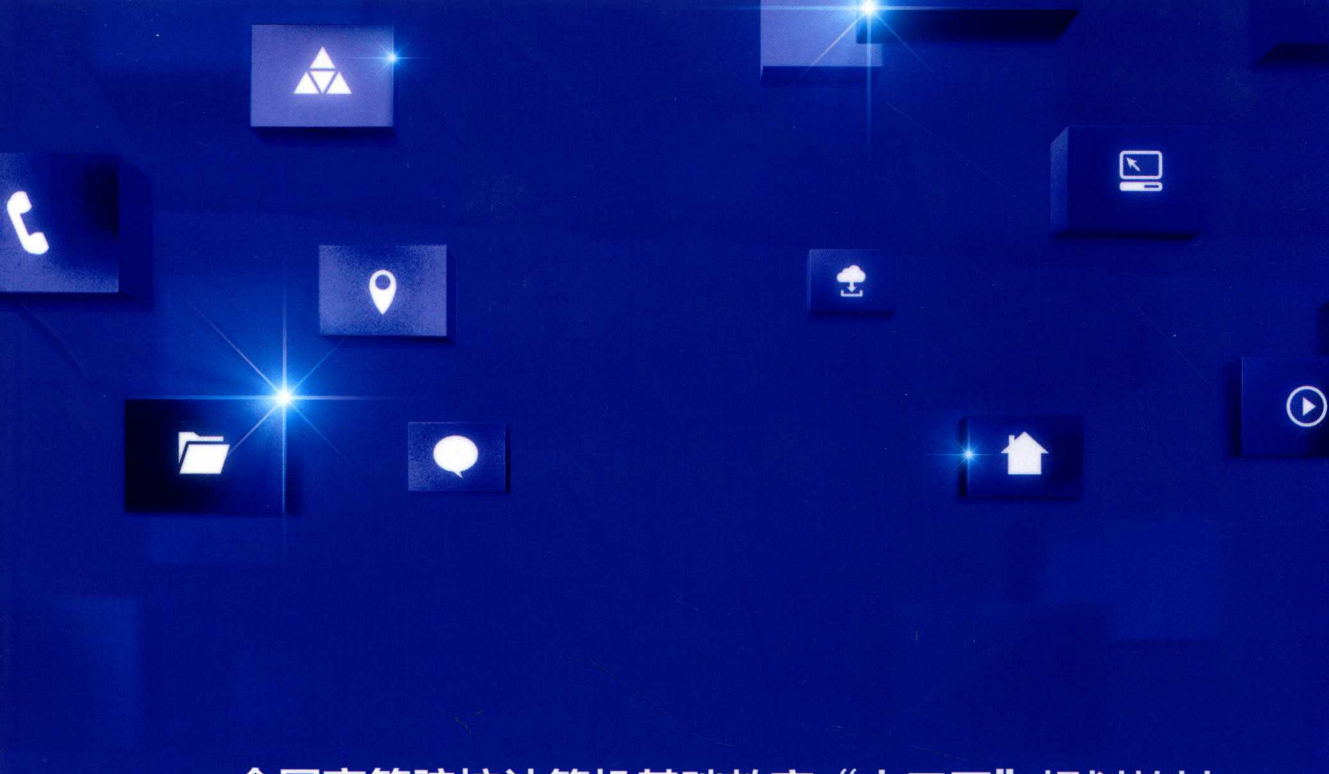

全国高等院校计算机基础教育“十三五”规划教材

程序设计与实践（C）实验教程

CHENGXU SHEJI YU SHIJIAN (C) SHIYAN JIAOCHENG

臧劲松　主　编
黄小瑜　刘丽霞　胡春燕　杨　赞　副主编
夏　耘　主　审

中国铁道出版社有限公司
CHINA RAILWAY PUBLISHING HOUSE CO., LTD.

全国高等院校计算机基础教育“十三五”规划教材

程序设计与实践（C）实验教程

臧劲松 主 编

黄小瑜 刘丽霞 胡春燕 杨 赞 副主编

夏 耘 主 审

中国铁道出版社有限公司
CHINA RAILWAY PUBLISHING HOUSE CO., LTD.

CONTENTS

目录

第一部分 实 验

第二部分 练 习

第三部分　基本算法与应用

第一部分

实　验

实验0 系统安装

学习一门编程语言的最好方法是编写程序。一般而言，首先需要解决的问题是将需要的信息输出在屏幕上，例如，输出“你好，世界!”。要实现上述输出，需要考虑程序在哪里输入，如何成功编译程序，如何加载程序，以及如何运行程序。解决上述问题后对程序的学习就变得比较容易了。

C 语言程序的工作环境是指集应用程序的编辑、编译、连接、运行和调试等功能以及可视化软件开发为一体的集成开发环境。下面介绍 Code:Blocks 集成开发环境。

Code::Blocks 是一款免费开源的 C/C++ IDE，支持 GCC、MSVC++ 等多种编译器，还可以导入 Dev-C++ 的项目。Code::Blocks 的优点是：跨平台，在 Linux、Mac、Windows 上都可以运行，且自身体积小，安装非常方便。安装 Code::Blocks 与安装普通软件一样，完全的傻瓜式操作，远没有安装 Visual Studio 那么复杂。对于 Windows 用户，其 Code::Blocks 17.12 下载地址如下：

https://sourceforge.net/projects/codeblocks/files/Binaries/17.12/Windows/codeblocks-17.12mingw-setup.exe/download

对于 Mac 用户，其 Code::Blocks 13.12 的下载地址如下：

https://sourceforge.net/projects/codeblocks/files/Binaries/13.12/MacOS/CodeBlocks-13.12-mac.zip/download

一、Code::blocks 集成开发环境的使用

Code::Blocks 的最新版本 codeblocks-17.12 for windows 可以在官网 www.codeblocks.org/downloads 上免费下载。下载时需先选择 Download the binary release 项，然后选择自带 MinGW 编译器的 codeblocks-17.12mingw-setup.exe 下载项。下面以 codeblocks-13.12 为例介绍该环境下开发 C 语言程序的过程。

Code::Blocks 集成开发环境如图 1-0-1 所示。同 Visual Studion 类似，除常规的标题栏、菜单栏、工具栏和状态栏，其还包括工作区、编辑区、日志区。

Code::Blocks 以工作区（Workspace）形式管理工程（Project），一个工作区中可包含多个工程，每个工程只能包含一个入口函数 main()，在 Workspace 选项卡中可以看到包含的工程名称及其工程中所包括的资源文件。编辑区用于源文件的编辑，以不同颜色来强调程序中包含的关键字（深蓝色）、字符串（浅蓝色）、符号（红色）等。日志区用于显示编译、连接时的提示，以及相关文件的路径、执行的时间等信息。

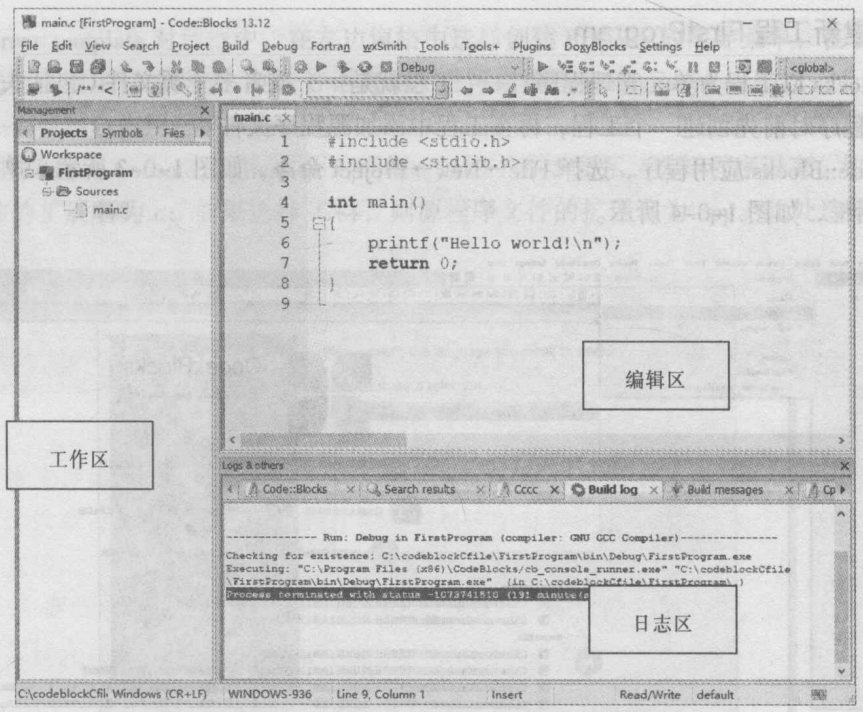

图 1-0-1 Code::Blocks 集成开发环境

Code::Blocks 可以创建单独的 C 程序文件，如图 1-0-2 所示。但单独的文件无法使用调试器，因为 Code::Blocks 的调试器需要一个完整的工程才可以启动。

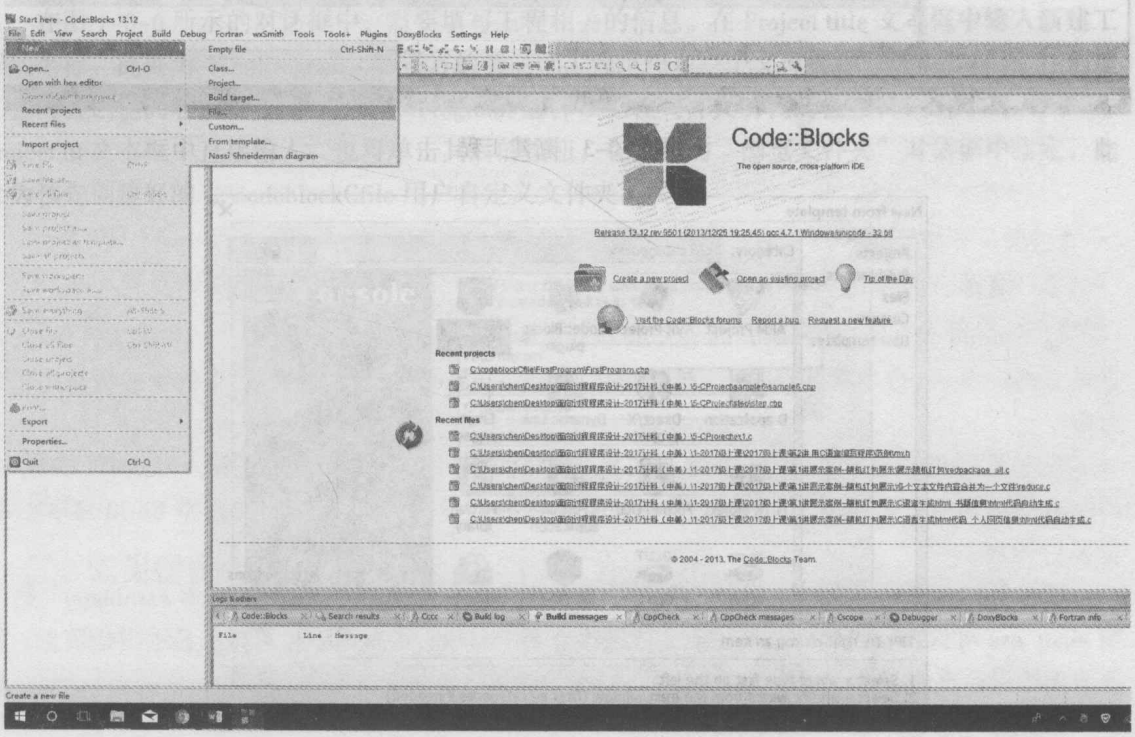

图 1-0-2 创建单独的 C 程序

1. 创建新工程 FirstProgram

Code::Blocks 以工程为单元管理程序，一个工程就是一个或者多个源文件（包括头文件）的集合。创建 C 程序时前先创建一个工程，再在工程中添加源程序文件。

启动 Code::Blocks 应用程序，选择 File→New→Project 命令，如图 1-0-3 所示，弹出 New from template 对话框，如图 1-0-4 所示。

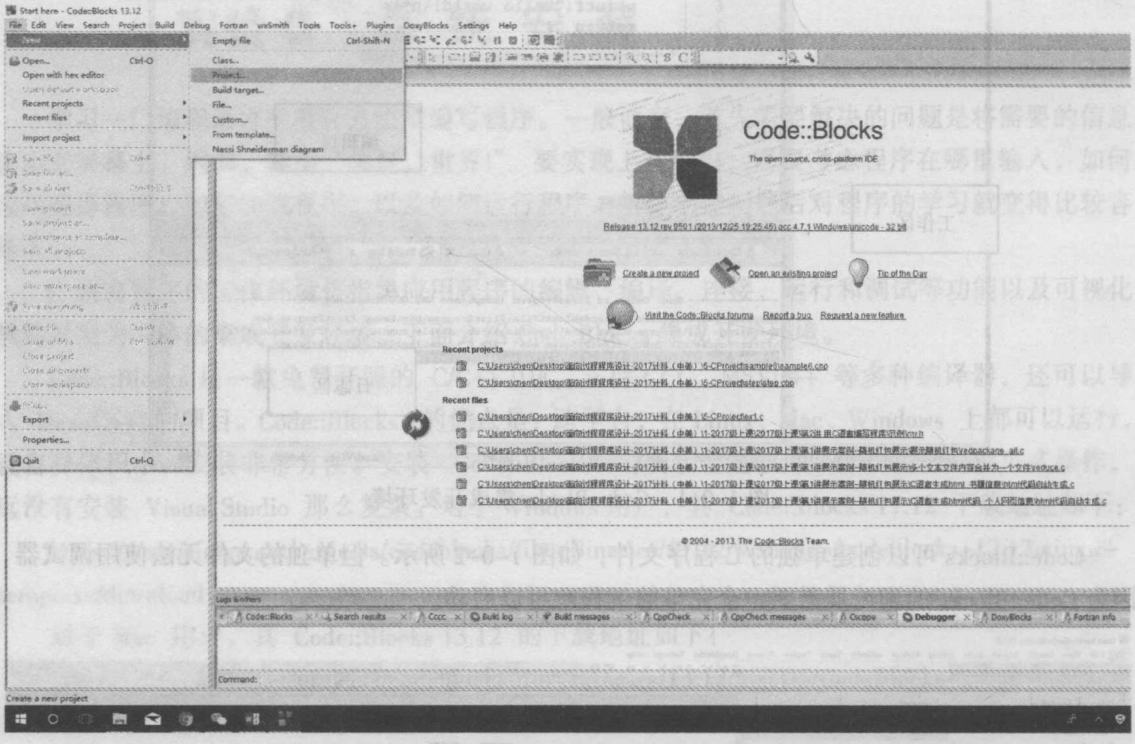

图 1-0-3 新建工程

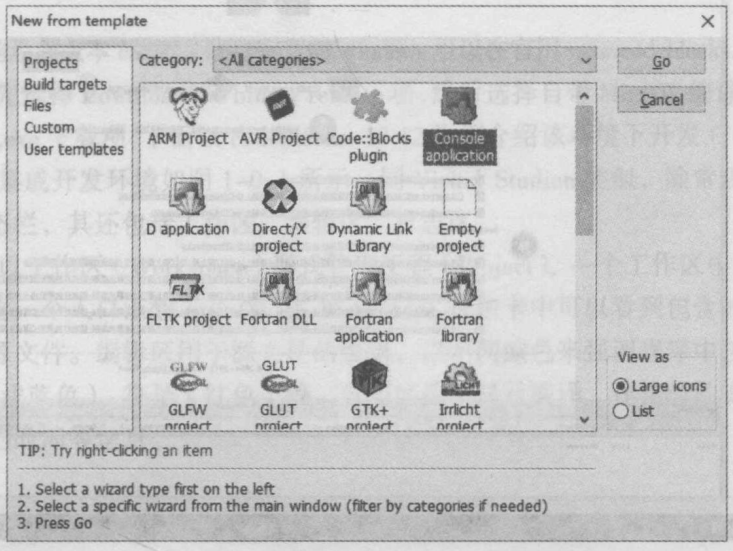

图 1-0-4 新建工程对话框

在 New from template 对话框中，在左边窗格中选择创建 Projects 的向导菜单，在右边的工程类别列表中选择创建工程的类型。创建 C 语言的工程可以选择 Empty project，创立一个空的工程；也可以创建一个控制台应用程序。在此选择 Console application 创建一个控制台应用程序。

单击 Go 按钮进入 Console application 对话框，如图 1-0-5 所示，选择程序语言。如果选择 C，则源程序文件的扩展名为.c；如果选择 C++，则源程序文件的扩展名为.cpp。此处选择 C，单击 Next 按钮。

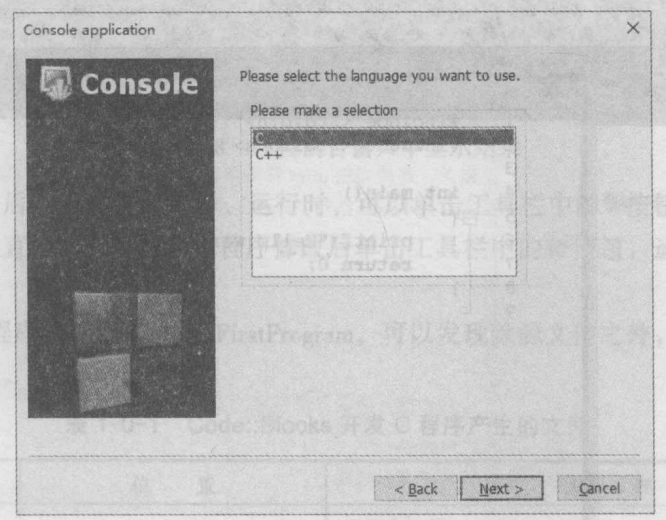

图 1-0-5 Console application 对话框

在图 1-0-6 所示的对话框中，需要填写工程相关的信息。在 Project title 文本框中输入新建工程的名称，此处为 FirstProgram，C 语言的每一个工程组织在一个文件夹中，Code::Blocks 在创建工程时会创建同名的工程文件夹 FirstProgram。选择该工程文件夹所在的位置，可在 Folder to create project in 文本框中直接输入，也可单击其后的按钮，在弹出的“浏览文件夹”对话框中指定，此处为预先创建好的 C:\codeblockCfile 用户自定义文件夹。

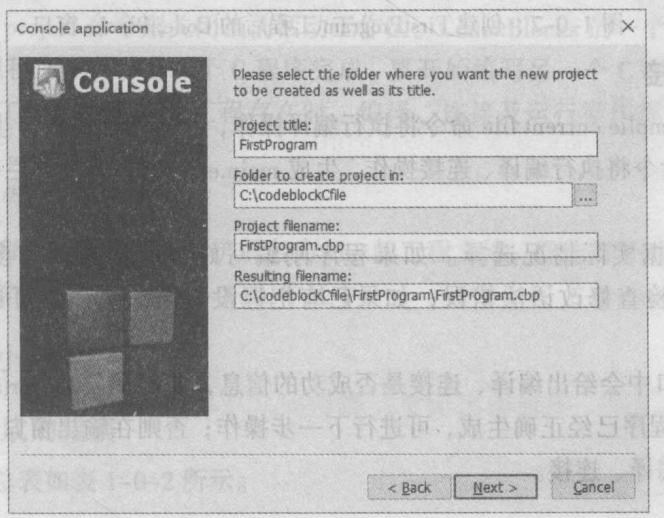

图 1-0-6 填写工程信息

单击 Next 按钮继续，后续的对话框中保留默认值，并单击 Finish 按钮结束，进入 Code::Blocks 集成环境。左边的 Management 窗口中显示了工程的组织结构，如图 1-0-7 所示。工作区 Workspace 用来管理工程，可以包含多个工程，此处有只有工程 FirstProgram。工程 FirstProgram 下的 Sources 用来管理工程中的文件，此处有一个自动生成的源文件 main.c。双击打开 main.c，在右边的编辑区显示第一个 C 程序 main.c 的源程序代码，可以编辑修改代码以实现相应功能。

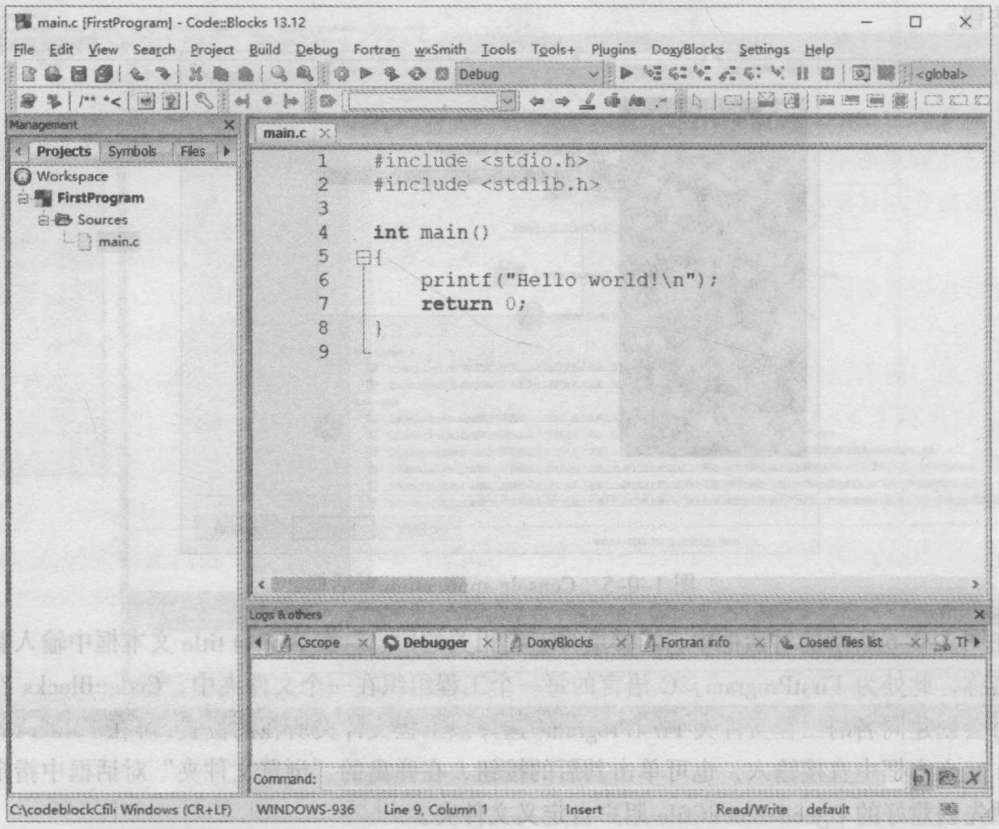

图 1-0-7　创建 FirstProgram 工程后的 Code::Blocks 窗口

2. 编译、连接

选择 Build→Complie current file 命令将执行编译操作，检查语法错误，生成中间文件 main.o。选择 Build→Build 命令将执行编译、连接操作，生成 main.exe 文件 ，工具栏中的 ⚙ 按钮与此命令对应。

程序员可以根据实际情况选择。如果程序刚编写好，可能错误较多，可选择 Complie current file 命令，检查修改语法错误；如果已有把握没有语法错误，可选择 Build 命令，准备执行程序。

在 Bulid log 窗口中会给出编译、连接是否成功的信息，如出现“0 error(s), 0 warning(s)”则说明 main .exe 可执行程序已经正确生成，可进行下一步操作；否则在输出窗口中显示出错信息，需要改正错误后重新编译、连接。

3. 运行程序

编译完成后可以选择 Build→Run 命令，或是单击工具栏中的 ▶ 按钮执行程序。Code::Blocks

会在控制台窗口中显示运行的结果，如图 1-0-8 所示。第一行的文本是程序的输出结果。第二行是程序运行的信息，包括返回值、运行时间等，按任意键关闭窗口。

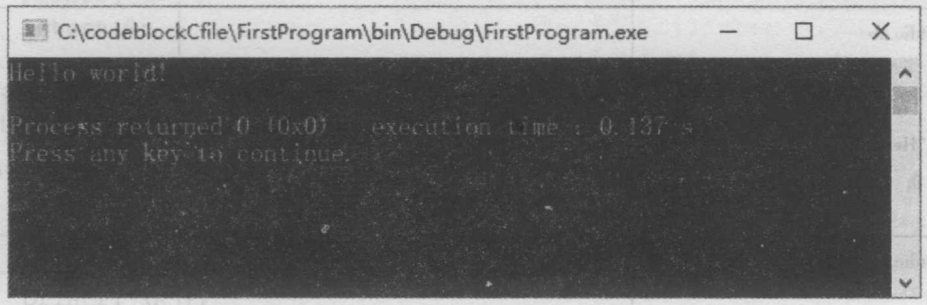

图 1-0-8 在控制台窗口中显示结果

当源程序修改过后，需要再次编译、运行时，可以单击工具栏中的▶按钮，会先编译、连接源程序，没有错误就直接运行。如果源程序修改后单击工具栏中的▶按钮，运行的仍然是前一次生成的可执行文件。

打开保存 C 源程序的工程文件夹 FirstProgram，可以发现除源文件之外，还生成了表 1-0-1 所示的文件和文件夹。

表 1-0-1 Code::Blocks 开发 C 程序产生的文件

文 件 名	位 置	解 释
main.c	FirstProgram\	源程序文件
FirstProgram.cbp	FirstProgram\	工程文件
FirstProgram.layout	FirstProgram	关于开发环境的参数文件
main.o	FirstProgram\obj\Debug	编译产生的中间文件
FirstProgram.exe	FirstProgram\bin\Debug	生成的可执行工程文件

4. 关闭工程文件

选择 File→Close project 命令则关闭当前活动工作区。Code::Blocks 的一个工程中只能包含一个含有 main 函数的源程序文件，当一个 C 程序完成，要开始编写另一个 C 程序时，一定要再新建一个工程，在一个工作区中有多个工程存在时，编译、连接及运行操作都是针对当前选中的工程。

5. 打开工程文件

选择 File→Open 命令，选择相应的扩展名为.cbp 的工程文件，即能打开对应的工程，也可以直接拖动.cbp 文件的图标到 Code::Blocks 的工作区中。

二、常见错误小结

1. 程序代码造成的错误

常见代码错误汇总表如表 1-0-2 所示。

表 1-0-2 常见代码错误汇总表

常见错误实例	错 误 描 述	错 误 类 型

#include <stdio.h>
int main()
{
printf("Hello C!\n")
return 0;
}
	printf("Hello C!\n")语句结束后缺少分号（;）。C 语言要求以分号作为语句的结束符号	语法错误，编译时出错

#include <stdio.h>
int main()
{
print("Hello C!\n");
return 0;
}
	print("Hello C!\n");语句中的函数名 printf 拼写错误，连接器把 print 符号串看成是外部函数，但又找不到该函数，产生一个连接错误	语法错误，连接时出错

#include <stdio.h>
int main()
{
int n,sum;
double ave;
n=10;
sum=43;
ave=sum/n;
printf("average=%.1f",ave);
return 0;
}
	该程序本要求屏幕显示结果如下：
	average=4.3
	但实际显示结果为：
	average=4.0
	因为 sum/n 表达式中两个整数相除的结果为整数，截取了小数部分，所以结果不正确
		逻辑错误，运行结果不正确

2. 开发程序操作不当造成的错误

（1）当源程序文件被修改过后，需要再次经过编译、连接再运行，否则运行的仍是前一次生成的可执行文件。

（2）Code::Blocks 集成环境中，开发 C 程序时经常会出现再次新建的含有 main()函数的程序文件没有任何错误，但却不能正常编译和运行的情况。这是由于新建的程序文件依然建在前一个程序文件的 Project 中，导致一个 Project 包含了一个以上的 main()函数，而 Visual Studio 和 Code::Blocks 都要求一个 Project 中只能包含一个 main()函数。因此必须牢记不同的程序文件要放在不同的 Project 中。

三、自测题

1. 运行下列程序，根据输出信息，理解转义字符

源程序 1：

```
#include <stdio.h>
int main()
{
```

```
    printf("hello, ");
    printf("world");
    printf("\n");
    return 0;
}
```

源程序 2:

```
#include <stdio.h>
int main()
{
    printf("hello,\n ");
    printf("world\n");
    printf("\n");
    return 0;
}
```

源程序 3:

```
#include <stdio.h>
int main()
{
    printf("\n\thello, ");
    printf("world!\n");
    return 0;
}
```

2. 体验下列程序的功能

源程序:

```
#include <stdio.h>
#include <windows.h>
void color(short x) //自定义函根据参数改变颜色
{
    if(x>=0 && x<=15)//参数在 0-15 的范围颜色
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),x); //只有一
个参数，改变字体颜色
    else//默认的颜色白色
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),7);
}
int main()
{
    int i;
    printf("此处为没调用颜色函数之前默认的颜色\n");
    //调用自定义 color(x)函数 改变的颜色
    color(0);    printf("黑色\n");
    color(1);    printf("蓝色\n");
    color(2);    printf("绿色\n");
    color(3);    printf("湖蓝色\n");
    color(4);    printf("红色\n");
    color(5);    printf("紫色\n");
    color(6);    printf("黄色\n");
    color(7);    printf("白色\n");
    color(8);    printf("灰色\n");
    color(9);    printf("淡蓝色\n");
```

```
    color(10);   printf("淡绿色\n");
    color(11);   printf("淡浅绿色\n");
    color(12);   printf("淡红色\n");
    color(13);   printf("淡紫色\n");
    color(14);   printf("淡黄色\n");
    color(15);   printf("亮白色\n");
    color(16);    //因为这里大于15，恢复默认的颜色
    printf("回到原来颜色\n");
    //直接使用颜色函数
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),FOREGROUND_RED
| FOREGROUND_INTENSITY | BACKGROUND_GREEN | BACKGROUND_INTENSITY);
    printf("红色字体   前景加强 绿色背景 背景加强\n");
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15 | 8 | 128 |
64);
    printf("亮白色字体 前景加强 红色背景 背景加强\n");
    //声明句柄再调用函数
    HANDLE JB=GetStdHandle(STD_OUTPUT_HANDLE);//创建并实例化句柄
    SetConsoleTextAttribute(JB, 2 | 8);
    printf("颜色及对应数字表: \n");
    for(i=0;i<1000;i++){
        //color(16);printf(" ");
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),i);
        printf("%-3d",i);
        color(16);printf(" ");
        if(i%16==0)   printf("\n");
    }
    color(16);
    return 0;
    //类似的函数还有 system("color XX");（X是十六进制0~F之间的数，不过这种函数改变的
是整个画面，而不能让多处局部变色
}
```

实验1 认识程序

一、知识导图

本实验知识导图如图 1-1-1 所示。

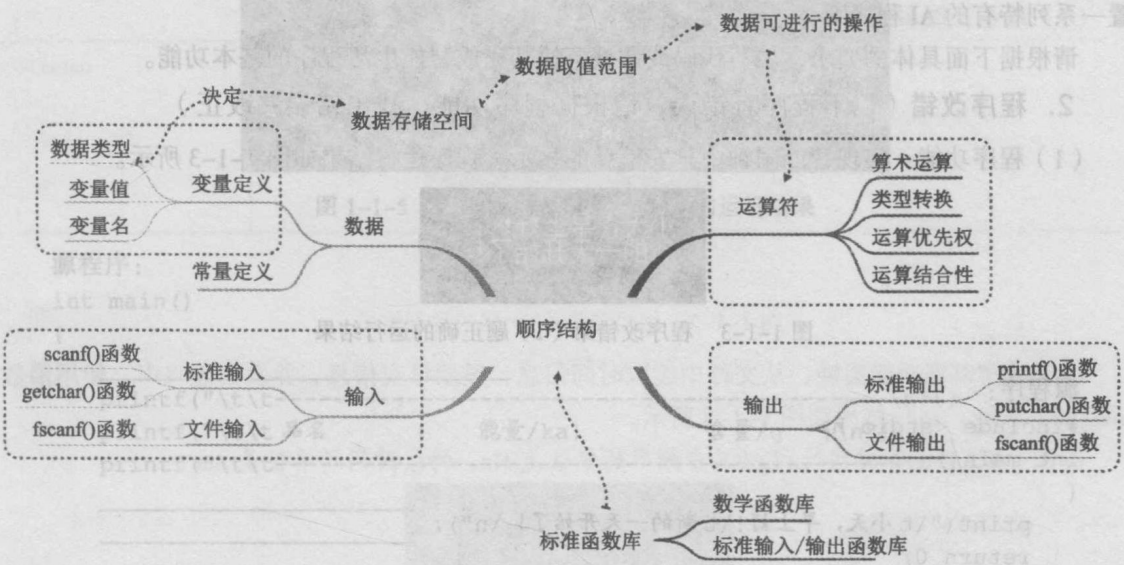

图 1-1-1 知识导图

二、实验目的

- 熟悉 C 语言编程的顺序编程框架。
- 熟悉数据的定义。
- 熟悉数据的输入和输出。
- 熟悉简单的数据运算操作。
- 了解文件的基本操作。

三、实验内容

1. 问题背景

人工智能正在深刻地改变着生活与社会，智能程序无处不在，它渗透于生活中的点点滴滴。

设想每天您在智能闹钟（见图 1-1-2）的个性呼唤中醒来："小天，现在是 2020 年 4 月 1 日早上 7 点，新的一天开始了！"。洗漱之后，如果是过去，您肯定还在烦恼早餐如何解决，而现在，智能的烹饪机器人已经为您将一切都准备好了。

图 1-1-2　智能闹钟

贴心的烹饪机器人婷婷从健康监测系统中获取到数据，并根据您的口味及爱好，为您准备好了一份营养早餐：一杯牛奶、一个鸡蛋、一盘水果沙拉、两个您爱吃的煎饼果子。

健康的早餐让您精力充沛、心情愉悦，然后您可以启动智能小车——小智，开始一天的工作。只需您通过语音说出密码，即可开启座驾，小智将根据您预计的使用时间，精确地判断出您到公司的时间……

这一切都如此贴心，让人感觉到您的 AI 朋友无处不在。现在就模仿这个情景，帮您的朋友设置一系列特有的 AI 程序吧。

请根据下面具体的要求，填写代码或调试，使其完成智能生活程序的基本功能。

2. 程序改错（以下程序有错误，请根据程序功能，找出错误并改正）

（1）程序功能：实现智能闹钟中早安信息的显示。程序运行结果如图 1-1-3 所示。

```
小天，早上好！
新的一天开始了！
```

图 1-1-3　程序改错第（1）题正确的运行结果

源程序：

```
#include <stdio.h>
int main()
{
    print("\t 小天，早上好!\t 新的一天开始了！ \n");
    return 0;
}
```

（2）程序功能：智能小车小智设置了问题提问系统。问题是询问年龄，只有输入正确的年龄，小车才能正确启动。编程实现这个小车问题启动程序。输入年龄，输出图 1-1-4 所示的信息。

```
        上班时间到了，快去呼叫小智吧！
        请输入密码.
你多大了？
25
        欢迎使用小智，新的一天启动了！
```

图 1-1-4　程序改错第（2）题正确的运行结果

源程序：

```
#include <stdio.h>
int mian()
{
    int ages;
    printf("\t 上班时间到了，快去呼叫小智吧！ \n");
    printf("\t 请输入密码.\n 你多大了？ \n")
```

```
    scanf("%2d",age);  //输入函数，输入整数年龄值
    if(age=PWD)
        printf("\t 欢迎使用小智，新的一天启动了！\n");
    else
        printf("\t 对不起，您不是小智的主人！\n");
    return 0;

}
```

3. 程序填空（根据程序功能，将空白处补充完整）

（1）程序功能：利用智能烹饪程序将主人的早餐清单显示出来。程序运行结果如图 1-1-5 所示。

```
主人：你好！
早餐时间到了，这是您的早餐清单，祝您好胃口！
-----------------------------------------
品名        |    能量/ka|          含量/g   |
-----------------------------------------
鸡蛋        |      20.1|            1.1|
牛奶        |     100.7|           10.5|
水果        |      11.4|            2.3|
煎饼        |      50.0|           15.7|
-----------------------------------------
```

图 1-1-5 程序填空第（1）题正确的运行结果

源程序：

```
int main()
{
    _____
    printf("\t\t-------------------------------------------\n");
    printf("\t\t 品名        |    能量/ka|          含量/g   |\n");
    printf("\t\t-------------------------------------------\n");
    _____
    _____
    _____
    _____
    printf("\t\t-------------------------------------------\n");
    return 0;
}
```

（2）程序功能：启动智能汽车程序后，可输入预计时长，统计并计算预计到达公司的时间。程序运行结果如图 1-1-6 所示。

```
现在时间是：07:05
输入预计使用的时间(分钟)：56
预计到达公司时间为：08:01
```

图 1-1-6 程序填空第（2）题正确的运行结果

```
int main()
{
    int use;
    FILE *fp;
    int hour,min;
```