



§ . 输入输出重定向及管道运算符

1. 预备知识
2. 输入输出重定向
3. 用输入输出重定向测试程序
4. 管道运算符的使用
5. 文本比较工具的使用

本文档分别对应5个视频文件，都放在附件中



§ . 输入输出重定向及管道运算符

1. 预备知识

1. 1. 准备工作:

用VS在D盘根目录下建立“test”解决方案，其中包含“demo”和“demo2”两个项目

- ★ 创建之前，观察D盘根目录下现有的文件夹
- ★ 创建之后，观察D盘根目录下现有的文件夹

★ 要求：各位同学按需在自己需要的位置建立解决方案，并且能够找到对应位置



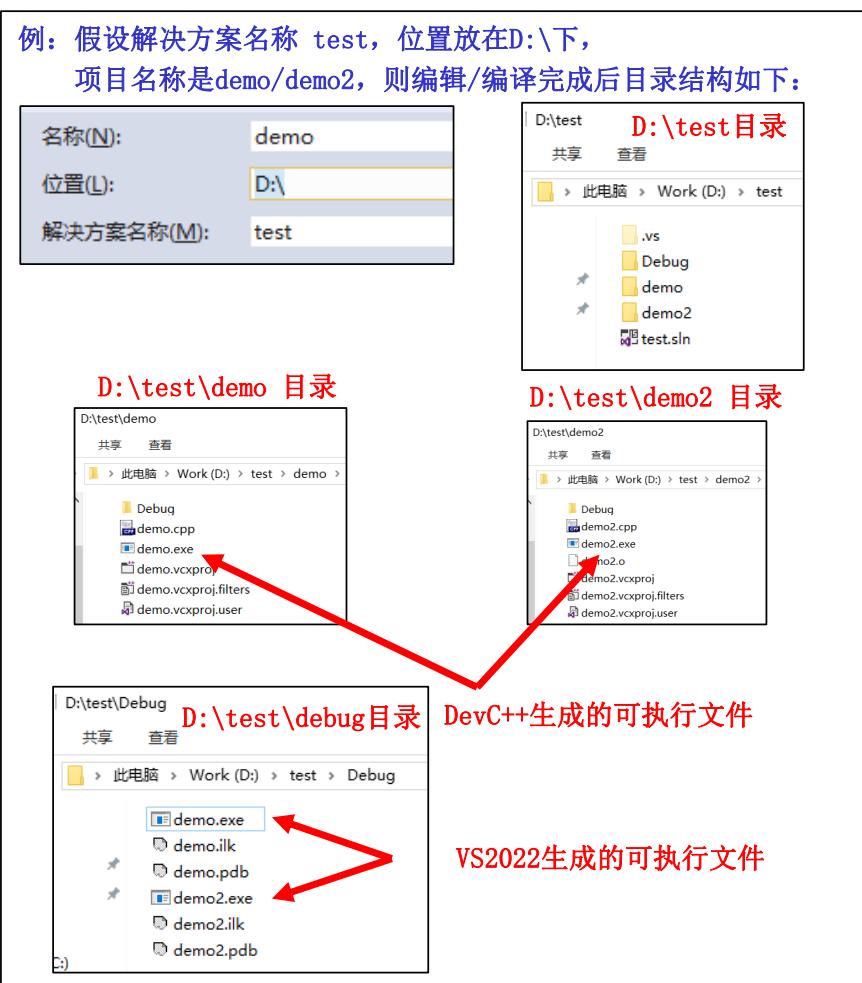
§ . 输入输出重定向及管道运算符

1. 预备知识

1. 2. VS2022/Dev C++编译生成的可执行文件(*.exe)的位置说明

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：



说明：

- 1、本文档后续说明均以此目录结构为例，实际使用时要对应到自己的相应目录下
- 2、VS2022生成的可执行文件，在与项目(Demo/Demo2)同级的Debug目录中
- 3、DevC++生成的可执行文件名，与源程序(*.cpp)在同一目录下



§ . 输入输出重定向及管道运算符

1. 预备知识

1. 3. 在图形化界面中双击exe执行导致“闪退”的原因解释

★ 在集成环境下运行与直接双击exe文件名方式运行的差异

{ 在集成环境下运行 : 直接 CTRL+F5(VS) / F11(Dev) 运行, 结束后提示按键关闭
直接双击exe文件名运行: “闪退”

★ “闪退”的现象及原因解释

双击exe后, 会自动开启一个cmd窗口, 随后运行程序, 运行完成后立即关闭

★ 解决“闪退”的方法

- 方法1: 在程序最后加一句 `system("pause")` / 其它方式的读键盘语句
=> 多人批量运行时, 完成时必须按键才能继续下一个人, 因此会“卡死”, 导致批处理无法自动执行
(仅限于演示, 不允许作业中使用, 否则0分)
- 方法2: 先手动启动一个cmd窗口, 在窗口中用命令行方式运行 (1. 4~1. 6, 必须掌握)

★ 从即日起, 下发的所有参考exe, 均为命令行方式执行

★ **再次强调:** 提交的作业不要加`system("pause")`等暂停/读键语句, 否则会因为检查作业时卡死而被判为死循环, 得分为0



§ . 输入输出重定向及管道运算符

1. 预备知识

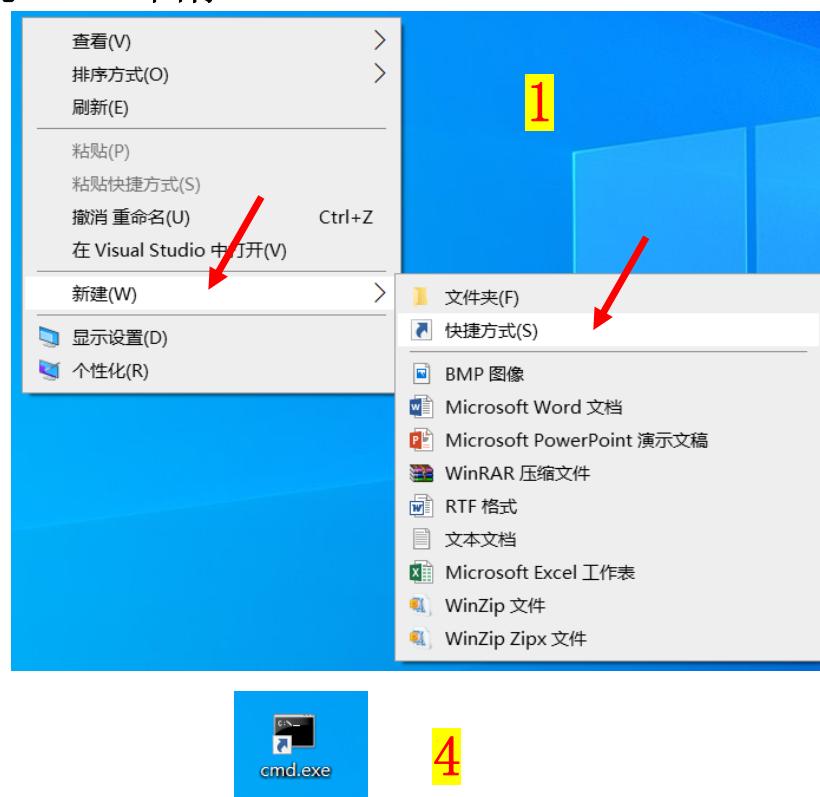
1. 4. 在桌面上建立cmd. exe的快捷方式

Step1: 桌面空白处右键 - 新建 - 快捷方式

Step2: 出现的对话框中输入cmd, 按“下一步”

Step3: 输入该快捷方式的名称, 按“完成”

Step4: 桌面出现cmd. exe图标





§ . 输入输出重定向及管道运算符

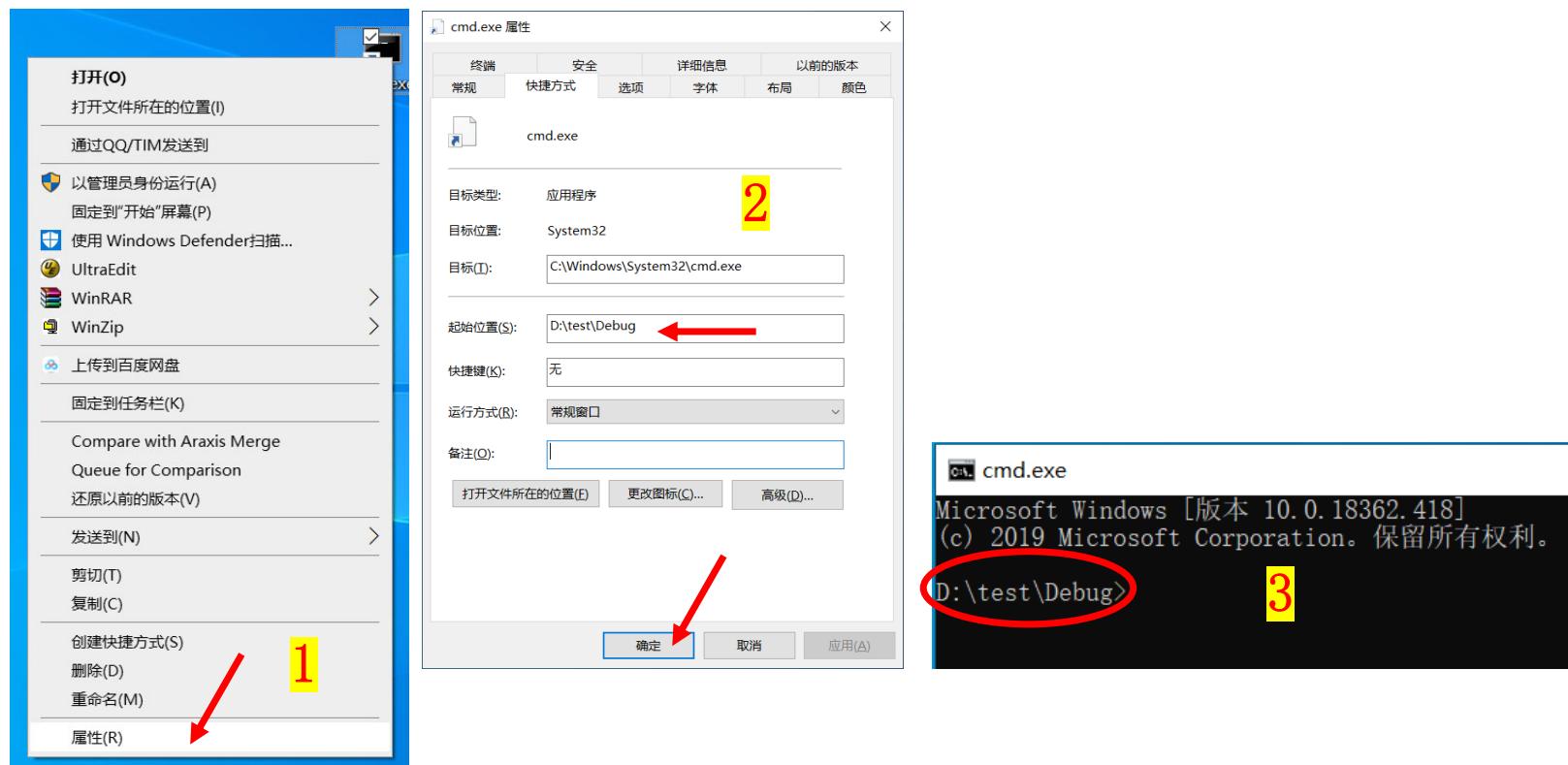
1. 预备知识

1. 5. 设置启动cmd. exe快捷方式后缺省进入指定目录(本文档以D:\test\debug为例, 可对应修改)

Step1: 在cmd. exe快捷方式上右键 - 属性

Step2: 出现的对话框中, 将“起始位置”改为指定目录, 按“确定”

step3: 双击cmd. exe快捷方式, 确认进入D:\test\Debug





§ . 输入输出重定向及管道运算符

1. 预备知识

1. 6. 启动cmd. exe快捷方式后，手工输入命令，进入任意目录的方法(本文档以D:\test\debug为例，可对应修改)

Step1：双击0. 2建立的cmd. exe快捷方式(缺省对应的目录是 C:\Windows\system32)

cmd. exe
Microsoft Windows [版本 10.0.22621.1413]
(c) Microsoft Corporation. 保留所有权利。
C:\Windows\System32> 1

Step2：出现的cmd窗口中，输入D:，按回车

cmd. exe
Microsoft Windows [版本 10.0.22621.1413]
(c) Microsoft Corporation. 保留所有权利。
C:\Windows\System32>d: 2
D:\>

step3：出现 D:\> 后，再输入cd d:\test\debug，按回车

cmd. exe
Microsoft Windows [版本 10.0.22621.1413]
(c) Microsoft Corporation. 保留所有权利。
C:\Windows\System32>d:
D:\>cd D:\test\Debug 3
D:\test\Debug>

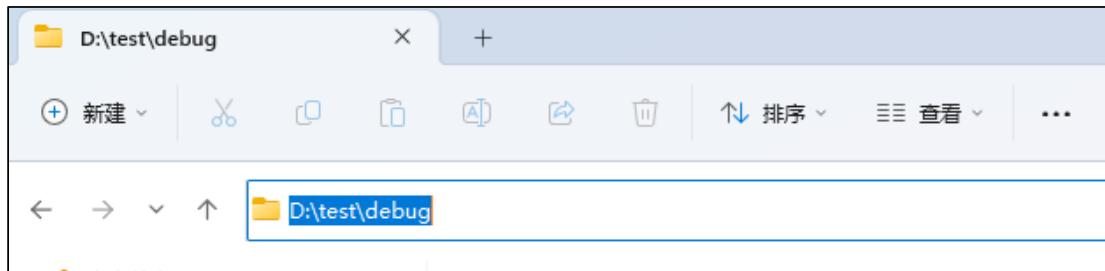
注：这种方法可以进入任意目录，缺点是每次启动cmd. exe后都要手工输入，具体采用1. 5还是1. 6的方法，各人按需选择即可



§ . 输入输出重定向及管道运算符

1. 预备知识

1. 7. 通过文件管理器启动cmd. exe，直接进入对应目录的方法(本文档以D:\test\debug为例，可对应修改)



```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.22621.1413]
(c) Microsoft Corporation。保留所有权利。

D:\test\debug>
```

注：这种方法进入指定目录后，还可以按1. 6的方法，再进入其它目录



§ . 输入输出重定向及管道运算符

2. 输入输出重定向

2. 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



§ . 输入输出重定向及管道运算符

2. 输入输出重定向

2. 2 将输出重定向到文件中

2. 2. 1. 输出的分类

cout: 标准输出

cerr: 错误输出

clog: 错误输出

例：观察下面程序的运行结果

(1) 集成环境下

(2) cmd窗口中

注：VS和Dev生成的exe在不同目录下，为方便，将Dev的exe文件移动到VSexe文件所在的目录中，改名为demo.dev.exe

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

int main()
{
    cout << "标准输出(cout)" << endl;
    cerr << "错误输出(cerr)" << endl;
    clog << "错误输出(clog)" << endl;

    return 0;
}
```



§ . 输入输出重定向及管道运算符

2. 输入输出重定向

2. 2 将输出重定向到文件中

2. 2. 2. 标准和错误输出重定向到文件中

★ 将2. 1中VS2022/DevC++生成的exe文件在cmd窗口下运行，分别输入不同的命令，观察运行结果

进入cmd窗口，分别输入以下命令，观察运行结果

- 1、demo
- 2、demo >a.txt
- 3、demo 1>a.txt
- 4、demo 2>a.txt
- 5、demo 1>a.txt 2>b.txt
- 6、demo 1>a.txt 2>&1

注：1、cmd下运行exe时，“.exe”后缀可以不加

2、观察运行果时，注意目录下的文件变化，以及a.txt/b.txt等文件中内容的变化

- dir : 用来列出当前目录下的文件
- notepad a.txt : 用“记事本”查看a.txt文件

3、自行替换为demo.dev.exe，观察结果（应相同）



§ . 输入输出重定向及管道运算符

2. 输入输出重定向

2. 2. 将输出重定向到文件中

2. 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

进入cmd窗口，分别输入以下命令，观察运行结果

- 1、[demo](#)
- 2、[demo >a. txt](#) (尝试一次>，多次>>)
- 3、[demo 1>a. txt](#) (尝试一次>，多次>>)
- 4、[demo 2>a. txt](#) (尝试一次>，多次>>)
- 5、[demo 1>a. txt 2>b. txt](#) (尝试一次>，多次>>)
- 6、[demo 1>a. txt 2>&1](#) (尝试一次>，多次>>)

注：1、cmd下运行exe时，“. exe”后缀可以不加

2、观察运行果时，注意目录下的文件变化，以及a. txt/b. txt等文件中内容的变化

- dir : 用来列出当前目录下的文件
- notepad a. txt : 用“记事本”查看a. txt文件

3、自行替换为demo. dev. exe，观察结果（应相同）



§ . 输入输出重定向及管道运算符

- 2. 输入输出重定向
- 2. 2. 将输出重定向到文件中
- 2. 2. 4. 在图形化界面中双击exe执行导致“闪退”的原因解释

见1.3



§ . 输入输出重定向及管道运算符

2. 输入输出重定向

2. 3. 将输入重定向为来自文件中

例：观察下面程序的运行结果

- (1) 集成环境下
- (2) cmd窗口中

注：VS和Dev生成的exe在不同目录下，为方便，将Dev的exe文件移动到VSexe文件所在的目录中，改名为demo.dev.exe

```
#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;
}
```

进入cmd窗口，分别输入以下命令，观察运行结果

- 1、 [demo](#)
- 2、 [demo < z.dat](#)

注：用记事本编辑z.dat，写入两个整数，再保存

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？



§ . 输入输出重定向及管道运算符

2. 输入输出重定向

2. 4. 同时进行输入/输出重定向

★ 上例的exe，将命令组合即可（顺序任意），观察运行结果

```
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
```

● 也可以自行将>换成>>，观察结果



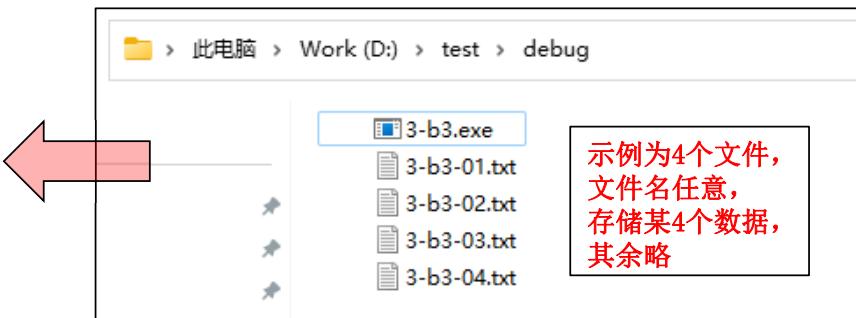
§ . 输入输出重定向及管道运算符

3. 用输入/输出重定向测试程序(以浮点数分解3-b3为例)

3. 1. 单次测试

★ 准备工作：在D:\test\Debug(本文档示例目录，自行替换)下建立24个文件，将要求的24个数据分别放入，
每个文件一个数据

9999999999. 99	9999999999. 90	9999999999. 09	9900000000
8912003005. 78	2501200350. 03	1203056740. 00	203056740. 20
23000056. 82	3051200. 72	301000. 35	10001. 34
8070. 23	9876. 54	803. 03	12. 30
10. 03	9. 30	7. 03	0. 35
0. 30	0. 07	0. 03	0



★ 依次运行：生成3-b4. exe后在cmd下用输入重定向方式运行，观察结果(可多个)

```
D:\test\debug>3-b3
请输入[0-100亿)之间的数字:
9999999999. 99
十亿位 : 9
亿位 : 9
千万位 : 9
百万位 : 9
十万位 : 9
万位 : 9
千位 : 9
百位 : 9
十位 : 9
圆 : 9
角 : 9
分 : 9
D:\test\debug>
```

```
D:\test\debug>3-b3 < 3-b3-01.txt
请输入[0-100亿)之间的数字:
十亿位 : 9
亿位 : 9
千万位 : 9
百万位 : 9
十万位 : 9
万位 : 9
千位 : 9
百位 : 9
十位 : 9
圆 : 9
角 : 9
分 : 9
D:\test\debug>
```



§. 输入输出重定向及管道运算符

3. 用输入/输出重定向测试程序(以浮点数分解3-b3为例)

3. 1. 单次测试

错误和不建议的用法:

```
cmd.exe
Microsoft Windows [版本 10.0.22621.1413]
(c) Microsoft Corporation。保留所有权利。

C:\Windows\System32>
建立cmd快捷方式时，如未做
0.4步骤的“起始位置”设置，则
缺省为“C:\Windows\System32”
```

```
cmd.exe
Microsoft Windows [版本 10.0.22621.1413]
(c) Microsoft Corporation。保留所有权利。

C:\Windows\System32>D:\test\debug\3-b3.exe < 3-b3-01.txt
系统找不到指定的文件。
如果cmd窗口的起始位置不是exe文件所在目录，则虽然可以采用
“鼠标拖曳方式将exe拖入cmd窗口+输入重定向文件名手工输入”运行，
但后面的输入重定向文件会因为不在起始位置目录而报错
```

示例为4个文件，
文件名任意，
存储某4个数据，
其余略

```
cmd.exe
Microsoft Windows [版本 10.0.22621.1413]
(c) Microsoft Corporation。保留所有权利。

C:\Windows\System32>D:\test\debug\3-b3.exe < D:\test\debug\3-b3-01.txt
请输入[0-100亿)之间的数字： 1234567890
十亿位 : 1
亿位 : 2
千万位 : 3
百万位 : 4
十万位 : 5
万位 : 6
千位 : 7
百位 : 8
十位 : 9
圆 : 9
角 : 9
分 : 9

如果cmd窗口的起始位置不是exe文件所在目录，则虽然可以采用
“鼠标拖曳方式将exe拖入cmd窗口+鼠标拖曳输入重定向件名”运行，
但不建议!!!
```

另：后续的作业会给出供参考的exe，如果要求
exe在cmd下运行，同样不建议用鼠标拖曳!!!



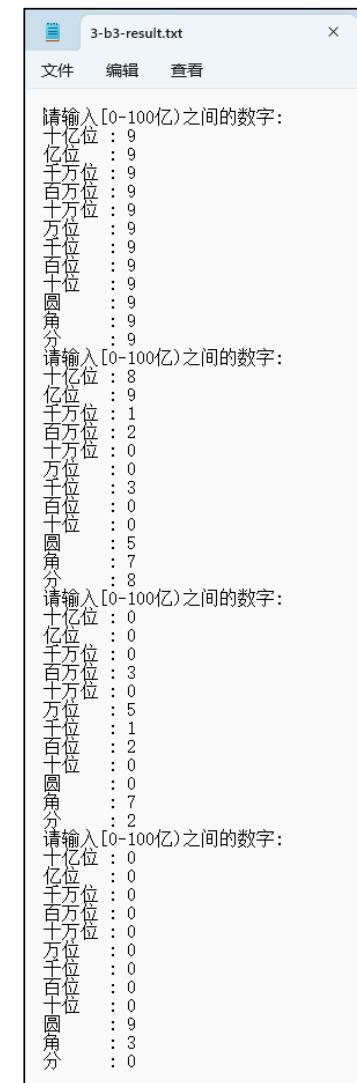
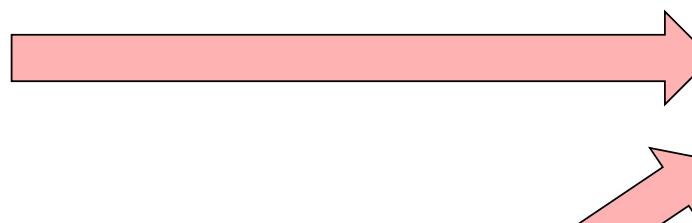
§ . 输入输出重定向及管道运算符

3. 用输入/输出重定向测试程序(以浮点数分解3-b3为例)

3. 2. 在Windows下用输入/输出重定向测试程序批量测试(以3-b3为例)

★ 批量运行：生成3-b4.exe后在cmd下连续运行多个，输出都放入同一个文件中

```
D:\test\debug>3-b3 < 3-b3-01.txt > 3-b3-result.txt  
  
D:\test\debug>3-b3 < 3-b3-02.txt >> 3-b3-result.txt  
  
D:\test\debug>3-b3 < 3-b3-03.txt >> 3-b3-result.txt  
  
D:\test\debug>3-b3 < 3-b3-04.txt >> 3-b3-result.txt  
  
D:\test\debug>  
D:\test\debug>notepad 3-b3-result.txt
```



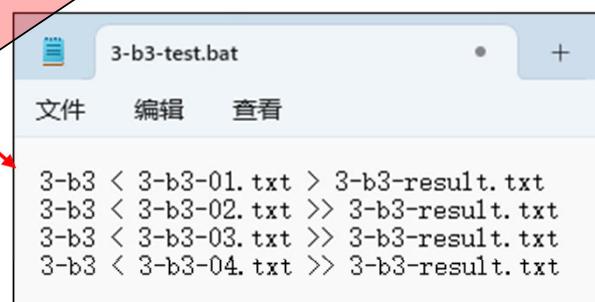
3-b3-result.txt

文件 编辑 查看

请输入 [0-100亿)之间的数字:
十亿位 : 9
亿位 : 9
千万位 : 9
百万位 : 9
十万位 : 9
万位 : 9
千位 : 9
百位 : 9
十位 : 9
圆角分 : 9
分 : 9
请输入 [0-100亿)之间的数字:
十亿位 : 8
亿位 : 9
千万位 : 1
百万位 : 2
十万位 : 0
万位 : 0
千位 : 3
百位 : 0
十位 : 5
圆角分 : 7
分 : 8
请输入 [0-100亿)之间的数字:
十亿位 : 0
亿位 : 0
千万位 : 0
百万位 : 3
十万位 : 0
万位 : 5
千位 : 1
百位 : 2
十位 : 0
圆角分 : 2
分 : 2
请输入 [0-100亿)之间的数字:
十亿位 : 0
亿位 : 0
千万位 : 0
百万位 : 0
十万位 : 0
万位 : 0
千位 : 0
百位 : 0
十位 : 9
圆角分 : 3
分 : 0

★ 用批处理方式批量运行：用notepad编辑3-b3-test.bat并运行

```
D:\test\debug>notepad 3-b3-test.bat  
  
D:\test\debug>  
D:\test\debug>3-b3-test.bat  
  
D:\test\debug>3-b3 0<3-b3-01.txt 1>3-b3-result.txt  
  
D:\test\debug>3-b3 0<3-b3-02.txt 1>>3-b3-result.txt  
    黄色框中为bat自动执行后的显示  
    0< 1>是系统自动添加的，不必纠结  
D:\test\debug>3-b3 0<3-b3-03.txt 1>>3-b3-result.txt  
D:\test\debug>3-b3 0<3-b3-04.txt 1>>3-b3-result.txt  
D:\test\debug>  
D:\test\debug>notepad 3-b3-result.txt
```



3-b3-test.bat

文件 编辑 查看

3-b3 < 3-b3-01.txt > 3-b3-result.txt
3-b3 < 3-b3-02.txt >> 3-b3-result.txt
3-b3 < 3-b3-03.txt >> 3-b3-result.txt
3-b3 < 3-b3-04.txt >> 3-b3-result.txt



§ . 输入输出重定向及管道运算符

4. 管道运算符的使用

4. 1. 管道运算符的基本使用

★ 管道运算符的作用：将前一个程序的输出当做后一个程序的输入

★ 可级联使用

例：两个程序，用管道运算符级联运算

程序1：输入两个正整数，输出最大值 (demo. exe)

程序2：输入一个正整数，输出其平方根 (demo2. exe)

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

int main()
{
    int a, b;

    cin >> a >> b;
    cout << (a>b?a:b) << endl;

    return 0;
}
//注：去除了各种输入输出提示
```

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

int main()
{
    int x;
    cout << "请输入一个整数" << endl;
    cin >> x;
    cout << "平方根是：" << sqrt(x) << endl;

    return 0;
}
```

```
D:\test\Debug>notepad z.dat
D:\test\Debug>type z.dat
10 15

D:\test\Debug>type z.dat | demo
15

D:\test\Debug>type z.dat | demo | demo2
请输入一个整数
平方根是: 3.87298
D:\test\Debug>
```



§ . 输入输出重定向及管道运算符

4. 管道运算符的使用

4. 2. 用工具get_input_data.exe并配合管道运算符测试程序(以3-b3为例)

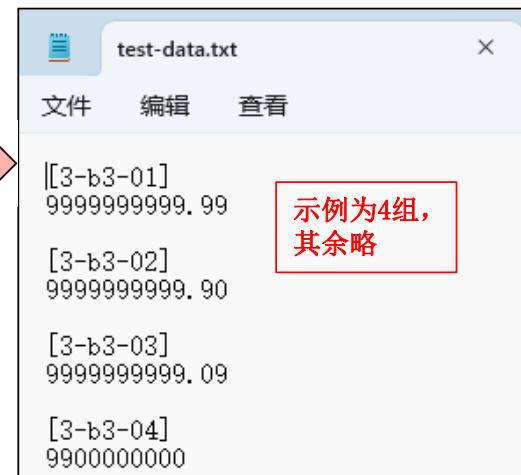
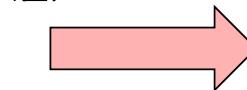
★ 准备工作: 1、将get_input_data.exe放入D:\test\Debug下

2、用notepad编辑test-data.txt文件, 建立32个组,
将要求的32个数据分别放入, 每组一个数据

- 如果放在C:\Windows下, 则更通用
- 认真阅读附件test-data.txt中的说明
- 允许新建, 使用其它文件名

★ 依次运行: 在cmd下用管道运算符依次运行, 观察结果(可多个)

```
D:\test\debug>get_input_data test-data.txt 3-b3-01  
999999999.99  
  
D:\test\debug>get_input_data test-data.txt 3-b3-01 | 3-b3  
请输入[0-100亿)之间的数字:  
十亿位 : 9  
亿位 : 9  
千万位 : 9  
百万位 : 9  
十万位 : 9  
万位 : 9  
千位 : 9  
百位 : 9  
十位 : 9  
圆 : 9  
角 : 9  
分 : 9  
  
D:\test\debug>
```



管道运算符方式

"get_input_data+管道运算符"相比输入重定向的优点:
1、所有数据(某题的若干数据, 若干题的若干数据)可以放在同一个文件中, 方便编辑和维护
2、后期某些特殊情况, 需要用一个程序的输入做另一个程序的输出时, 可以级联



§ . 输入输出重定向及管道运算符

4. 管道运算符的使用

4. 2. 用工具get_input_data.exe并配合管道运算符测试程序(以3-b3为例)

★ 批量运行：在cmd下连续运行多个测试数据，输出都放入同一个文件中

```
D:\test\debug>get_input_data test-data.txt 3-b3-01 | 3-b3 > 3-b3-result.txt
```

```
D:\test\debug>get_input_data test-data.txt 3-b3-02 | 3-b3 >> 3-b3-result.txt
```

```
D:\test\debug>get_input_data test-data.txt 3-b3-03 | 3-b3 >> 3-b3-result.txt
```

```
D:\test\debug>get_input_data test-data.txt 3-b3-04 | 3-b3 >> 3-b3-result.txt
```

```
D:\test\debug>
```

```
D:\test\debug>notepad 3-b3-result.txt
```

```
3-b3-result.txt  
文件 编辑 查看  
请输入[0-100亿)之间的数字:  
十位 : 9  
亿位 : 9  
千万位 : 9  
百万位 : 9  
十万位 : 9  
万位 : 9  
千位 : 9  
百位 : 9  
十位 : 9  
圆角分 : 9  
请输入[0-100亿)之间的数字:  
十位 : 9  
亿位 : 9  
千万位 : 9  
百万位 : 9  
十万位 : 9  
万位 : 9  
千位 : 9  
百位 : 9  
十位 : 9  
圆角分 : 9  
请输入[0-100亿)之间的数字:  
十位 : 9  
亿位 : 9  
千万位 : 9  
百万位 : 9  
十万位 : 9  
万位 : 9  
千位 : 9  
百位 : 9  
十位 : 9  
圆角分 : 9  
请输入[0-100亿)之间的数字:  
十位 : 9  
亿位 : 9  
千万位 : 9  
百万位 : 9  
十万位 : 9  
万位 : 9  
千位 : 9  
百位 : 9  
十位 : 9  
圆角分 : 0
```

★ 用批处理方式批量运行：用notepad编辑3-b4-test-pipe.bat并运行

```
D:\test\debug>notepad 3-b3-test-pipe.bat
```

```
D:\test\debug>3-b3-test-pipe.bat
```

```
D:\test\debug>get_input_data test-data.txt 3-b3-01 | 3-b3 1>3-b3-result.txt
```

```
D:\test\debug>get_input_data test-data.txt 3-b3-02 | 3-b3 1>>3-b3-result.txt
```

黄色框中为bat自动执行后的显示

```
D:\test\debug>get_input_data test-data.txt 3-b3-03 | 3-b3 1>>3-b3-result.txt
```

OK 1是系统自动添加的，不必纠结

```
D:\test\debug>get_input_data test-data.txt 3-b3-04 | 3-b3 1>>3-b3-result.txt
```

```
D:\test\debug>
```

```
D:\test\debug>notepad 3-b3-result.txt
```

思考并查询资料：
不让黄框中的四行在
执行时显示，应在bat的
最开始加什么？

```
3-b4-test-pipe.bat - 记事本  
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)  
get_input_data test-data.txt 3-b4-01 | 3-b4 > 3-b4-result.txt  
get_input_data test-data.txt 3-b4-02 | 3-b4 >> 3-b4-result.txt  
get_input_data test-data.txt 3-b4-03 | 3-b4 >> 3-b4-result.txt  
get_input_data test-data.txt 3-b4-04 | 3-b4 >> 3-b4-result.txt
```



§ . 输入输出重定向及管道运算符

5. 文本比较工具的使用

5. 1. comp命令

要求必须完全相同

5. 2. fc命令

用/w参数可以忽略空格

5. 3. txt_compare工具

自用工具，多种参数组合，自行研究