

专题：月光族

【应用背景】

小明是个月光族，而且到了月底也总是想不清楚钱是怎么花的。因此希望通过记账来分析自己的支出情况。目前基本的想法是把支出分为“基本支出”（包括日常衣食住行水电燃气等必需的支出）、“学习支出”（包括学费和学习相关的书籍和设备的购买等）、“医药支出”、“休闲支出”（休闲饮食、游戏、娱乐旅游、聚会等）、“其它支出”（无法归入上述各类的支出）。每天在自己的本子上记录当天的支出，一笔支出一条记录，需要时就检查记录，进行分析。

各问摘要：

- 第1问，专题理解，程序形式回答基础问题。
- 第2问，程序调试，输入并统计各支出类别及金额。
- 第3问，规范处理，程序代码的规范化处理。
- 第4问，数据文件，设计函数将输入数据追加到文件。
- 第5问，配置统计，函数实现类别配置与支出统计，使用结构体指针数组。
- 第6问：应用提升，增设函数实现数据输入和支出排序功能。

【第1问，专题理解】

按照应用背景，回答以下问题。同时编写一个程序，直接以printf语句输出问题的回答，输出格式形如“(1)... (2)... (3)... (4)...”，保存新程序为C:\KS\A_1_1.c。

- (1) 如果本月总共有3笔“休闲支出”，分别为100元、50元、50元，则本月的“休闲支出”总和为_____。
A. 50元 B. 100元 C. 150元 D. 200元
- (2) 由于支出金额单位为元，而实际每次支出金额会有角甚至分的零头，为了记账精确，支出数据的范围应为_____。
A. 自然数 B. 实数 C. 虚数 D. 整数
- (3) 如果要用文件保存支出记录，未来输入新的支出以追加方式加入文件，则用函数“fopen”打开文件时的选项应该是_____。
A. “r+” B. “w” C. “a” D. “w+”
- (4) 在用循环结构把本月各笔支出进行累加时，_____。
A. 只能使用while循环 B. 只能使用do-while循环
C. 只能使用for循环 D. 可以用任一种合法的循环

【第2问，程序调试】

下列分类统计支出的程序中包含4个错误，请按题中的功能要求，打开C:\KS\A_1_2.c，调试并修改该程序(在所修改语句后加“/*_*/”或“//_”作为标记)，使其运行能得到正确的结果。修改后的程序仍保存为C:\KS\A_1_2.c。

运行示例：

```

1 50.55
1 50
2 90.01
1 60.01
-1 0
基本支出: 0.00
学习支出: 160.56
医药支出: 90.01
休闲支出: 0.00
其它支出: 0.00

```

【第3问，规范处理】

进一步修改上述程序，实现以下附加要求，将修改后的程序另保存为C:\KS\A_1_3.c。

(1) 边界检测、友好交互

- ① 输入数据错误时发出提示。
- ② 如果可能，在用户输入之前，输出信息提示用户。
- ③ 如果可能，输出数据的前后加上名称、单位等。

(2) 更好的代码风格

- ① 赋值运算符两边至少一个空格。
- ② if/for/while/switch之后至少一个空格。
- ③ 函数参数之间（逗号之后）至少一个空格。
- ④ 表达式各主要项之间空格隔离。
- ⑤ 预定义时最好加括号，如：#define PRICE (0.6)
- ⑥ 代码的第一行应为注释，描述程序名称、程序功能、编写日期等。
- ⑦ 使用有意义的变量名称，变量声明时，注释说明每一个变量的功能/作用。
- ⑧ 同一系列的一组同类型变量用数组代替。

运行示例：

```

请输入支出类型和支出金额
1 50.55
请输入支出类型和支出金额
1 50
请输入支出类型和支出金额
5 55.55
错误的支出类型
请输入支出类型和支出金额
2 90.01
请输入支出类型和支出金额
1 60.01
请输入支出类型和支出金额
-1 0
基本支出: 0.00元
学习支出: 160.56元
医药支出: 90.01元
休闲支出: 0.00元
其它支出: 0.00元

```

【第4问，数据文件】

编程要求：连续输入支出类型和支出金额，直至输入类型为-1，输入的数据追加到文本文件 booking.txt 中。编写后的程序保存为 C:\KS\A_1_4.c。要求设计并使用以下函数：

```
int input(void);
```

函数 input 运行时以追加方式打开文件 booking.txt，如果无法打开，则以新建方式打开该文件，如果新建方式仍然无法打开，则报告“无法创建 booking.txt”，退出程序运行。如果能够打开文件 booking.txt，则接受从键盘输入的支出记录，并向文件中加入输入的记录。文件 booking.txt 的结构如下：

```
1 50.55
1 50
2 90.01
1 60.01
```

运行示例：

输入：

请输入支出类型和支出金额

```
1 50.55
```

请输入支出类型和支出金额

```
1 50
```

请输入支出类型和支出金额

```
2 90.01
```

请输入支出类型和支出金额

```
1 60.01
```

请输入支出类型和支出金额

```
-1 0
```

输出：

运行后文件 booking.txt 内容如下：

```
1 50.55
1 50.00
2 90.01
1 60.01
```

【第5问，配置统计】

编程要求：打开文件 C:\KS\A_1_5.c，按题中的功能要求在其中完成以下2个函数，调试程序得到正确的结果。编写后的程序仍然保存为 C:\KS\A_1_5.c。

```
int configType(char spendType[TYPENUM][20]);
```

```
int summarize(struct sumStruct sums[TYPENUM]);
```

程序功能：

(1) 函数 configType 从文件 spendingtype.txt 中读入支出分类信息，要求：

- ① 如果无法打开这个文件，则报告“找不到 spendingtype.txt 文件”，然后结束程序运行。
- ② 如果能够正确打开文件 spendingtype.txt，则把读入的每一个支出分类名称依次保存在

各个字符数组spendType[i]中。

③ 返回文件spendingtype.txt中实际给出的支出分类数。

④ 规定用字符串表示的每一个支出分类的名称(长度不超过20个字符)。同时用常数TYPENUM定义数组spendType的最大行数,即最多允许多少个支出分类。

⑤ 文件spendingtype.txt中第一个整数n表示当前实际的支出分类数,规定n不能大于TYPENUM,后面n行依次为对应分类的字符串名称,该文件内容格式示例如下(如果C:\KS\下没有该文件,请考生自行创建):

```
5
基本支出
学习支出
医药支出
休闲支出
其它支出
```

(2) 为了更好关联支出类型和支出金额,将统计结果用结构体数组表达,同时为避免重复存储分类名称,引入指针指向可共享的字符串。

定义结构体如下:

```
struct sumStruct {
    char *typePtr;
    double amount;
};
```

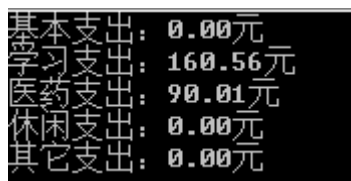
(3) 函数 summarize 从文件booking.txt中读取数据进行统计,如果无法打开则报告“没有找到数据文件”。

运行示例:

如果文件booking.txt中的内容如下:

```
1 50.55
1 50
2 90.01
1 60.01
```

运行结果如下:



```
基本支出: 0.00元
学习支出: 160.56元
医药支出: 90.01元
休闲支出: 0.00元
其它支出: 0.00元
```

【第6问: 应用提升】

编程要求: 打开文件C:\KS\A_1_6.c, 请按题中的功能要求在其中完成以下5个函数, 调试程序得到正确的结果。编写后的程序仍然保存为C:\KS\A_1_6.c。

```
int configType(char spendType[TYPENUM][20]);
int summarize(struct sumStruct sums[TYPENUM]);
int processInput(struct sumStruct sums[TYPENUM], int n);
int sort(struct sumStruct *p[], int realTypeNum);
```

```
int main();
```

程序功能:

- (1) 结构体sumStruct、函数configType和summarize沿用第5问（参考第5问）。
- (2) 完成函数processInput，用函数configType返回的支出分类数作为参数n的值传入。
 - ① 以追加方式打开文件booking.txt，如果无法打开，则以新建方式打开该文件，如果新建方式仍然无法打开，则报告“无法创建booking.txt”，然后退出程序运行。
 - ② 从键盘输入新的支出记录，对输入的每一笔支出，加入文件booking.txt中，并把该笔支出累加到sums数组的对应项中。输入时如果支出类型不对则提示支出类型不对，并要求继续输入。
- (3) 完成排序函数sort，可以使用任意一种排序算法，对通过参数传入的p数组所指向的各项支出的统计结果按金额降序排序。
- (4) 编写主函数main调用上述函数，完成以下步骤。
 - ① 调用 configType 取得支出分类的名称和分类数；
 - ② 使结构体数组 sums 中各元素的 typePtr 成员指向对应的 spendType 数组元素；
 - ③ 调用函数 summarize 对现有booking.txt文件中的支出进行统计；
 - ④ 调用函数 processInput 对追加的支出进行记录和统计；
 - ⑤ 定义结构体指针数组p，并使每一项p[i]指向对应的sums[i]；
 - ⑥ 调用函数sort对p所指的支出降序排序；
 - ⑦ 输出各类支出的名称及金额。

运行示例1:

在文件booking.txt不存在时，运行效果如下:

```
分类名称: 基本支出: 分类编号: 0
分类名称: 学习支出: 分类编号: 1
分类名称: 医药支出: 分类编号: 2
分类名称: 休闲支出: 分类编号: 3
分类名称: 其它支出: 分类编号: 4
“-1 0”表示输入结束
没有找到数据文件
请输入支出类型和支出金额
1 50.55
请输入支出类型和支出金额
1 50
请输入支出类型和支出金额
2 90.01
请输入支出类型和支出金额
1 60.01
请输入支出类型和支出金额
-1 0
=====以下为统计得到的各项支出累计=====
分类名称: 学习支出: 累计支出: 160.56元
分类名称: 医药支出: 累计支出: 90.01元
分类名称: 基本支出: 累计支出: 0.00元
分类名称: 休闲支出: 累计支出: 0.00元
分类名称: 其它支出: 累计支出: 0.00元
```

运行示例2:

运行示例1成功运行后，紧接着再次运行，效果如下:

```

分类名称: 基本支出: 分类编号: 0
分类名称: 学习支出: 分类编号: 1
分类名称: 医药支出: 分类编号: 2
分类名称: 休闲支出: 分类编号: 3
分类名称: 其它支出: 分类编号: 4
“-1 0”表示输入结束
请输入支出类型和支出金额
-1 0
=====以下为统计得到的各项支出累计=====
分类名称: 学习支出: 累计支出: 160.56元
分类名称: 医药支出: 累计支出: 90.01元
分类名称: 基本支出: 累计支出: 0.00元
分类名称: 休闲支出: 累计支出: 0.00元
分类名称: 其它支出: 累计支出: 0.00元

```

运行示例3:

运行示例2成功运行后，紧接着再次运行，效果如下:

```

分类名称: 基本支出: 分类编号: 0
分类名称: 学习支出: 分类编号: 1
分类名称: 医药支出: 分类编号: 2
分类名称: 休闲支出: 分类编号: 3
分类名称: 其它支出: 分类编号: 4
“-1 0”表示输入结束
请输入支出类型和支出金额
0 500
请输入支出类型和支出金额
6 1000
错误的支出类型
请输入支出类型和支出金额
3 1000
请输入支出类型和支出金额
4 100.01
请输入支出类型和支出金额
-1 0
=====以下为统计得到的各项支出累计=====
分类名称: 休闲支出: 累计支出: 1000.00元
分类名称: 基本支出: 累计支出: 500.00元
分类名称: 学习支出: 累计支出: 160.56元
分类名称: 其它支出: 累计支出: 100.01元
分类名称: 医药支出: 累计支出: 90.01元

```