数据结构课程设计

项目说明文档

银行业务

软件工程

张靖凯

2151396

目录

项目内容

设某银行有 A，B 两个业务窗口，且处理业务的速度不一样，其中 A 窗口处理速度是 B 窗口的 2 倍----即当 A 窗口每处理完 2 个顾客是，B 窗口处理完 1 个顾客。给定到达银行的顾客序列，请按照业务完成的顺序输出顾客序列。假定不考虑顾客信后到达的时间间隔，并且当不同窗口同时处理完 2 个顾客时，A 窗口的顾客优先输出。

项目功能要求

1 输入说明：输入为一行正整数，其中第一数字 N（N<=1000）为顾客总数，后面跟着 N 位顾客的编号。编号为奇数的顾客需要到 A 窗口办理业务，为偶数的顾客则去 B 窗口。数字间以空格分隔。

2 输出说明：按照业务处理完成的顺序输出顾客的编号。数字键以空格分隔，但是最后一个编号不能有多余的空格。

3 测试用例：

序号	输入	输出	说明
1	8 2 1 3 9 4 11 13 15	1 3 2 9 11 4 13 15	正常测试，A 窗口人多
2	8 2 1 3 9 4 11 12 16	1 3 2 9 11 4 12 16	正常测试，B 窗口人多
3	1 6	6	最小 N

项目设计

数据结构类设计

用顺序储存结构，封装动态数组来实现队列，其中配有简单迭代器，即 int 型 index。默认初始容量为 10。实现了 push 和 pop 函数，即入队出队操作；实现了 front，back 操作；size 返回容器中元素的个数；extend 函数对 capacity 进行扩容操作

Private Attributes

value_type *	**_data** 元素顺序容器 More...
int	**begin** 头指针（同迭代器） More...
int	**_end** 尾指针 More...
size_t	**_capacity** 容器的容量 （size的求法由函数实现，不再单独设置size数据） More...

Public Types

using	**value_type** = T

Public Member Functions

	Queue ()
	~Queue ()
void	**push** (const **value_type** &e)
void	**pop** ()
value_type &	**front** () const
value_type &	**back** () const
size_t	**size** () const
bool	**empty** () const
void	**extend** () 当size超过_capacity时翻倍，模拟STL More...

数据结构关键代码

扩容操作：

```
template<typename T>
    void Queue<T>::extend()
    {
        size_t sz = size();
        if (sz == _capacity - 1)
        {
            value_type* tmp = new value_type[_capacity << 1];

            for (int i = begin, j = 0; i != _end; i++, j++)
                tmp[j] = _data[i];
            begin = 0;
            _end = sz;
            delete[]_data;
            _data = tmp;
            _capacity <<= 1;
        }
    }
```

解题思路

设置两个队列，一个是奇数一个是偶数，通过输入分别将奇数和偶数入队。总共循环两次，循环结构为并列，第一次是在 q1 队列不为空的情况下每次循环弹出 2 个奇数，弹出一个偶数。当 q1 为空后，有两种情况：

1. 奇数全出，偶数有剩余。
2. 奇数全出，偶数没剩余。

从而看偶数队列是否为空，依次出队。

解题关键代码

```
int i = 0;    ///< i 的目的是为了防止最后一个数据输出空格
while (!q1.empty())
    {
        int cnt = 2;
        // 弹出两个奇数
        while (cnt-- && !q1.empty())
        {
            if (i++)
                cout << " ";
            cout << q1.front();
            q1.pop();
        }
        // 弹出一个偶数
        if (!q2.empty())
        {
            cout << " " << q2.front();
            q2.pop();
        }
    }

    /// 剩余的偶数出队
    while (!q2.empty())
    {
        if (i++)
            cout << " ";
        cout << q2.front();
        q2.pop();
    }
```

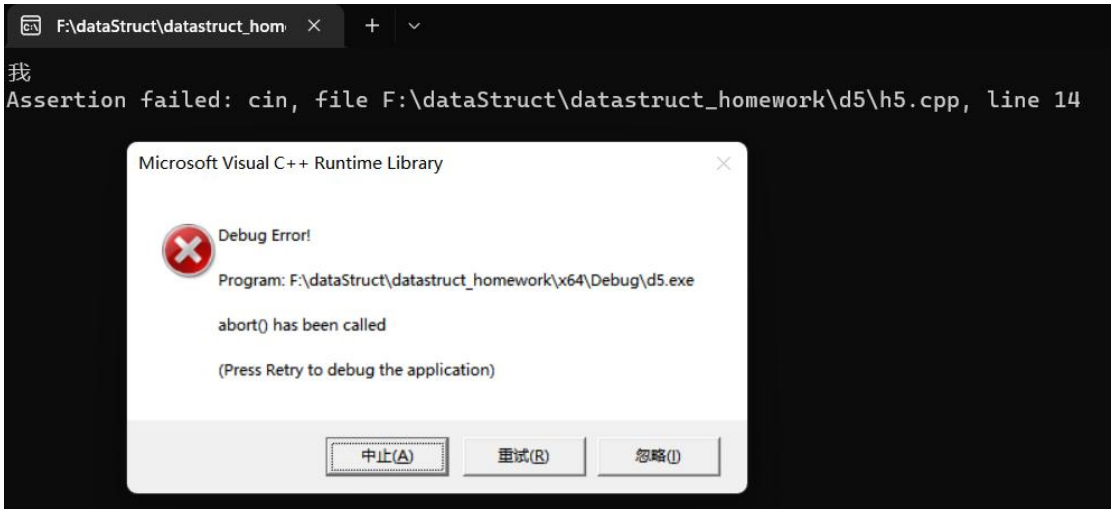
设计亮点

代码注释规范

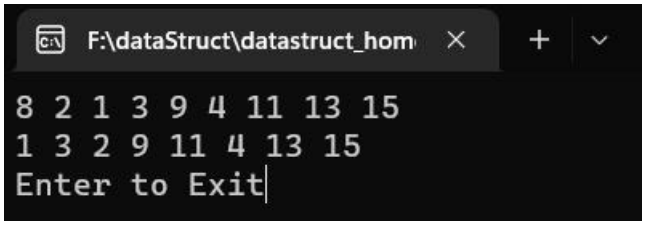
采用了 doxygen 注释规范，对类、函数等有简要说明，命名采用驼峰命名法，类内的各个声明规范，方便本人回顾之前写过的代码和 code reviewer 查看，使 API 规范，帮助这个开发流程高效、规范地进行。

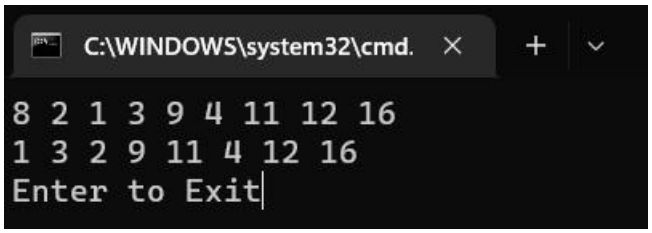
```cpp
/// @brief 简单队列
/// @detail 五类构造函数只实现第一个 其他暂未实现
template <typename T>
class Queue
{
public:
    using value_type = T;
    static const int _default_size = 10;
public:
    Queue() :begin(0), _end(0), _capacity(_default_size) { _data = new value_type[_default_size]; };
    ~Queue() { delete[]_data; };
private:
    value_type* _data;  ///< 元素顺序容器
    int begin;          ///< 头指针（同迭代器）
    int _end;           ///< 尾指针
    size_t _capacity;   ///< 容器的容量    （size的求法由函数实现，不再单独设置size数据）
```

输入错误断言处理



代码测试





Linux 下测试（Ubuntu）