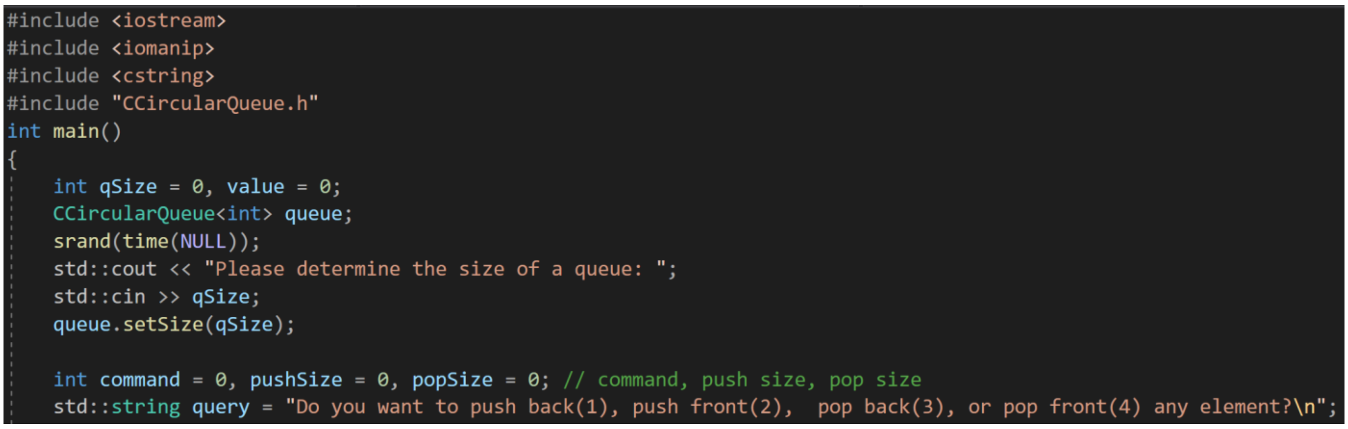
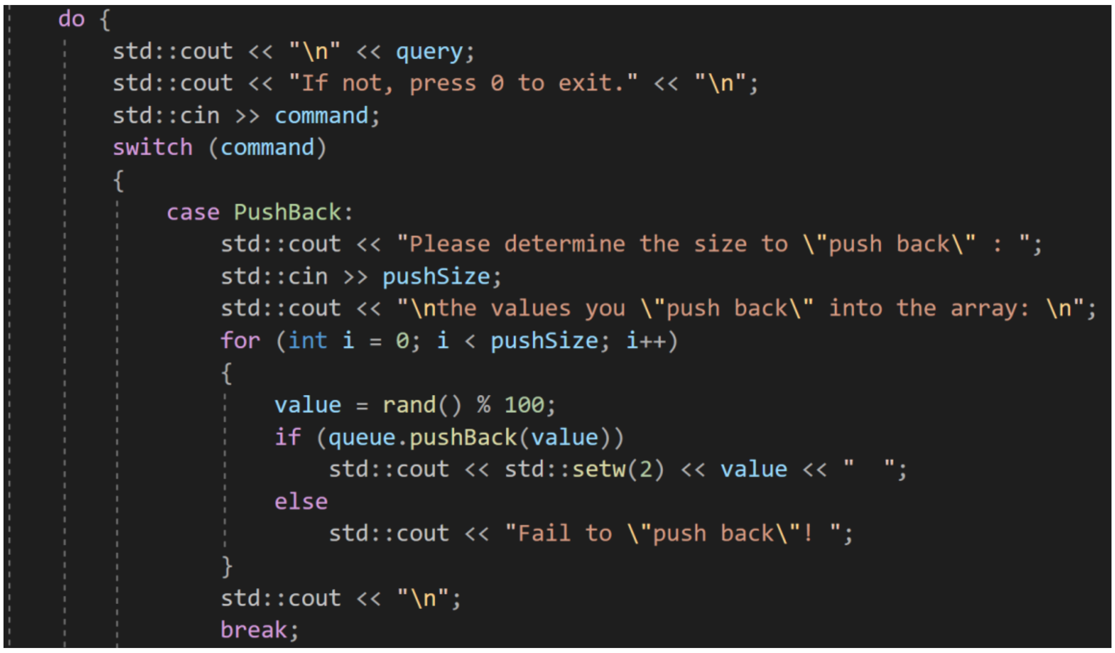
HW3: 頭尾皆能存入與取出元素

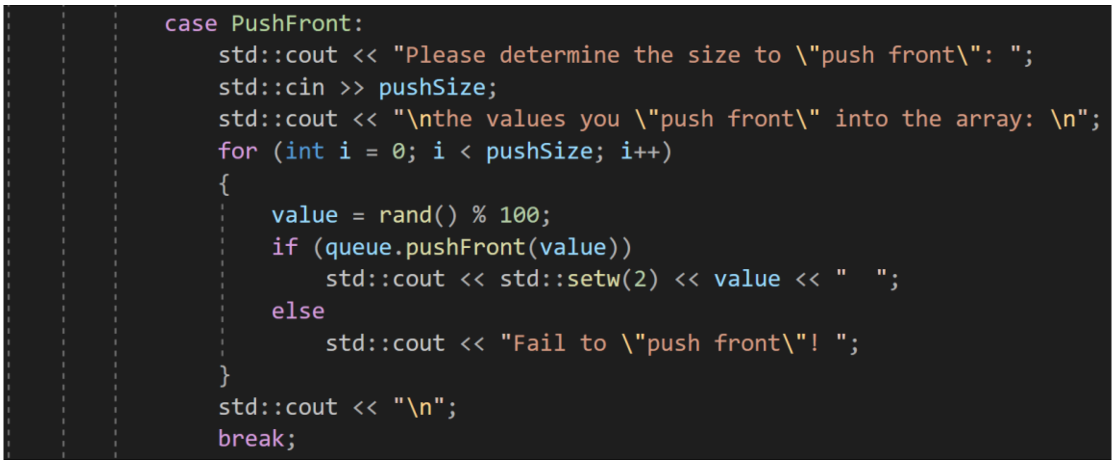
HW3CPP.cpp



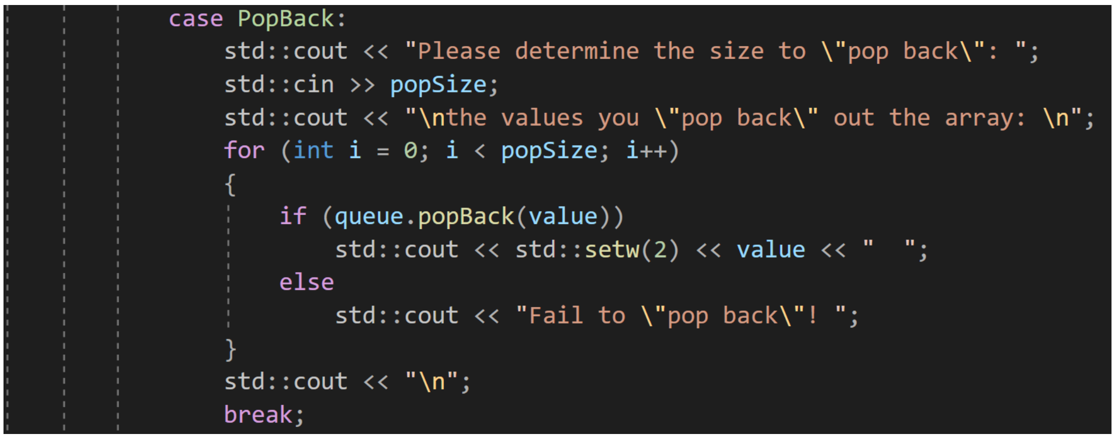
```cpp
#include <iostream>
#include <iomanip>
#include <cstring>
#include "CCircularQueue.h"
int main()
{
    int qSize = 0, value = 0;
    CCircularQueue<int> queue;
    srand(time(NULL));
    std::cout << "Please determine the size of a queue: ";
    std::cin >> qSize;
    queue.setSize(qSize);

    int command = 0, pushSize = 0, popSize = 0; // command, push size, pop size
    std::string query = "Do you want to push back(1), push front(2),  pop back(3), or pop front(4) any element?\n";
```



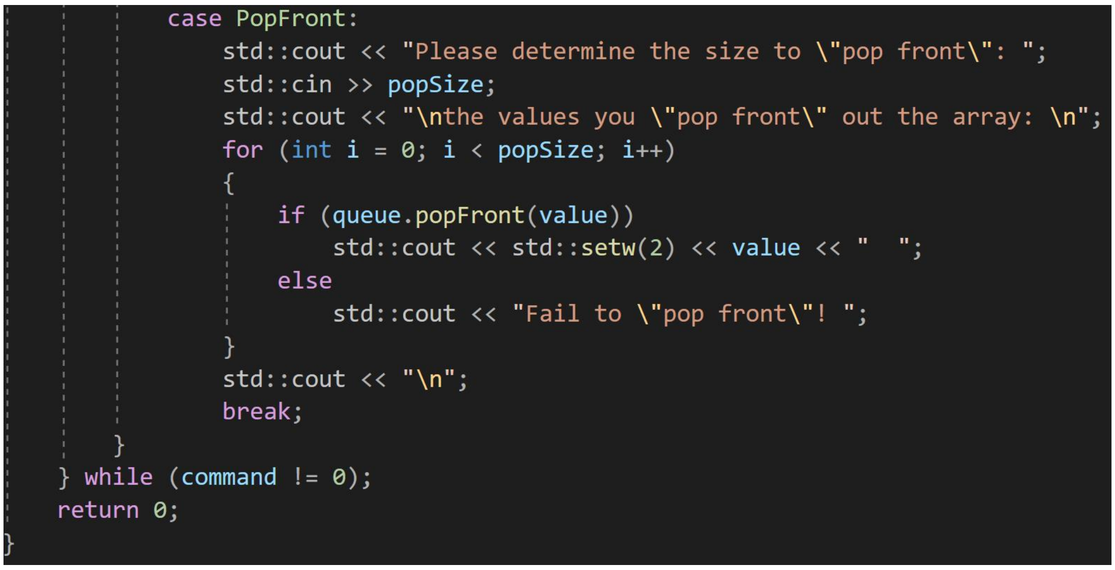
```cpp
    do {
        std::cout << "\n" << query;
        std::cout << "If not, press 0 to exit." << "\n";
        std::cin >> command;
        switch (command)
        {
            case PushBack:
                std::cout << "Please determine the size to \"push back\" : ";
                std::cin >> pushSize;
                std::cout << "\nthe values you \"push back\" into the array: \n";
                for (int i = 0; i < pushSize; i++)
                {
                    value = rand() % 100;
                    if (queue.pushBack(value))
                        std::cout << std::setw(2) << value << "  ";
                    else
                        std::cout << "Fail to \"push back\"! ";
                }
                std::cout << "\n";
                break;
```



```cpp
            case PushFront:
                std::cout << "Please determine the size to \"push front\": ";
                std::cin >> pushSize;
                std::cout << "\nthe values you \"push front\" into the array: \n";
                for (int i = 0; i < pushSize; i++)
                {
                    value = rand() % 100;
                    if (queue.pushFront(value))
                        std::cout << std::setw(2) << value << "  ";
                    else
                        std::cout << "Fail to \"push front\"! ";
                }
                std::cout << "\n";
                break;
```

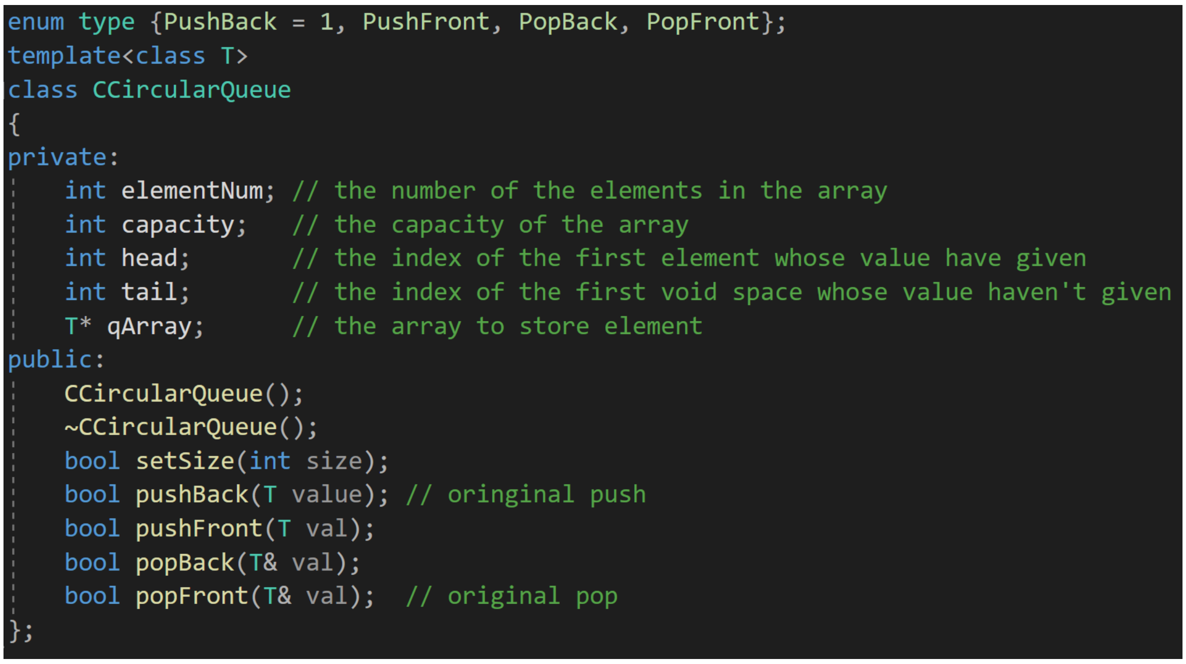


```
            case PopBack:
                std::cout << "Please determine the size to \"pop back\": ";
                std::cin >> popSize;
                std::cout << "\nthe values you \"pop back\" out the array: \n";
                for (int i = 0; i < popSize; i++)
                {
                    if (queue.popBack(value))
                        std::cout << std::setw(2) << value << "  ";
                    else
                        std::cout << "Fail to \"pop back\"! ";
                }
                std::cout << "\n";
                break;
```

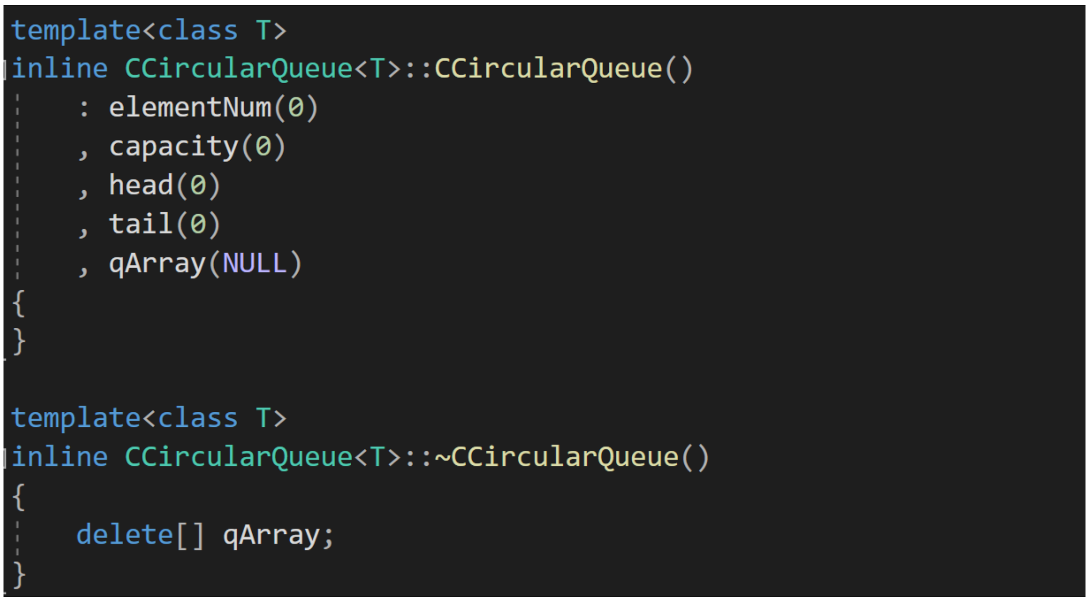


```
            case PopFront:
                std::cout << "Please determine the size to \"pop front\": ";
                std::cin >> popSize;
                std::cout << "\nthe values you \"pop front\" out the array: \n";
                for (int i = 0; i < popSize; i++)
                {
                    if (queue.popFront(value))
                        std::cout << std::setw(2) << value << "  ";
                    else
                        std::cout << "Fail to \"pop front\"! ";
                }
                std::cout << "\n";
                break;
        }
    } while (command != 0);
    return 0;
}
```

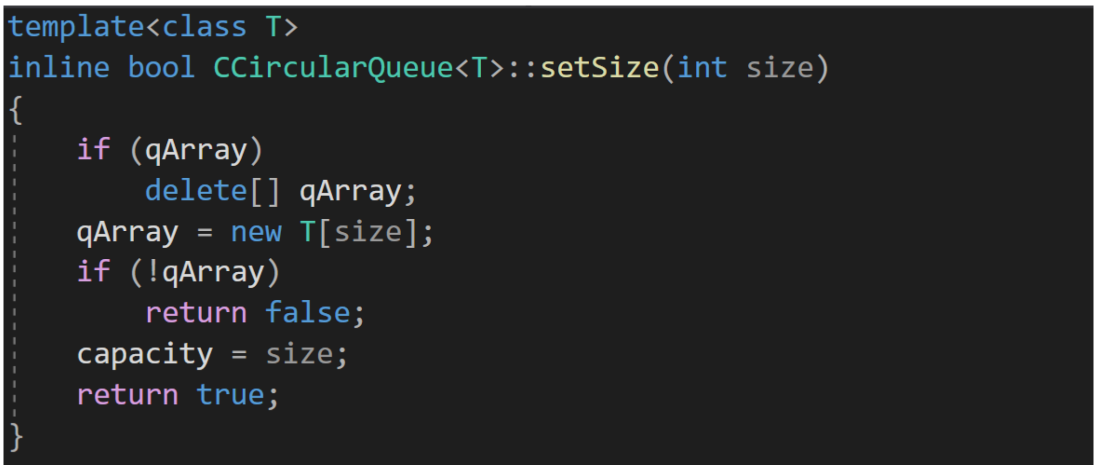
CCircularQueue.h



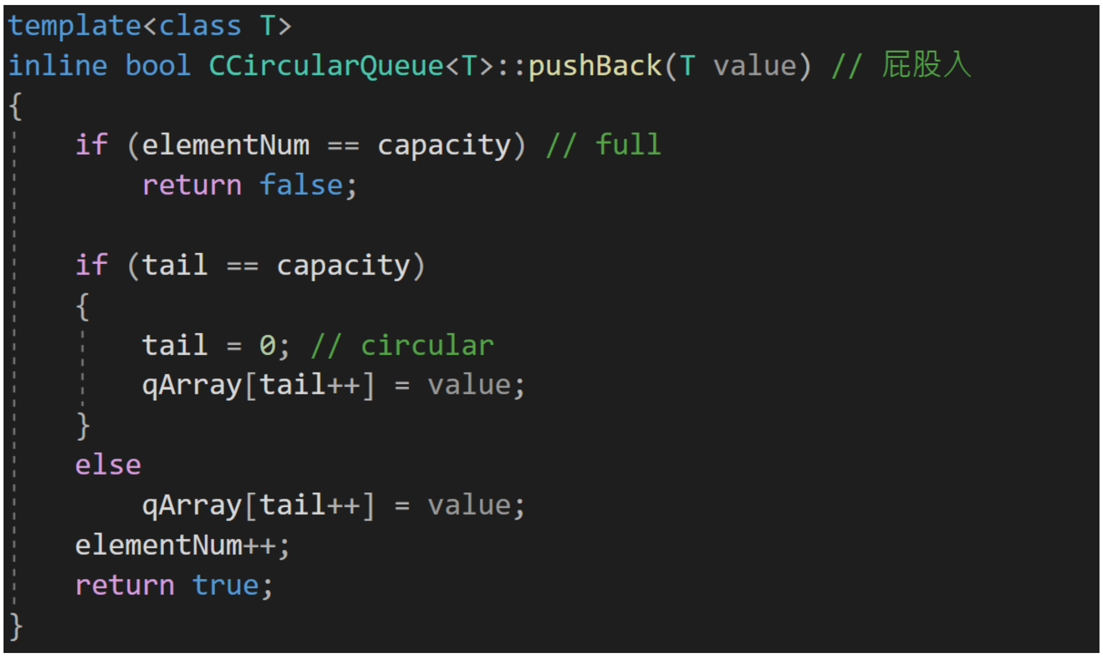
```cpp
enum type {PushBack = 1, PushFront, PopBack, PopFront};
template<class T>
class CCircularQueue
{
private:
    int elementNum; // the number of the elements in the array
    int capacity;   // the capacity of the array
    int head;       // the index of the first element whose value have given
    int tail;       // the index of the first void space whose value haven't given
    T* qArray;      // the array to store element
public:
    CCircularQueue();
    ~CCircularQueue();
    bool setSize(int size);
    bool pushBack(T value); // oringinal push
    bool pushFront(T val);
    bool popBack(T& val);
    bool popFront(T& val);  // original pop
};
```



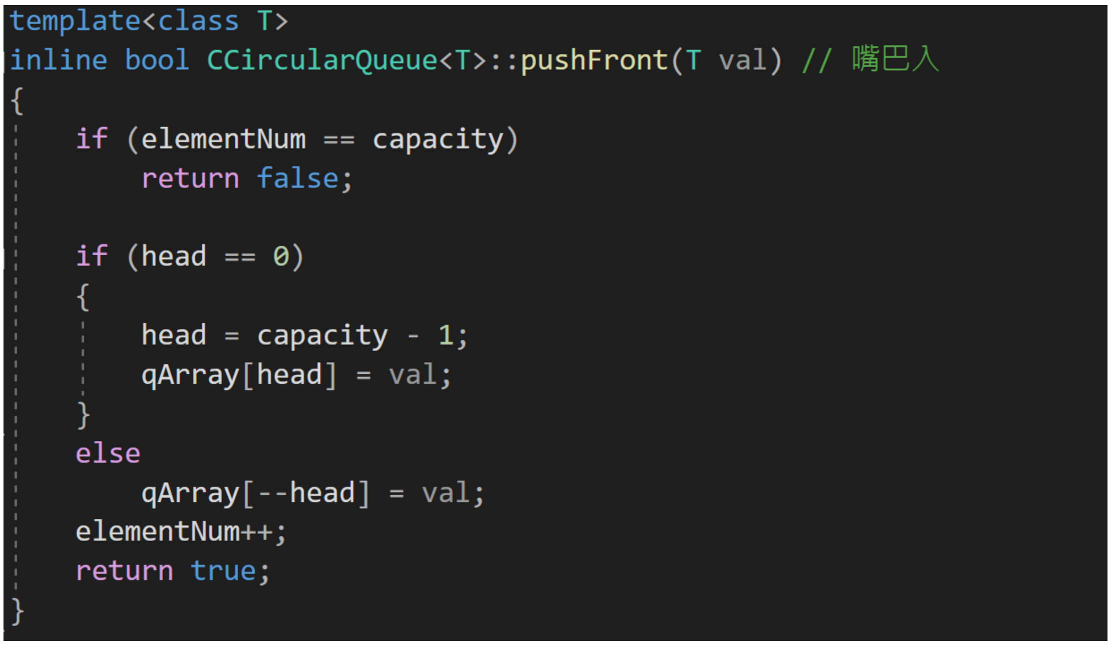
```cpp
template<class T>
inline CCircularQueue<T>::CCircularQueue()
    : elementNum(0)
    , capacity(0)
    , head(0)
    , tail(0)
    , qArray(NULL)
{
}

template<class T>
inline CCircularQueue<T>::~CCircularQueue()
{
    delete[] qArray;
}
```



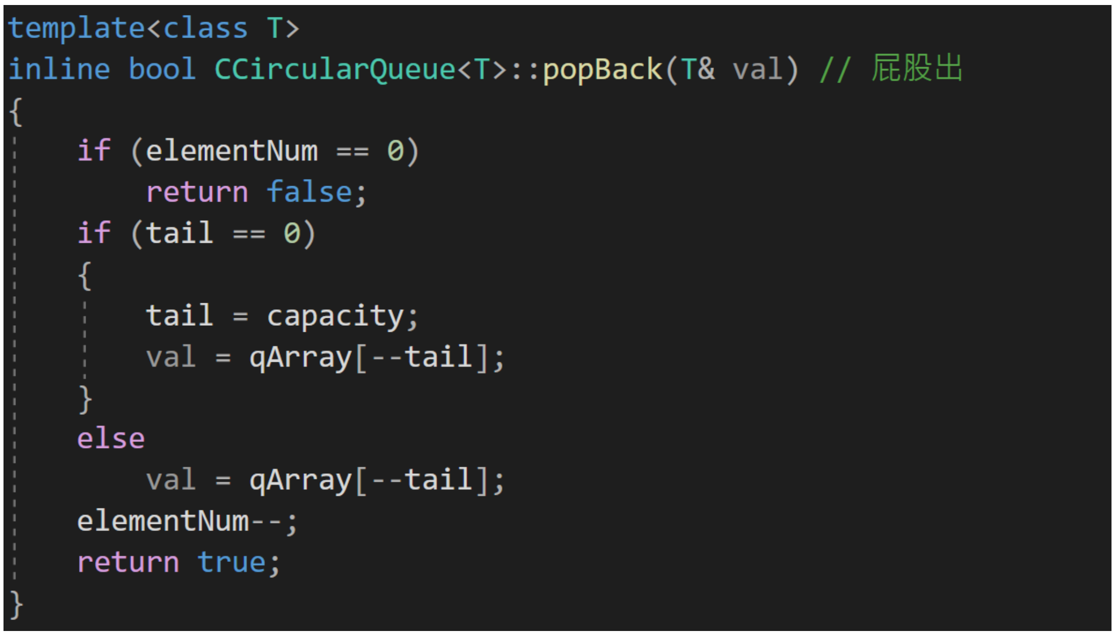
```cpp
template<class T>
inline bool CCircularQueue<T>::setSize(int size)
{
    if (qArray)
        delete[] qArray;
    qArray = new T[size];
    if (!qArray)
        return false;
    capacity = size;
    return true;
}
```



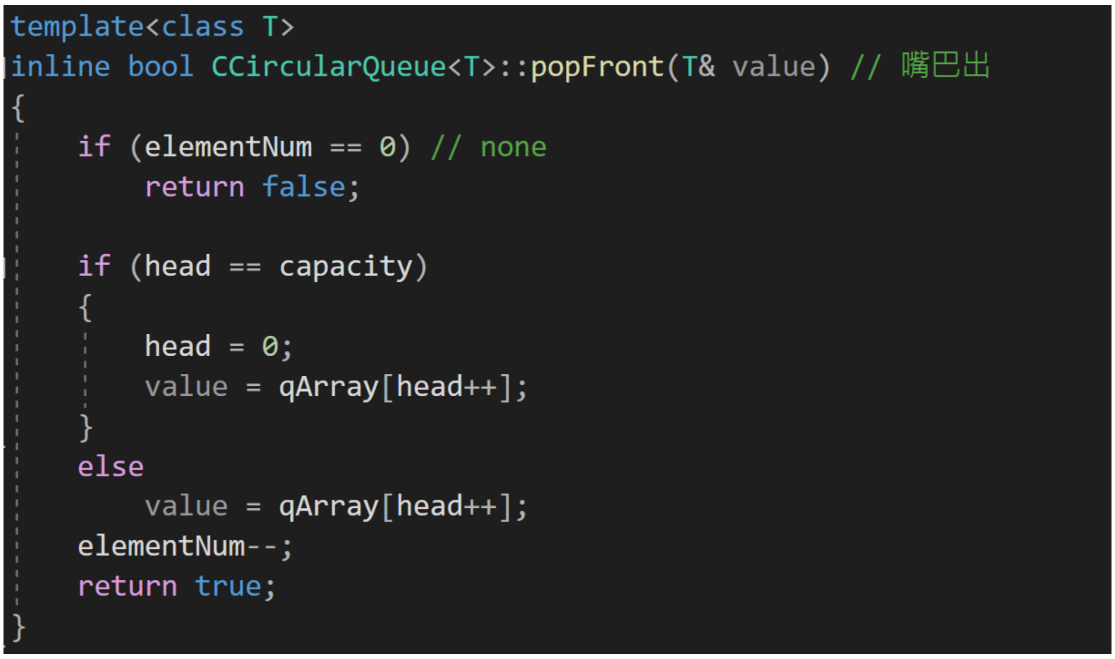
```cpp
template<class T>
inline bool CCircularQueue<T>::pushBack(T value) // 屁股入
{
    if (elementNum == capacity) // full
        return false;

    if (tail == capacity)
    {
        tail = 0; // circular
        qArray[tail++] = value;
    }
    else
        qArray[tail++] = value;
    elementNum++;
    return true;
}
```



```cpp
template<class T>
inline bool CCircularQueue<T>::pushFront(T val) // 嘴巴入
{
    if (elementNum == capacity)
        return false;

    if (head == 0)
    {
        head = capacity - 1;
        qArray[head] = val;
    }
    else
        qArray[--head] = val;
    elementNum++;
    return true;
}
```

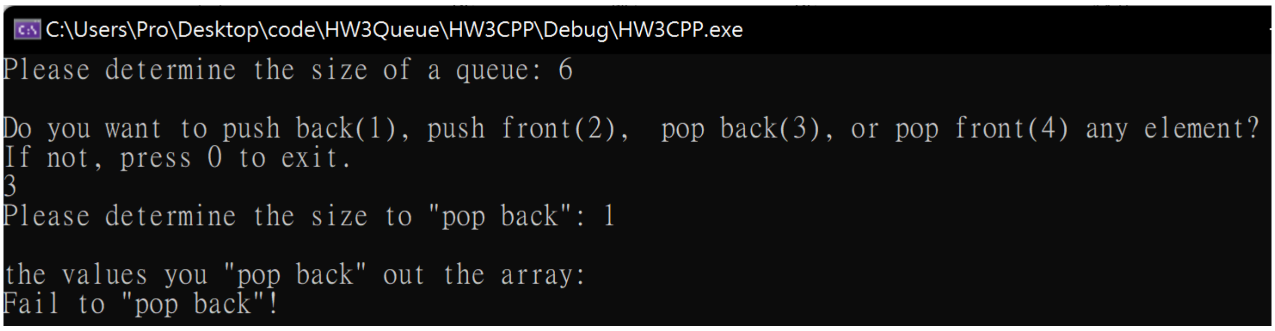


```cpp
template<class T>
inline bool CCircularQueue<T>::popBack(T& val) // 屁股出
{
    if (elementNum == 0)
        return false;
    if (tail == 0)
    {
        tail = capacity;
        val = qArray[--tail];
    }
    else
        val = qArray[--tail];
    elementNum--;
    return true;
}
```



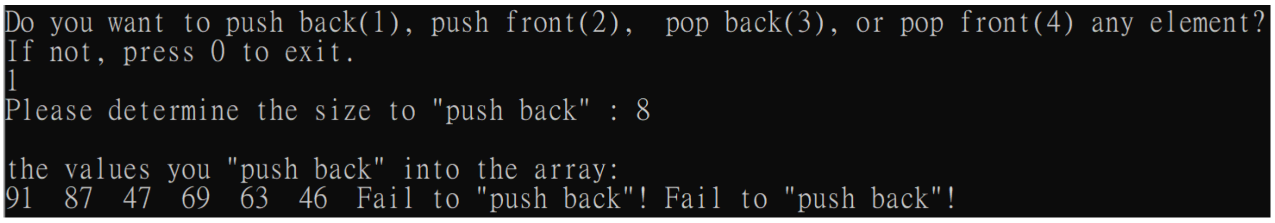
```cpp
template<class T>
inline bool CCircularQueue<T>::popFront(T& value) // 嘴巴出
{
    if (elementNum == 0) // none
        return false;

    if (head == capacity)
    {
        head = 0;
        value = qArray[head++];
    }
    else
        value = qArray[head++];
    elementNum--;
    return true;
}
```

1. give the initial size of the queue, try to pop back, and show the error message because there is no value in the queue initially



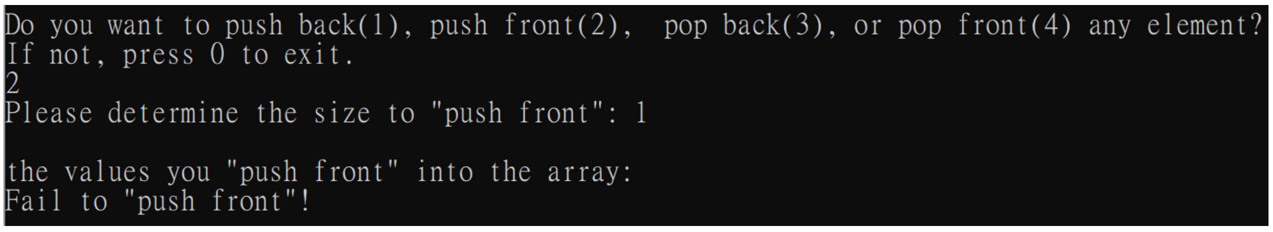
```
C:\Users\Pro\Desktop\code\HW3Queue\HW3CPP\Debug\HW3CPP.exe
Please determine the size of a queue: 6

Do you want to push back(1), push front(2),  pop back(3), or pop front(4) any element?
If not, press 0 to exit.
3
Please determine the size to "pop back": 1

the values you "pop back" out the array:
Fail to "pop back"!
```

2. determine the size to push back, show the values user push back, and the error message because the queue is full so the redundant elements won't be stored into the queue



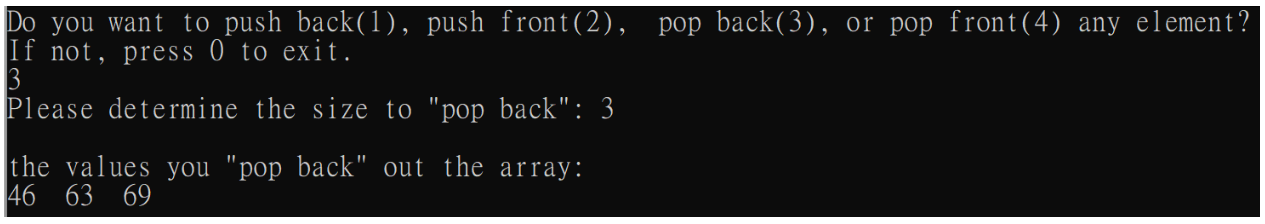
```
Do you want to push back(1), push front(2),  pop back(3), or pop front(4) any element?
If not, press 0 to exit.
1
Please determine the size to "push back" : 8

the values you "push back" into the array:
91  87  47  69  63  46  Fail to "push back"! Fail to "push back"!
```

3. determine the size to push front and fail to push front because the queue is full



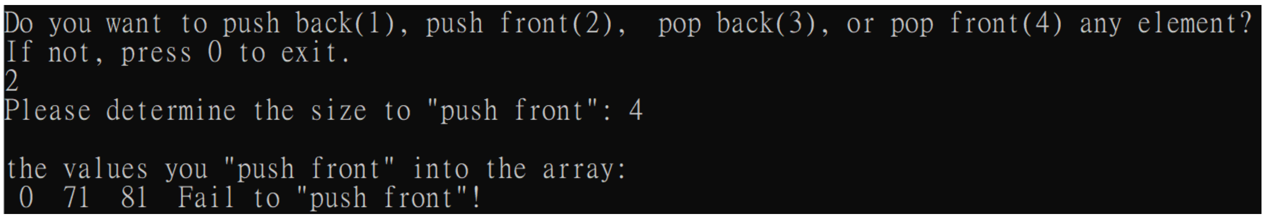
```
Do you want to push back(1), push front(2),  pop back(3), or pop front(4) any element?
If not, press 0 to exit.
2
Please determine the size to "push front": 1

the values you "push front" into the array:
Fail to "push front"!
```

4. determine the size to pop back and show the result



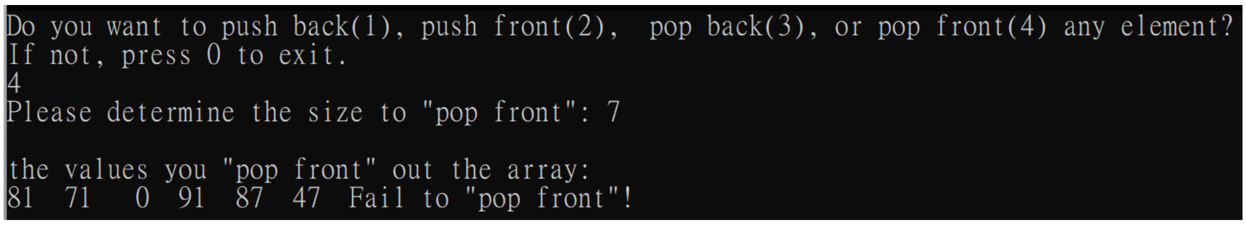
```
Do you want to push back(1), push front(2),  pop back(3), or pop front(4) any element?
If not, press 0 to exit.
3
Please determine the size to "pop back": 3

the values you "pop back" out the array:
46  63  69
```

5. determine the size to push front, show the result, and error message because the number of element user intend to push front is one more than the max capacity of the queue



```
Do you want  to push back(1), push front(2),  pop back(3), or pop front(4) any element?
If not, press 0 to exit.
2
Please determine the size to "push front": 4

the values you "push front" into the array:
 0  71  81  Fail to "push front"!
```

6. determine the size to pop front and show the result, and the error messages because the number of element user intend to pop front is one more than that in the queue



```
Do you want  to push back(1), push front(2),  pop back(3), or pop front(4) any element?
If not, press 0 to exit.
4
Please determine the size to "pop front": 7

the values you "pop front" out the array:
81  71   0  91  87  47  Fail to "pop front"!
```