HW3 : Dry

1- We can use some queue as non const queue and some queue as const queue. The queue is composed of Nodes. If a queue is const as the nodes in the queue also const. An iterator is also a pointer to one of the node of the queue, so a const iterator can't change the value that they point to. Because of this we need a non const iterator who can do that if we want/need.

2- We supposed the type of the template in the constructor, in the copy constructor, in the destructor, in the assignment operator in the pushback and in all of functions that called for pushback.
Because we are creating a new node with m_data in the node we are supposing that T he have a constructor and a destructor and every function and fields that allowed us to addition him, subtract him or compare it with something else.
For example in filter we supposed that we can use iterators on T.
And in transform we supposed that the type of T is matching with the function.

3- The error is in the linking step because the compiler needs to have all the information of the template in the same file. The linking can only be achieved if the template arguments are known. Imagine a scenario where you have a template function declared in Queue.h and defined in a Queue.cpp and used in main.cpp. When Queue.cpp is compiled, it is not necessarily known that the upcoming compilation main.cpp will require an instance of the template. We can put template in .cpp only if we used it the same file.

4- If the number is only known in the running step. We have to declare a new template, because the template will take the value of int S only in the running time and will do the correct operation . Like this :

```
template<int S>
        bool isEven(int n)
        {
                return (n % S) == 0;
        }
```

Itay Elkeslassy
Ruben Timsit