天津大学

《高级语言程序设计》实验报告 1

学　　院　理学院

专　　业　应用物理学（强基计划）

年　　级　2020

姓　　名　贺卓诚

2022 年 4 月 28 日

实验八

一、 实验内容

1.向 my_list.h 中添加如下函数:

1).拷贝构造函数:

```
template <typename T>
my_list<T>(const my_list<T> &);
```

2).拷贝赋值:

```
template<typename T>
my_list<T>& operator=(const my_list<T>&);
```

3).移动构造函数:

```
template<typename T>
my_list<T>(my_list<T>&&);
```

4).移动复制:

```
template<typename T>
my_list<T>& operator=(my_list<T>&&);
```

2.在 my_list.cpp 中，添加以上函数的测试代码;

二、 实验任务

1. 拷贝构造函数:

创建一个新的 list 依次加入值

```
    my_list(const my_list<T> &other) : my_list()
    {
        for (node_ptr<T> it = other.head; it != other.tail; it = it->next)
        {
            this->push_back(*it);
        }
        cout << "copy construct" << endl;
    }
```

2. 拷贝赋值:

```
my_list<T> &operator=(const my_list<T> &other)
    {
        if (&other == this)
        {
            cout << "copy assignment, but already equal!!!" << endl;
            return *this;
        }
        clear();
        for (node_ptr<T> it = other.head; it != other.tail; it = it->next)
        {
            this->push_back(*it);
```

```
        }
        cout << "copy assignment" << endl;
        return *this;
    }
```

3. 移动构造函数:

头对头，尾对尾，中间自动跟着过去了

```
my_list<T>(my_list<T> &&other) : my_list()
    {
        std::swap(head, other.head);
        std::swap(tail, other.tail);
        cout << "move construct" << endl;
    }
```

4. 移动复制:

```
my_list<T> &operator=(my_list<T> &&other)
    {
        std::swap(head, other.head);
        std::swap(tail, other.tail);
        cout << "move assignment" << endl;
        return *this;
    }
```

5. 驱动函数测试:

```
void print()
    {
        for (auto it = cbegin(); it != cend(); it++)
        {
            cout << *it << " ";
        }
        cout << endl;
    }
```

其余主函数比较简单易懂，就不往这里复制了。