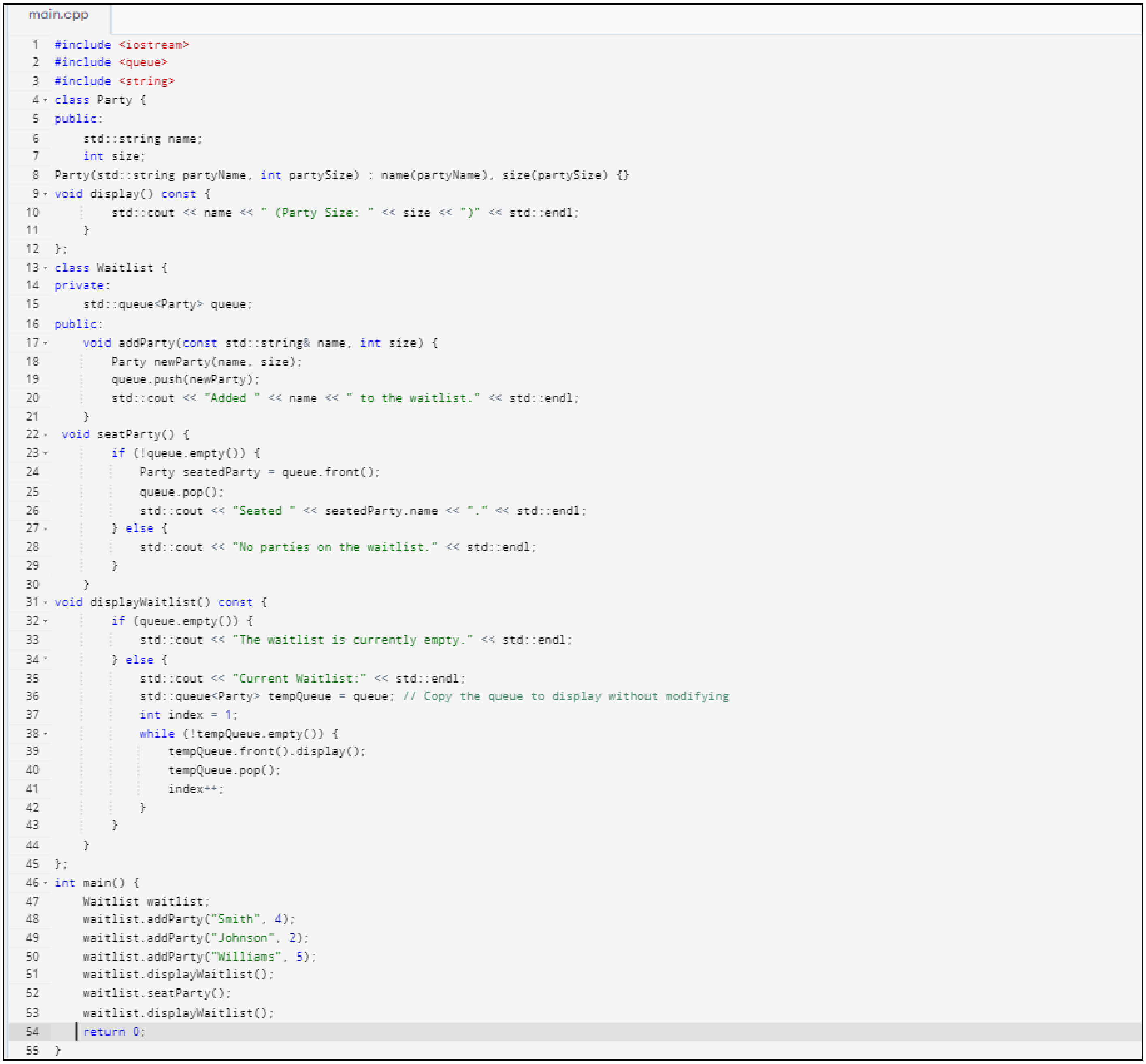
A) Implement a restaurant waitlist system using the queue data structure. Restaurant waitlist provide following facility:
a. Add Party to Waitlist
b. Seat Party
c. Display Waitlist

```
#include <iostream>
#include <queue>
#include <string>
class Party {
public:
   std::string name;
   int size;

   Party(std::string partyName, int partySize) : name(partyName), size(partySize) {}

   void display() const {
      std::cout << name << " (Party Size: " << size << ")" << std::endl;
   }
};
class Waitlist {
private:
   std::queue<Party> queue;

public:
   void addParty(const std::string& name, int size) {
      Party newParty(name, size);
      queue.push(newParty);
      std::cout << "Added " << name << " to the waitlist." << std::endl;
   }
```

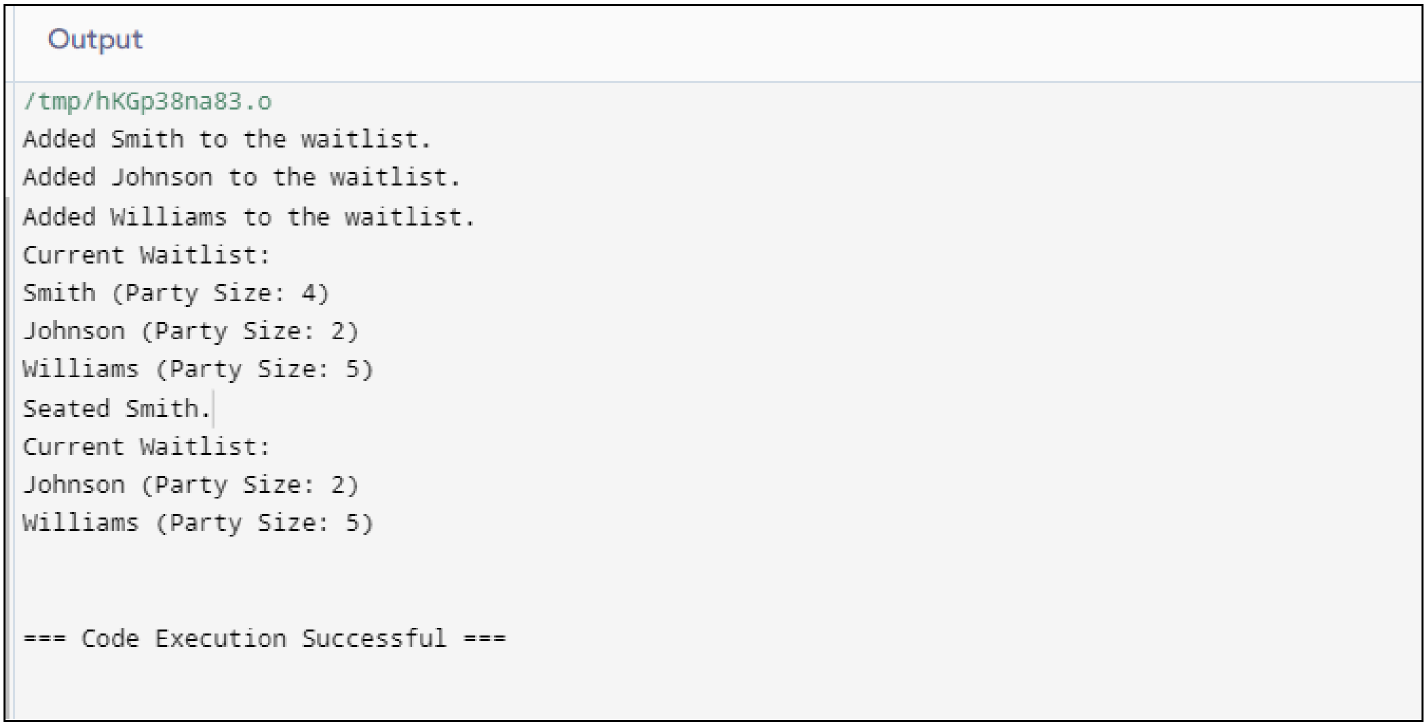
```
    void seatParty() {
        if (!queue.empty()) {
            Party seatedParty = queue.front();
            queue.pop();
            std::cout << "Seated " << seatedParty.name << "." << std::endl;
        } else {
            std::cout << "No parties on the waitlist." << std::endl;
        }
    }

    void displayWaitlist() const {
        if (queue.empty()) {
            std::cout << "The waitlist is currently empty." << std::endl;
        } else {
            std::cout << "Current Waitlist:" << std::endl;
            std::queue<Party> tempQueue = queue; // Copy the queue to display without
modifying
            int index = 1;
            while (!tempQueue.empty()) {
                tempQueue.front().display();
                tempQueue.pop();
                index++;
            }
        }
    }
};

int main() {
    Waitlist waitlist;
    waitlist.addParty("Smith", 4);
    waitlist.addParty("Johnson", 2);
    waitlist.addParty("Williams", 5);
    waitlist.displayWaitlist();
    waitlist.seatParty();
    waitlist.displayWaitlist();
    return 0;
}
```

OUTPUT :

```
Added Smith to the waitlist.
Added Johnson to the waitlist.
Added Williams to the waitlist.
Current Waitlist:
Smith (Party Size: 4)
Johnson (Party Size: 2)
Williams (Party Size: 5)
Seated Smith.
Current Waitlist:
Johnson (Party Size: 2)
Williams (Party Size: 5)
```



```cpp
#include <iostream>
#include <queue>
#include <string>
class Party {
public:
    std::string name;
    int size;
Party(std::string partyName, int partySize) : name(partyName), size(partySize) {}
void display() const {
        std::cout << name << " (Party Size: " << size << ")" << std::endl;
    }
};
class Waitlist {
private:
    std::queue<Party> queue;
public:
    void addParty(const std::string& name, int size) {
        Party newParty(name, size);
        queue.push(newParty);
        std::cout << "Added " << name << " to the waitlist." << std::endl;
    }
 void seatParty() {
        if (!queue.empty()) {
            Party seatedParty = queue.front();
            queue.pop();
            std::cout << "Seated " << seatedParty.name << "." << std::endl;
        } else {
            std::cout << "No parties on the waitlist." << std::endl;
        }
    }
void displayWaitlist() const {
        if (queue.empty()) {
            std::cout << "The waitlist is currently empty." << std::endl;
        } else {
            std::cout << "Current Waitlist:" << std::endl;
            std::queue<Party> tempQueue = queue; // Copy the queue to display without modifying
            int index = 1;
            while (!tempQueue.empty()) {
                tempQueue.front().display();
                tempQueue.pop();
                index++;
            }
        }
    }
};
int main() {
    Waitlist waitlist;
    waitlist.addParty("Smith", 4);
    waitlist.addParty("Johnson", 2);
    waitlist.addParty("Williams", 5);
    waitlist.displayWaitlist();
    waitlist.seatParty();
    waitlist.displayWaitlist();
    return 0;
}
```



```
/tmp/hKGp38na83.o
Added Smith to the waitlist.
Added Johnson to the waitlist.
Added Williams to the waitlist.
Current Waitlist:
Smith (Party Size: 4)
Johnson (Party Size: 2)
Williams (Party Size: 5)
Seated Smith.
Current Waitlist:
Johnson (Party Size: 2)
Williams (Party Size: 5)


=== Code Execution Successful ===
```