输入输出重定向

1. 基本概念

输出重定向：程序执行时，系统默认的输出设备是显示器，如果改为其他设备/文件，则称为输出重定向

输入重定向：程序执行时，系统默认的输入设备是键盘，如果改为其他设备/文件，则称为输入重定向

2. 将输出重定向到文件中

2.1. 输出的分类

cout：标准输出

cerr：错误输出

clog：错误输出

第13章的内容，使用方法相同

★ 下面这个例子，在正常输出为屏幕时，看到的结果没有任何差别

```
#include <iostream>
using namespace std;
int main()
{
    cout << "标准输出(cout)ex1" << endl;
    cerr << "错误输出(cerr)ex1" << endl;
    clog << "错误输出(clog)ex1" << endl;
    return 0;
}
```

输入输出重定向

2. 将输出重定向到文件中

2.2. 标准和错误输出重定向到文件中

```
#include <iostream>
using namespace std;
int main()
{
    cout << "标准输出(cout)ex1" << endl;
    cerr << "错误输出(cerr)ex1" << endl;
    clog << "错误输出(clog)ex1" << endl;
    return 0;
}
```

★ 操作步骤（脱离集成编译环境，以VS2017为例）

step1 ： 正确输入程序并编译通过
可执行文件在 D:\test\debug 下

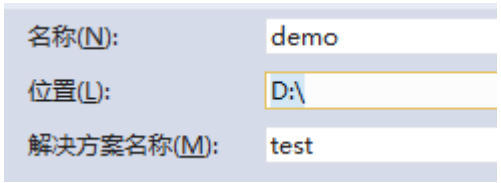
step2 ： 进入到cmd窗口

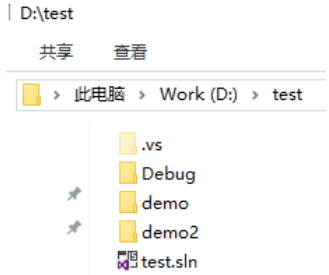
step3 ： 切换到程序所在盘符(D:)

step4 ： 进入到可执行文件程序所在目录
cd D:\test\debug

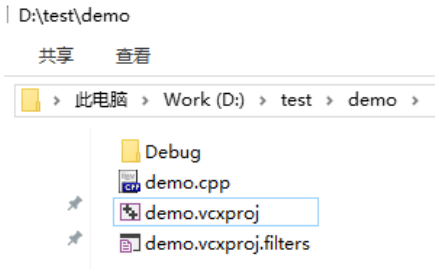
step5 ： 分别输入以下命令，观察运行结果
demo
demo >a.txt
demo 1>a.txt
demo 2>a.txt
demo 1>a.txt 2>b.txt
demo 1>a.txt 2>&1

例：假设解决方案名称 test，位置放在D:\下，
项目名称是demo/demo2，则编辑/编译完成后目录结构如下：

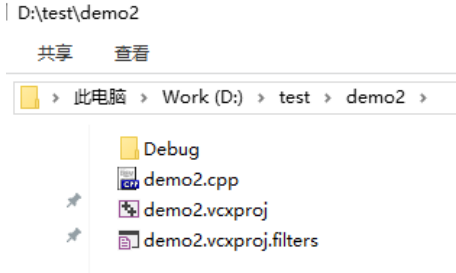




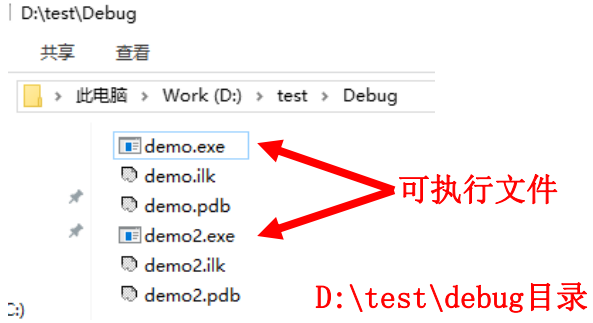
D:\test目录



D:\test\demo 目录



D:\test\demo2 目录



可执行文件

D:\test\debug目录

2.3. 重定向文件的追加

引入：上例中，若命令反复执行，则a.txt或b.txt的内容仅会保留最后一次

解决：将>换为>>即可不断追加而不清空原有内容

输入输出重定向

3. 将输入重定向为来自文件中

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

int main()
{
    int a, b;

    cout << "请输入两个整数" << endl;
    cin >> a >> b;
    cerr << "a=" << a << "  b=" << b << endl;
    cout << "大数是：" << (a>b?a:b) << endl;

    return 0;
}
```

★ 操作步骤（脱离集成编译环境，以VS2017为例）

step1 ： 正确输入程序并编译通过

可执行文件在 D:\test\debug 下

step2 ： 进入到cmd窗口

step3 ： 切换到程序所在盘符(D:)

step4 ： 进入到可执行文件程序所在目录

cd D:\test\debug

step5 ： 分别输入以下命令，观察运行结果

demo

step6 ： 用记事本编辑z.dat，写入两个整数，再输入以下命令，观察运行结果

demo < z.dat

问题：

如果z.dat中 (1) 仅有1个整数

(2) 3个及以上整数

(3) 不是整数（类似于12a34这种）

(4) 不是整数（字母或符号开头）

demo < z.dat：运行结果？

4. 同时进行输入/输出重定向

★ 命令组合即可

```
demo <z.dat 1>a.txt
demo 1>a.txt <z.dat
demo >a.txt <z.dat
demo 1>a.txt 2>b.txt <z.dat
demo 1>a.txt 2>&1 <z.dat
```