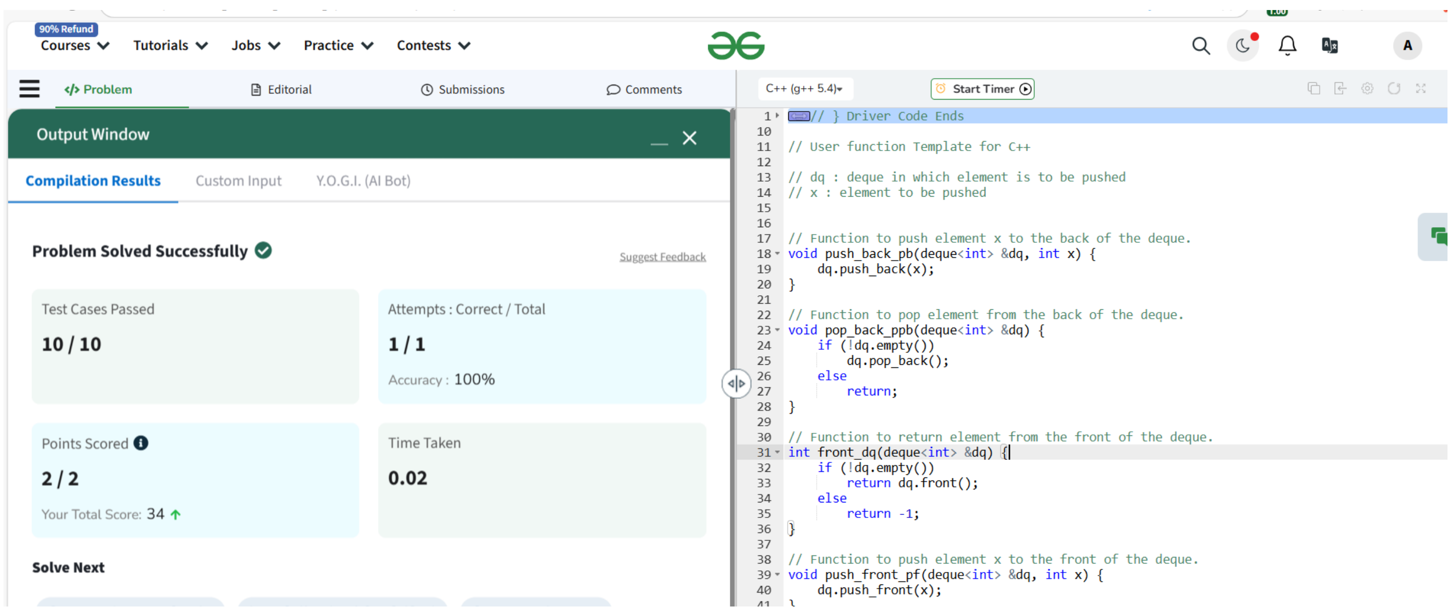
Dequeue using stack

```
// Function to push element x to the back of the deque.
void push_back_pb(deque<int> &dq, int x) {
    dq.push_back(x);
}

// Function to pop element from the back of the deque.
void pop_back_ppb(deque<int> &dq) {
    if (!dq.empty())
        dq.pop_back();
    else
        return;
}

// Function to return element from the front of the deque.
int front_dq(deque<int> &dq) {
    if (!dq.empty())
        return dq.front();
    else
        return -1;
}

// Function to push element x to the front of the deque.
void push_front_pf(deque<int> &dq, int x) {
    dq.push_front(x);
}
```



Problem Solved Successfully

Test Cases Passed
10 / 10

Attempts : Correct / Total
1 / 1
Accuracy : 100%

Points Scored
2 / 2
Your Total Score: 34

Time Taken
0.02

Solve Next

```
// } Driver Code Ends

// User function Template for C++

// dq : deque in which element is to be pushed
// x : element to be pushed


// Function to push element x to the back of the deque.
void push_back_pb(deque<int> &dq, int x) {
    dq.push_back(x);
}

// Function to pop element from the back of the deque.
void pop_back_ppb(deque<int> &dq) {
    if (!dq.empty())
        dq.pop_back();
    else
        return;
}

// Function to return element from the front of the deque.
int front_dq(deque<int> &dq) {
    if (!dq.empty())
        return dq.front();
    else
        return -1;
}

// Function to push element x to the front of the deque.
void push_front_pf(deque<int> &dq, int x) {
    dq.push_front(x);
}
```