Queue Implementation using Array

Introduction:

This document explains the logic behind implementing a queue using a static array in C++. The queue follows the FIFO (First In, First Out) principle. The implementation provides essential queue operations like push, pop, front, size, and display.

Queue Class Implementation:

```
#include<iostream>
using namespace std;

class Queue{
  public:
  int f; // Front pointer
  int b; // Back pointer
  int arr[5]; // Static array of size 5

  Queue(){
    f = 0;
    b = 0;
  }

  // Push operation
  void push(int val){
    if(b == 5){
      cout << "Queue is full!" << endl;
      return;
    }
    else{
      arr[b] = val;
      b++;
    }
  }

  // Pop operation
  void pop(){
    if(b - f == 0){
      cout << "Queue is empty!" << endl;
      return;
    }
    else f++;
  }

  // Front operation
  void front(){
    if(b - f == 0){
      cout << "Queue is empty!" << endl;
      return;
    }
    else {
      cout << arr[f] << endl;
    }
  }

  // Size operation
  void size(){
    cout << b - f << endl;
  }

  // Display function
  void display(){
    if(b - f == 0){
      cout << "Queue is empty!" << endl;
      return;
    }
    for(int i = f; i < b; i++){
      cout << arr[i] << " ";
    }
    cout << endl;
  }
};

// Main function to test the queue operations
int main(){
  Queue q;
  q.push(1);
  q.push(2);
  q.push(3);
  q.push(4);
  q.push(5);
  q.display(); // Expected output: 1 2 3 4 5

  q.pop();
  q.display(); // Expected output: 2 3 4 5

  q.front();  // Expected output: 2
  q.size();   // Expected output: 4

  return 0;
}
```

Explanation of Each Function:

1. Constructor initializes front (f) and back (b) pointers to 0 and defines a static array of size 5.
2. push(int val): Adds an element at position b. Increments b after insertion.
3. pop(): Removes the front element by increasing f.
4. front(): Displays the first element.
5. size(): Returns the current number of elements (b - f).
6. display(): Prints all elements from f to b - 1.

Conclusion:

- The queue follows FIFO order correctly.
- The static array has a fixed size of 5.
- The front pointer moves forward when an element is removed.
- For better efficiency, a circular queue should be used.

Future Improvements:

- Implementing a circular queue to optimize space.
- Using dynamic memory allocation for queue expansion.
- Using STL (std::queue) for flexibility.