作者：董佳昕

一、问题描述

实现一个模板类**Vector**，包含以下成员变量及成员函数

```
template <class T>
class Vector {
public:
  Vector();                       // creates an empty vector
  Vector(int size);               // creates a vector for holding 'size' elements
  Vector(const Vector& r);        // the copy ctor
  ~Vector();                      // destructs the vector
  T& operator[](int index);       // accesses the specified element without bounds checking
  T& at(int index);               // accesses the specified element, throws an exception of type 'std::out_of_range' when index <0 or >=m_nSize
  int size() const;          // return the size of the container
  void push_back(const T& x);     // adds an element to the end
  void clear();                   // clears the contents
  bool empty() const;             // checks whether the container is empty
private:
  void inflate();                 // expand the storage of the container to a new capacity, e.g. 2*m_nCapacity
  T *m_pElements;                 // pointer to the dynamically allocated storage
  int m_nSize;                    // the number of elements in the container
  int m_nCapacity;                // the number of elements that can be held in currently allocated storage
};

```

二、实现思路

2.1 成员变量

```
//容器中的元素个数
int m_nSize;
//容器的最大容量
int m_nCapacity;
//指向动态开辟的内存
T* m_pElements;
```

2.2 倍增内存操作

```
//当容器的容量不够时,倍增容器容量
void inflate() {
    cout << "inflate, size = " << yellow << m_nCapacity << blue << endl;
    T* now = new T[m_nCapacity * 2]; //now指向新开辟的内存
    T* plast = m_pElements;          //plast指向原来的内存
    T* pnow = now;                   //pnow 指向新开辟的内存

    int capacity = m_nCapacity;
    while (capacity > 0) {//将原来的元素拷贝到新的内存中
        *pnow = *plast;
        plast++, pnow++;
        capacity--;
    }

    delete m_pElements;
    m_pElements = now;
    m_nCapacity *= 2;
}
```

2.3 构造函数及析构函数

```
//创建一个空的vector
Vector() {
    m_pElements = new T;
    m_nCapacity = 1;
    m_nSize = 0;
    cout << "vector()" << endl;
}
//创建一个初始容量为size的vector
Vector(int size) {
    m_pElements = new T[size];
    m_nCapacity = size;
    m_nSize = 0;
}
//创建一个vector,与r的所有参数相同
Vector(const Vector& r) {
    m_nSize = r.m_nSize;
    m_nCapacity = r.m_nCapacity;

    T* now = new T[m_nCapacity]; //now指向自己开辟的内存
    T* plast = r.m_pElements;    //plast指向r的内存
    T* pnow = now;               //pnow指向自己开辟的内存

    int capacity = m_nCapacity;
    while (capacity > 0) {//将r中的元素拷贝到自己的内存中
        *pnow = *plast;
        plast++, pnow++;
        capacity--;
    }
    m_pElements = now;
}
//析构函数
~Vector() {
    delete m_pElements;
}
```

2.4 获取第index个元素

```cpp
//获取第index个元素,不包含边界检查
T& operator[](int index) {
    return m_pElements[index];
}
//获取第index个元素,如果index<0 || index>=m_nSize,输出异常'std::out_of_range'
T& at(int index) {
    if (index >= m_nSize || index < 0) {
        throw std::out_of_range("index out of range");
    }
    else return m_pElements[index];
}
```

2.5 其它功能

```cpp
//返回容器的大小
int size() const {
    return m_nSize;
}
//向容器的末尾添加一个元素
void push_back(const T& x) {
    if (m_nSize >= m_nCapacity) inflate();
    m_pElements[m_nSize] = x;
    m_nSize++;
}
//清空容器中的所有元素
void clear() {
    m_nSize = 0;
}
//判断容器是否为空
bool empty() const {
    return m_nSize == 0;
}
```

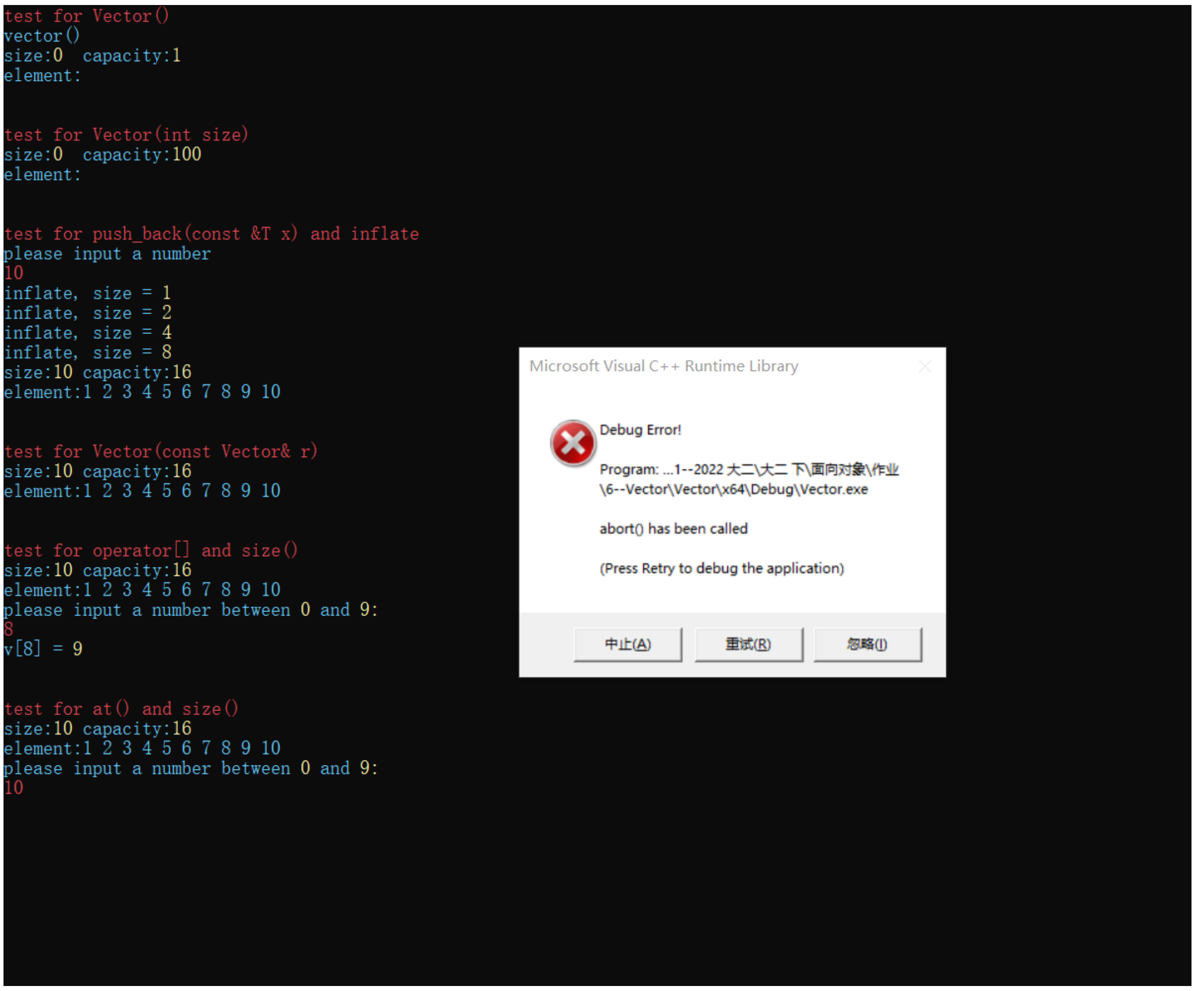
三、测试样例

在main函数中，依次检查每一个成员函数的功能，测试代码如下

```cpp
int main() {
    //检测:Vector()
    cout << red << "test for Vector()\n" << blue;
    Vector<int>v;
    v.Debug();
    cout << blue << "\n\n";

    //检测:Vector(int size)
    cout << red << "test for Vector(int size)\n"<< blue;
    Vector<int>v1(100);
    v1.Debug();
    cout << blue << "\n\n";

    //检测:push_back(),inflate
    cout << red << "test for push_back(const &T x) and inflate\n"<< blue;
    cout << "please input a number\n" << red;
    int size = 0;
    cin >> size;
    cout << blue;
    for (int i = 1; i <= size; i++)
        v.push_back(i);
    v.Debug();
    cout << blue << "\n\n";

    //检测:Vector(const Vector& r)
    cout << red << "test for Vector(const Vector& r)\n" << blue;
    Vector<int>v2(v);
    v2.Debug();
    cout << blue << "\n\n";

    //检测:operator[], size()
    cout << red << "test for operator[] and size()\n" << blue;
    v.Debug();
    cout << "please input a number between " << yellow << "0" << blue << " and " << yellow << v.size() - 1 << ":\n" << red;
    int index = 0;
    cin >> index;
    cout << blue << "v["  << index  << "] = " << yellow << (int)v[index] << blue << endl;
    cout << blue << "\n\n";

    //检测:at()
    cout << red << "test for at() and size()\n" << blue;
    v.Debug();
    cout << "please input a number between " << yellow << "0" << blue << " and " << yellow << v.size() - 1 << ":\n" << red;
    cin >> index;
    cout << blue << "v[" << index << "] = " << yellow << (int)v.at(index) << blue << endl;
    cout << blue << "\n\n";

    //检测:empty()
    cout << red << "test for empty()\n" << blue;
    v.Debug();
    cout << "v.empty() = " << yellow << v.empty() << blue << "\n";
    cout << blue << "\n\n";

    //检测:clear()
    cout << red << "test for clear()\n" << blue;
    v.clear();
    v.Debug();
    cout << "v.empty() = " << yellow << v.empty() << blue << "\n";
    cout << blue << "\n\n";

    return 0;
}
```

测试结果：

1. **at()**抛出异常：



```
test for Vector()
vector()
size:0  capacity:1
element:


test for Vector(int size)
size:0  capacity:100
element:


test for push_back(const &T x) and inflate
please input a number
10
inflate, size = 1
inflate, size = 2
inflate, size = 4
inflate, size = 8
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10


test for Vector(const Vector& r)
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10


test for operator[] and size()
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10
please input a number between 0 and 9:
8
v[8] = 9


test for at() and size()
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10
please input a number between 0 and 9:
10
```

Microsoft Visual C++ Runtime Library

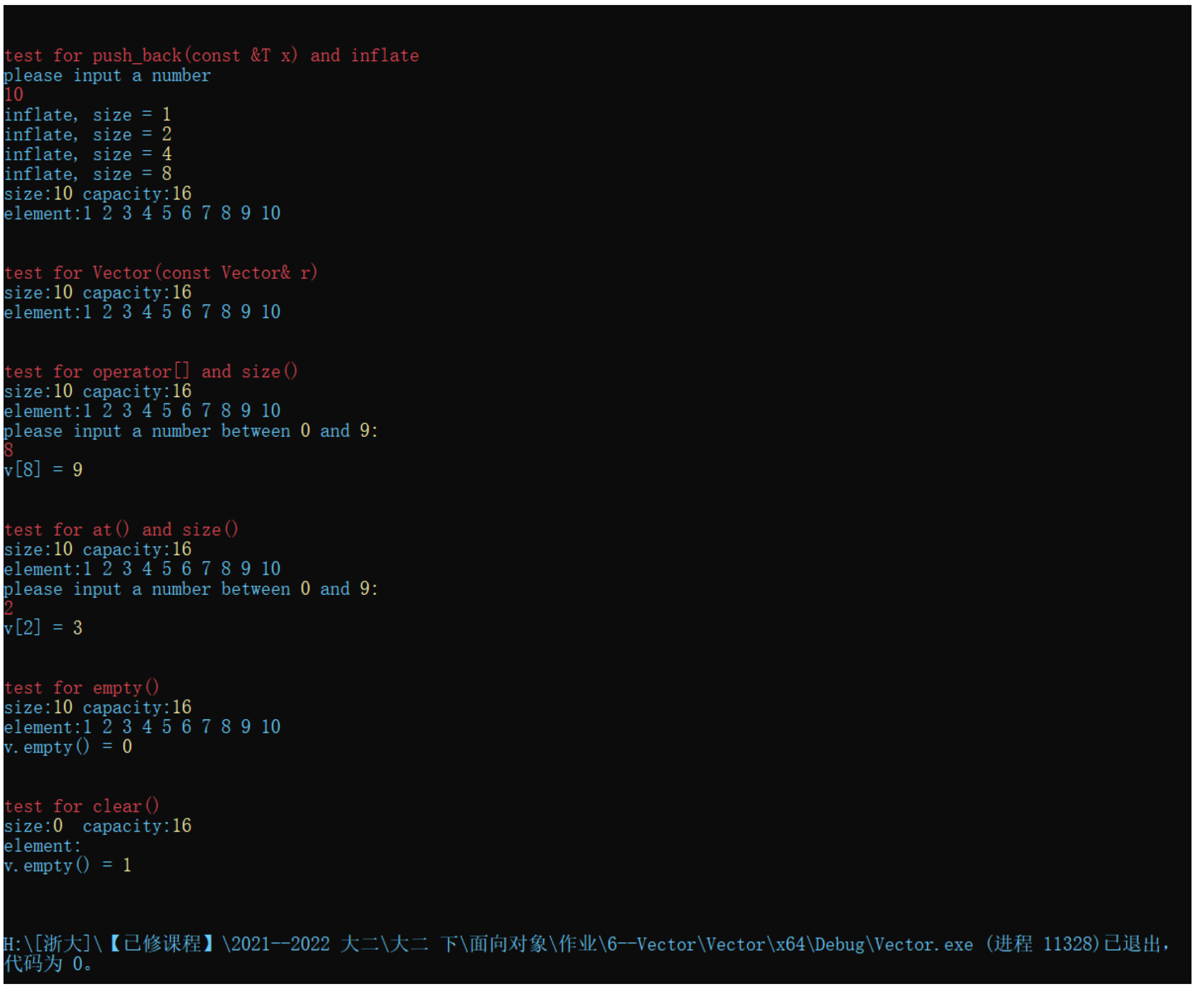
Debug Error!

Program: ...1--2022 大二\大二 下\面向对象\作业\6--Vector\Vector\x64\Debug\Vector.exe

abort() has been called

(Press Retry to debug the application)

2. **at()**函数未抛出异常：



```
test for push_back(const &T x) and inflate
please input a number
10
inflate, size = 1
inflate, size = 2
inflate, size = 4
inflate, size = 8
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10


test for Vector(const Vector& r)
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10


test for operator[] and size()
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10
please input a number between 0 and 9:
8
v[8] = 9


test for at() and size()
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10
please input a number between 0 and 9:
2
v[2] = 3


test for empty()
size:10 capacity:16
element:1 2 3 4 5 6 7 8 9 10
v.empty() = 0


test for clear()
size:0  capacity:16
element:
v.empty() = 1


H:\[浙大]\【已修课程】\2021--2022 大二\大二 下\面向对象\作业\6--Vector\Vector\x64\Debug\Vector.exe (进程 11328)已退出，
代码为 0。
```