

西安电子科技大学计算机科学与技术学院

1. **实验目的**

完成第12章习题1和10，并实现栈模板类。

1. **实验环境**

IDE：clion cmake mingw

os：windows

硬件：Dell g15 5511笔记本

1. **实验内容**
2. 根据题目要求的类继承关系编写程序并运行，查看结果。
3. 定义父类Shape，然后定义其三个子类Circle、Square、Triangle。要求定义一个函数intersect(Shape\* s1, Shape\* s2)，使得输入两个形状之后，能运行对应的判断两个形状是否重合的算法。
4. 实现Stack模板类，能够实现一定功能。
5. **实验步骤**

1、

令Base有两个子类Son和Dau。测试不同对象对iam函数的调用结果；测试Base\*指针指向子类对象时调用iam函数的结果。

2、

父类Shape中定义四个纯虚函数：

virtual bool intersect\_sec(Shape\* const s) = 0;

virtual bool judgeIntersect(Circle& s) = 0;

virtual bool judgeIntersect(Square& s) = 0;

virtual bool judgeIntersect(Triangle& s) = 0;

子类各自实现其对应函数。

intersect(Shape\* s1, Shape\* s2)调用s1->intersect\_sec(s2)，根据父类指针指向子类对象的性质，这个调用会自动指向s1的具体形状的子类的intersect\_sec函数，以完成对s1形状的判断与分配。

然后，intersect\_sec函数调用s->judgeIntersect(\*this)，同理完成对s2的形状的判断与分配。

最后，在s2的类里调用与s1类型（即\*this对应的类型）对应的judgeIntersect函数，即寻找到了s2与s1匹配的判断重合的函数。

主要思想即：将第一个父类指针指向子类对象匹配到对应的子类中，然后将第二个父类指针指向子类对象匹配到对应的子类中，最后通过同名函数自动匹配比较函数。到最后自动匹配时，已经没有父类指针参与了。

3、

以栈中数据类型T来建立该模板类，即template <class T>。

1. 以int pos来指示栈顶位置，栈为空时pos为-1。
2. 使用T\* dataList=new T[SIZE];来讲数据存储在堆中的数组里，SIZE初值为4。
3. 限定数据存储最大大小MAXSIZE=1024，若数据量超出了1024，则由assert(isMax());进行限制。
4. 若dataList数组已满，则令SIZE\*=2，然后new一个SIZE大小的临时数组进行复制，然后交换并delete[]原数组，实现数组扩大。
5. void push(const T &item);可以把一个T类型对象压入栈，使用assert(isMax());限制大小，并判定如果dataList满的话就扩展数组。
6. void pop();弹出栈顶，T& top();获得栈顶的引用。都使用assert(!isEmpty());进行保护。
7. void clear();用于清空栈。另SIZE=4;然后delete[] dataList;并dataList=new T[SIZE];使得dataList回归原始状态，并另pos=-1。
8. int size();返回数组大小，即pos+1。
9. **实验结果**

1、

main函数如下：

Base b; Son son; Dau dau;

b.iam();

son.iam();

dau.iam();

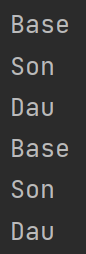
Base\* p;

p = &b; p->iam();

p = &son; p->iam();

p = &dau; p->iam();

运行结果如下：



达到了预期结果，子类成功覆盖父类的virtual函数；父类指针指向子类对象时正确调用子类函数。

2、

main函数如下：

Circle circle;

Square square;

Triangle triangle;

intersect(&circle,&circle);

intersect(&square, &square);

intersect(&triangle, &triangle);cout<<endl;

intersect(&circle, &square);

intersect(&square, &circle);cout<<endl;

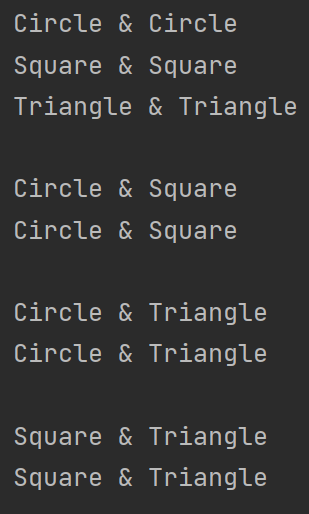
intersect(&circle, &triangle);

intersect(&triangle, &circle);cout<<endl;

intersect(&square, &triangle);

intersect(&triangle, &square);cout<<endl;

结果如下：



正确调用了对应的判断函数。

3、

main函数如下：

Stack<int> s;

s.push(1);s.push(2);s.push(3);

s.push(4);s.push(5);s.push(6);

cout << "size=" << s.size() << endl;

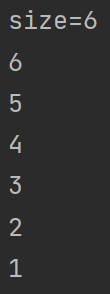
while (!s.isEmpty()) {

cout << s.top() << endl;

s.pop();

}

结果为：



成功实现栈功能。

1. **实验总结**

阐述通过实验你对对本次实验的认识和理解（200字左右）。

1. 明确了解到，cpp与java不同，若子类要覆盖父类的函数的时候，必须在父类的对应方法前加上virtual，十分严格。
2. 父类指针指向子类对象使得程序逻辑更为简洁，但有时候需要把对象的具体类型区分开，而语法提供了Base\*->function()的方式使得访问具体的对象类型的函数、变量成为可能（本质上是子类与父类的逻辑存储关系所致）。而第二题巧妙地把这种方法发挥到极致，通过双层筛选使得两个对象被识别出到底是哪一类的同时没有失去两个对象的紧密关联，筛选完后使用同名不同变量的函数直接分配对应所需函数，实在巧妙。另外，\*this能够直接返回最严格的类型而非更基层的类型也是非常重要的一点。
3. Stack模板类的实现是非常基础的模板类实践，应当熟练掌握。