

Implement dequeue using queue

// 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 back of the deque.

```
void pop_back_ppb(deque<int> &dq) {  
    if (!dq.empty())  
        dq.pop_back();  
}
```

// Function to return element from 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);  
}
```

The screenshot displays a C++ IDE interface. On the left, the 'Output Window' shows 'Compilation Results' for a problem solved successfully. It indicates that 10 out of 10 test cases passed, with 2 out of 2 attempts correct and 100% accuracy. The time taken was 0.01 seconds. A message at the bottom states: 'You get marks only for the first correct submission if you solve the problem without viewing the full solution.' Below this is a 'Solve Next' button.

The main editor area shows the C++ code for the deque implementation. The code includes the following functions:

- `push_back_pb`: Pushes element `x` to the back of the deque.
- `pop_back_ppb`: Pops element from the back of the deque.
- `front_dq`: Returns the element from the front of the deque, or `-1` if empty.
- `push_front_pf`: Pushes element `x` to the front of the deque.

```
1 // } Driver Code Ends  
2  
3 // User function Template for C++  
4 // dq : deque in which element is to be pushed  
5 // x : element to be pushed  
6  
7  
8 // Function to push element x to the back of the deque.  
9 void push_back_pb(deque<int> &dq, int x) {  
10     dq.push_back(x);  
11 }  
12  
13 // Function to pop element from back of the deque.  
14 void pop_back_ppb(deque<int> &dq) {  
15     if (!dq.empty())  
16         dq.pop_back();  
17 }  
18  
19 // Function to return element from front of the deque.  
20 int front_dq(deque<int> &dq) {  
21     if (!dq.empty())  
22         return dq.front();  
23     else  
24         return -1;  
25 }  
26  
27 // Function to push element x to the front of the deque.  
28 void push_front_pf(deque<int> &dq, int x) {  
29     dq.push_front(x);  
30 }  
31  
32 // } Driver Code Ends
```