《C语言程序设计：现代方法》（第2版）

以下是本人（黄国洪Samwee）学习《C语言程序设计：现代方法》这本书的学习笔记。

封面：

课后习题参考答案：

> Answers to Selected Exercises in C Programming: A Modern Approach - Second Edition
>
> Answers to selected exercises in the book C Programming: A Modern Approach, Second Edition
>
> http://knking.com/books/c2/answers/index.html

1 C语言概述

2 C语言基本概念

每一章后面的练习题、编程题都动手做

3 格式化输入/输出

一些关于printf ， scanf等的输入输出，以及相关规范

4 表达式

讲了一些++,- -,*,/,%等的使用和计算，不是很难

5 选择语句

5.1 逻辑表达式

5.2 if 语句

然而，C语言遵循的规则是else 子句应该属于离它最近的且还未和其他else 匹配的if 语句。

5.3 switch 语句

6 循环

6.1 while语句

6.2 do语句，

两种语句的区别是，do 语句的循环体至少要执行一次，而while 语句在控制表达式初始为0 时会完全跳过循环体。

6.3 for语句

6.4 退出循环

break, continue, goto

break 语句的目标是包含该语句的循环结束之后的那一点，而continue 语句的目标是循环结束之前 的那一点。goto 语句则可以跳转到函数中任何有标号的语句处。

```
n = 0;
sum = 0;
while (n < 10) {
  scanf("%d", &i);
  if (i == 0)
    continue;
  sum += i;
  n++;
  /* continue jumps to here */
```

```
}
```

6.5 空语句

7 基本类型

7.1 整数类型

7.2 浮点类型

7.3 字符类型

7.4 类型转换 (int)f 表示把f 的值转换成int 类型后的结果

7.5 类型定义

8 数组

8.1 一维数组

8.2 多维数组

9 函数

9.1 函数的定义和调用

9.2 函数声明

9.3 实际参数

9.4 return语句

9.5 程序终止

exit函数

9.6 递归

快速排序算法

10 程序结构

讨论一个程序包含多个函数时所产生的几个问题

10.1 局部变量

10.2 外部变量

10.3 程序块

10.4 作用域

10.5 构建C程序

11 指针

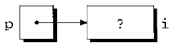
本章将从内存地址及其与指针变量的关系入手（11.1节），然后11.2节介绍取地址运算符和间接寻址运算符，11.3节是有关指针赋值的内容，11.4节说明给函数传递指针的方法，而11.5节则讨论从函数返回指针。

11.1 指针变量

11.2 取地址运算符和间接寻址运算符

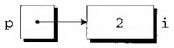
只要p 指向i ，*p 就是i 的**别名** 。*p 不仅拥有和i 相同的值，而且对*p 的改变也会改变i 的值。（*p 是左值，所以对它赋值是合法的。）下面的例子说明了*p 和i 的等价关系，这些图显示了在计算中不同的点上p 和i 的值。

```
p = &i;
```




```
i = 1;
```


```
printf("%d\n", i);    /* prints 1 */
printf("%d\n", *p);   /* prints 1 */
*p = 2;
```




```
printf("%d\n", i);    /* prints 2 */
printf("%d\n", *p);   /* prints 2 */
```

11.3 指针赋值

11.4 指针作为参数

11.5 指针作为返回值

可以使用单词const 来表明函数不会改变指针参数所指向的对象。

● 2. 如果i 是int 类型变量，且p 和q 是指向int 的指针，那么下列哪些赋值是合法的？

(a) p = i;　　(b) *p = &i;　　(c) &p =q;

(d) p = &q;　　(e) p = *&p;　　(f) p = q;

(g) p = *q;　　(h) *p = q;　　(i) *p = *q;

(e)、(f) 和 (i) 是合法的。

(a) 是非法的，因为`p`它是一个指向整数的指针并且`i`**是一个整数**。

(b) 是非法的，**因为`*p`它是一个整数**并且`&i`是一个指向整数的指针。

(c) 是非法的，因为`&p`是一个指向整数的指针，并且`q`是一个指向整数的指针。

(d) 由于与 (c) 类似的原因是非法的。

(g) 是非法的，因为`p`它是一个指向整数的指针并且`*q`**是一个整数**。

(h) 是非法的，**因为`*p` 它是一个整数**并且`q`是一个指向整数的指针。

（c）(d)我不太懂，因为根据（f）来看，指针应该是可以赋值给指针的。

12 指针和数组

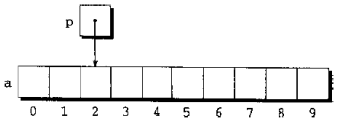
12.1 指针的算术运算

12.1.1 指针加上整数

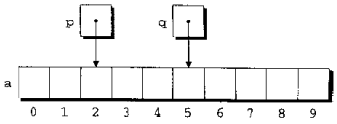
指针p 加上整数j 产生指向特定元素的指针，这个特定元素是p 原先指向的元素后的j 个位置。更确切地说，Q&A 如果p 指向数组元素a[i] ，那么p+j 指向a[i + j] （当然，前提是a[i + j] 必须存在）。

下面的示例说明指针的加法运算，插图说明计算中p 和q 在不同点的值。

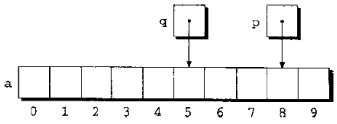
```
p = &a[2];
```




```
q = p + 3;
```




```
p += 6;
```



12.2 指针用于数组处理

12.3 用数组名作为指针

通常情况下，a + i 等同于&a[i] （两者都表示指向数组a 中元素i 的指针），

12.4 指针和多维数组

12.5 C99中的指针和变长数组

13 字符串

13.1 字符串字面量

13.2 字符串变量

字符串变量就是那些格子

13.3 字符串的读和写

13.4 访问字符串中的字符

13.5 使用C语言的字符串库

1 strcpy函数 字符串复制

2 strlen函数 求字符串长度

3 strcat函数 字符串拼接

4 strcmp函数 字符串比较

13.6 字符串惯用法

13.7 字符串数组

2. 假设p 的声明如下：

```
char *p = "abc";
```

下列哪些函数调用是合法的？请说明每个合法的函数调用的输出，并解释为什么其他的是非法的。

(a) putchar(p);　　(c) puts(p);

(b) putchar(*p);　　(d) puts(*p);

2. [was #2]

(a) Illegal; p is not a character.
(b) Legal; output is a.
(c) Legal; output is abc.
(d) Illegal; *p is not a pointer.

14 预处理器

14.1 预处理器的工作原理

14.2 预处理指令

14.3 宏定义

14.4 条件编译

14.5 其他命令

15 编写大型程序

15.1 源文件

15.2 头文件

15.3 把程序划分成多个文件

15.4 构建多文件程序

1 makefile

16 结构、联合和枚举

结构是可能具有不同类型的值（成员）的集合。

联合的成员共享同一存储空间。

枚举是一种整数类型，它的值由程序员来命名。

16.1 结构变量

16.2 结构类型

16.3 嵌套的数组和结构

16.4 联合

16.5 枚举

以扑克牌四种花色为例，就很好理解

17 指针的高级应用

17.1 动态存储分配

17.1.1　内存分配函数

malloc 函数——分配内存块，但是不对内存块进行初始化。
calloc 函数——分配内存块，并且对内存块进行清零。
realloc 函数——调整先前分配的内存块大小。

17.1.2　空指针

17.2 动态分配字符串

17.3 动态分配数组

17.4 释放存储空间

1 free函数

2 “悬空指针”问题

虽然free 函数允许收回不再需要的内存，但是使用此函数会导致一个新的问题：悬空指针 （dangling pointer）。调用free(p) 函数会释放p 指向的内存块，但是不会改变p 本身。如果忘记了p不再指向有效内存块，混乱可能随即而来

17.5 链表

1 声明结点类型

2 创建结点

3 ->运算符

4 在链表的开始处插入结点

5 搜索链表

6 从链表中删除结点

7 有序链表

17.6 指向指针的指针

17.7 指向函数的指针

17.8 受限指针

17.9 灵活数组成员

18 声明

18.1 声明的语法

18.2 存储类型

18.3 类型限定符

C语言中一共有两种类型限定符：const 和volatile 。（ C99还有第三种类型限定符，即restrict ，它只用于指针（受限指针➤17.8节）。）

18.4 声明符

18.5 初始化式

18.6 内联函数

19 程序设计

19.1 模块

19.2 信息隐藏

19.3 抽象数据类型

19.4 栈抽象数据类型

19.5 抽象数据类型的设计问题

20 底层程序设计

20.1 位运算符

20.2 结构中的位域

20.3 其他底层技术

21 标准库

22 输入/输出

22.1 流 stream

22.2 文件操作

22.3 格式化的输入/输出

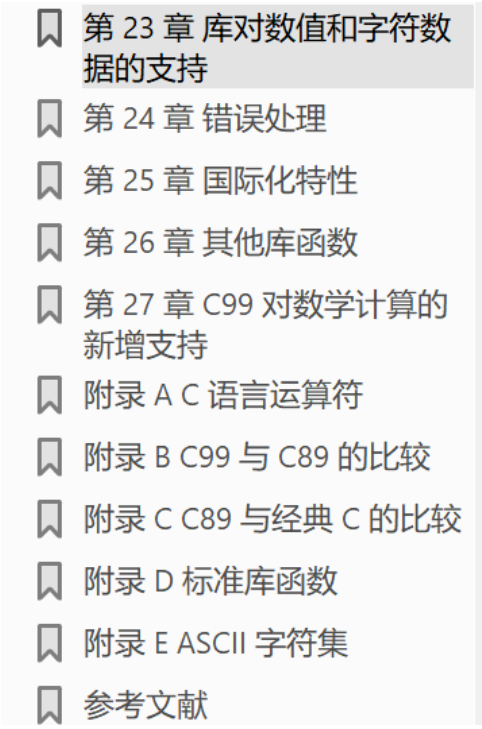
22.4 字符的输入/输出

22.5 行的输入/输出

22.6 块的输入/输出

22.7 文件定位

22.8 字符串的输入/输出



23章及以后多是罗列一些库、函数之类的，有需要用到再google查询即可，全部罗列出来也没法记。

读后感想

优点：这本书还是挺不错的，外国人写书的思维还是要更好一些，娓娓道来，充分考虑新手认知新事物的顺序，非常值得推荐。

缺点：这本书确实很厚，看到后面人很疲惫。