写入数据操作函数	说　　明
fputc(int ch,FILE *fp)	写一个字符到文件中
fputs(char *str , FILE *fp)	写一个字符串到文件中
fprintf(FILE *fp,char *format,arg_list)	将变量数据用指定的格式写入文件中

（4）对文件进行写数据举例

```
//定义 writedata 函数
void writedata()
{
    //w 代表对文件进行写操作
    FILE *fp = fopen("test.txt" , "w");
    //写一个字符到文件中
    fputc('a' , fp);
    //写一个字符串到文件中
    char *s="可以写文件了，我真的非常高兴！";
    fputs(s , fp);
    //按格式 %s %s %d 写数据到文件中
    fprintf(fp , "%s %s %d" , "张三" ,"男" ,18);
    //关闭文件
    fclose(fp);
}
int main()
{
    //调用函数
    writedata();
    return 0;
}
```

说明：

对文件写操作单独编写 writedata 函数，下面进行详细介绍。

① FILE *fp = fopen("test.txt" , "w");

这行代码可以“覆盖”的方式打开项目中的 test.txt 文件。

② fputc('a' , fp);

这行代码可向 test.txt 文件中写一个'a'字符。

③ char *s="可以写文件了，我真的非常高兴！";

fputs(s , fp);

这两行代码就是将 s 这个字符串的内容写到 test.txt 文件中。

④ fprintf(fp , "%s %s %d" , "张三" ,"男" ,18);

这行代码可按格式 **%s %s %d** 将"张三"、"男"、18 这三个数据写到 test.txt 文件中。

学习活动 3　制定方案

实现本任务方案

● 实现思路

通过对本任务的分析及相关知识学习，制定方案如下：

（1）定义一个结构体，包含书名、作者、定价三个成员；

（2）创建一个新的函数 saveData()，接收参数为书本信息结构体数组，返回一个整型数据。

（3）在 main()中实现对 saveData()的调用。

● 实现步骤

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 writeData。

（2）在项目的 main.c 文件中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 writeData。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()之前创建描述书本信息的结构体代码，其代码如下：

```
//定义书本信息结构体
struct book
{
    char title[50];   //书名，一个字符串
    char author[50];  //作者，一个字符串
    float value;   //定价，一个浮点数
};
```

（4）编写 saveData()。

在定义好的结构体下方创建该函数。

接收参数：书本信息结构体数组；

函数功能：将接收参数传入结构体中的书本信息保存到文件中；

函数返回：返回一个整型数据（1 为成功，0 为失败），其代码如下：

```
//写文件函数
int   saveData(struct   book D[4] )
{
    int jieguo=1;
    FILE *fp=fopen("bookinfo.txt" , "a"); //a :追加方式
    if(!fp){
        jieguo=0;   //操作文件出错
    }
```

```
    //循环一次写一本书的信息
    int i;
    for(i=0;i<4;i++)   //采用格式化写入函数
    {
        fprintf(fp,"%s %s %.1f \n",D[i].title, D[i].author, D[i].value);
    }
    //关闭文件
    fclose(fp);
    //函数返回
    return jieguo;
}
```

（5）编写 main()实现调用。

在完成以上操作后，编写 main()中的代码，实现对 saveData()的调用，以完成本任务。

```
//主函数中实现数据整理有调用函数实现写数据
int main()
{
    //定义 1 个结构体数组
    struct book bookList[4]=
    {
        {"红楼梦","曹雪芹",38.6},
        {"三国演义","罗贯中",54.3},
        {"水浒传","施耐庵",33.6},
        {"西游记","吴承恩",49.4}
    };
    //调用函数将结构体数据写入文件保存
    int   fankui=saveData(bookList);
    if(fankui==1){
        printf("数据写入成功！");
    }else{
        printf("数据写入失败！");
    }
    return 0;
}
```

实现本任务后完整的代码如下。

```
//定义书本信息结构体
struct book
{
    char title[50];   //书名，一个字符串
    char author[50];   //作者，一个字符串
    float value;   //定价，一个浮点数
};
//写文件函数
int   saveData(struct   book D[4] )
{
```

```
    int jieguo=1;
    FILE *fp=fopen("bookinfo.txt" , "a");   //a：追加方式
    if(!fp){
        jieguo=0;   //操作文件出错
    }
    //循环一次写入一本书的信息
    int i;
    for(i=0;i<4;i++)   //采用格式化写入函数
    {
        fprintf(fp,"%s %s %.1f \n",D[i].title, D[i].author, D[i].value);
    }
    //关闭文件
    fclose(fp);
    //函数返回
    return jieguo;
}
//在主函数中实现数据整理，并调用函数实现写数据
int main()
{
    //定义 1 个结构体数组
    struct book bookList[4]=
    {
        {"红楼梦","曹雪芹",38.6},
        {"三国演义","罗贯中",54.3},
        {"水浒传","施耐庵",33.6},
        {"西游记","吴承恩",49.4}
    };
    //调用函数将结构体数据写入文件保存
    int   fankui=saveData(bookList);
    if(fankui==1){
        printf("数据写入成功！");
    }else{
        printf("数据写入失败！");
    }
    return 0;
}
```

补充说明：

（1）打开文件说明。

FILE *fp=fopen("bookinfo.txt","a");

以 a（追加）方式打开 bookinfo.txt 文件，这里为什么要使用追加方式呢？

大家想一下，结构体数组中存放的是四大名著，也就是 4 本书的信息。写文件应循环 4 次，第 1 次循环后，将第 1 本书的 3 个信息写到文件；第 2 次循环时写第 2 本书的信息，这里如果使用 w（覆盖）就会把之前写的数据覆盖，所以这里必须使用 a（追加）方式。

（2）写文件说明。

写文件的代码如下：

```
for(i=0;i<4;i++)   //采用格式化写入函数
    {
        fprintf(fp,"%s %s %.1f \n",D[i].title, D[i].author, D[i].value);
    }
```

这里使用循环实现对 4 本书信息的写入过程。

因为函数接收的参数是一个 book 结构体数组 D，所以开始循环时，从结构体数组的第 0 个元素开始，依次完成对 4 本书信息的写入操作。

学习活动 5　测试验收

任务测试验收单

● **实现效果**

编写 C 语言程序，实现对文件的写入操作。

按制定的方案进行任务实现，在正确的情况下，其效果如图 8.2 所示。

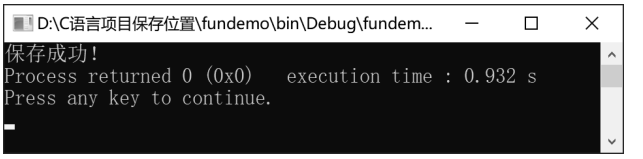


图 8.2　任务运行的效果

文件中写入数据的效果如图 8.3 所示。

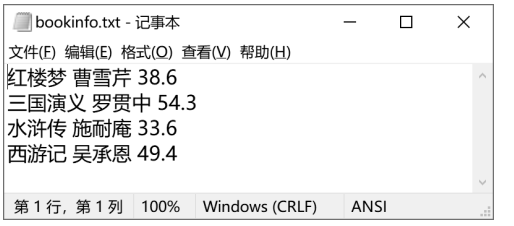


图 8.3　数据写入文件的效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	开发环境安装与置换情况					
3	掌握知识的情况					
4	程序运行情况					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

本任务实现将四大名著信息的结构体数据，保存到一个文本文件中。

● **技术层面**

数据写入文本文件的操作流程。

在 C 语言中写入数据的相关函数。

● **课程思政**

同学们掌握了 C 语言文件操作中写文件的相关知识。

本模块以“日出而作，日入而息”作为开场，引出对应的读取文件数据和将数据写入文件中的过程。“日出而作，日入而息”呈现出的是一幅自力更生的农作场景，所以希望同学们能够传承勤劳的优良传统，做一个自力更生的人。

● **教学拓展**

通过本任务的学习，同学们对写数据到文本文件的过程有了一定的了解，试着优化本任务，让其完成不止写入 4 本书的功能。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

本任务完成了数据的写入

下一回：起床了都出来露个脸

（读取文件中数据）

任务 2　起床了都出来露个脸

目标描述

任务描述
● 编写程序实现 将上次任务写入文件的四大名著信息读出来并显示。 读文件、显示书本信息功能，单独以函数实现。 ● 技术层面 掌握读取文本文件数据的操作流程。 掌握读取文本文件数据的相关知识。 ● 课程思政 劳动教育。 自力更生。

学习活动 1　接领任务

领任务单
● 任务确认 编写 C 语言程序，将上次任务写入文件的四大名著信息读取出来。 具体要求如下： （1）程序能正确将文件中的数据读取，并显示在界面上； （2）从文件中读取数据功能，单独以自定义函数实现； （3）显示书本信息功能，单独以自定义函数实现； （4）掌握 C 语言代码的使用规范（变量命名及注释说明）； （5）程序能正确运行，并具有可扩展性。 ● 确认签字

学习活动 2　分析任务

编写 C 语言程序，实现将上次任务写入文件中的四大名著信息读取出来，实现读文件的功能。

知识学习：C 语言读取文件的操作

知识学习

学习笔记

文件读取操作的流程和写文件是一样的。先把文件打开，然后进行读取操作，最后关闭文件。

1．对文件进行读取操作的函数

实现对打开的文件进行读取操作的函数如下：

读取数据操作函数	说　　明
fgetc (FILE *fp)	从文件中读取一个字符
fgets(char *str , int n , FILE *fp)	从文件中读取一个指定长度的字符串
fscanf(FILE *fp,char *format,arg_list)	格式化读取文件中数据，并存放于指定的变量参数中

说明：

文件的读取函数与写入函数是一一对应的。

另外，与读取文件数据相关的还有一个特别的函数，即 feof()，它是文件尾函数。

这个函数返回值为逻辑值，如果到文件尾则返回真（1），否则返回假（0）。所以，利用这个函数可判断数据是否已经读取完了。如果 feof()返回为真，则说明文件中的数据已全部读取完。

2．对文件进行读取数据举例

```
/*  读文件示例*/
void readdata()
{
    FILE *fp = fopen("test.txt","r");    //r 只读操作
    //读取一字节数据
    char a;
    a = getc(fp);
    printf("%c\n",a);
    //读取一个字符串
    char s[30];
    fgets( s , 31, fp);    //读取 30 个字符
    printf("%s\n",s);
    //按指定格式读取字符
    char sex[10]; char name[20]; int age;
    fscanf(fp,"%s %s %d",&name,&sex,&age); printf("%s\t%s\t%d",name,sex,age);
    //关闭文件
    fclose(fp);
}

int main()
{
    //调用函数
    readdata();
    return 0;
}
```

说明：

（1）a = getc(fp);从 test.txt 文件中读取一个字符，并赋值给 a 变量。

（2）fgets(s , 31, fp);从 test.txt 文件中读取 30 个字符，因为上一行代码已读了一个字符，所以这里从第 2 个字符开始读 30 个，即为 31。

（3）fscanf(fp,"%s %s %d",&name,&sex,&age);从 test.txt 文件中，以"%s %s %d"格式读取 3 个数据，并分别赋值给 name、sex、age 3 个变量。

特别说明：本任务要基于上次任务写入的数据。

学习活动 3　制定方案

实现本任务方案

● **实现思路**

通过对本任务的分析及相关知识学习，制定方案如下。

（1）定义一个结构体，包含书名、作者、定价三个成员。

（2）调用两个函数，

readData()用于从文件中读取数据，将数据保存到结构体中。

showData()将返回的结构体数组中的数据显示出来。

（3）在 main()中实现对 showData()的调用，如图 8.4 所示。

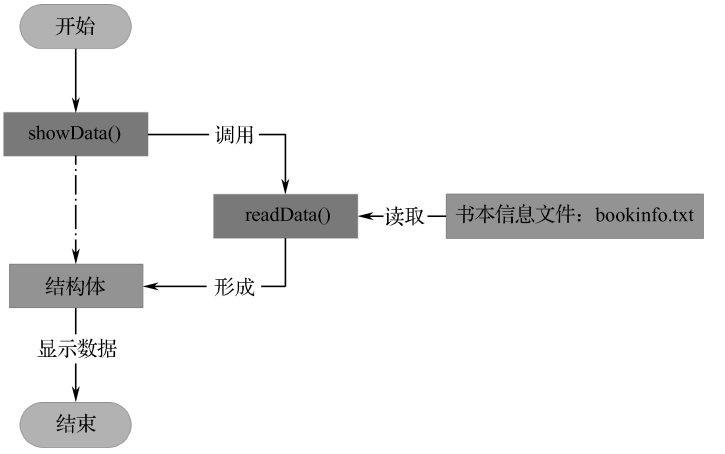


图 8.4　对 showData()的调用流程

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 readData。

（2）在项目的 main.c 文件中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● 实现代码

（1）打开 CodeBlocks 软件，创建一个新的控制台项目，项目名称输入为 readData。

（2）打开项目中的 main.c 文件，进入编辑界面。

（3）在 main()之前创建结构体及结构体全局变量，其代码如下。

```
//定义书本信息结构体
struct book
{
    char title[50];   //书名，一个字符串
    char author[50];   //作者，一个字符串
    float value;   //定价，一个浮点数
};

//定义一个全局结构体数组
struct book BOOKM[4];
```

（4）编写 readData()。

从文件中读取数据，并直接保存到 BOOKM 中，其代码如下：

```
/* 功能：从文件中读取数据 */
void   readData()
{
    FILE *fr = fopen("bookinfo.txt","r");
    if(fr == NULL) {
        return;
    }
    //将文件中的数据写入结构体数组中
    int i=0;   //记录书本个数
    while(!feof(fr)) {
        //按指定格式读取数据，并保存到结构体数组中
        fscanf(fr,"%s %s %f\n",
                &BOOKM[i].title ,
                &BOOKM[i].author ,
                &BOOKM[i].value);
        i++;
    }
    fclose(fr);   //关闭文件
}
```

（5）编写 showData()，并实现显示。

将保存在 BOOKM 中的数据显示在界面中。

```
/* 功能：显示书本信息 */
void showData()
{
```

```
        printf("**************** 书本信息 ****************\n\n");
        printf("书名\t\t     作者\t\t     单价\n----------------------------------------\n");

        //调用函数，实现从文件中读取书本信息
        readData();

        //显示书本信息
        int i;
        for(i=0; i<4; i++)
        {
            printf("%s    \t     %s \t     %.1f \n",
    BOOKM[i].title,BOOKM[i].author,BOOKM[i].value);
        }
        printf("----------------------------------------\n");
    }
```

（6）在 main()中调用，实现本任务。

在 main()中调用 showData()，以完成本任务。

```
    //主函数中实现数据整理，并调用函数实现写数据
    int main()
    {
        //调用显示数据函数
         showData();
         return 0;
    }
```

补充说明：

本任务实现了以下两个自定义函数：

① void readData()：从文件中读取数据。

② void showData()：显示书本信息。

（1）全局结构体数组。

```
struct book BOOKM[4];   //定义一个全局结构体数组
```

因本任务要将文件中的四大名著信息读取出来，所以这里定义了一个全局的 book 结构体数组，数据读取出来就保存到这个结构体数组中，然后就可以直接从这个结构体数组中取数进行显示。

（2）读取数据。

读取文件中数据的代码如下：

```
while(!feof(fr)) {
            //按指定格式读取数据，并保存到结构体数组中
            fscanf(fr,"%s %s %f\n",
        &BOOKM[i].title ,
        &BOOKM[i].author ,
        &BOOKM[i].value);
            i++;
        }
```

这里使用 while 循环语句来读取。循环的结束标志就是到读取文件的尾部，这里使用!feof(fr) 做循环条件，即不到文件尾时进行循环，到了文件尾时结束循环。

fscanf 函数按“%s %s %f\n”格式，一次读取书的 3 个信息，分别保存到全局结构体数组的 title、author、value 成员中。

（3）显示数据。

显示数据的主要代码如下：

```
//调用函数，实现从文件中读取图书的信息
    readData();

    //显示图书信息
    int i;
    for(i=0; i<4; i++)
    {
        printf("%s    \t     %s \t     %.1f \n",
BOOKM[i].title,BOOKM[i].author,BOOKM[i].value);
    }
```

在显示数据的函数中调用 readData()，先实现从文件中读取数据，保存到全局的结构体数组中，再利用 for 循环，将全局的结构体数组中的数据进行显示。

学习活动 5　测试验收

任务测试验收单

● 实现效果

本任务实现将保存在文件中的数据读取，并显示在界面上。

按制定的方案进行任务实现，在正确的情况下，其效果如图 8.5 所示。

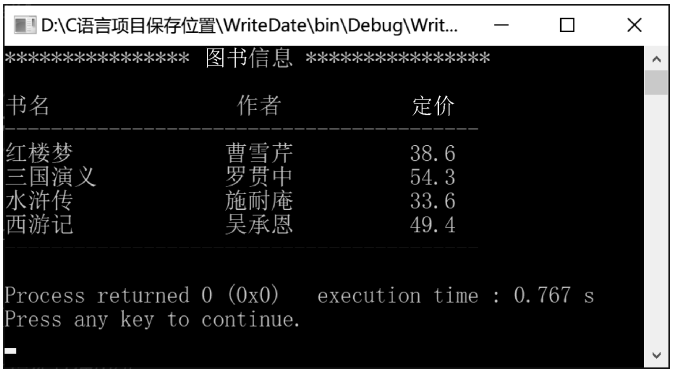


图 8.5　显示数据界面

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

实现了将从保存四大名著信息的文本文件中，读取数据并显示。读取数据和显示数据以功能进行区分，并单独以函数实现。

● 技术层面

将以上次任务写入的四大名著信息，完成读取数据。

（1）掌握数据读取文本文件的操作流程。

（2）掌握数据读取文本文件的相关知识。

（3）单独以函数实现（模块化设计思路）。

● 课程思政

同学们掌握了 C 语言文件操作中读取文件的相关知识。本模块以“日出而作，日入而息”作为开场，引出对应的读取文件数据和将数据写入文件的过程。

那么，“日出而作，日入而息”呈现出的是一幅自力更生的劳作场景，所以希望同学们能传承中华民族勤劳的优良传统，做一个自力更生的人。

● 教学拓展

同学们尝试着将写文件和读文件这两个任务相结合，以完成书本信息的添加与显示功能。

● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

现在已将文件、结构体、函数都集合了

下一回：与结构体和函数一起玩玩

（为后续的综合案例做好准备）

任务 3　与结构体和函数一起玩玩

目标描述

任务描述
● 编写程序实现 **实现学生信息的添加与显示功能。** 具体要求如下： （1）结合结构体、函数、文件操作的知识； （2）存储：采用文件来存储学生信息； （3）业务：单独采用头文件（H 文件）的形式； （4）表示：数据以结构体的形式传递。 ● 技术层面 综合应用结构体、函数、文件的操作知识。 ● 课程思政 做好职业规划。

学习活动 1　**接领任务**

领任务单
● 任务确认 编写 C 语言程序，将结构体、函数、文件操作相关知识结合实现学生信息的添加与显示功能。

具体要求如下：

（1）结合结构体、函数、文件操作知识实现；

（2）存储：采用文件来存储学生信息；

（3）业务：单独利用头文件（H 文件）的形式；

（4）表示：数据以结构体的形式传递；

（5）掌握 C 语言代码的使用规范（变量命名及注释说明）；

（6）程序能正确运行，并具有可扩展性。

● **确认签字**

学习活动 2　分析任务

要求结合结构体、函数、文件操作知识，实现学生信息的添加与显示功能。为后续的综合项目做好准备。

（1）存储层：采用文件来存储学生信息；

（2）业务逻辑层：单独利用头文件（H 文件）的形式；

（3）表示层：数据以结构体的形式传递。

学习活动 3　制定方案

实现本任务方案

● **实现思路**

通过对本任务的分析及相关知识的学习，制定实现方案如图 8.6 所示。

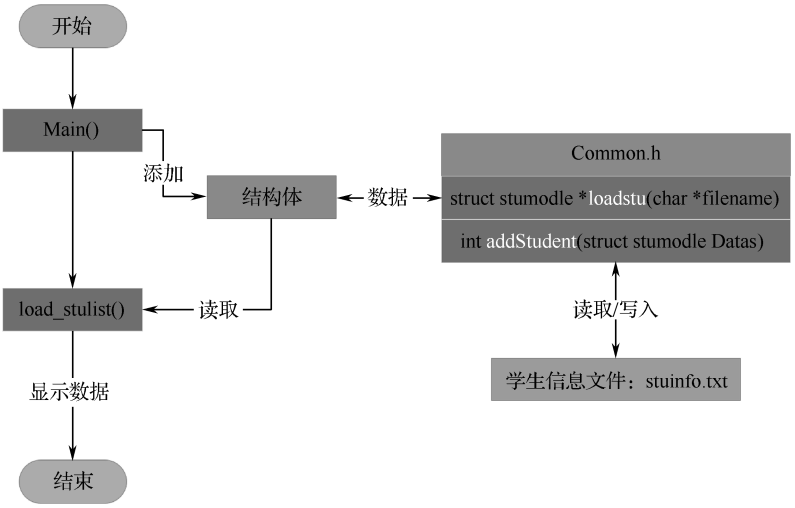


图 8.6　制定实现方案示意

说明：

（1）表示层设计。

在 main()中实现创建结构体数据，并调用业务层中的 addStudent()以实现添加学生信息。

实现 load_stulist()是从文件中读取的学生信息，并进行显示。

（2）业务逻辑层设计。

将实现对文件操作的功能，采用单独利用函数的方式在 H 文件中实现，以实现业务功能层。

使用 Loadstu()实现从文件中读取数据，并以结构体返回。

使用 addStudent()实现结构体数据的接收，并保存到文件中。

（3）数据存储层设计。

将学生信息保存到 StudentInfo.txt 文件中。

● **实现步骤**

（1）在 CodeBlocks 软件中创建一个新项目，项目名称为 StuManageDemo。

（2）在项目中按实现思路编写代码。

学习活动 4　实施实现

任务实现

● **实现代码**

（1）打开 Code Block 软件，创建项目 StuManageDemo，其具体步骤如下。

① 执行“File”→“New”→“Project…”，如图 8.7 所示。

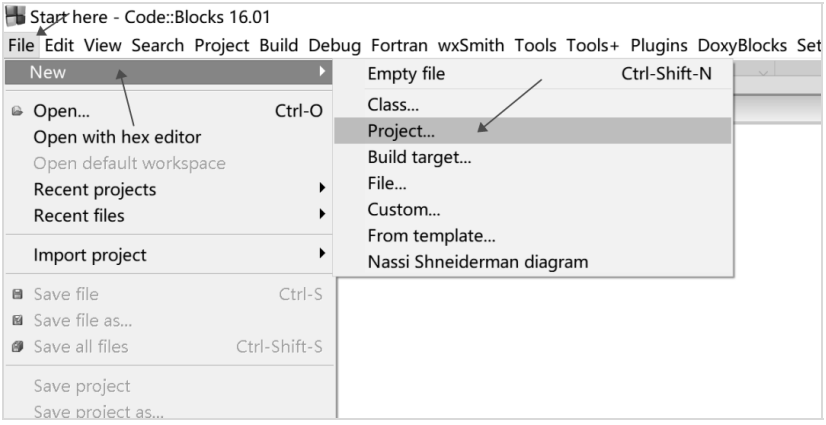


图 8.7　新建项目

② 选择“Console application”选项，单击“Go”按钮，如图 8.8 所示。

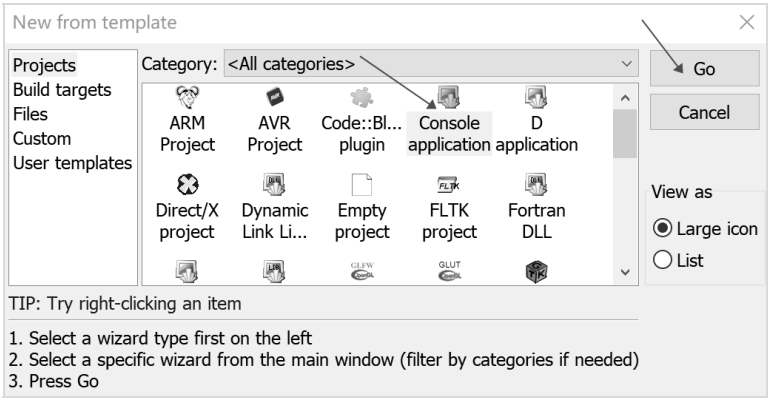


图 8.8　选择项目类型

③ 选择“C”选项，单击“Next”按钮，如图 8.9 所示。

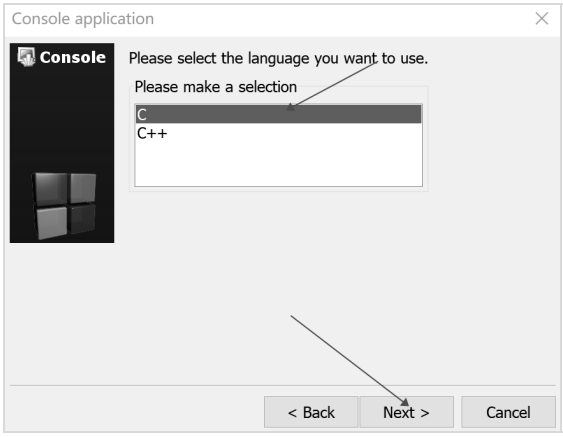


图 8.9　选择 C 语言

④ 输入项目名称为 StuManageDemo，单击“Next”按钮，如图 8.10 所示。

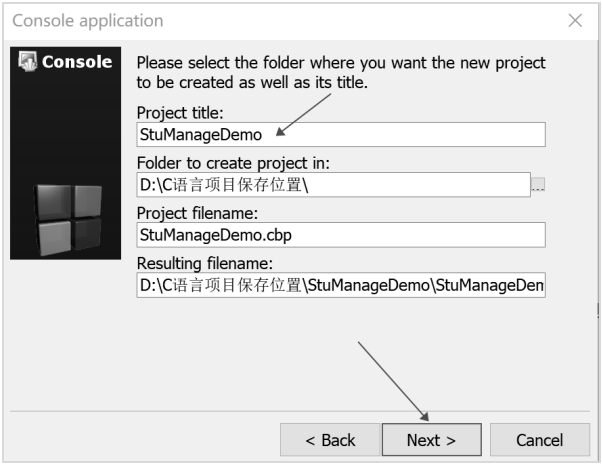


图 8.10　输入项目信息

（2）创建头文件为 Common.h，具体步骤如下。

① 单击项目名称，以选中项目，如图 8.11 所示。

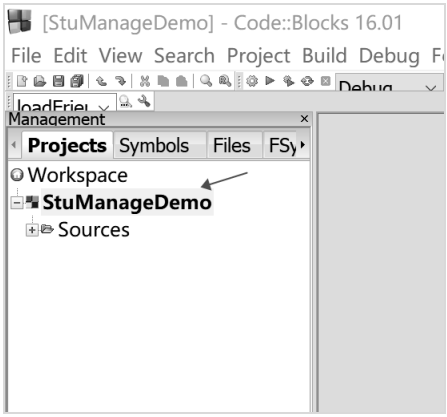


图 8.11　单击项目名称

② 执行“File”→“New”→“File…”，新建 C 语言头文件，如图 8.12 所示。

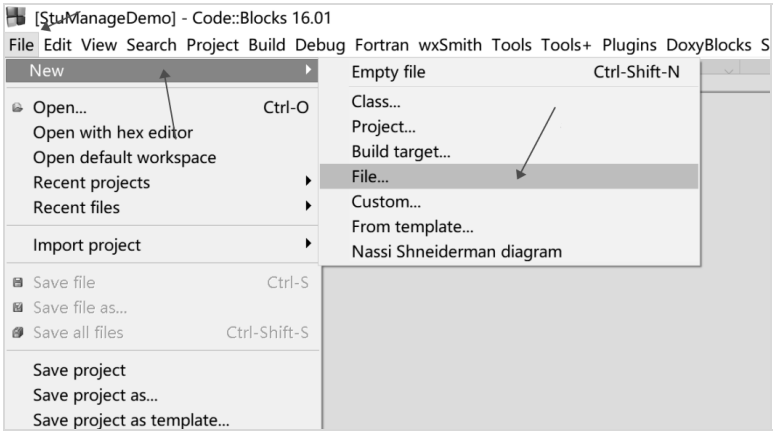


图 8.12　新建头文件

③ 选择“C/C++ header”选项，单击“Go”按钮，如图 8.13 所示。

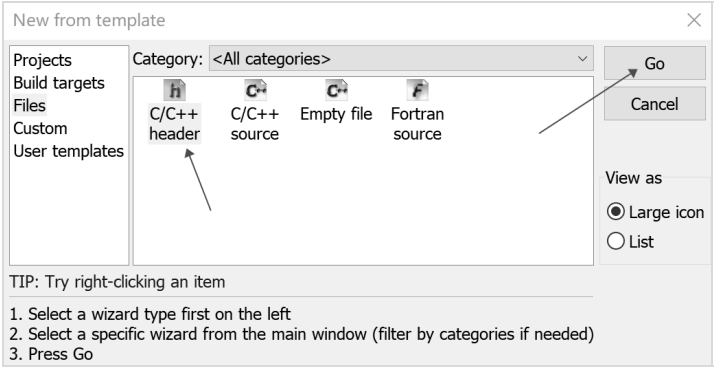


图 8.13　选择头文件类型

④ 单击浏览按钮，设置头文件保存位置，如图 8.14 所示。

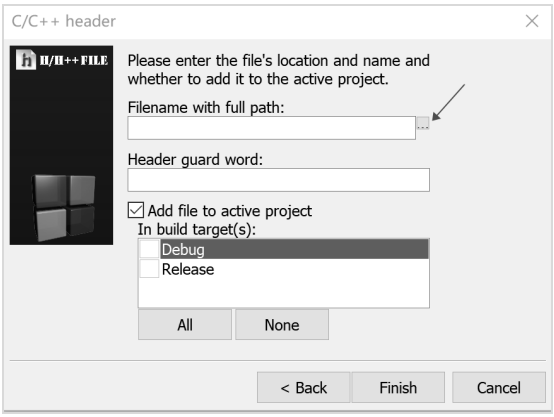


图 8.14　设置头文件保存位置

⑤ 输入头文件名为 Common.h，然后单击“保存”按钮，如图 8.15 所示。

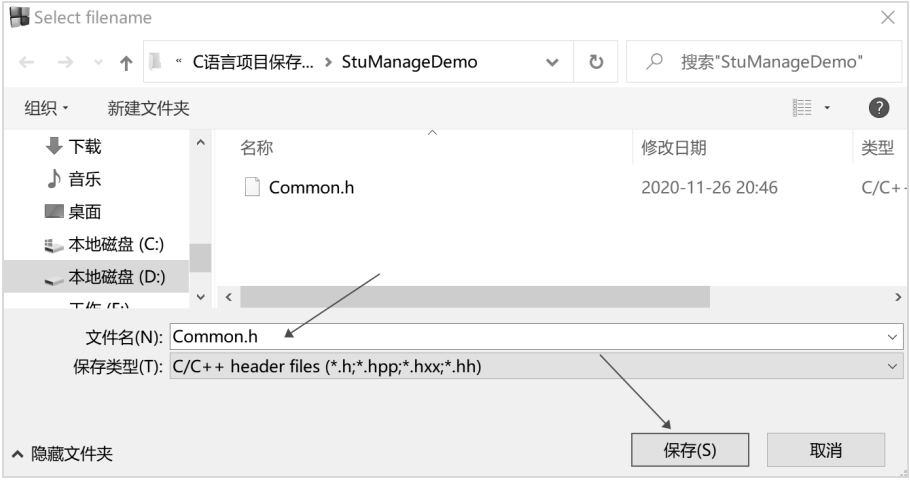


图 8.15　输入头文件名

⑥ 单击“Finish”按钮完成创建，如图 8.16 所示。

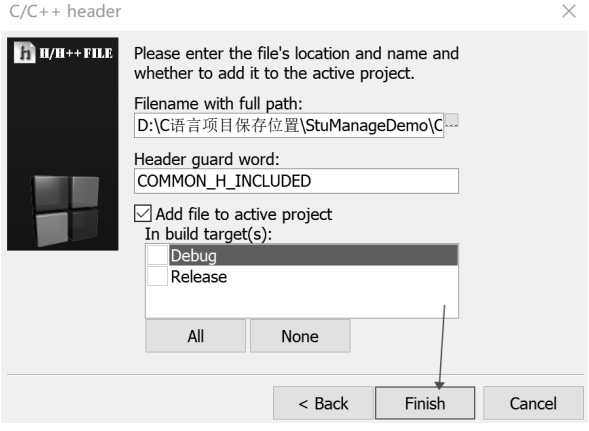


图 8.16　完成创建头文件

⑦ 项目出现如图 8.17 所示的界面，说明 H 头文件创建成功。

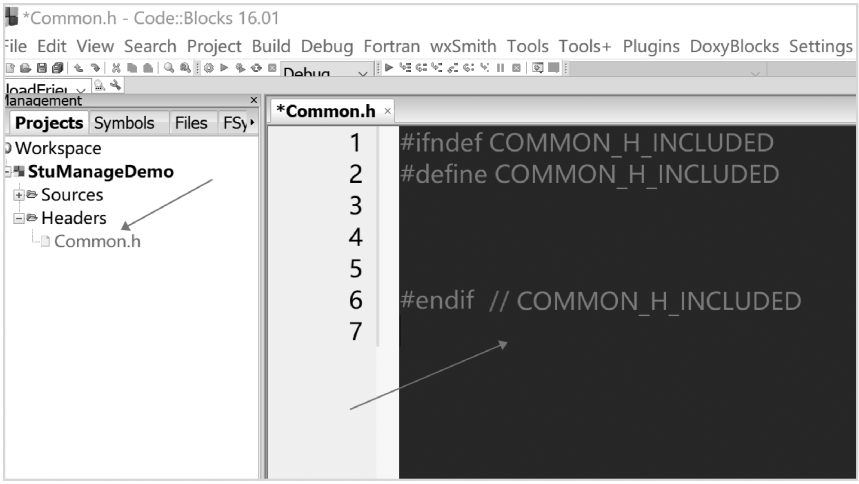


图 8.17　项目头文件结构

● **实现数据存储**

通过上面的步骤操作，我们已经完成项目及 H 头文件的创建功能，下面设计本任务实现数据存储文件的创建。

（1）找到项目保存在硬盘的位置。

用鼠标右击项目名称，选择“Open Project Folder in File Browser”选项，打开项目所保存的位置，如图 8.18 和图 8.19 所示。

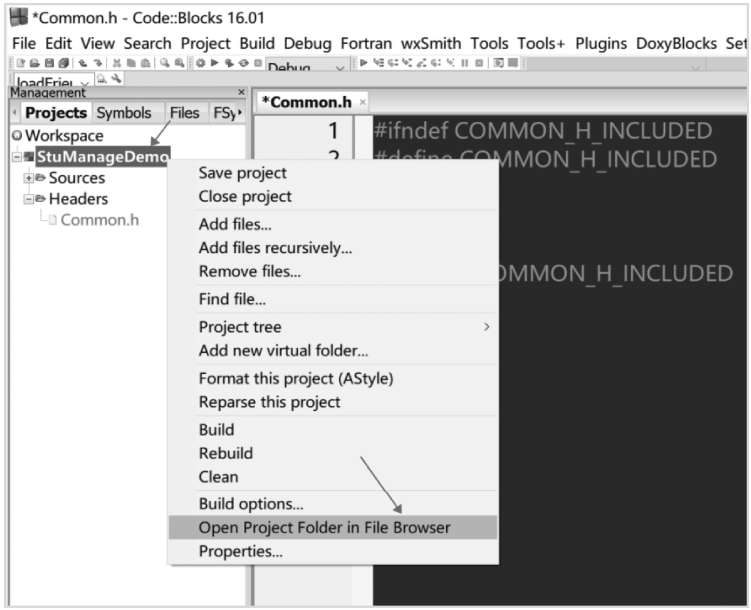


图 8.18　打开项目保存位置

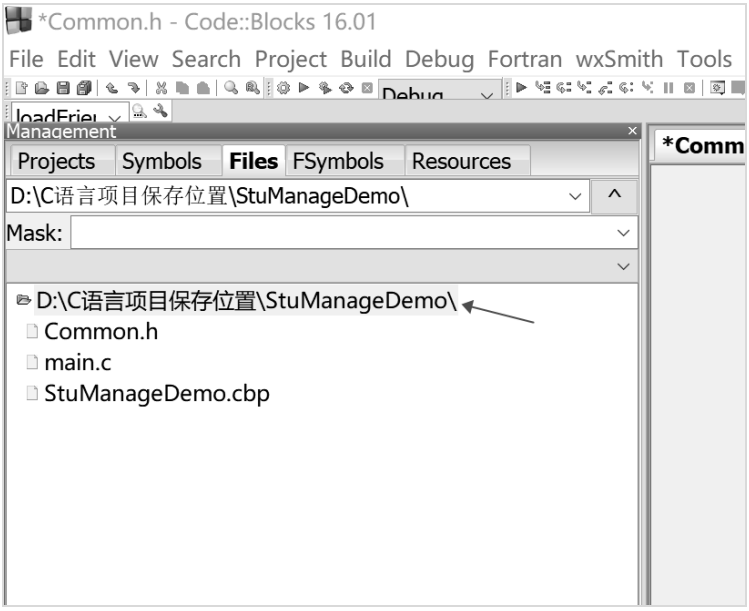


图 8.19　项目保存位置

在这里，我们可以看到该项目是保存在“D:\C 语言项目保存位置\StuManageDemo\”目录下的。

（2）打开计算机，进入 D:\C 语言项目保存位置\StuManageDemo 目录下，如图 8.20 所示。创建存储文件为 StudentInfo.txt。

图 8.20　进入项目保存位置

在该目录下，用鼠标右击空白处，执行“新建”→“文本文档”，如图 8.21 所示。

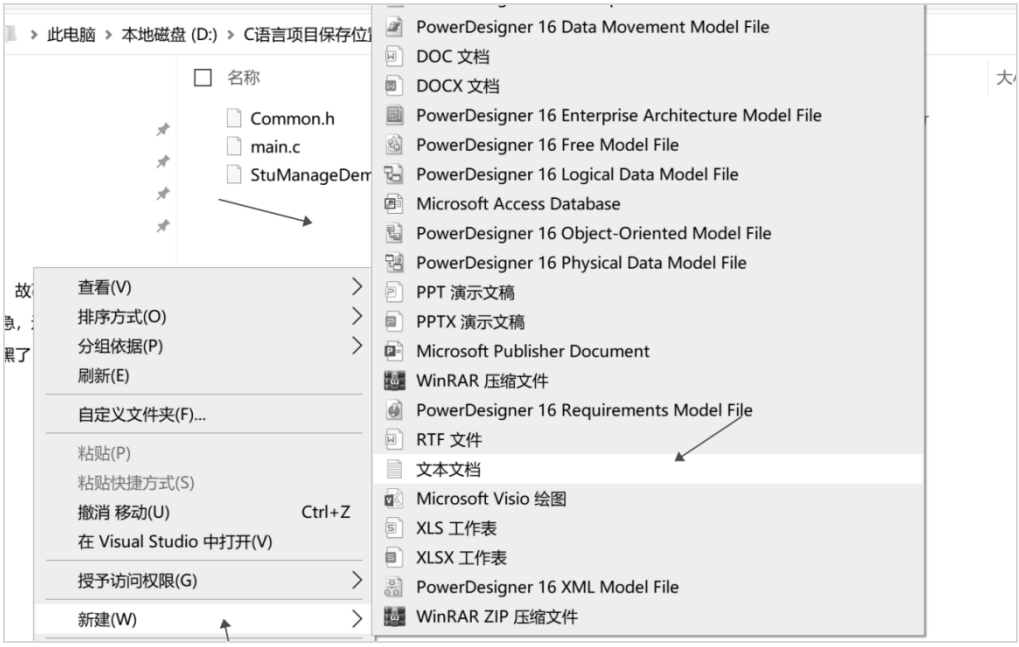


图 8.21　新建存储文件

将新建立的文本文档，改名为 StudentInfo，如图 8.22 所示。

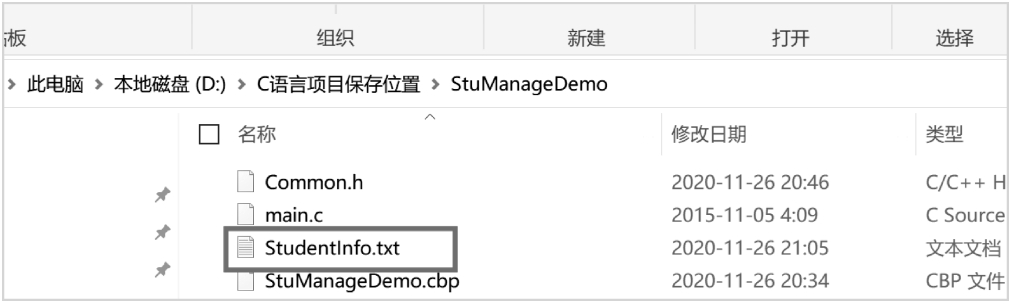


图 8.22　改数据存储文件名

● **实现业务层功能**

通过上面的步骤，我们已经实现了以下内容：

① 创建项目；

② 创建 Common.h 头文件；

③ 在项目文档目录中，创建存储数据的文件为 StudentInfo.txt。

万事已经具备，接下来，我们就一起来实现该程序的业务功能函数。

1．实现 addStudent()

（1）打开 Common.h 头文件。

选中“Projects”选项卡，展示项目列表下的 Header 目录，然后双击“Common.h”文件进行编写，如图 8.23 所示。

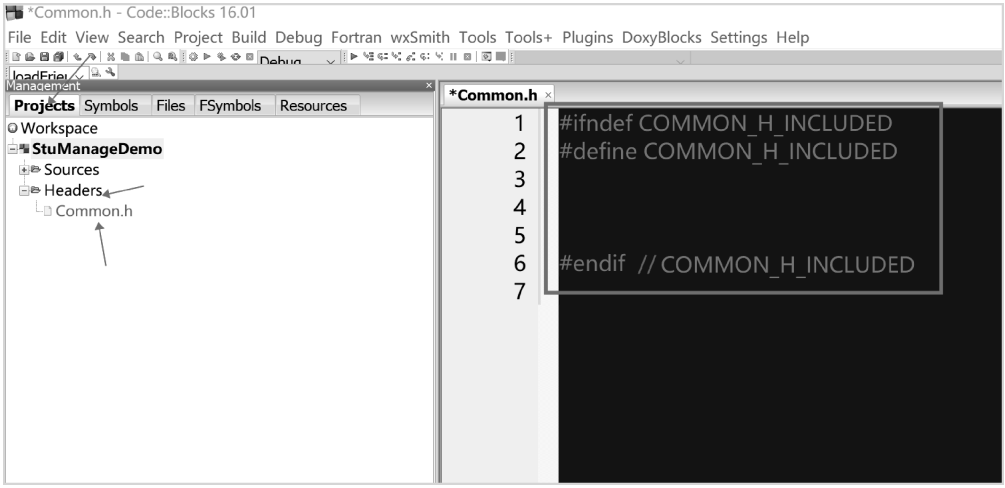


图 8.23　打开头文件内容

（2）删除多余的代码。

图 8.23 的右边框线中的代码是系统自动产生的，需要全部删除。

（3）实现结构体。

实现一个结构体 stumodle 用于中传数据，其代码如下：

```
#include <io.h>
#include <direct.h>

//本系统存放学生总数
int    MAXSIZE=100;
//用于记录已经存放学生的真实个数
int    TOTALCOUNT=0;

//学生结构体
struct stumodle {
      char name[20];   //姓名
      char sex[20];      //性别
      char tel[20];       //电话
      char age[20];      //年龄
};
```

（4）实现 addStudent()。

在上一步创建的结构体中，实现添加学生信息的功能函数 addStudent()，其代码如下：

```
/**
* 功能：添加一个学生
* @Datas：输入参数，stumodle 结构体变量
* 返回   1 为成功，0 为失败
*/
int addStudent(struct stumodle Datas)
{
    //创建文件
```

```
        FILE *fp=fopen("StudentInfo.txt","a");
        if(!fp)
        {
            printf("errror!\n");
            return 0;
        }
        //写入文件
        fprintf(fp,"%s %s %s %s\n",Datas.name,Datas.sex,Datas.tel,Datas.age);
        fclose(fp);   //关闭文件
        return 1;
    }
```

2．实现 loadstu()

在上一步实现添加学生信息 addStudent()的下方，实现读取学生信息的功能函数 loadstu()，其代码如下。

```
    /**
    * 功能：读取指定文件的内容*
    * 返回：若对应结构体则成功，否则失败
    */
    struct stumodle *loadstu()
    {
        FILE *fr = fopen("StudentInfo.txt","r");
        if(fr == NULL) {
            return;
        }
        //定义结构体
        struct stumodle sMod[MAXSIZE];
        //将文件中的数据写入结构体数组中
    int n=0;   //用于记录学生的数量
        while(!feof(fr)) {
            fscanf(fr,"%s%s%s%s\n",sMod[n].name,sMod[n].sex,sMod[n].tel,sMod[n].age);
            n++;
        }
        //记录已有学生的真实数量
        TOTALCOUNT=n;
        //关闭文件
        fclose(fr);
        //返回存放了数据的结构体
        return sMod;
    }
```

● **实现表示层功能**

通过上面步骤，我们已经实现 Common.h 头文件添加、读取学生信息功能函数的编码实现，下面进入表示层功能的实现，具体步骤如下。

1．实现表示层显示数据

（1）引入 Common.h 文件。

将实现添加学生与读取学生的功能单独以函数的方式在 Common.h 文件中实现，那么在

表示层中使用其中的函数，就必须引用 Commom.h 文件，否则调用不到其相关功能函数。

展示项目列表，双击 main.c 文件以进入表示层功能的编辑区，具体如图 8.24 所示。

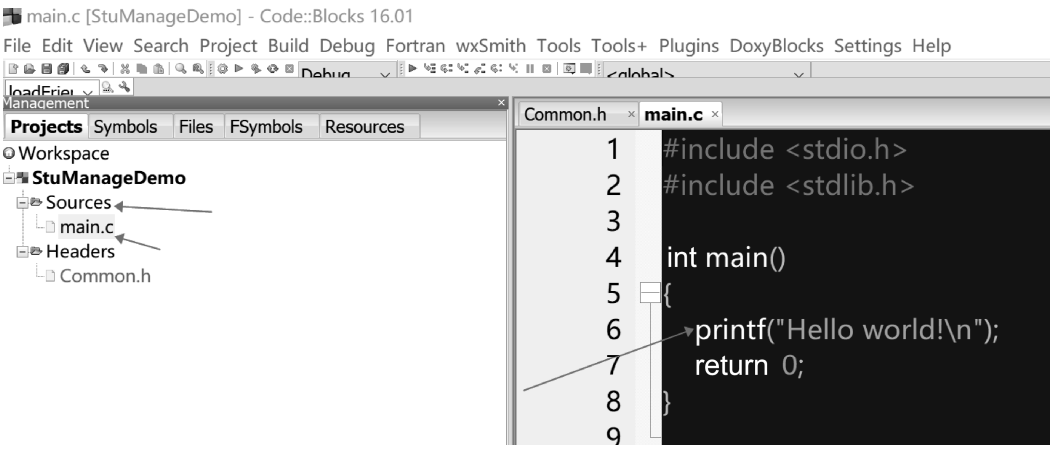


图 8.24　打开 main.c 文件

在 main()前，引入 Common.h 头文件的代码，其代码如下。

```
#include <stdio.h>
#include <stdlib.h>

//调用自定义的 Common.h 头文件
#include "Common.h"

int main()
{

    return 0;
}
```

（2）实现表示层显示学生信息的实现。

在调用 Common.h 文件与 main()之间创建一个新函数为 load_stulist()，其代码如下。

```
//加载所有学生并显示
void load_stulist()
{
    //清除屏幕
    system("cls");
    printf("**********  学生列表  **********\n\n");
    printf("\n 序号\t 姓名\t\t 性别\t\t 电话\t\t 年龄\n-------------------------------------------------------------\n");

    //定义学生结构体变量
    struct stumodle *_stumodle;
    //调用 H 文件中的 loadstu()，以结构体返回所有学生信息，并赋值给_stumodle
    _stumodle=loadstu();

    //显示所有学生
```

```
    int i;
    for(i=0; i<TOTALCOUNT; i++)   //在 Common.h 中已经赋值
    {
        printf("%2d%12s\t\t%s\t\t%s\t\t%s\n",i+1,_stumodle[i].name,_stumodle[i].sex,
_stumodle[i].tel,_stumodle[i].age);
    }
    printf("-------------------------------------------------------------------\n");
}
```

2．实现表示层添加数据

在 main()中实现一个结构体，调用 Common.h 文件中的学生信息函数，以实现添加学生的功能，其代码如下。

```
int main()
{
    //创建结构体，该结构体在 Common.h 中已经创建
    struct stumodle _stumodle;

//将"林小"字符串赋值给_stumodle.name
    strcpy(_stumodle.name,"林小");
    strcpy(_stumodle.sex,"男");
    strcpy(_stumodle.tel,"1131313");
    strcpy(_stumodle.age,"20");
    //调用 H 文件中的 addStudent 函数，将结构体以参数传入，以实现添加学生信息的目的
    addStudent(_stumodle);
    //调用 load_stulist 函数，显示学生信息
    load_stulist();
    return 0;
}
```

学习活动 5　测试验收

任务测试验收单

- **实现效果**

实现了学生信息管理的添加和显示功能。在正确的情况下，运行程序应如图 8.25 所示。

```
D:\C语言项目保存位置\StuManageDemo\bin\Debug\StuManageDemo.exe
********** 学生列表 **********

序号    姓名          性别          电话          年龄
-------------------------------------------------------------------
 1       林小          男          1131313       20
-------------------------------------------------------------------
```

图 8.25　程序运行的效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

结合结构体、函数、文件操作的知识，实现了学生信息的添加与显示功能。

同时，采用软件三层架构的思路进行设计：

存储：采用文件来存储学生信息；

业务：单独采用头文件（H 文件）的形式实现；

表示：数据以结构体的形式传递。

● 技术层面

结构体、函数和文件操作。

软件的三层架构。

● 课程思政

利用软件三层架构的思路，实现了学生信息的添加与显示的功能。

在实现的过程中，利用流程图的方式先做好软件的设计，然后根据设计逐步进行编码实现（见图 8.6）。

通过本任务的学习，充分说明设计的重要性。有怎样的设计，就会有怎样的结果。从某种角度来说也验证了“不打无准备的仗”的意义。

对于未来的软件从业人员，我们更应该明白设计对人生意义，所以希望同学们做好人生的职业规划，并坚持着实现自己人生的价值。

● **教学拓展**

本任务实现了对学生信息的添加与显示功能，同学们尝试将本任务学生信息的删除功能进行实现。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

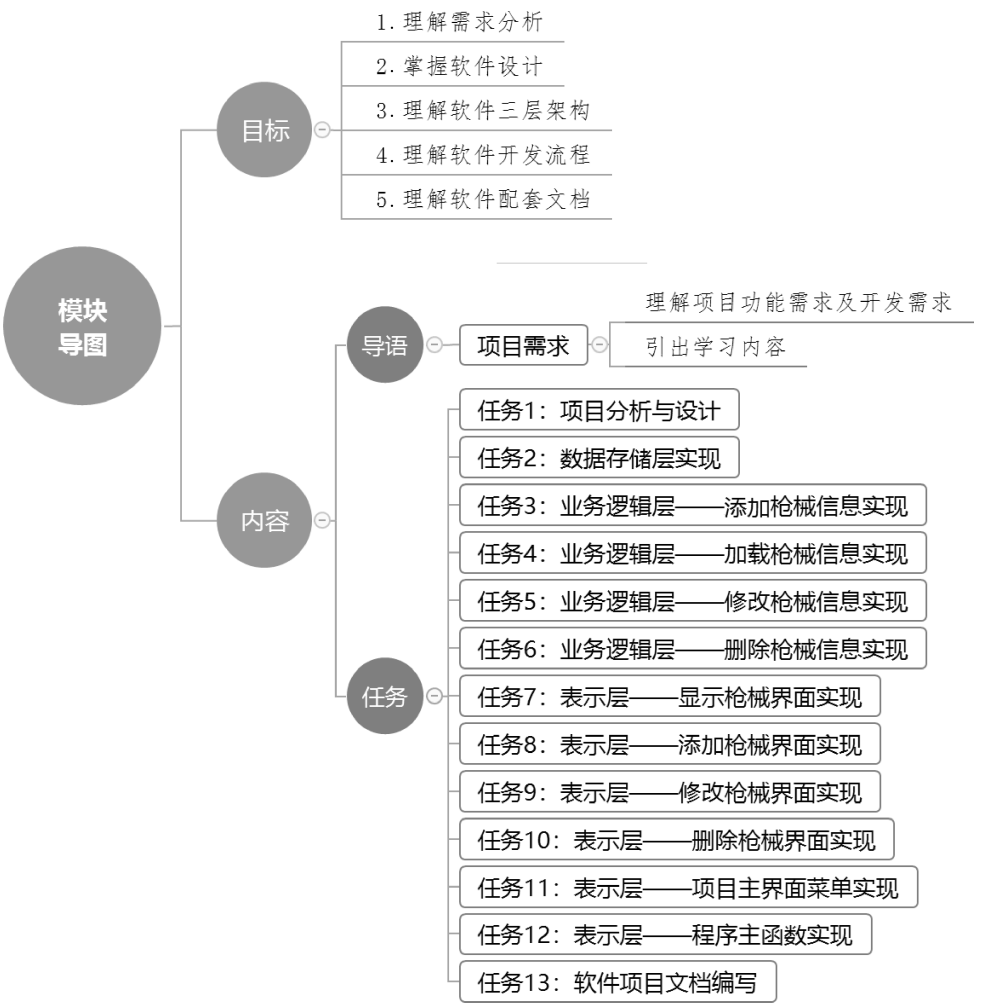
“小试牛刀”不过瘾

下一回：进入模块 9　C 语言综合项目实现

（综合项目实现吃鸡游戏的枪械信息管理）

模块 9

C 语言程序综合项目实现



项目需求

本模块进入了 C 语言第三阶段的学习。将按完整的软件项目开发流程进行设计，综合应用 C 语言的相关知识完成一个综合性的软件项目。

项目名称：吃鸡游戏枪械信息管理系统。

1. 功能需求

功能 1：实现枪械信息的显示；

功能 2：实现枪械信息的添加；

功能 3：实现枪械信息的修改；

功能 4：实现枪械信息的删除。

2. 开发要求

（1）项目架构：采用软件三层架构开发。

表示层：实现枪械信息的显示、添加、修改、删除、菜单等界面；

业务逻辑层：实现枪械信息的加载、添加、修改、删除等业务功能；

数据存储层：采用本地文本文档实现枪械信息的存储。

（2）软件文档：编写软件项目配套需求说明书、概要设计文档、用户操作手册等文档。

3. 参考效果

该项目实现“吃鸡游戏枪械信息管理系统”后的功能参考如图 9.1 至图 9.4 所示。

```
**********************************************************吃鸡游戏枪械信息管理**********************************************************
编号    类型          名称          弹药          弹夹容量    伤害    归零距离    射速    射击模式        射击间隔
  1     手抢          P18C          9mm           17          19      25-25       375     单发/全自动     0.060 s
  2     冲锋抢        MicroUZI      9mm           25          23      100-200     350     单发/全自动     0.048 s
  3     步抢          M416          5.56mm        30          41      100-600     880     单发/全自动     0.086 s
  4     狙击枪        Kar98k        7.62m         5           72      100-600     760     单发            1.900 s
-------------------------------------------------------------------------------------------------------------------------------

 功能操作
        1.添加枪械
        2.修改枪械
        3.删除枪械
        0.退出系统
 请输入你要操作的功能号:
```

图 9.1　主界面示意

```
枪械管理>添加枪械界面
请输入枪械类型：霰弹抢
请输入枪械名称：S12K
请输入枪械弹药：12gauge
请输入弹夹容量：5
请输入伤害值：22
请输入归零距离：25-25
请输入射速：350
请输入射击模式：单发
请输入射击间隔：0.250
添加成功！是否继续添加(Y/N)?
```

图 9.2　添加界面示意

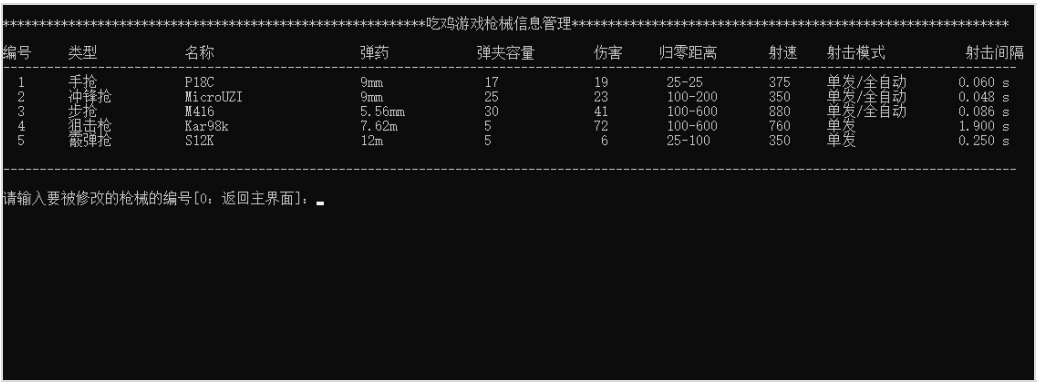


图 9.3　修改界面示意

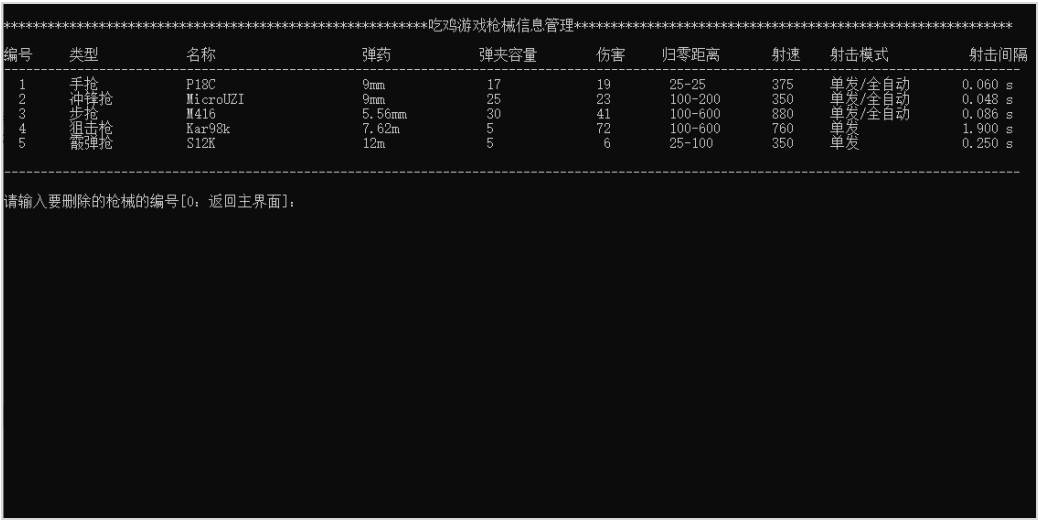


图 9.4　删除界面示意

任务 1　项目分析与设计

目标描述

任务描述
● 目标及要求 通过对“吃鸡游戏枪械信息管理系统”项目的需求分析完成系统设计。 具体要求如下： （1）完成项目架构的设计； （2）完成各功能流程图的设计； （3）完成项目的创建。

学习活动 1　接领任务

领任务单
● 任务确认 完成“吃鸡游戏枪械信息管理系统”项目的系统设计。 具体要求如下： （1）完成项目软件三层架构的设计，并绘制相应设计图； （2）利用专业绘制软件完成各功能流程图的设计； （3）在 CodeBlocks 软件中完成项目的创建。 ● 确认签字 ----------

学习活动 2　分析任务

分析任务
本任务要实现项目设计，就要按要求实现以下内容。 **1．完成项目软件三层架构的设计** “吃鸡游戏枪械信息管理系统”项目采用软件三层架构开发。 （1）表示层：实现项目的界面操作（在 main.c 中实现），可实现枪械信息的显示、添加、修改、删除、菜单等界面函数。 （2）业务逻辑层：实现项目业务功能（单独在头文件中实现），可实现枪械信息的加载、添加、修改、删除功能函数。 （3）数据存储层：采用本地文本文档实现枪械信息的存储（保存在文本文件中）。 **2．完成各功能流程图的设计** 用专业的绘制软件完成项目所有功能流程图的设计与绘制。 **3．完成项目的创建** 在 CodeBlocks 软件中按设计完成项目的创建，做好后续开发的准备。

学习活动 3　制定方案

实现本任务方案
● 实现思路 通过对本任务的分析及相关知识学习，制定方案如下： （1）设计项目对应三层架构设计； （2）绘制各功能业务流程图； （3）在 CodeBlocks 软件中进行项目的创建，并完成三层框架的设计。

● 实现步骤

（1）在 Visio 软件中绘制三层架构图，并确定各层的文件名及相应函数名。

（2）在 Visio 软件中绘制添加、修改、删除、加载功能的详细业务流程图。

（3）在 CodeBlocks 软件中完成项目的创建，并创建三层架构对应的文件。

学习活动 4　实施实现

任务实现

● 实现参考

1．项目三层架构设计的实现

通过以上分析，对“吃鸡游戏枪械信息管理系统”项目的架构设计如图 9.5 所示。

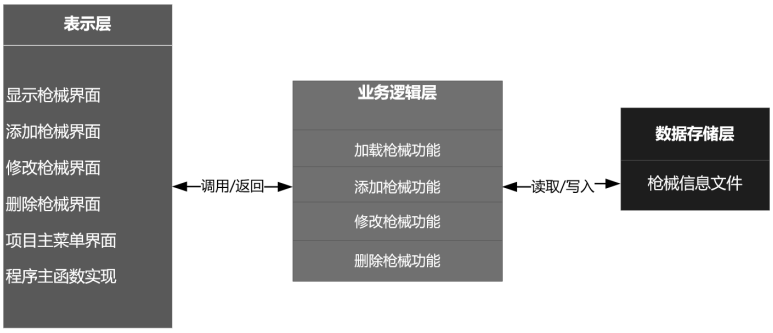


图 9.5　三层架构设计示意

表示层：对应 main.c 文件，在其中实现添加、修改、删除等界面功能；

业务逻辑层：对应 GunManage.h 文件，在其中实现具体的加载添加、删除、修改、功能；

数据存储层：对应文件 guninfo.txt 文件，用于存放数据。

具体功能都以单独的函数进行实现，详细设计如图 9.6 所示。

图 9.6　三层架构的设计

2．各功能业务流程设计实现

三层架构的设计完成后，再对项目中各功能模块的详细业务流程进行设计，具体如图 9.7 至图 9.10 所示。

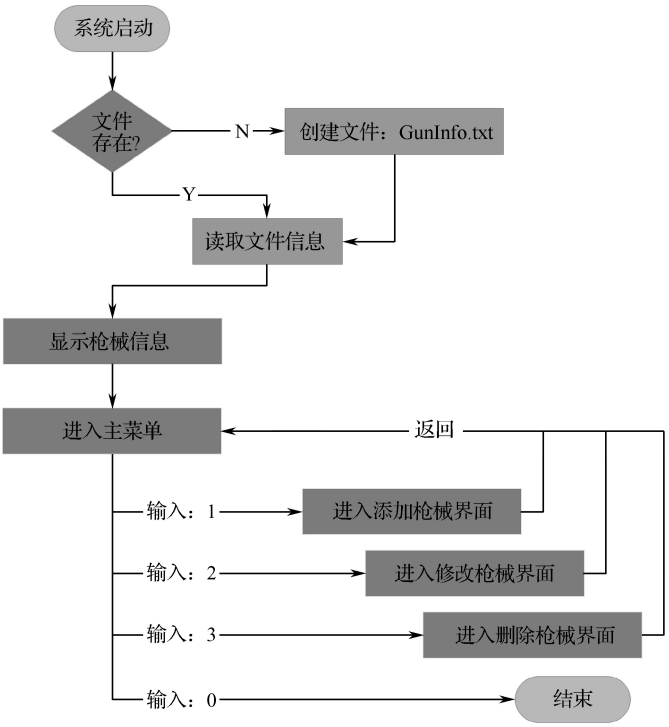


图 9.7　系统主界面流程

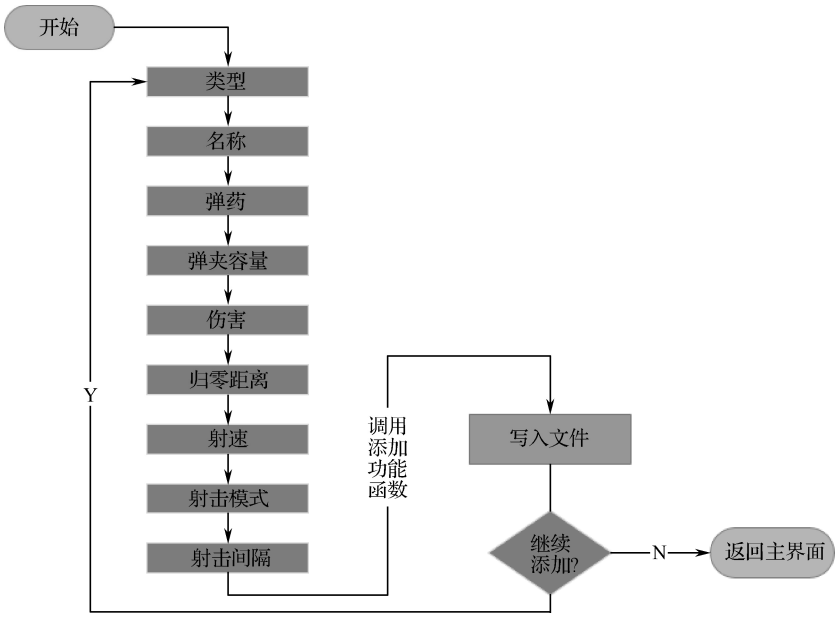


图 9.8　添加的流程

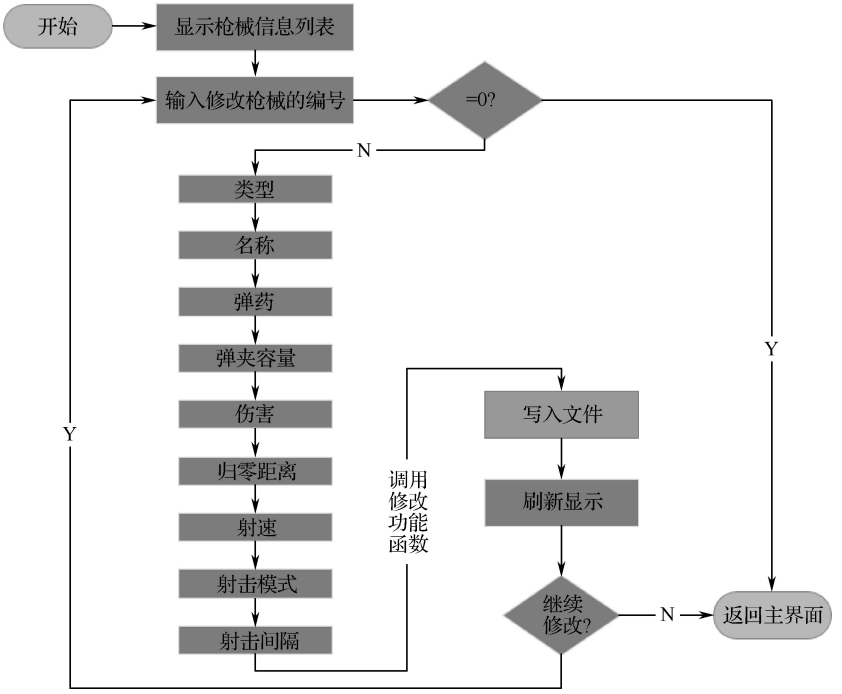


图 9.9　修改流程

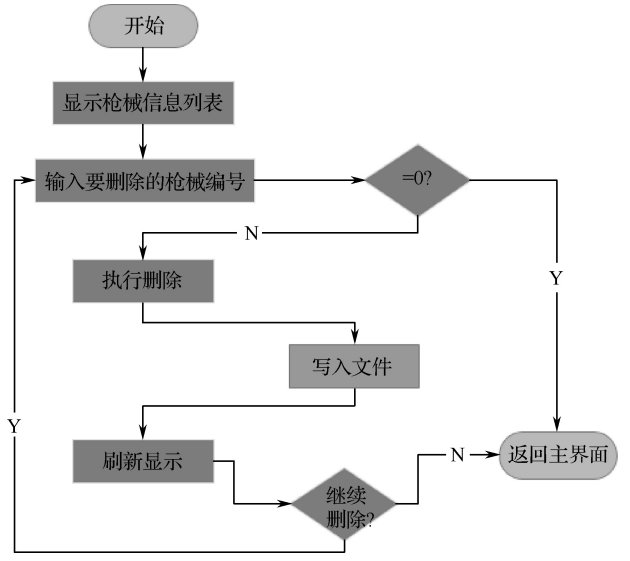


图 9.10　删除流程

3．创建项目实现

（1）打开 CodeBlocks 软件，创建项目 theGunInfo，具体操作步骤如下。

① 执行“File”→“New”→“Project…”，如图 9.11 所示。

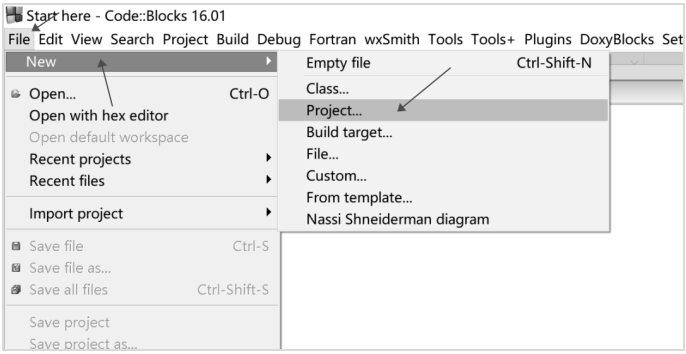


图 9.11　创建项目

② 选择“Console application”→“Go”，如图 9.12 所示。

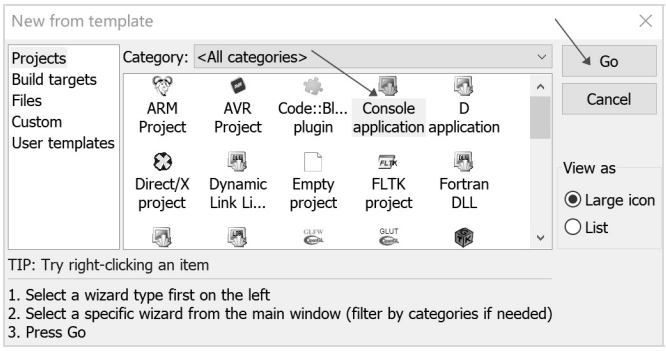


图 9.12　选择项目类型

③ 选择“C”语言，单击“Next”按钮，如图 9.13 所示。

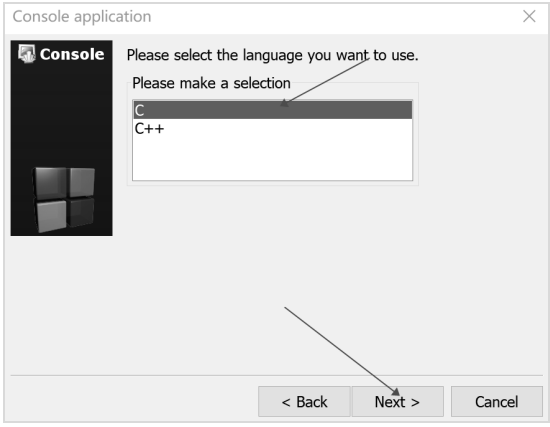


图 9.13　选择开发语言

④ 输入项目名称 theGunInfo，单击“Next”按钮，如图 9.14 所示。

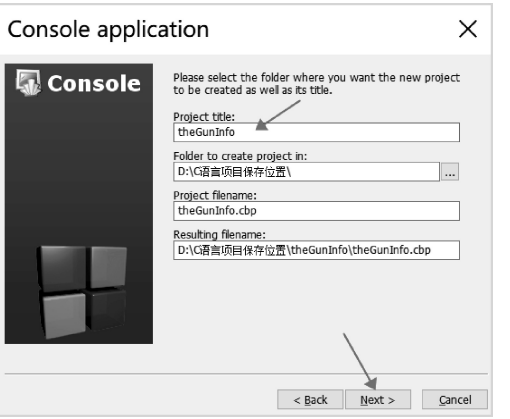


图 9.14　填写项目信息

（2）创建头文件 GunManage.h，具体步骤如下：

① 选中项目名称。

② 执行“File”→“New”→“File…”，如图 9.15 所示。

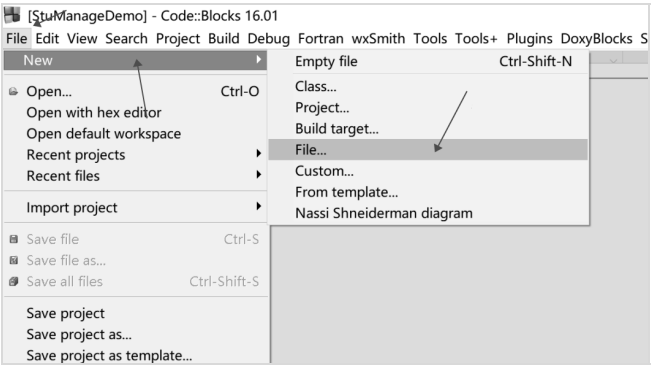


图 9.15　创建头文件

③ 选择“C/C++ header”→“Go”，如图 9.16 所示。

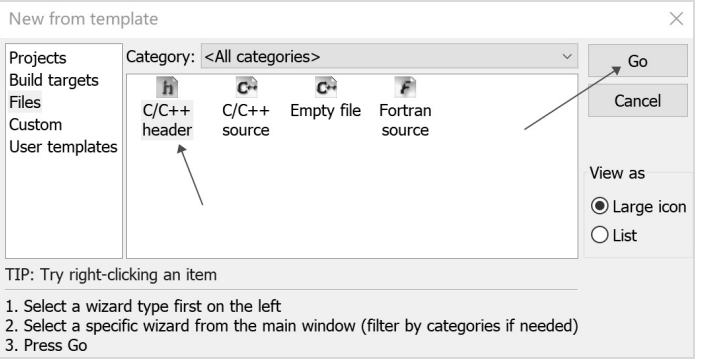


图 9.16　选择头文件

④ 单击浏览按钮，如图 9.17 所示。

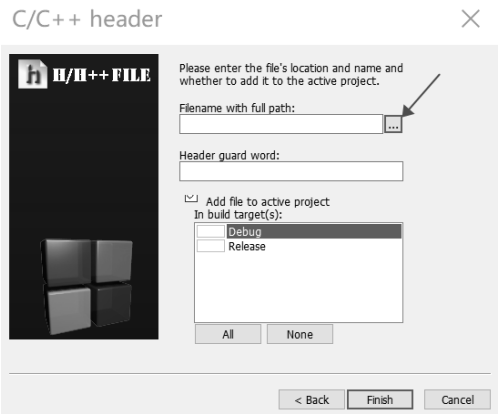


图 9.17　选择文件保存位置

⑤ 输入文件名 GunManage.h，并单击“保存”按钮，如图 9.18 所示。

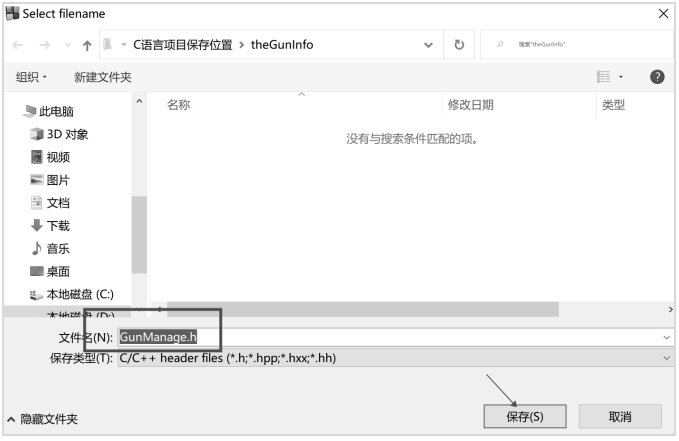


图 9.18　保存头文件

⑥ 直接单击“Finish”按钮，如图 9.19 所示。

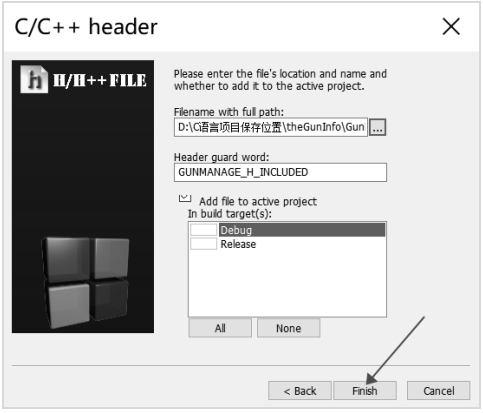


图 9.19　完成创建

⑦ 项目已完成头文件的创建，在项目中增加了一个 Headers 目录，在这个目录中就是新创建的头文件 GunManage.h。将文件内容选中并删除，如图 9.20 所示。

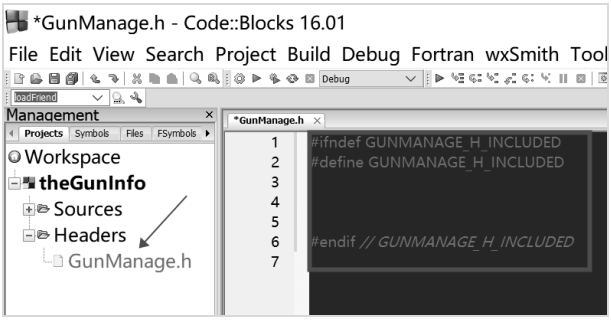


图 9.20　删除头文件中默认的代码

学习活动 5　测试验收

任务测试验收单

● **实现效果**

“吃鸡游戏枪械信息管理系统”项目创建已完成，运行程序没有报错，并能看到“Hello World!”即表示成功。

按制定方案进行任务实现，在正确的情况下，任务运行效果如图 9.21 所示。

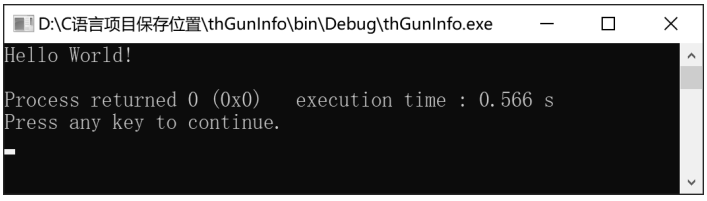


图 9.21　任务运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	流程图绘制情况					
4	项目创建实现情况					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

通过对“吃鸡游戏枪械信息管理系统”项目的需求分析，完成系统设计。

（1）完成项目三层架构的设计；

（2）完成各功能流程图的设计；

（3）完成项目的创建。

● 技术层面

利用 Visio 软件实现流程图绘制。

理解软件三层架构。

● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 2　数据存储层实现

目标描述

任务描述

● 目标及要求

根据项目设计，实现数据存储层。

具体要求如下：

（1）数据结构的设计；

（2）数据文件的创建；

（3）结构体的创建。

学习活动 1　接领任务

领任务单
● 任务确认 本任务可实现项目的数据存储层，具体要求实现如下： （1）分析数据，并完成系统数据结构的设计； （2）完成数据文件的创建； （3）根据数据结构，在程序中完成对应数据结构体的创建。 ● 确认签字

学习活动 2　分析任务

分析任务
在“吃鸡游戏枪械信息管理系统”项目中已完成 5 类枪械（手枪、冲锋枪、霰弹枪、步枪、狙击枪）的相关信息管理。本任务要求完成枪械信息的个数及对应程序中数据类型的确定。

学习活动 3　制定方案

实现本任务方案
● 实现思路 通过对本任务的分析及相关知识学习，制定方案如下： （1）确定枪械信息的结构，暂定为 9 个方面的信息； （2）在上次创建的项目中创建保存数据的文件； （3）在项目中编写对应的结构体。 ● 实现步骤 （1）分析枪械信息的结构，确定有 9 个方面的信息； （2）找到项目所在位置，创建保存数据的文件； （3）在 CodeBlocks 软件中打开上次创建的项目，并在 GunManage.h 中完成结构体的创建。

学习活动 4　实施实现

任务实现
● 实现参考 **1．数据结构设计实现** 实现对“吃鸡游戏枪械信息管理系统”项目中枪械信息的管理，具体如表所示。

枪械信息数据结构

类型	名称	弹药	弹夹容量	伤害	归零距离	射速	射击模式	射击间隔
手枪	P18C	9mm	17	19	25～25	375	单发/全自动	0.060s
冲锋枪	MicroUZI	9mm	25	23	100～200	350	单发/全自动	0.048s
霰弹枪	S12K	12m	5	6	25～100	350	单发	0.250s
步枪	M416	5.56mm	30	41	100～600	880	单发/全自动	0.086s
狙击枪	Kar98k	7.62m	5	72	100～600	760	单发	1.900s

2．数据文件的实现

上面已经确定本任务将管理枪械 9 个方面的信息，这些信息采用文本文件形式保存。数据存储文件名定义为 guninfo.txt。在文本文件中一个枪械的 9 个方面的信息保存为一行，各信息之间以一个空格分隔开，如图 9.22 所示。

图 9.22　数据文件示意图

实现步骤如下：

（1）在计算机中找到枪械信息管理项目，并进入该目录中；

（2）在空白区域右击，执行“新建”→“文本文档”，以创建文件，如图 9.23 所示。

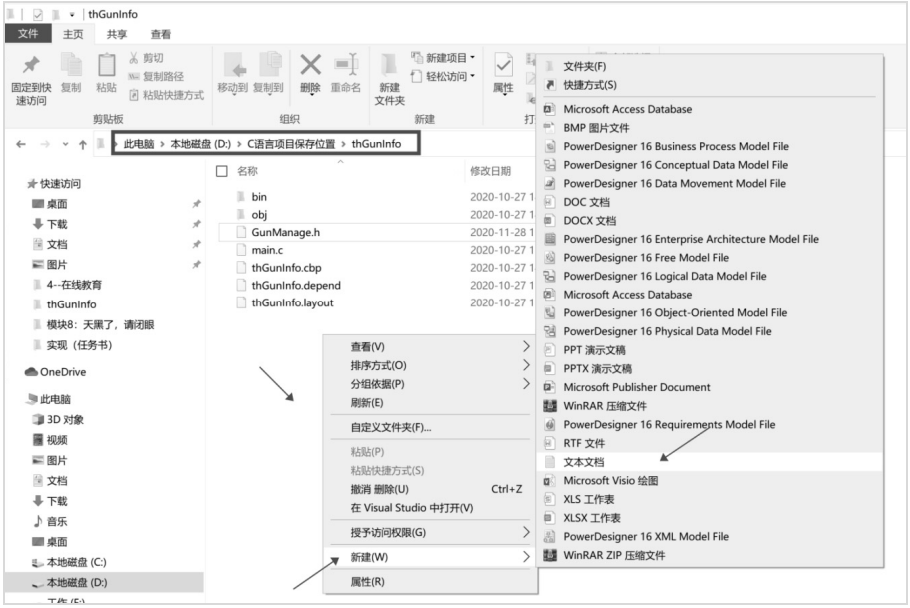


图 9.23　创建文件

（3）将文件名修改为 guninfo，即可，如图 9.24 所示。

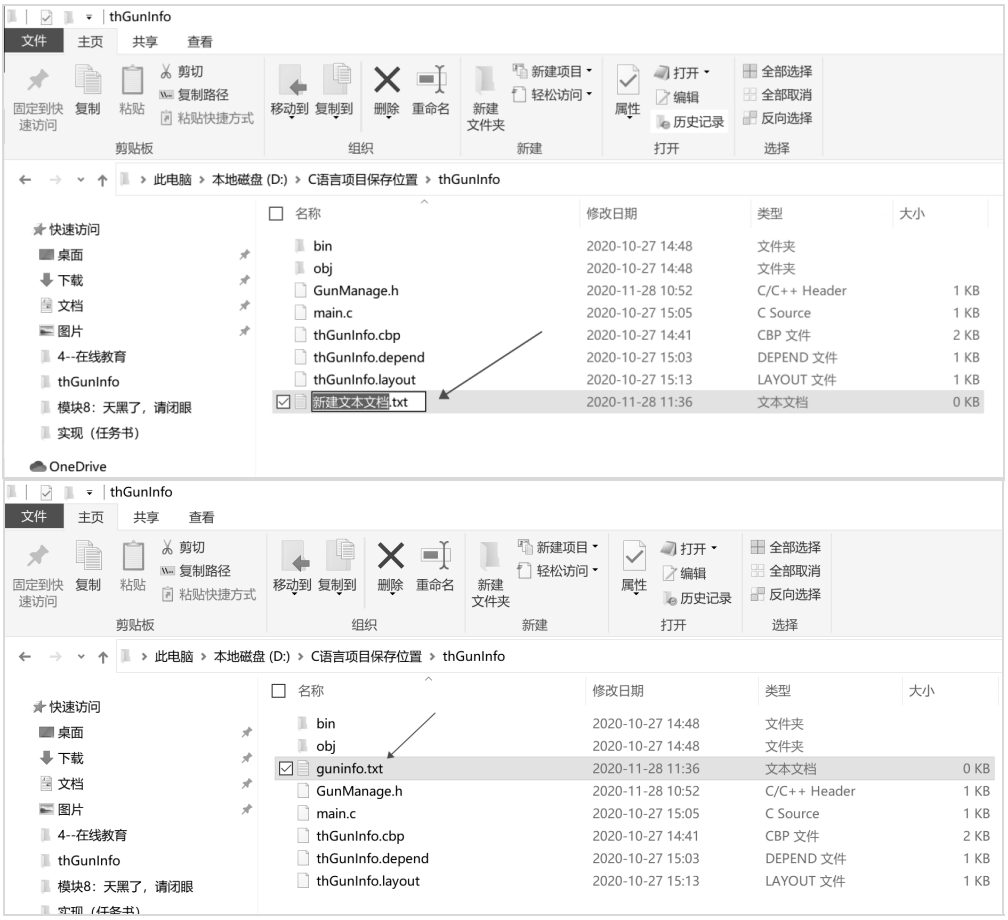


图 9.24　修改文件名

温馨提示：

如果你的计算机默认没有显示文件的扩展名，则可以选择“查看”选项卡，勾选“文件扩展名”复选框，如图 9.25 所示。

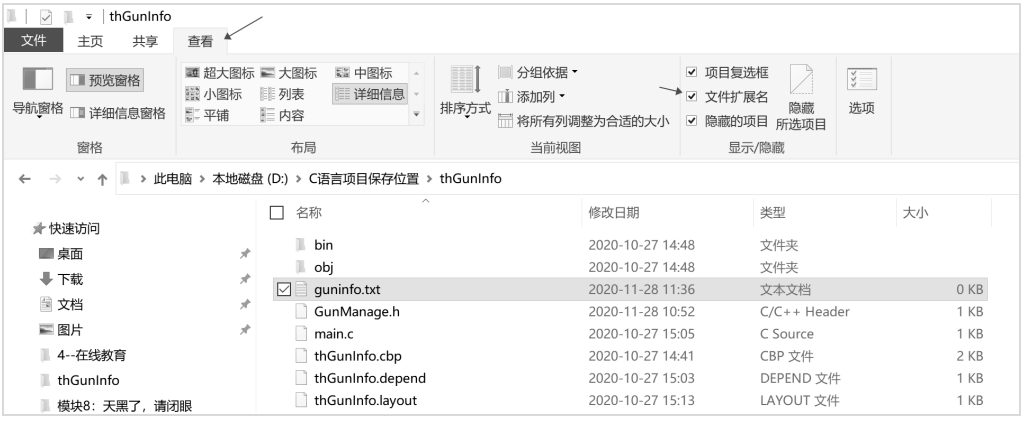


图 9.25　勾选“文件扩展名”复选框

3．项目结构体实现

通过上面的步骤，在项目目录中创建了存储数据的文件为 guninfo.txt。

接下来，在 CodeBlocks 软件中打开项目，即可完成枪械信息结构体的实现，具体步骤如下。

（1）打开项目

启动 CodeBlocks 软件，执行“File”→“Open…”，弹出“Open file”对话框，找到项目保存的位置，进入项目的目录，勾选项目的 CBP 文件，单击“打开”按钮以打开项目，如图 9.26 和图 9.27 所示。

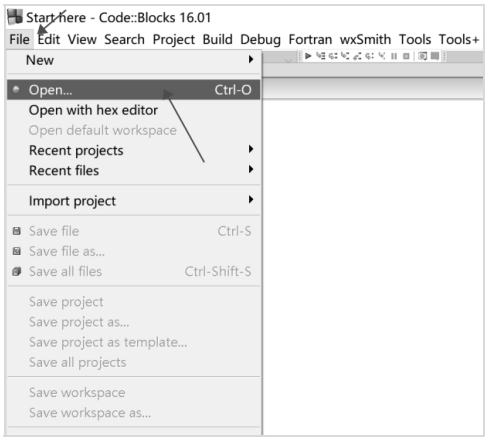


图 9.26　打开项目

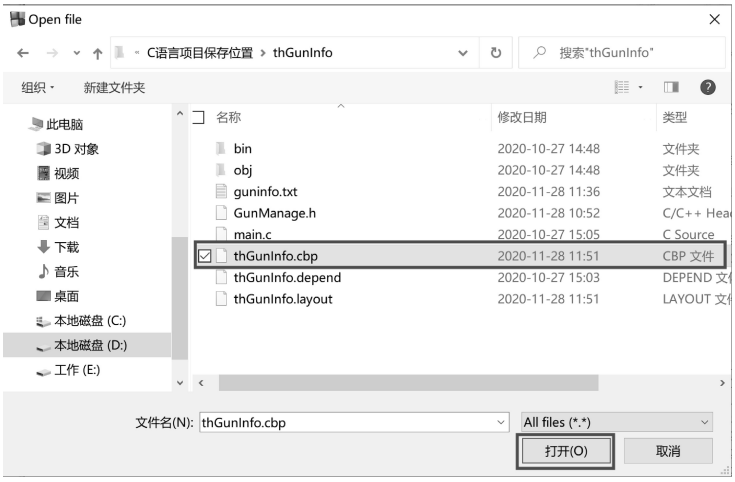


图 9.27　选择项目文件

（2）打开 GunManage.h 文件

展示项目的 Headers，双击 GunManage.h 文件，进入编辑状态，如图 9.28 所示。

（3）实现结构体

由于在任务 1 中创建 GunManage.h 文件时，已将文件中的内容删除了，所以本任务进入该文件时，里面是一片空白。

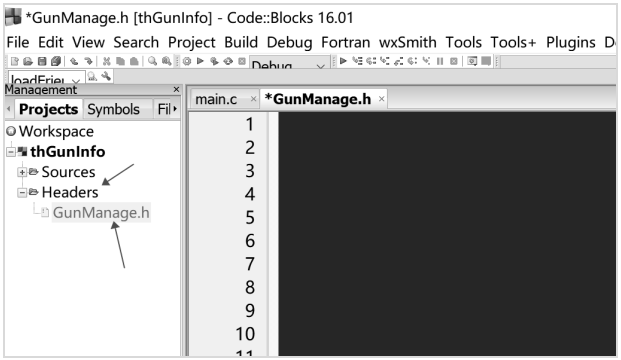


图 9.28　编辑文件

下面将在该文件中实现“吃鸡游戏枪械信息管理系统”项目的结构体，即完成对枪械的 9 个方面的信息对应程序中的数据类型及名称的实现。

实现的结构体如下：

```
//定义用于描述枪械信息的结构体
struct  GunModel
{
    char   NX[50];          //类型（如手枪、冲锋枪...）
    char   MC[50];          //名称（如 M416）
    char   DY[50];          //弹药（如 5.56mm）
    int    RL;              //弹夹容量（如 30）
    int    SH;              //伤害（如 41）
    char   GLJL[50];        //归零距离（如 100～600）
    int    SS;              //射速（如 880）
    char   SJMS[50];        //射击模式（如单发/全自动）
    float  SJJG;            //射击间隔（0.086s）
};
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

实现了项目数据存储文件的创建，以及对应结构体的设计。如果运行程序能够看到以下界面，则说明本任务顺利完成，如图 9.29 所示。

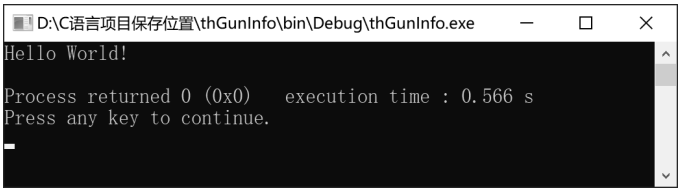


图 9.29　任务运行效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	数据结构设计情况					
4	结构体定义情况					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

完成“吃鸡游戏枪械信息管理系统”项目数据存储层的实现，完成内容如下：

（1）枪械信息数据结构的设计；

（2）存放枪械信息数据文件的创建；

（3）枪械信息结构体的创建。

● 技术层面

数据结构设计。

数据文件创建。

结构体定义。

● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 3　业务逻辑层——添加枪械信息实现

目标描述

任务描述
● 目标及要求 根据项目设计，完成业务逻辑层添加枪械信息的功能。 具体要求如下： 在业务逻辑层（GunManage.h）中，实现添加枪械信息的功能函数。

学习活动 1　接领任务

领任务单
● 任务确认 实现业务逻辑层添加功能。 具体要求如下： （1）正确实现业务逻辑层添加枪械信息功能函数； （2）命名规范，注释清晰。 ● 确认签字

学习活动 2　分析任务

分析任务
实现“吃鸡游戏枪械信息管理系统”项目业务逻辑层添加枪械信息的功能，该功能能够接收表示层的数据，将数据正确写入文件中，并进行保存，从而实现信息的添加。 以单独函数的方式实现。

学习活动 3　制定方案

实现本任务方案
● 实现思路 实现“吃鸡游戏枪械信息管理系统”项目中业务逻辑层添加枪械信息的功能函数，具体思路如下。 （1）函数名称：addGunInfo。 （2）函数输入：保存枪械信息的结构体。

（3）函数功能：将接收结构体中的数据按指定格式写入文件中。

（4）函数返回：操作完成后，返回操作，具体如图 9.30 所示。

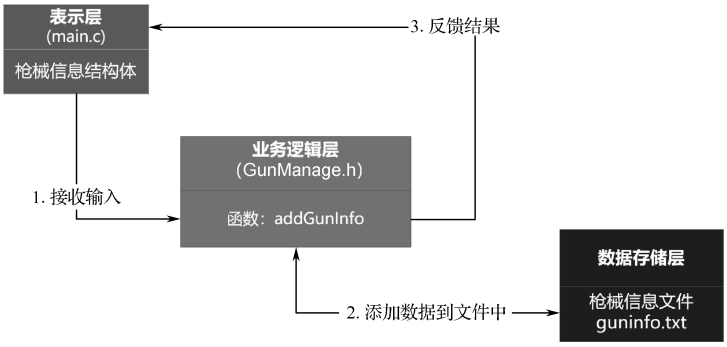


图 9.30　业务逻辑层添加功能示意

● 实现步骤

（1）打开之前创建好的项目；

（2）在业务逻辑层（GunManage.h）中完成。

学习活动 4　实施实现

任务实现

● 实现参考

通过上面的分析，进入 CodeBlocks 软件实现添加枪械信息功能函数，实现步骤如下。

（1）打开 GunManage.h 文件。展示项目的 Headers，双击 GunManage.h 文件，进入编辑状态。

（2）实现两个全局变量。

在已实现的结构体定义下方定义两个全局变量，为后续的开发做好准备，具体如图 9.31 所示。

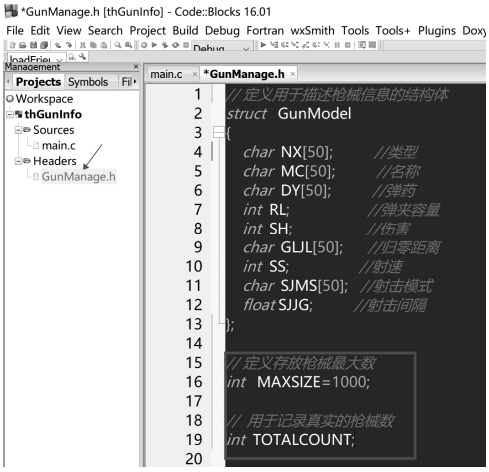


图 9.31　定义全局变量

（3）实现添加枪械信息功能函数。

根据上以分析，完成函数的功能编写，其代码如下。

```
int    MAXSIZE=1000;    //定义存放枪械的最大数
int   TOTALCOUNT;       //用于记录真实的枪械数
```

说明：

（1）写入文件时，一个枪械的 9 个方面的信息为一行数据。

（2）9 个方面的信息之间使用一个空格间隔。

（3）后续任务一样。

```
/**
*函数功能：实现将接收的结构体中的数据按指定格式写入文件中
*函数输入：@gData，保存枪械信息的结构体
*函数返回：1 为操作成功，0 为操作失败
*/
int addGunInfo(struct   GunModel   gData)
{
    int jieguo=1;   //用于返回状态
    FILE *fp=fopen("guninfo.txt","a");   //打开文件
    if(!fp)
    {
        jieguo=0;   //如果对文件操作有误
    }
    //按指定的格式将数据写入文件中
    fprintf(fp,"%s %s %s %d %d %s %d %s %.3f\n",
    gData.NX,gData.MC,gData.DY,gData.RL,gData.SH,gData.GLJL,
    gData.SS,gData.SJMS,gData.SJJG);
    //关闭文件
    fclose(fp);
    //返回操作结果
    return jieguo;
}
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

实现了业务逻辑层添加枪械信息功能函数。此时，运行程序应该能够看到如图 9.32 所示的界面，说明本任务已顺利完成。

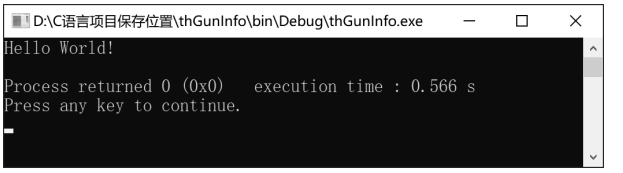


图 9.32　任务运行效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

在业务逻辑层（GunManage.h）中，完成“吃鸡游戏枪械信息管理系统”项目添加枪械信息的功能函数。

● 技术层面

分析设计。

函数定义。

● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 4　业务逻辑层——加载枪械信息实现

目标描述

任务描述
● 目标及要求 根据项目设计，完成业务逻辑层加载枪械信息的功能。 具体要求如下： 在业务逻辑层（GunManage.h）中，实现加载枪械信息的功能函数。

学习活动 1　接领任务

领任务单
● 任务确认 实现业务逻辑层加载的功能，其具体要求如下： （1）实现业务逻辑层加载枪械信息的功能函数； （2）应命名规范，注释清晰。 ● 确认签字

学习活动 2　分析任务

分析任务
实现“吃鸡游戏枪械信息管理系统”项目的业务逻辑层加载枪械的信息功能，将保存在文件中的枪械信息读取出来，并以结构体的形式返回给表示层，从而实现信息的加载。 以单独函数的方式实现。

学习活动 3　制定方案

实现本任务方案
● 实现思路 实现“吃鸡游戏枪械信息管理系统”项目的业务逻辑层加载枪械信息的功能函数，其具体实现思路如下。 （1）函数名称：loadGunInfo； （2）函数输入：无； （3）函数功能：读取文件中的枪械信息，并保存到结构体数组中；

（4）函数返回：结构体数组，如图 9.33 所示。

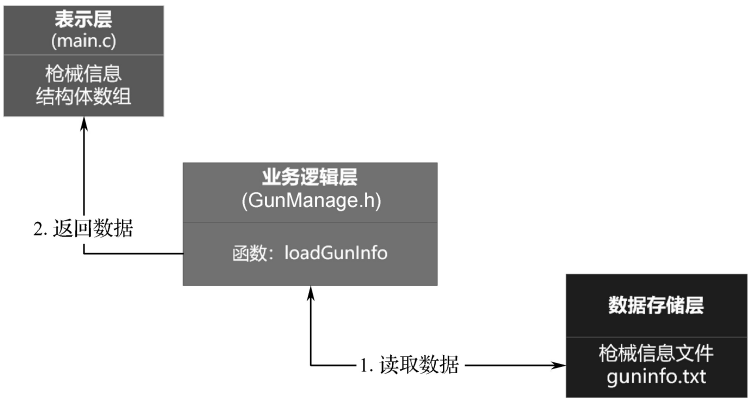


图 9.33 业务逻辑层加载功能示意

● 实现步骤

（1）打开之前创建好的项目；

（2）在业务逻辑层（GunManage.h）中完成。

学习活动 4 实施实现

任务实现

● 实现参考

通过上面的分析，进入 CodeBlocks 软件实现加载枪械信息的功能函数，其步骤如下。

（1）打开 GunManage.h 文件。

展示项目的 Headers，双击“GunManage.h”文件，进入编辑。

（2）实现加载枪械信息的功能函数。

根据上以分析，完成函数的功能编写，其代码如下：

```
/**
*函数功能: 实现读取文件中的枪械信息，并保存到结构体数组中返回
*函数输入：无
*函数返回：结构体数组
*/
struct  GunModel  *loadGunInfo()
{
    FILE  *fp=fopen("guninfo.txt","r");  //以只读方式，打开文件
    //定义保存枪械信息的结构体数组
    struct  GunModel  gM[MAXSIZE];
    //读取到文件尾
    int  i=0;
    while(!feof(fp))
    {
        if(fscanf(fp,"%s %s %s %d %d %s %d %s %f\n",
```

```
                    &gM[i].NX,&gM[i].MC,&gM[i].DY,&gM[i].RL,&gM[i].SH,
                    &gM[i].GLJL,&gM[i].SS,&gM[i].SJMS,&gM[i].SJJG)==9)
            {
                i++;
            }
        }
        //记录真实的枪械数目
        TOTALCOUNT=i;
        //关闭文件
        fclose(fp);
        //返回结构体数组
        return  gM;
    }
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

实现了业务逻辑层加载枪械信息的功能函数。

此时，运行程序能够看到如图 9.34 所示的界面，说明本任务已顺利完成。

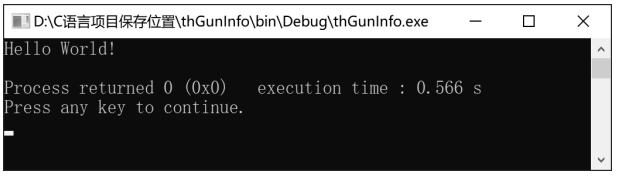


图 9.34　任务运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字

学习活动 6　总结拓展

任务总结与拓展
● 实现效果 在业务逻辑层（GunManage.h）中，实现了“吃鸡游戏枪械信息管理系统”项目加载枪械信息的功能函数。 ● 技术层面 分析设计。 函数定义。 ● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等） --- --- ---

任务 5　业务逻辑层——修改枪械信息实现

目标描述

任务描述
● 目标及要求 根据项目设计，完成业务逻辑层修改枪械信息的功能。 具体要求如下： 在业务逻辑层（GunManage.h）中，实现修改枪械信息的功能函数。

学习活动 1　接领任务

领任务单
● 任务确认 实现业务逻辑层的修改功能。具体要求如下： （1）正确完成业务逻辑层修改枪械信息的功能函数； （2）命名规范，注释清晰。 ● 确认签字 ---

学习活动 2　分析任务

分析任务
实现“吃鸡游戏枪械信息管理系统”项目业务逻辑层修改枪械的信息功能，该功能可将表示层传出的新数据重新写入文件中，从而实现信息的修改。 以单独函数的方式实现。

学习活动 3　制定方案

实现本任务方案
● 实现思路 实现“吃鸡游戏枪械信息管理系统”项目业务逻辑层修改枪械信息的功能函数，具体实现思路如下。 （1）函数名称：editGunInfo； （2）函数输入：结构体数组等内容； （3）函数功能：将结构体中的新数据更新到文件中； （4）函数返回：操作成功/失败，具体如图 9.35 所示。

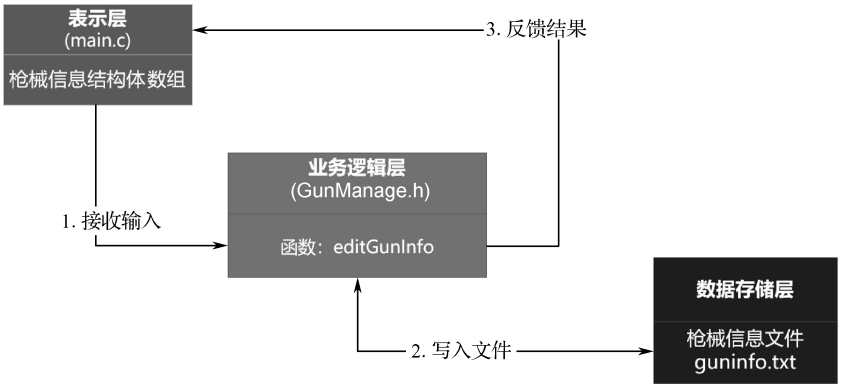


图 9.35　业务逻辑层修改功能示意

● 实现步骤

（1）打开之前创建好的项目；

（2）在业务逻辑层（GunManage.h）中完成。

学习活动 4　实施实现

任务实现
● 实现参考 通过上面的分析，进入 CodeBlocks 软件实现修改枪械信息功能函数，实现步骤如下。

（1）打开 GunManage.h 文件。

展示项目的 Headers，双击 GunManage.h 文件进入编辑。

（2）实现修改枪械信息功能函数。

根据上以分析，完成函数的功能编写，其代码如下：

```
/**
*函数功能：将结构体中的新数据更新到文件中
*函数输入：
*                @gM: 保存枪械信息的一个结构体
*                @totalcount: 枪械总数
*函数返回：1 为成功，0 为操作失败
*/
int editGunInfo(struct GunModel *gM, int totalcount)
{
    int   jieguo=1;
    //打开文件
    FILE *fp=fopen("guninfo.txt","w");
    if(!fp)
    {
        jieguo=0;   //如果对文件操作有误
    }
    //更新文件数据
    int i;
    for(i=0; i<totalcount; i++)
    {
        fprintf(fp,"%s %s %s %d %d %s %d %s %.3f\n",
            gM[i].NX,gM[i].MC,gM[i].DY,gM[i].RL,gM[i].SH,
            gM[i].GLJL,gM[i].SS,gM[i].SJMS,gM[i].SJJG);
    }
    //关闭文件
    fclose(fp);
    return jieguo;
}
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

实现了业务逻辑层修改枪械信息的功能函数。

此时，运行程序能够看到如图 9.36 所示的界面，说明本任务已顺利完成。

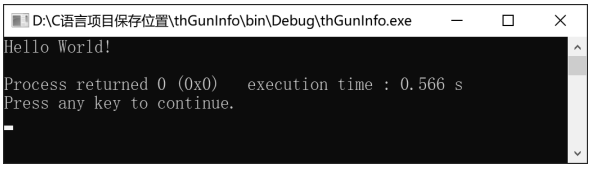


图 9.36　任务运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

在业务逻辑层（GunManage.h）中，完成“吃鸡游戏枪械信息管理系统”项目修改枪械信息的功能函数。

● **技术层面**

分析设计。

函数定义。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 6　业务逻辑层——删除枪械信息实现

目标描述

任务描述
● 目标及要求 根据项目设计，实现业务逻辑层删除枪械信息的功能，具体要求如下： 在业务逻辑层（GunManage.h）中，实现删除枪械信息的功能函数。

学习活动 1　接领任务

领任务单
● 任务确认 实现业务逻辑层的删除功能，具体要求如下： （1）完成业务逻辑层删除枪械信息的功能函数； （2）命名规范，注释清晰。 ● 确认签字

学习活动 2　分析任务

分析任务
实现“吃鸡游戏枪械信息管理系统”项目业务逻辑层删除枪械信息的功能函数，该功能可将表示层传出的新数据重新写入文件中，从而实现信息的删除。 以单独函数的方式实现。

学习活动 3　制定方案

实现本任务方案
● 实现思路 实现“吃鸡游戏枪械信息管理系统”项目业务逻辑层删除枪械信息的功能函数，具体思路如下： （1）函数名称：delGunInfo； （2）函数输入：结构体数组等内容； （3）函数功能：删除枪械信息； （4）函数返回：操作成功/失败，具体如图 9.37 所示。

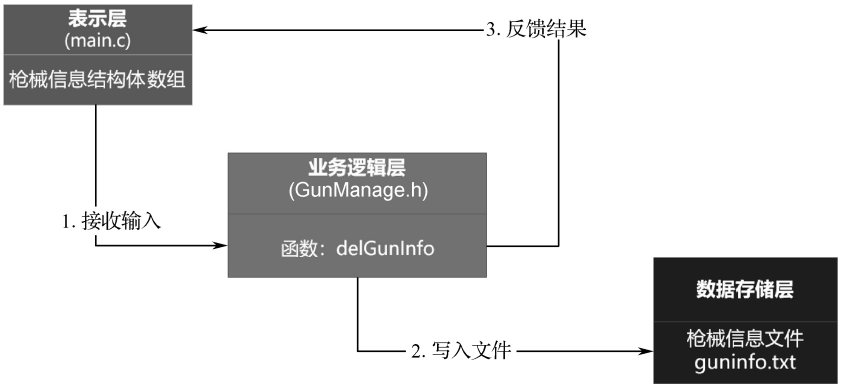


图 9.37　业务逻辑层删除功能示意

● 实现步骤

（1）打开之前创建好的项目；

（2）在业务逻辑层（GunManage.h）中完成。

学习活动 4　实施实现

任务实现

● 实现参考

通过上面的分析，进入 CodeBlocks 软件实现删除枪械信息的功能函数，其步骤如下：

（1）打开 GunManage.h 文件。

展示项目的 Headers，双击 GunManage.h 文件进入编辑状态。

（2）实现删除枪械信息的功能函数。

根据以上分析，完成函数的功能编写，其代码如下：

```
/**
*函数功能：实现删除枪械信息
*函数输入：
*@gM：枪械信息的结构体
*@id：被删除的枪械的编号
*@totalcount：枪械总数
*函数返回：1 为成功，0 为失败
*/
int delGunInfo(struct  GunModel  *gM,int id,int totalcount)
{
    int jieguo=1;
    //打开文件
    FILE *fp=fopen("guninfo.txt","w");
    if(!fp)
    {
        jieguo=0;
    }
```

```
        //删除功能的实现
        int i;
        for(i=0; i<totalcount; i++)
        {
            //id：删除枪械编号。跳过这个编号，不对其进行写操作，从而实现删除
            if(i==id-1)
            {
                continue;
            }
            //写数据
            fprintf(fp,"%s %s %s %d %d %s %d %s %.3f\n",
                gM[i].NX,gM[i].MC,gM[i].DY,gM[i].RL,gM[i].SH,
                gM[i].GLJL,gM[i].SS,gM[i].SJMS,gM[i].SJJG);
        }
        //关闭文件
        fclose(fp);
        return jieguo;
    }
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

实现了业务逻辑层删除枪械信息的功能函数。

此时，运行程序看到如图 9.38 所示的界面，说明本任务已顺利完成。

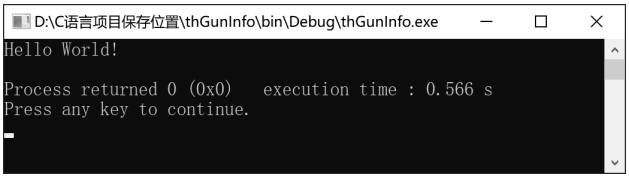


图 9.38　任务运行效果

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展
● 实现效果 在业务逻辑层（GunManage.h）中，完成“吃鸡游戏枪械信息管理系统”项目删除枪械信息的功能函数。 ● 技术层面 分析设计。 函数定义。 ● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等） --- --- ---

任务 7　表示层——显示枪械界面实现

目标描述

任务描述
● 目标及要求 根据项目设计完成项目表示层，显示枪械信息的界面实现，具体要求如下： 在表示层（main.c）中，实现调用业务逻辑层（GunManage.h）中加载枪械信息的功能函数，获取数据，并显示在界面上。

学习活动 1　接领任务

领任务单
● 任务确认 实现表示层显示枪械信息的界面，具体要求如下：

（1）正确完成表示层枪械信息界面的显示；
（2）单独以函数实现；
（3）命名规范，注释清晰。

● **确认签字**

学习活动 2　分析任务

分析任务

实现表示层（main.c）显示枪械信息的界面，具体分析如下：
（1）在表示层中以函数实现，函数名称为 showGunView()；
（2）调用业务逻辑层（GunManage.h）中的函数 loadGunInfo 获取数据；
（3）设计显示界面，将枪械信息进行显示，具体如图 9.39 所示。

图 9.39　显示枪械信息界面示意

学习活动 3　制定方案

实现本任务方案

● **实现思路**

实现表示层显示枪械信息的界面，采用函数实现，具体实现思路如下：
（1）函数名称：showGunView()；
（2）函数输入：无；
（3）函数功能：将调用业务逻辑层 loadGuninfo()返回的结构体数据进行显示；
（4）函数返回：无。

● **实现步骤**

（1）打开之前创建好的项目；
（2）在表示层（main.c）中完成。

学习活动 4　实施实现

任务实现

● **实现参考**

通过上面的分析，进入项目实现显示枪械界面的函数。

● **实现步骤**

（1）打开 main.c 文件。

展示项目的“Sources”，双击“main.c”文件，进入编辑状态。

（2）实现显示枪械界面的函数。

从本任务开始将进行表示层的界面实现，进入 main.c 文件中。

首先，定义一个枪械信息结构体的全局变量，使用该变量交互表示层与业务逻辑层的枪械信息。

其次，在全局结构体变量与 main()之间实现本任务，如图 9.40 所示。

```
#include <stdio.h>
#include <stdlib.h>

// 调用GunManage.h
#include "GunManage.h"

// 定义全局的结构体变量
struct GunModel *GUN;


int main()
{
    // 调用显示系统主界面
    mainMenuView();
    return 0;
}
```

图 9.40　定义全局结构体变量

实现枪械信息显示的界面函数为 showGunView，其代码如下：

```
/* 显示枪械信息界面 */
void    showGunView()
{
    system("cls"); //清屏幕
    printf("\n");
    printf("************************************************************吃鸡游戏枪械信息管理");
    printf("*************************************************************\n\n");
    //显示标题
    printf("编号 \t 类型 \t\t 名称 \t\t 弹药\t\t 弹夹容量 \t 伤害 \t 归零距离 \t 射速 \t\t 射击
        模式 \t 射击间隔 \n");
    printf("-----------------------------------------------------------------------------------------------------\n");
    //调用业务逻辑层加载枪械信息的函数
    GUN=loadGunInfo();
    //循环显示所有枪械信息
    int i;
```

```
        for(i=0; i<TOTALCOUNT; i++)
        {
            printf("%3d \t %s \t\t %s \t\t %s \t\t %d \t %10d \t %s \t %d \t %s \t %.3f s\n",i+1, GUN[i].NX,
                    GUN[i].MC,GUN[i].DY,GUN[i].RL,GUN[i].SH,GUN[i].GLJL,GUN[i]. SS,
                    GUN[i].SJMS,GUN[i].SJJG);
        }
        printf("-------------------------------------------------------------------------------------------------------\n");
    }
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

在 main 主函数中调用此函数，运行程序看到如图 9.41 所示的界面，说明本任务已顺利完成。

图 9.41　任务运行效果

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展
● 实现效果 在“吃鸡游戏枪械信息管理系统”项目表示层（main.c）中实现了显示枪械界面函数。 ● 技术层面 分析设计。 函数定义。 ● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 8　表示层——添加枪械界面实现

目标描述

任务描述
● 目标及要求 根据项目设计表示层，实现添加枪械界面，具体要求如下： （1）在表示层（main.c）中通过函数实现添加枪械界面。 （2）调用业务逻辑层（GunManage.h）中添加枪械信息的功能函数，实现写入文件。

学习活动 1　接领任务

领任务单
● 任务确认 实现表示层添加枪械界面，具体要求如下： （1）正确完成表示层添加枪械界面； （2）单独以函数实现； （3）命名规范，注释清晰。 ● 确认签字

学习活动 2　分析任务

分析任务

实现表示层（main.c）添加枪械界面，具体分析如下：

（1）在表示层中以函数实现，函数名称为 addGunView()；

（2）调用业务逻辑层（GunManage.h）中的 addGunInfo()添加数据；

（3）设计添加界面，添加枪械信息，具体如图 9.42 所示。

图 9.42　表示层添加枪械界面示意

学习活动 3　制定方案

实现本任务方案

● **实现思路**

实现表示层添加枪械界面，采用函数实现，具体实现思路如下：

（1）函数名称：addGunView()；

（2）函数输入：无；

（3）函数功能：调用业务逻辑层 addGuninfo()实现添加数据到文件中；

（4）函数返回：无。

● **实现步骤**

（1）打开之前创建好的项目；

（2）在表示层（main.c 中）完成。

学习活动 4　实施实现

任务实现

● **实现参考**

通过上面分析进入项目，实现表示层添加枪械界面的函数。

● **实现步骤**

（1）打开 main.c 文件。

展示项目的“Sources”，双击“main.c”文件，进入编辑状态。

（2）实现枪械信息添加的界面函数。

实现表示层枪械信息添加的界面函数为 addGunView()，其代码如下：

```
/*   添加枪械界面   */
void addGunView()
{
    //清除屏幕
    system("cls");
    printf("\n 枪械管理>添加枪械界面\n");
    //1 定义记录枪械的信息结构体
    struct   GunModel   _gunM;

    while(1)   //死循环
    {
        //2 接收用户的输入
        printf("\n 请输入枪械类型：");
        scanf("%s",&_gunM.NX);
        printf("\n 请输入枪械名称：");
        scanf("%s",&_gunM.MC);
        printf("\n 请输入枪械弹药：");
        scanf("%s",&_gunM.DY);
        printf("\n 请输入弹夹容量：");
        scanf("%d",&_gunM.RL);
        printf("\n 请输入伤害值：");
        scanf("%d",&_gunM.SH);
        printf("\n 请输入归零距离：");
        scanf("%s",&_gunM.GLJL);
        printf("\n 请输入射速：");
        scanf("%d",&_gunM.SS);
        printf("\n 请输入射击模式：");
        scanf("%s",&_gunM.SJMS);
        printf("\n 请输入射击间隔：");
        scanf("%f",&_gunM.SJJG);
        //调用业务层添加枪械的函数，并将用户输入的信息写入文件
        if(addGunInfo(_gunM)==1)   //操作成功
        {
            char   test;
            printf("\n 添加成功！是否继续添加(Y/N)?");
            scanf("%s",&test);
            //判断用户输入的是否是 y
            if(test=='Y' || test=='y')
            {
                //继续添加
                continue;
            }
            else
            {
                //返回主界面
```

```
                    break;   //跳出循环
                }
            }
            else      //操作失败
            {
                printf("\n 添加失败！");
                break; //跳出循环
            }
        }
    }
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

实现了表示层添加枪械界面的函数。在主函数 main 中，调用该函数测试是否能完成将添加的信息写入文件，如果能正确写入则表示成功，否则表示失败。

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

在“吃鸡游戏枪械信息管理系统”项目表示层（main.c）中实现了添加枪械界面的函数。

● **技术层面**

分析设计。

函数定义。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 9　表示层——修改枪械界面实现

目标描述

任务描述
● **目标及要求** 根据项目设计表示层，实现修改枪械界面，具体要求如下： （1）在表示层（main.c）中通过函数实现修改枪械界面。 （2）调用业务逻辑层（GunManage.h）中修改枪械信息的功能函数实现写文件。

学习活动 1　**接领任务**

领任务单
● **任务确认** 实现表示层修改枪械界面，具体要求如下： （1）正确完成表示层修改枪械界面； （2）单独以函数实现； （3）命名规范，注释清晰。 ● **确认签字**

学习活动 2　**分析任务**

分析任务
实现表示层（main.c）修改枪械界面，具体分析如下： （1）在表示层中以函数实现，函数名称为 editGunView()； （2）调用业务逻辑层（GunManage.h)中的 editGunInfo()修改数据； （3）设计修改界面，对枪械信息进行修改，具体如图 9.43 所示。

图 9.43　表示层修改枪械界面示意

学习活动 3　制定方案

实现本任务方案

● **实现思路**

实现表示层修改枪械界面，采用函数实现，具体实现思路如下：

（1）函数名称：editGunView()；

（2）函数输入：无；

（3）函数功能：调用业务逻辑层 editGuninfo()实现将新数据修改到文件中；

（4）函数返回：无。

● **实现步骤**

（1）打开之前创建好的项目；

（2）在表示层（main.c）中完成。

学习活动 4　实施实现

任务实现

● **实现参考**

实现表示层修改枪械界面，具体步骤如下：

（1）打开 main.c 文件。

展示项目的“Sources”，双击“main.c”文件，进入编辑状态。

（2）实现枪械信息修改的界面函数。

实现枪械信息修改的界面函数为 editGunView，其代码如下：

```
/*   修改枪械界面   */
void editGunView()
{
    //清除屏幕
    system("cls");
    //定义变量用来记录被修改枪械的编号
    int id;
    //显示枪械列表
    showGunView();
```

```
//开始循环（因为不确定用户要修改几次）
while(1)
{
    //初始化变量的初始值
    id =0;
    //提示输入修改的编号
    printf("\n 请输入要被修改的枪械的编号[0：返回主界面]：");
    scanf("%d",&id);   //被修改的枪械编号
    if(id==0)   //如果编号为 0，则返回主界面
    {
        break;
    }
    //判断输入编号有效性
    if(id>0 &&   id<=TOTALCOUNT)
    {
        //获取新信息并保存到对应的结构体数组的元素中（修改编号  id -1)
        printf("\n 请输入枪械类型：");
        scanf("%s",&GUN[id-1].NX);
        printf("\n 请输入枪械名称：");
        scanf("%s",&GUN[id-1].MC);
        printf("\n 请输入枪械弹药：");
        scanf("%s",&GUN[id-1].DY);
        printf("\n 请输入弹夹容量：");
        scanf("%d",&GUN[id-1].RL);
        printf("\n 请输入伤害值：");
        scanf("%d",&GUN[id-1].SH);
        printf("\n 请输入归零距离：");
        scanf("%s",&GUN[id-1].GLJL);
        printf("\n 请输入射速：");
        scanf("%d",&GUN[id-1].SS);
        printf("\n 请输入射击模式：");
        scanf("%s",&GUN[id-1].SJMS);
        printf("\n 请输入射击间隔：");
        scanf("%f",&GUN[id-1].SJJG);

        //调用业务逻辑层修改函数
        if(editGunInfo(GUN,TOTALCOUNT)==1)
        {
            //刷新显示
            showGunView();
            //询问是否继续
            char test;
            printf("\n 修改成功！是否继续修改(Y/N)?");
            scanf("%s",&test);
            //判断用户输入的是否是 y
            if(test=='Y' || test=='y')
            {
                //继续修改
                continue;
```

```
                    }
                    else
                    {
                        //返回主界面
                        break;   //跳出循环
                    }
                }
                else
                {
                    //失败
                    printf("\修改失败！ ");
                }
            }
            else
            {
                printf("\n 请输入正确的编号");
            }
        }
    }
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

实现了表示层修改枪械界面的函数。在主函数 main 中测试是否能完成将修改信息写到文件中，如果写入正确则成功，否则失败。

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展
● 实现效果 本任务在“吃鸡游戏枪械信息管理系统”项目表示层（main.c）中实现了修改枪械界面的函数。 ● 技术层面 分析设计。 函数定义。 ● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等） --- --- ---

任务 10　表示层——删除枪械界面实现

目标描述

任务描述
● 目标及要求 根据项目设计表示层，实现删除枪械界面，具体要求如下： （1）在表示层（main.c）中通过函数实现删除枪械界面。 （2）调用业务逻辑层（GunManage.h）中删除枪械信息的功能函数，实现写入文件。

学习活动 1　接领任务

领任务单
● 任务确认 实现表示层删除枪械界面，具体要求如下： （1）正确完成表示层删除枪械界面； （2）单独以函数实现； （3）命名规范，注释清晰。 ● 确认签字 ---

学习活动 2　分析任务

分析任务

实现表示层（main.c）删除枪械界面，具体分析如下：

（1）在表示层中以函数实现，函数名称为 delGunView()；

（2）调用业务逻辑层（GunManage.h)中的 delGunInfo()删除数据；

（3）设计删除界面，删除枪械信息，具体如图 9.44 所示。

图 9.44　删除枪械界面示意

学习活动 3　制定方案

实现本任务方案

● **实现思路**

实现表示层删除枪械界面，采用函数实现，具体思路如下：

（1）函数名称：delGunView()；

（2）函数输入：无；

（3）函数功能：调用业务逻辑层 delGuninfo()实现数据文件中数据的删除；

（4）函数返回：无。

● **实现步骤**

（1）打开之前创建好的项目；

（2）在表示层（main.c）中完成。

学习活动 4　实施实现

任务实现

● **实现参考**

实现表示层删除枪械界面，具体步骤如下：

（1）打开 main.c 文件。

展示项目的“Sources”，双击“main.c”文件，进入编辑状态。

（2）实现枪械信息删除的界面函数。

实现枪械信息删除的界面函数为 delGunView()，其代码如下：

```
/*   实现删除枪械信息的界面   */
void delGunView()
{
    //定义被删除的枪械的编号
    int id;

    //显示枪械信息列表
    showGunView();

    while(1)
    {
        printf("\n 请输入要删除的枪械编号[0：返回主界面]：");
        scanf("%d",&id);

        //判断输入编号是否在正常范围内
        if( id>0 && id<=TOTALCOUNT)
        {
            //调用业务逻辑层的函数删除枪械信息
            if(delGunInfo(GUN,id,TOTALCOUNT)==1)
            {
                //操作成功
                //刷新数据
                showGunView();

                char flag;
                printf("\n  删除成功！是否继续删除(y/n)?");
                scanf("%s",&flag);
                if(flag=='Y' || flag=='y')
                {
                    continue;   //继续
                }
                else
                {
                    break;   //跳出循环，结束
                }
            }
            else
            {
                //操作失败
                printf("\n  删除失败！");
                continue;
            }
        }
        else
        {
```

```
                printf("\请输入正确的编号");
                continue;
            }
        }
    }
```

学习活动 5　测试验收

任务测试验收单

● **实现效果**

实现了表示层删除枪械界面的函数。在主函数 main 中测试是否能完成删除。

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

在“吃鸡游戏枪械信息管理系统”项目表示层（main.c）中，实现了删除枪械界面的函数。

● **技术层面**

分析设计。

函数定义。

● **任务小结**（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 11　表示层——项目主界面菜单实现

目标描述

任务描述
● **目标及要求** 完成项目主界面菜单实现，具体要求如下： （1）在表示层（main.c）中通过函数实现主界面。 （2）实现主界面菜单的选择，并调用对应已实现的业务界面函数。

学习活动 1　**接领任务**

领任务单
● **任务确认** 实现项目主界面菜单界面，具体要求如下： （1）实现信息显示及菜单显示，并实现选择进入对应界面的功能； （2）单独以函数实现； （3）命名规范，注释清晰。 ● **确认签字**

学习活动 2　**分析任务**

分析任务
实现项目的主界面菜单功能，如图 9.45 所示。 系统启动进入界面，显示枪械信息，然后实现系统的主菜单，即输入 1 进入添加界面，输入 2 时进入修改界面，输入 3 时进入删除界面，输入 0 时退出系统。

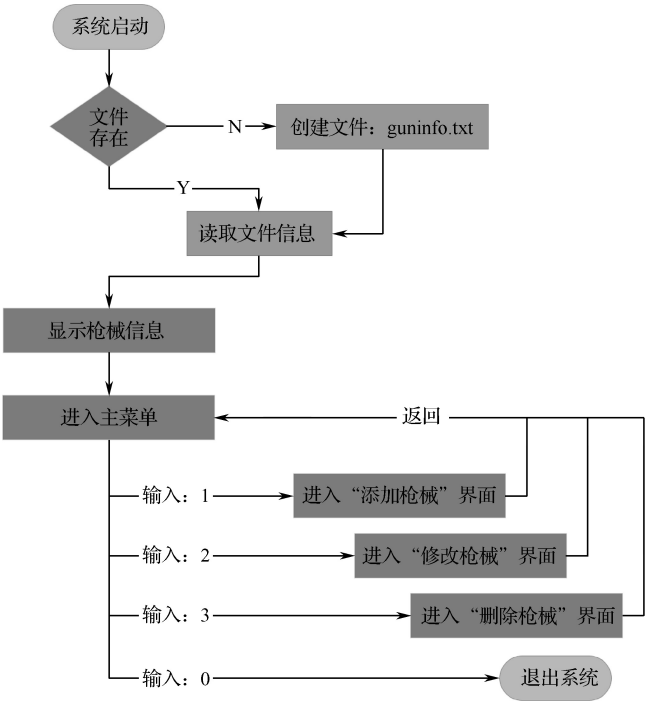


图 9.45　项目主界面菜单示意

学习活动 3　制定方案

实现本任务方案
● 实现思路 实现表示层项目主界面菜单的功能，具体思路如下： （1）函数名称：mainMenuView()； （2）函数输入：无； （3）函数功能：主要实现主界面菜单的功能； （4）函数返回：无。 ● 实现步骤 （1）打开之前创建好的项目； （2）在表示层（main.c）中完成。

学习活动 4　实施实现

任务实现
● 实现参考 通过上面的分析进入项目，实现项目主界面函数。

● 实现步骤

（1）打开 main.c 文件。

展示项目的“Sources”，双击“main.c”文件，进入编辑状态。

（2）实现项目主界面菜单函数。

实现项目主界面菜单函数 mainMenuView，其代码如下：

```
/* 系统主界面 */
void mainMenuView()
{
    //定义功能号变量
    int functionno;

    while(1)
    {
        //调用显示枪械信息界面，以显示枪械信息
        showGunView();
        //显示功能操作菜单
        printf("\n\n 功能操作");
        printf("\n\n \t 1.添加枪械");
        printf("\n\n \t 2.修改枪械");
        printf("\n\n \t 3.删除枪械");
        printf("\n\n \t 0.退出系统");

        printf("\n\n 请输入你要操作的功能编号:");
        scanf("%d",&functionno);
        //判断输入的菜单编号
        if(functionno >=0 && functionno<=3)
        {
            if(functionno==0)
            {
                //结束程序
                break;
            }
            else if(functionno==1)
            {
                //调用添加
                addGunView();
            }
            else if(functionno==2)
            {
                //调用修改
                editGunView();
            }
            else if(functionno==3)
            {
                //调用删除
                delGunView();
```

```
                }
            }
            else
            {
                printf("\n 输入功能编号有误，请重新输入");
            }
        }
        printf("\n  再见！ ");
    }
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

实现了项目主界面菜单的功能，包含调用显示枪械信息界面、显示系统操作菜单。同时能够根据输入的菜单编号正确进入对应的操作界面（输入 1 时进入添加枪械界面；输入 2 时进入修改枪械界面；输入 3 时进入删除枪械界面；输入 0 时退出系统）。

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

在“吃鸡游戏枪械信息管理系统”项目表示层（main.c）中实现了主界面函数，融合了之前实现的所有表示层的函数（显示枪械信息、添加枪械信息、修改枪械信息、删除枪械信息等）。

● 技术层面

分析设计。

函数定义。

● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 12　表示层——程序主函数实现

目标描述

任务描述
● 目标及要求 完成项目的主函数调用，以最终实现项目开发，具体要求如下： 在表示层（main.c）的主函数中，调用上次任务实现的主界面菜单函数以完成本项目。

学习活动 1　接领任务

领任务单
● 任务确认 调用主界面菜单函数，以实现系统的所有开发，具体要求实现如下： （1）调用上次任务实现的主界面菜单函数； （2）命名规范，注释清晰。 ● 确认签字

学习活动 2　分析任务

分析任务
在上一次任务中已完成“吃鸡游戏枪械信息管理系统”项目的所有功能开发。 对该项目采用按软件三层架构的思路进行设计开发，各功能均使用单独函数进行实现，即模块化实现。这样可增加系统的扩展性和可读性。 在主函数中调用已实现的项目主界面函数，以完成系统的真正开发。

学习活动 3　制定方案

实现本任务方案

● 实现思路

（1）找到项目的主函数；

（2）调用 mainMenuView()。

● 实现步骤

（1）打开之前已创建的项目；

（2）在表示层（main.c）中完成。

学习活动 4　实施实现

任务实现

● 实现参考

通过上次任务的实现，该项目所有功能已开发完成，接下来进入主函数调用，以mainMenuView()实现项目最后的功能。

代码参考如下：

```
int main()
{
    //调用显示枪械界面的函数
    mainMenuView();

    return 0;
}
```

学习活动 5　测试验收

任务测试验收单

● 实现效果

实现了本项目的所有开发，能够正确运行，说明开发成功，具体效果如图 9.46 至图 9.49 所示。

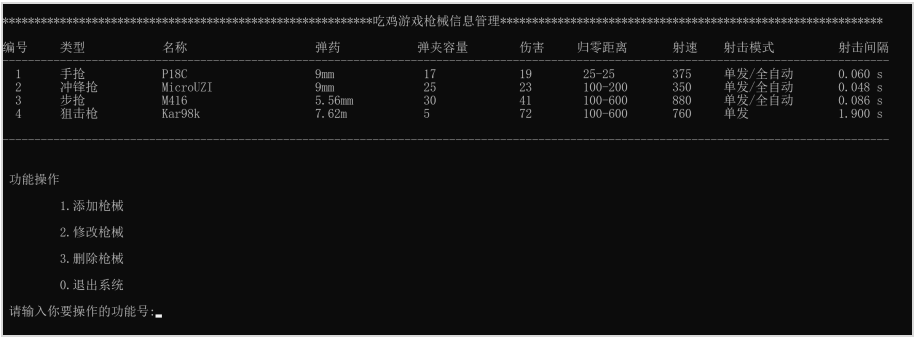


图 9.46　项目主界面

在界面中，输入 1 时进入“添加枪械”的界面。

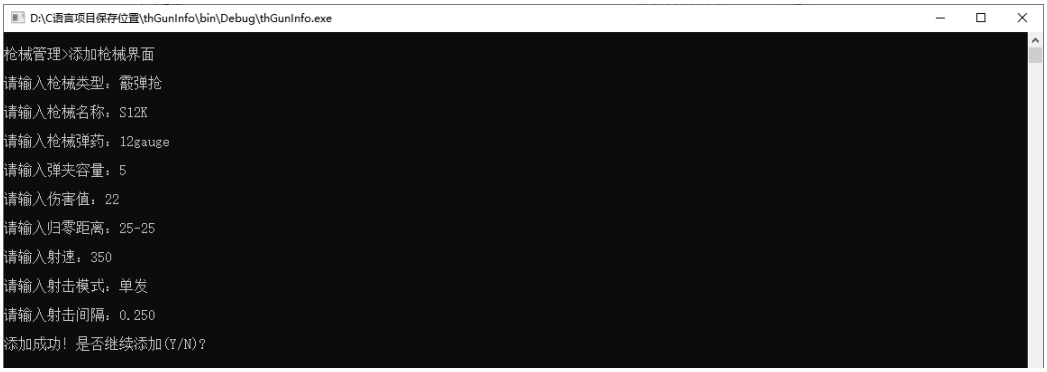


图 9.47　添加信息界面

在界面中，输入 2 时进入“修改枪械”的界面。

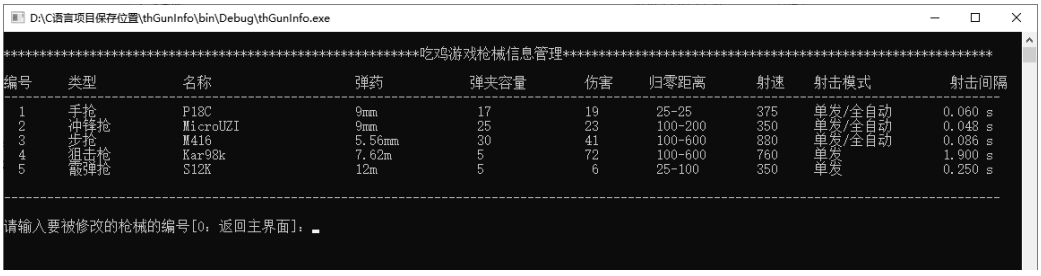


图 9.48　修改信息界面

在界面中，输入 3 时进入“删除枪械”的界面。

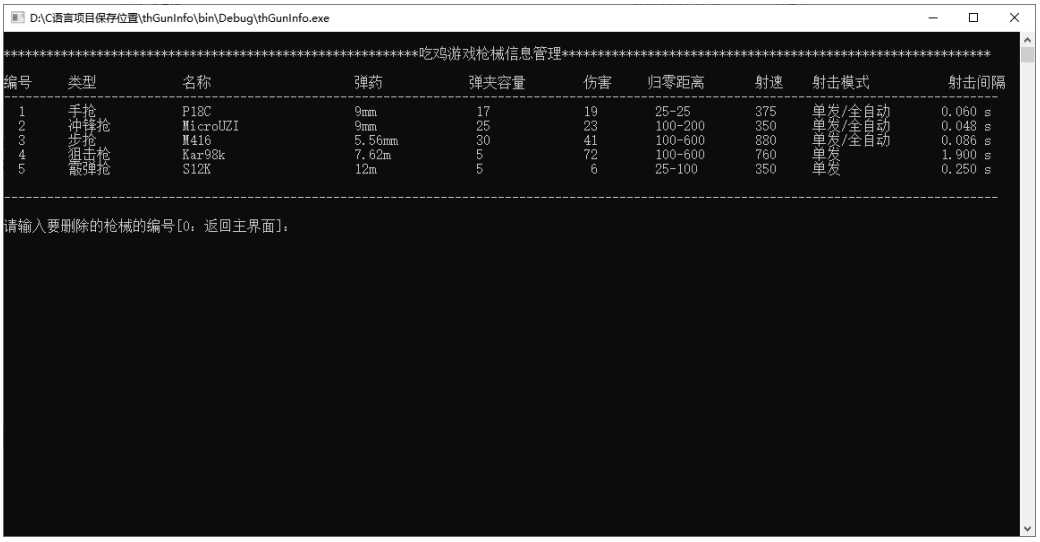


图 9.49　删除信息界面

在界面中，输入 0 时则退出系统。

● 验收结果

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的功能实现情况					
2	使用代码的规范性（变量命名、注释说明）					
3	掌握知识的情况					
4	程序性能及健壮性					
5	团队协作					

说明：在实现效果对应等级中打“√”。

● 验收评价

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● 实现效果

完成“吃鸡游戏枪械信息管理系统”项目中主函数功能的调用，以实现项目的所有开发任务。

本项目按软件三层架构进行设计，所有功能都以单独的函数实现，根据需要进行调用，从而实现了代码利用性，使程序变得更加模板化，并具有很好的扩展性。

同时，整个开发都是按照最初的项目设计逐步实现的。

从另一个角度也引申出：“做你所说，说你所做”。

● 技术层面

分析设计。

函数定义。

● 任务小结（请在此记录你在本任务中对所学知识的理解与实现本任务的感悟等）

任务 13　软件项目文档编写

目标描述

任务描述
● 目标现实 本次开发的项目：吃鸡游戏枪械信息管理系统，主要实现功能如下。 功能 1：实现枪械信息的展示； 功能 2：实现枪械信息的添加； 功能 3：实现枪械信息的修改； 功能 4：实现枪械信息的删除。 到现在已经完成开发，下面将实现对该项目相关软件文档的编写。 ● 具体要求如下： 完成需求分析说明书编写； 完成概要设计文档的编写； 完成用户操作手册的编写。

学习活动 1　接领任务

领任务单
● 任务确认 完成项目软件文档的编写，具体要求如下： （1）按需求分析说明文档格式，完成该项目需求分析文档的编写； （2）按概要设计文档格式，完成该项目概要设计文档的编写； （3）文档格式排版规范。 ● 确认签字

学习活动 2　分析任务

分析任务
一个完整的软件除了有完善的功能，还应配套详细的软件文档。本任务要求简单实现软件需求说明书、概要设计文档、用户操作手册。 （1）软件需求说明书。 说明书的编制是为了使用户和软件开发者对该软件的初始规定有一个共同的理解，使之成为整个开发工作的基础。其中包括硬件、功能、性能、输入与输出、接口需求、警示信息、

保密安全、数据与数据库、文档和法规的要求等。

通俗来讲，编写需求说明书的目的就是说明软件是“做什么的，有什么样的功能”，同时再对软件相关要求与规定进行约束。

（2）概要设计文档。

概要设计文档是一个软件设计师根据用户交互过程和用户需求来形成交互框架和视觉框架的过程，其结果往往以反映交互控件布置、界面元素分组，以及界面整体版式的页面框架图的形式来呈现。

这是一个在用户研究和设计之间架起的桥梁，使用户研究和设计无缝结合，将对用户目标与需求转换成具体界面设计解决方案的重要阶段。

概要设计文档的主要任务是，把需求分析得到的系统扩展用例图转换为软件结构和数据结构。设计软件结构是将一个复杂系统按功能进行模块划分、建立模块的层次结构及调用关系、确定模块间的接口及人机界面等。数据结构设计包括数据特征的描述、确定数据的结构特性、数据库的设计。

概要设计文档的目标是概括设计出系统的逻辑模型。

（3）用户操作手册。

用户手册是详细描述软件的功能、性能和用户界面，使用户了解如何使用该软件，即指导用户使用软件的手册。

学习活动 3　制定方案

实现本任务方案

● **实现思路**

（1）查找文档格式的要求；

（2）结合之前的设计与实现过程完成文档的编写。

● **实现步骤**

（1）将之前在 Visio 等软件中绘制的流程图进行整理；

（2）在 Word/WPS 软件中编写实现。

学习活动 4　实施实现

任务实现

● **实现参考**

1．需求分析说明书内容要求

一般软件需求说明书的内容及要求如下所示。

1．引言

1.1 编写目的

说明编写软件需求说明书的目的，指出预期的读者。

1.2 背景

说明：

a．待开发的软件系统的名称；

b．本项目的任务提出者、开发者、用户及实现该软件的计算中心或计算机网络；

c．该软件系统同其他系统或其他机构基本的相互来往关系。

1.3 定义

列出本文件中用到的专门术语定义和外文首字母组词的原词组。

1.4 参考资料

列出所需的参考资料，如：

a．本项目经核准的计划任务书或合同、上级机关的批文；

b．属于本项目的其他已发表的文件；

c．本文件中引用的文件、资料，包括所要用到的软件开发标准。列出这些文件资料的标题、文件编号、发表日期和出版单位，说明得到这些文件资料的来源。

2．任务概述

2.1 目标

叙述该项软件开发的意图、应用目标、作用范围，以及其他应向读者说明的有关该软件开发的背景材料。解释被开发软件与其他有关软件之间的关系。如果本软件产品是一项独立的软件，而且全部内容自含，则应加以说明。如果所定义的产品是一个更大系统的一个组成部分，则应说明本产品与该系统中其他各组成部分之间的关系，可使用方框图来说明该系统的组成和本产品同其他各部分的联系和接口。

2.2 用户的特点

列出本软件的最终用户的特点，充分说明操作人员、维护人员的教育水平和技术专长，以及本软件的预期使用频度。这些内容是软件设计工作的重要约束。

2.3 假定和约束

列出进行本软件开发工作的假定和约束，如经费限制、开发期限等。

3．需求规定

3.1 对功能的规定

用列表的方式（如 IPO 表即输入、处理、输出表的形式），逐项定量和定性地叙述对软件所提出的功能要求，说明输入什么量、经怎样的处理、得到什么输出，以及软件应支持的终端数和并行操作的用户数。

3.2 对性能的规定

3.2.1 精度

说明对该软件的输入、输出数据精度的要求，包括传输过程中的精度。

3.2.2 时间特性要求

说明对于该软件的时间特性要求，如：

a．响应时间；

b．更新处理时间；

c．数据的转换和传送时间；

d．解题时间。

3.2.3 灵活性

说明对该软件的灵活性的要求，即当需求发生某些变化时，该软件对这些变化的适应能力，如：

a．操作方式上的变化；

b．运行环境的变化；

c．同其他软件的接口变化；

d．精度和有效时限的变化；

e．计划的变化或改进。

对于为了提供这些灵活性而进行专门设计的部分应该加以标明。

3.3 输入和输出要求

解释各输入和输出数据类型，并逐项说明其媒体、格式、数值范围、精度等。对软件的数据输出及必须标明的控制输出量进行解释并举例，包括对复制报告（正常结果输出、状态输出及异常输出），以及

图形或显示报告的描述。

3.4 数据管理能力要求

说明需要管理的文卷和记录的个数、表和文卷的大小规模，要按可预见的增长对数据及其分量的存储要求进行估算。

3.5 故障处理要求

列出可能的软件、硬件故障，以及对各项性能而言所产生的后果和对故障处理的要求。

3.6 其他专门要求

如用户单位对安全保密的要求、使用方便的要求、可维护性、可补充性、易读性、可靠性、运行环境和可转换性的特殊要求等。

4 运行环境规定

4.1 设备

列出运行该软件所需要的硬设备。说明其中新型设备及其专门功能，包括：

a. 处理器型号及内存容量；

b. 外存容量、联机或脱机、媒体及其存储格式，以及设备的型号和数量；

c. 输入和输出设备的型号、数量，以及联机或脱机；

d. 数据通信设备的型号和数量；

e. 功能键及其他专用硬件。

4.2 支持软件

列出支持软件，包括操作系统、编译（或汇编）程序、测试支持软件等。

4.3 接口

说明该软件同其他软件之间的接口、数据通信协议等。

4.4 控制

说明控制该软件运行的方法和控制信号，以及这些控制信号的来源。

2．软件概要设计内容要求

一般软件概要设计内容及要求如下。

1．引言

1.1 编写目的

说明编写这个概要设计说明书的目的，指出预期的读者。

1.2 背景

说明：

a. 待开发软件系统的名称；

b. 列出此项目的任务提出者、开发者、用户，以及将运行该软件的计算站（中心）。

1.3 定义

列出本文件中用到的专门术语的定义和外文首字母组词的原词组。

1.4 参考资料

列出有关的参考文件，如：

a. 本项目经核准的计划任务书或合同，以及上级机关的批文；

b. 属于本项目的其他已发表文件；

c. 本文件中各处引用的文件、资料，包括所要用到的软件开发标准。列出这些文件的标题、文件编号、发表日期和出版单位，说明能够得到这些文件资料的来源。

2．总体设计

2.1 需求规定

说明对本系统主要的输入和输出项目、处理的功能性能要求。

2.2 运行环境

简要说明对本系统的运行环境（包括硬件环境和支持环境）的规定。

2.3 基本设计概念和处理流程

说明本系统的基本设计概念和处理流程，尽量使用图表的形式。

程序设计的基本概念有程序、数据、子程序、子例程、协同例程、模块，以及顺序性、并发性、并行性和分布性等。

2.4 结构

用一览表及框图的形式说明本系统的系统元素（各层模块、子程序、公用程序等）的划分，扼要说明每个系统元素的标识符和功能，分层次地给出各元素之间的控制与被控制关系。

2.5 功能需求与程序的关系

使用矩阵图的形式，说明各项功能需求的实现同各块程序的分配关系。

2.6 人工处理过程

说明在本软件系统的工作过程中不得不包含的人工处理过程（如果有的话）。

2.7 尚未解决的问题

说明在概要设计过程中尚未解决而设计者认为在系统完成之前必须解决的各个问题。

3．接口设计

3.1 用户接口

说明将向用户提供的命令和语法结构，以及软件的回答信息。

3.2 外部接口

说明本系统同外界的所有接口的安排，包括软件与硬件之间的接口，以及本系统与各支持软件之间的接口关系。

3.3 内部接口

说明本系统内各个系统元素之间的接口安排。

4．运行设计

4.1 运行模块组合

说明对系统施加不同的外界运行控制时所引起的各种不同的运行模块组合，以及每种运行所历经的内部模块和支持软件。

4.2 运行控制

说明每种外界运行控制的方式方法和操作步骤。

4.3 运行时间

说明每种运行模块组合将占用各种资源的时间。

5．系统数据结构设计

5.1 逻辑结构设计要点

给出本系统内所使用每个数据结构的名称、标识符，每个数据项、记录、文卷和系的标识、定义、长度，以及它们之间的层次或表格的相互关系。

5.2 物理结构设计要点

给出本系统内所使用每个数据结构中数据项的存储要求，以及访问方法、存取单位、存取的物理关系（索引、设备、存储区域）、设计考虑和保密条件。

5.3 数据结构与程序的关系

说明各个数据结构与访问这些数据结构的形式。

6．系统出错处理设计

6.1 出错信息

用一览表的方式说明每种可能的出错或故障情况出现时，系统输出信息的形式、含义及处理方法。

6.2 补救措施

说明故障出现后可能采取的变通措施，包括：

a. 说明准备采用的后备技术，当原始系统数据万一丢失时启用的副本建立和启动的技术，例如，周期性地把磁盘信息记录到磁带就是磁盘媒体的一种后备技术；

b. 采用降效技术作为后备技术，使用另一个效率稍低的系统或方法来求得所需结果的某些部分，如一个自动系统的降效技术可以是手工操作和数据的人工记录；

c. 恢复及再启动技术说明将使用的恢复再启动技术，使软件从故障点恢复执行或使软件从头开始重新运行的方法。

6.3 系统维护设计

说明为了系统维护的方便而在程序内部设计中做出的安排，包括在程序中专门安排用于系统的检查与维护的检测点和专用模块。各个程序之间的对应关系，可采用矩阵图的形式。

3．用户操作手册的编写要求

用户操作手册的编写没有特别固定的要求，但手册一定要详细描述软件的功能、性能和用户界面，使用户了解到如何使用该软件等。

学习活动 5　测试验收

任务测试验收单

● **实现效果**

按要求完成 3 个文档的编写，并提交验收。

● **验收结果**

序　号	验 收 内 容	实 现 效 果				
		A	B	C	D	E
1	任务要求的文档实现情况					
2	文档的规范性					
3	内容的正确性					
4	团队协作					

说明：在实现效果对应等级中打“√”。

● **验收评价**

验收签字 ---

学习活动 6　总结拓展

任务总结与拓展

● **实现效果**

完成项目的需求说明、概要设计、用户操作手册三个文档的编写。让同学们理解一个完整的软件项目，除有稳健的功能外，还要有翔实的软件文档。在真实的开发过程中，是先有文档，后做开发的。软件文档可起到规范与约定的作用。

● **技术层面**

软件文档的重要性（兵马未动，粮草先行）。