

**【注意:】**

- 1、本次作业**不允许**使用后续课程的知识点，包括但不限于数组、结构体、类等相关概念!!!
- 2、除明确要求外，已学过的知识中，不允许使用 goto
- 3、cstdio 及 cmath 中的系统函数**可以**直接使用，包括课上未介绍过的，具体可自行查阅相关资料
- 4、除明确要求外，所有 cpp 源程序不允许使用 scanf/printf 进行输入/输出
- 5、多编译器下均要做到“0 errors, 0 warnings”
- 6、部分题目要求 C 和 C++两种方式实现，具体见网页要求
- 7、输出为浮点数且未指定格式的，均要求为 double 型，C++为 cout 缺省输出，C 为 %lf 的缺省输出
- 8、认真阅读格式要求及扣分说明!!!

**【输出格式要求:】**

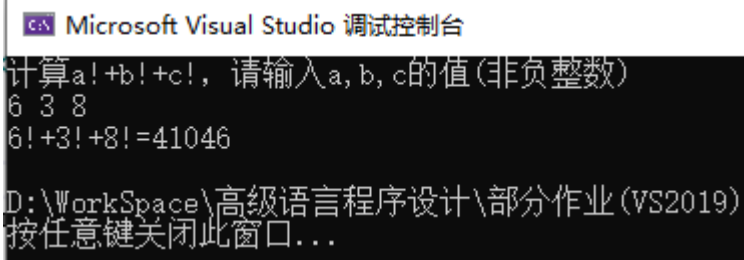
- 1、为方便机器自动判断正确性，作业有一定的输入输出格式要求（但不同于竞赛的无任何提示）
- 2、每个题目见具体说明，必须按要求输入和输出，不允许有偏差
- 3、没有特别说明的情况下，最后一行有效输出的最后有一个 endl

**【本次作业特别要求:】**

- 1、所有程序，除特别要求外，**不允许**出现任何形式的循环（for、while、do-while、if-goto），否则**得分为-20**
- 2、**不允许**使用静态局部变量及全局变量（题目另有说明的**例外**）
- 3、**不考虑**输入错误（目的是为了避免出现循环）
- 4、各种数字/符号的全半角不再详细说明，**以各题的 demo 为准**，要求你的程序与 demo 的输出**重定向结果 comp 一致**（部分 demo 处理了输入错误，作业**不需要**）
- 5、注意：屏幕显示与重定向的结果是不一样的，输出重定向文件不含输入信息
- 6、以上为本次作业的总体要求，若与每个题目的特殊要求冲突，**以每个题目的特殊要求为准**

**补充:**

- 5、求  $a!+b!+c!$  的值，要求用一个函数 fac(n) 求  $n!$ ，main 函数负责输入 abc 的值并输出结果

<p>输出格式要求：三行</p> <p>Line1: 输入提示</p> <p>Line2: 键盘输入的 abc 值</p> <p>Line3: <math>a!+b!+c!=**</math></p>	
--	--

- 【要求:】**
- 1、输入时人工控制 abc 的范围，使  $a!+b!+c!$  不超过 int 的最大表示范围即可
  - 2、提供 4-b5-demo.exe 供参考
  - 3、给出 4-b5.c 基准程序，按要求完成

- 6、用递归法求  $n$  阶 Legendre 多项式的值，递归公式如下：

$$P_n(x) = \begin{cases} 1 & (n=0) \\ x & (n=1) \\ ((2n-1) \cdot x \cdot P_{n-1}(x) - (n-1) \cdot P_{n-2}(x)) / n & (n>1) \end{cases}$$

输出格式要求：三行

Line1: 输入提示

Line2: 键盘输入的 x 和 n 值

Line3:  $\text{legendre}[n](x)=**$

Microsoft Visual Studio 调试控制台

计算legendre, 请输入x和n的值

1.9 2

legendre[2](1.9)=4.915

D:\WorkSpace\高级语言程序设计\部分作业  
按任意键关闭此窗口...

- 【要求:】
- 1、x 为浮点数, n 为非负整数
  - 2、输出为 double 型的正常格式即可, 不需要任何格式控制
  - 3、给出 4-b6-demo.exe 供参考
  - 4、给出 4-b6.cpp 基准程序, 按要求完成

- 7、用递归法将一个整数 n 按位分解后输出, 整数为 int 型 (**程序不允许用 64 位整数**), 分解后的每位以字符方式输出 (即输出形式为 `cout << char(...)`), 中间用空格分隔, 负数还需要输出负号

输出格式要求：三行

Line1: 输入提示, 任意

Line2: 键盘输入的 n 值

Line3: 转换后的输出(最后 1 位后面有空格)

Microsoft Visual Studio 调试控制台

请输入一个整数

2147483648

2 1 4 7 4 8 3 6 4 7

Microsoft Visual Studio 调试控制台

请输入一个整数

-2147483648

- 2 1 4 7 4 8 3 6 4 8

- 【要求:】
- 1、给出 4-b7-demo.exe 供参考
  - 2、给出 4-b7.cpp 基准程序, 按要求完成
  - 5、特别提示: -2147483648 的处理可能会与其它值不同, **允许**做特殊判断, 但**不允许**直接采用 `cout << "- 2 1 4 7 4 8 3 6 4 8"` 或其它形式的打表输出

- 8、题目同 4-b7, 仍然用递归法完成, 要求改为逆序输出, 其余要求及提示也同 4-b7

输出格式要求：同 4-b7

Microsoft Visual Studio 调试控制台

请输入一个整数

-2147483648

8 4 6 3 8 4 7 4 1 2 -

Microsoft Visual Studio 调试控制台

请输入一个整数

2147483647

7 4 6 3 8 4 7 4 1 2

- 【要求:】
- 1、给出 4-b8-demo.exe 供参考
  - 2、给出 4-b8.c 基准程序, 按要求完成
  - 3、特别提示: -2147483648 的处理可能会与其它值不同, **允许**做特殊判断, 但**不允许**直接采用 `cout << "- 2 1 4 7 4 8 3 6 4 8"` 或其它形式的打表输出

- 9、用递归法求 Fibonacci 数列, 要求函数参数是要求的项数, 返回为数列中该项的值

- 【要求:】
- 1、为避免歧义, 统一约定为  $F(1)=1, F(2)=1, F(n)=F(n-1)+F(n-2) \quad n \geq 3$
  - 2、不考虑运算结果溢出 int 上限的问题, 将测试项数人为控制在 [1..46] 即可
  - 3、为什么项数越大速度越慢, 请仔细思考并从中理解递归的执行过程及执行次数
  - 4、给出项数为 1-46 时递归函数的执行次数并给出前后项的递推公式(pdf 文档形式)
  - 5、给出 4-b9-demo.exe 供参考
  - 6、源程序 4-b9.cpp 已部分给出, 只允许修改首行及 fibonacci 函数, 其余不准改动
  - 7、本作业的输出重定向结果比对只看第 1~2 行是否匹配 (可使用 fc 命令比对且人工保证第 1-2 行匹配即可, 检查作业时会有另外的方法)

10、 写一个函数，求某个十进制正整数是否某个基数的幂

- 【要求:】1、函数形式定为 `int is_power(int num, int base)`，`num` 为十进制正整数，`base` 为基数（2 以上的正整数），返回值 1：是/0：否；要求以**递归函数**形式实现
- 2、`main` 函数负责输入十进制数和基数，并打印返回结果
- 3、参考测试数据如下

num	base	返回	num	base	返回	num	base	返回	num	base	返回
2048	2	1	24	2	0	729	9	1	243	9	0
81	3	1	54	3	0	1000	10	1	2000	10	0
125	5	1	100	5	0	4096	16	1	512	16	0
7776	6	1	108	6	0	1	2	1	1	8	1
2401	7	1	98	7	0	1	10	1	1	16	1
512	8	1	1024	8	0	注意：1 是任何基数的 0 次幂					

输出格式要求：三行

Line1: 输入提示

Line2: 键盘输入的 `num` 和 `base` 的值

Line3: `num` 是/不是 `base` 的幂

Microsoft Visual Studio 调试控制台

```
请输入整数num及基数base
81 3
81是3的幂
```

Microsoft Visual Studio 调试控制台

```
请输入整数num及基数base
96 3
96不是3的幂
```

【要求:】1、给出 4-b10-demo.exe 供参考

2、给出 4-b10.cpp 基准程序，按要求完成

11、 写一个程序，输入一个大写字母，打印从 A~该字母的菱形

输出格式要求：多行

Line1: 输入提示

Line2: 键盘输入的大写字母

Line3~: 输出的菱形（具体如图示）

Microsoft Visual Studio 调试控制台

```
请输入结束字符(A~Z)
F
=====
菱形(F->A)
=====
  A
 BAB
CBABC
DCBABCD
EDCBABCDE
FEDCBABCDEF
EDCBABCDE
DCBABCD
CBABC
 BAB
  A
```

【要求:】1、==的宽度与菱形的宽度相同（上例为 11 个=）

2、给出 4-b11-demo.exe 供参考

3、源程序 4-b11.cpp 已部分给出，按限制要求完成即可

- 12、 用递归法打印汉诺塔（Hanoi Tower）的移动步骤，汉诺塔的描述如下：
- 有三根柱子，编号分别为 ABC
  - 初始状态，在某根柱子（起始柱）上有 n 个大小不等的盘子从小到大依次叠放
  - 先要求，将起始柱的所有盘子都移动到另一个柱子（目标柱）上，移动规则如下
    - 每次只允许移动一个盘子
    - 任何时候，不允许大盘压小盘
    - 移动过程允许在三根柱子之间任意进行（第三根柱子称为中间柱）

现要求：键盘输入汉诺塔的层数、起始柱、目标柱，打印整个移动过程

输入格式要求：多行

Line1: 输入层数提示

Line2: 键盘输入的层数

Line3: 输入起始柱提示

Line4: 键盘输入的起始柱

Line5: 输入目标柱提示

Line6: 键盘输入的目标柱

输出格式要求：多行

Line1: 输出首行提示“移动步骤为:”

Line2~: 每步移动步骤

(盘号# 起始柱-->目标柱)

盘号宽度为 2，右对齐

```
D:\Workspace\高级语言程序设计
请输入汉诺塔的层数(1-16)
3
请输入起始柱(A-C)
A
请输入目标柱(A-C)
C
移动步骤为:
1# A-->C
2# A-->B
1# C-->B
3# A-->C
1# B-->A
2# B-->C
1# A-->C
```

注：本题**需要**考虑输入错误，处理规则约定如下

- 层数/起始柱/目标柱分三次输入
- 每次输入后无论正确与否，清空缓冲区内其余内容
- 考虑执行效率问题，层数限定在 1-16 之间
- 起始/目标柱的字母大小写均可，要检查正确性（仅 A~C）以及是否重合
- 层数及起始/目标柱的处理**允许**定义其它函数，这些函数中**允许**使用循环

【要求:】1、给出 4-b12-demo.exe 供参考

2、给出 4-b12.cpp 基准程序，按要求完成

- 13、 题目同 4-b12（汉诺塔），要求记录移动步数并打印出来

输入格式要求：多行

同 4-b12

输出格式要求：多行

Line1: 输出首行提示“移动步骤为:”

Line2~: 每步移动步骤

(编号: 盘号# 起始柱-->目标柱)

编号宽度 5 位，右对齐，英文冒号

编号与盘号间 1~2 个空格

->如果盘号为 1 位，则 2 个空格

->如果盘号为 2 位，则 1 个空格

```
510: 2# C-->B
511: 1# A-->B
512: 10# A-->C
513: 1# B-->C
514: 2# B-->A
```

```
D:\Workspace\高级语言程序设计
请输入汉诺塔的层数(1-16)
3
请输入起始柱(A-C)
b
请输入目标柱(A-C)
c
移动步骤为:
1: 1# B-->C
2: 2# B-->A
3: 1# C-->A
4: 3# B-->C
5: 1# A-->B
6: 2# A-->C
7: 1# B-->C
```

【要求:】1、本题为两个小题，分别采用全局变量(C)和静态局部变量(C++)来记录移动步数

2、给出 4-b13-demo.exe 供参考

3、给出 4-b13.cpp/4-b13.c 基准程序，按要求完成（两个小题输出相同）

### 【测试数据:】

附件的 test-data.txt 给出了本次的部分测试数据供参考，具体请自行阅读

### 【编译器要求:】

		编译器VS	编译器Dev	编译器Linux
4-b5.c	阶乘累加（C方式）	Y	Y	Y
4-b6.cpp	Legendre多项式	Y	Y	Y
4-b7.cpp	整数分解-正序	Y	Y	Y
4-b8.c	整数分解-逆序（C方式）	Y	Y	Y
4-b9.cpp	斐波那契数列(递归实现)	Y	Y	/
4-b10.cpp	判断是否为幂	Y	Y	Y
4-b11.cpp	输出字母菱形	Y	Y	Y
4-b12.cpp	汉诺塔-基本移动	Y	Y	Y
4-b13.c	汉诺塔-步数记录-全局变量（C方式）	Y	Y	Y
4-b13.cpp	汉诺塔-步数记录-静态局部变量	Y	Y	Y
4-b9.pdf	斐波那契数列(递归次数分析)	/	/	/

注：Linux 仅限计算机拔尖班（10069201/10071701）同学，其它班级忽略即可

### 【作业要求:】

- 1、11月4日前网上提交本次作业
- 2、每题所占平时成绩的具体分值见网页
- 3、超过截止时间提交作业会自动扣除相应的分数，具体见网页上的说明