9522 编程填空：学生信息处理程序
9520 奇怪的类复制
9521 返回什么才好呢
15997 超简单的复数类
15998 哪来的输出

****15997.cpp:

```
#include <iostream>
#include <cstring>
#include <cstdlib>
using namespace std;
/*
     第一种解法写了类型转换构造函数，使得 a = "5+6i"; 赋值号右边自动生成一个临时对象，再赋值给 a
     第二种解法返回引用，是因为要符合赋值号的使用习惯，并非必须。a=b 这个表达式返回 a 的引用，是未重载的赋值号的特性。
*/
class Complex {
private:
    double r,i;
public:
    void Print() {
        cout << r << "+" << i << "i" << endl;
    }
//your code starts here
//解法 1
    Complex() { };
    Complex ( const char * p) {
        r = p[0] - '0';
        i = p[2] - '0';
    }

    /*解法 2
    Complex & operator = ( const char * p) {
        r = p[0] - '0';
        i = p[2] - '0';
        return * this;
    }
    */

//your code ends here
};
```

```
int main() {
    Complex a;
    a = "3+4i"; a.Print();
    a = "5+6i"; a.Print();
    return 0;
}
```

****15998.cpp:

```
#include <iostream>
using namespace std;
class A {
    public:
        int i;
        A(int x) { i = x; }
//your code starts here
        ~A() {
            cout << i << endl;
        }
//your code ends here

};
int main()
{
    A a(1);
    A * pa = new A(2);
    delete pa;
    return 0;
}
```

****9520.cpp:

```
/*
程序填空，使得输出结果是
9
22
5

*/
#include <iostream>
using namespace std;
class Sample {
```

```
public:
    int v;
    //your code starts here
    Sample () { };
    Sample(int n):v(n) { };
    Sample(const Sample & x) { v = 2 +x.v; }
    //your code ends here
};
void PrintAndDouble(Sample o)
{
    cout << o.v;
    cout << endl;
}
int main()
{
    Sample a(5);
    Sample b = a;
    PrintAndDouble(b);
    Sample c = 20;
    PrintAndDouble(c);
    Sample d;
    d = a;
    cout << d.v;
    return 0;
}
```

****9522.cpp:

/*

实现一个学生信息处理程序

输入数据为一行：
姓名，年龄，学号（整数），第一学年平均成绩，第二学年平均成绩，第三学年平均成绩，第四学年平均成绩

输出：
姓名，年龄，学号，四年平均成绩

例如：

输入：Tom Hanks,18,7817,80,80,90,70

输出：Tom Hanks,18,7817,80

要求实现一个代表学生的类，并且所有成员变量都应该是私有的。

```
*/
// by Guo Wei
#include <iostream>
#include <cstring>
#include <cstdlib>
#include <string>
using namespace std;
class CStudent
{

    private:
        static const int COURSE_NUM = 4;
        char name[20];
        int age;
        int id;
        int scores[COURSE_NUM];
    public:
        int average() {
            int sum = 0;
            for( int i = 0;i < COURSE_NUM; ++i)
                sum += scores[i];
            return sum / COURSE_NUM;
        }
        void readInfo( ) {
            char buf[210];
            cin.getline(buf,200);
            char * p = strtok(buf,",");
            strcpy(name,p);
            p = strtok(NULL,",");
            age = atoi(p);
            p = strtok(NULL,",");
            id = atoi(p);
            for( int i = 0;i < COURSE_NUM; ++i ) {
                p = strtok(NULL,",");
                scores[i] = atoi(p);
            }
        }
        /* 另一写法：
        void readInfo() {
            char buf[110];
            cin.getline(buf,100);
            char * p = strchr(buf,',');
            p[0] = 0;
            strcpy( name,buf);
            sscanf(p + 1, "%d,%d,%d,%d,%d,%d",&id,&age,
                averageScore,averageScore+1,averageScore+2,
                averageScore+3);
        }
        */

        void printInfo() {
            cout << name <<"," << age << "," << id << "," << average() << endl;

        }
};
int main()
{
    CStudent s;
    s.readInfo();
    s.printInfo();
}

/* strtok 用法示例:

        char str[] ="- This, a sample string, OK.";
        //下面要从 str 逐个抽取出被" ,.-"这几个字符分隔的字串
        char * p = strtok (str," ,.-"); //请注意，" ,.-"中的第一个字符是空格
        while ( p != NULL) //只要 p 不为 NULL，就说明找到了一个子串
        {
            cout << p << endl;
            p = strtok(NULL, " ,.-");//后续调用，第一个参数必须是 NULL
        }
输出:
This
a
sample
string
OK

*/
```