编号： 第 1 章第 1 次

山东师范大学
SHANDONG NORMAL UNIVERSITY

信息科学与工程学院实验报告

《面向对象程序设计》

Object-Oriented Programming

姓名：	朱会琛
学号：	202311000202
班级：	计工本 2302
教师：	张庆科
时间：	2024 年 11 月 15 日

《面向对象程序设计》实验报告

报告要求：实验报告包含实验目的、实验内容、实验过程（详细操作流程）、实验结果（程序运行结果高清截图）、实验分析总结五个部分。报告中若涉及代码程序，请在附录部分提供完整程序源码及源码托管地址(基于 Highlight 软件导入源码)。报告撰写完毕后请将 PDF 格式版本上传到坚果云作业提交系统。

一、实验目的

- 理解 C++类的属性和方法的概念
- 掌握 C++类的构成和类的设计方法
- 掌握 this 指针的底层原理和用法
- 理解构造函数和析构函数的工作原理
- 能够熟练在类中使用构造函数和析构函数

二、实验内容

实验 1：三角形类的设计

请在 visual studio 环境下使用 C++语言设计一个三角形类 Triangle，其中三角形信息包括：各边边长； 三角形类对外能提供的接口功能包括：三角形初始化、 判断能否构成三角形、输出三角形周长、输出三角形面积、输出三角形类型（直角，锐角，还是钝角三角形）。

提示:面积计算可以通过海伦公式完成, 三角形面积 s = sqrt(p*(p-a)*(p-b)*(p-c)),其中 p 为三角形的半周长, a,b,c 为三角形的三条边长.

实验 2：栈类的设计

请采用 C++语言设计一个栈类 Stack, 通过该类实现与栈相关的各种操作。常见栈类功能包括： 栈内数据初始化、数据进栈、数据出栈、判断栈是空的还是满的、计算当前栈的大小、对象空间释放。

提示:栈内数据存储可以借助动态内存数组实现.

实验 3：字符串类的设计

请设计一个字符串类 Mystring, 通过该类实现与字符串相关的各种操作。 字符串类的主要功能包括:字符串长度计算、 字符串拼接、字符串大小比较、字符串元素查找和替换操作、对象空间释放。要求使用字符数组存储字符串,不允许使用 string 类型.

三、实验过程

任务一源码：

```
#include <iostream>
#include <algorithm>
#include <cmath>

using namespace std;

class Triangle {
private:
        int a,b,c;
        double Area;


public:
        void Sets(int x,int y,int z) {
                a = x;
                b = y;
                c = z;
        }
        double getArea() {
                double ans = (a + b + c) / 2;
                //cout << ans;
                return sqrt(ans * (ans - a) * (ans - b) * (ans - c));
        }

        void getShape() {
                if ((a * a + b * b) < (c * c)) {
                        cout << "钝角三角形" << endl;
                }
                else if ((a * a + b * b) > (c * c)) {
                        if (a == b && a == c) {
                                cout << "等边三角形" << endl;
                                return;
                        }
                        cout << "锐角三角形" << endl;
                }
                else {
                        cout << "直角三角形" << endl;
                }
        }

};
```

```
int main() {

        Triangle res;
        int cn[3];
        cout << "请输入三角形的三边:";
        for (int i = 0; i < 3; i++) {
                cin >> cn[i];
        }
        sort(cn, cn + 3);
        res.Sets(cn[0], cn[1], cn[2]);
        double Area = res.getArea();
        cout << "三角形的面积:" << Area << endl;
        cout << "三角形的形状是:" ;
        res.getShape();

        return 0;
}
```

任务二源码：

```
#include <iostream>
#include <cstring>
using namespace std;

const int N = 1000;
int s[N];
int top = -1;
//bool empty = false;
class mystack {
public:
        void push(int x) {
                s[++top] = x;
        }
        void pop() {
                top--;
        }
        void empty() {
                if (top == -1) {
                        cout << "YES" << endl;
                }
                else {
                        cout << "NO" << endl;
                }
        }
        void query() {
```

```
                cout << s[top] << endl;
        }
};

int main() {

        mystack ms;

        int n;
        cin >> n;
        while (n--) {
                string s;
                cin >> s;
                if (s == "push") {
                        cout << "入栈:" << endl;
                        int a;
                        cin >> a;
                        ms.push(a);
                }
                else if (s == "pop") {
                        cout << "出栈:" << endl;
                        ms.pop();
                }
                else if (s == "query") {
                        cout << "查询" << endl;
                        ms.query();
                }
                else {
                        cout << "判空:" << endl;
                        ms.empty();
                }
        }


        return 0;
}
```

任务三源码：

```
#include <iostream>
#include <cstring>

using namespace std;

class MyString {
```

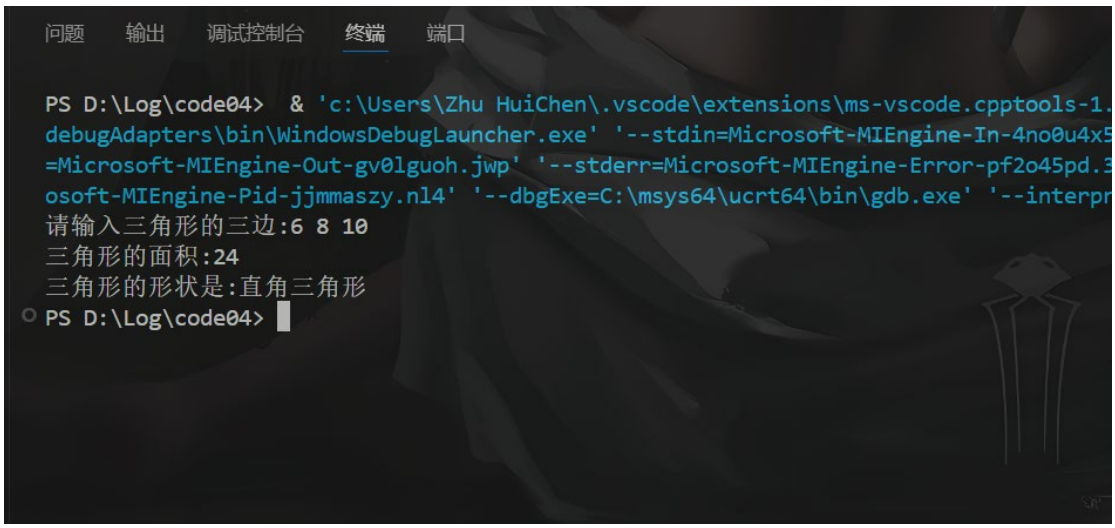
```
private:
    char* str;  // 动态分配的字符数组
    int len;    // 字符串的长度，使用 int 类型

public:
    // 默认构造函数
    MyString() : str(nullptr), len(0) {
        str = new char[1];  // 空字符串
        str[0] = '\0';
    }

    // 构造函数，初始化字符串
    MyString(const char* s) {
        if (s) {
            len = strlen(s);
            str = new char[len + 1];
            strcpy(str, s);
        } else {
            str = new char[1];  // 空字符串
            str[0] = '\0';
            len = 0;
        }
    }

    // 拷贝构造函数
    MyString(const MyString& other) {
        len = other.len;
        str = new char[len + 1];
        strcpy(str, other.str);
    }

    // 析构函数
    ~MyString() {
        delete[] str;
    }

    // 获取字符串的长度
    int length() const {
        return len;
    }

    // 字符串拼接
    MyString operator+(const MyString& other) const {
        MyString newStr;
```

```
        newStr.len = len + other.len;
        newStr.str = new char[newStr.len + 1];

        strcpy(newStr.str, str);           // 先拷贝当前字符串
        strcat(newStr.str, other.str);     // 拼接另外一个字符串

        return newStr;
    }

    // 字符串大小比较
    int compare(const MyString& other) const {
        return strcmp(str, other.str);
    }

    // 查找字符
    int find(char c) const {
        for (int i = 0; i < len; i++) {
            if (str[i] == c) {
                return i;
            }
        }
        return -1;  // 找不到返回-1
    }

    // 替换字符
    void replace(char oldChar, char newChar) {
        for (int i = 0; i < len; i++) {
            if (str[i] == oldChar) {
                str[i] = newChar;
            }
        }
    }

    // 输出字符串
    void print() const {
        cout << str << endl;
    }

    // 获取底层的字符数组
    const char* getCStr() const {
        return str;
    }
};
```

```
// 测试函数
int main() {
    MyString str1("I prefer c++");
    MyString str2(" to java !");
    MyString str3 = str1 + str2;  // 字符串拼接

    str3.print();

    cout << "Length of str3: " << str3.length() << endl;  // 输出 22

    cout << "Comparing str1 and str2: " << str1.compare(str2) <<
endl;  // 输出正数，表示 str 大于 str2

    int pos = str3.find('W');
    if (pos != -1) {
        cout << "'W' found at position: " << pos << endl;
    }

    str3.replace('!', '?');
    str3.print();  // 输出 " I prefer c++ to java ?"

    return 0;
}
```

四、实验结果

任务一运行结果：



任务二运行结果：

```
PS D:\Log\code04>  & 'c:\Users\Zhu HuiChen\.vscode\extensions\ms-vscode.cpptools-1.23.1-
debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-cfmu2d0o.wnj'
=Microsoft-MIEngine-Out-jb3xkrtz.1co' '--stderr=Microsoft-MIEngine-Error-jscs0a4g.zz0' '
osoft-MIEngine-Pid-zitnypaq.o1b' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=
3
push
入栈:
8
empty
判空:
NO
pop
出栈:
PS D:\Log\code04>
```

任务三运行结果：

```
PS D:\Log\code04>  & 'c:\Users\Zhu HuiChen\.vscode\extensions\ms-vscode.cpp
debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-
=Microsoft-MIEngine-Out-zzr1un3h.op3' '--stderr=Microsoft-MIEngine-Error-gc
osoft-MIEngine-Pid-wwzpsrns.3z2' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '-
I prefer c++ to java !
Length of str3: 22
Comparing str1 and str2: 1
I prefer c++ to java ?
PS D:\Log\code04>
```

五、实验总结

答：本次实验的目的是设计并实现一个自定义字符串类 MyString，用以模拟 C++ 中的字符串操作。我们通过使用字符数组存储字符串，避免使用 std::string 类型，深入理解字符串的内存管理和操作方法。类的基本功能包括计算字符串长度、字符串拼接、大小比较、查找字符和替换字符等。在设计 MyString 类时，我们使用动态内存分配来存储字符串，并通过构造函数和拷贝构造函数管理对象的初始化和拷贝。析构函数负责释放内存，以避免内存泄漏。length() 方法用于获取字符串的长度，operator+() 方法实现了字符串拼接，compare() 方法通过 strcmp 进行字符串比较，find() 方法查找字符的位置，replace() 方法则替换指定的字符。实验过程中，通过测试不同的字符串操作，我们验证了各个功能的正确性。所有测试均符合预期，程序能够正常处理字符串的基本操作，并且有效管理内存。

通过本次实验，我加深了对 C++ 字符串操作和内存管理的理解，并掌握了如何在没有标准库支持的情况下实现字符串相关功能。实验成功实现了一个功能完整的字符串类，且内存管理得当。