dian学习简历

重点：

1. 使用C语言的七个步骤：
2. 定义程序的目标
3. 设计程序
4. 编写代码
5. 编译
6. 运行程序
7. 测试与调试程序
8. 维护与修改代码
9. 使用#include共享库，方便程序的编写。
10. 提高程序可读性：.选择有意义的函数名与写注释。
11. 通过跟踪程序状态，或者跟踪一部分程序状态来调试代码，使用printf来跟踪数值的变化。
12. 语法错误：不遵循C语言的编写规则；语义错误，编写的程序在意义的执行上存在问题。
13. 变量的定义与储存：
14. 整数类型存储：以二进制存储整数，二进制的基本单位是bit，以0和1表示。
15. 浮点数类型存储：分为小数部分，指数部分以及符号部分。通常来说，浮点数范围更大，但是精度较小（数据过大或者过小时），运算速度较慢
16. 声明变量与变量初始值的赋予，使用=（赋值）符号
17. 其他整数类型：short int（16位）%h的前缀; long (int)（32位）%l的前缀; long long (int)(64位)%ll的前缀 ,unsigned int(0~65535)加上u的前缀 依据计算机字节长度而定。
18. Bool类型的运算，逻辑运算。&&与||
19. Char类型数组：连续多行的储存单元
20. Scanf()读取时遇到/n自动换行
21. 使用#define来定义一个常量（一般写在程序顶部），注意，无分号，使用const限制一个变量为只读。
22. Printf()转换说明表（很抱歉，暂时还没有全部背下来），转换说明和要打印的变量一定要一一对应。 由于参数传递的关系，只要有一个不匹配，可能就会导致输出的不正确。
23. Scanf读取输入时，每次读取一个，所以输入时多输入的会被之后的scanf继续读取导致错误，优点时能读取更多种类的字符。当scanf检测到文件末尾时会返回EOF，该内容将在之后的文件读取中详细解释。
24. 基本运算符：=（赋值）；+（加法）；-（减法）；\*（乘法）；/（除法）；%（求模）；++（递增）；--（递减）；==（恒等）
25. 运算符优先级：括号，递增减，乘除，加减，赋值
26. 类型转换：涉及两种类型的运算，结果会被变成较高级的那种；赋值表达式中计算的结果会被转成被赋值变量的值。
27. 强制类型转换（type）即可。
28. 如何编写函数：
29. 不带返回值的函数，使用void放在函数名的括号中
30. 带参数，就在函数的括号中按顺序用定义变量的方式定义形式参数，注意，此处的参数名仅在函数中可用。且形式参数的值由实际参数赋予。
31. 如果有返回值，函数之前的声明就是该返回值的类型。使用return返回值，一般赋给某个变量。
32. 使用指针在函数中通信时声明的形式参数也得是指针类型的。使用指针时，主调函数中的值也会被一起改变。
33. While入口条件循环：根据在括号的条件句判断是否进行下一个循环，考虑清如何终止循环。用运算关系符来写expression从而判断。
34. 运算符的优先级（）；++ -- sizeof；\* / %；+ -；< > <= >=; == !=; =;
35. For循环（计数循环）：集初始化、测试、更新为一体；更为灵活。
36. Do while(出口循环)：每次循环运行结束后判断是否进行下一次的循环，和while的区别主要体现在循环第一遍时是否需要判断。注意，该循环需要分号结尾。
37. 数组：按顺序按顺序存储的一系列相同类型的值，通过数据类型加函数名加中括号声明，例如：int test[100];声明通过下标访问数组中的元素，在循环中使用数组会使存储变得更加简单明了与便捷，查找也更加方便。初始化数组使用{}内部的数据用逗号隔开。在C99中,可以在{}内部使用下标来初始化内容。在使用时也可以直接用下标赋值。同时要防止下标超出边界，导致越界。在定义时用整数声明数组的大小。
38. 多维数组：定义方式与一维数组相似，几维数组就有几个中括号。初始化可以在一维数组的基础上每一行都用一个{}，也可以将花括号省略，只要顺序是对的就可以了。同时
39. 在只初始化了一部份数组的情况下，剩余的数组自动初始化为0.
40. 分支：
41. If语句：用expression来判断且当表达式为真时执行之后的内容，若表达式为假则会跳过语句
42. If else语句，提供两种选择，在if条件不满足时，会直接跳转到else中的语句。
43. else if同时，else后可以继续加if，形成多种条件下的分支，即多重判断语句。
44. swith 即另一种多重选择，使用case作为表达句之前的标志，尤其用于多种不同类型的分类选择。工作方式为读取一个变量，根据变量的不同的值进行不同的运算。
45. break与continue：用于辅助循环，前者跳出循环，一般和if语句搭配使用；后者用于重启循环，重新开始某一遍地循环，一般也和分支搭配使用。
46. 递归算法：使函数自己调用自己，在此过程中，函数的形式参数不断变化，每一级被调用的函数的参数都是私有的。执行顺序从低级到高级，有时也能用循环来替代（如斐波那契数列）。虽然思路简单但是会占用较大的内存空间。注意：一定要给递归的函数写结束条件！！！
47. 字符测试函数表：有效地在分支语句中作为判断条件使用（同样很遗憾，我还没有完全背下来。）
48. 缓冲区：1.为了节约时间2.在缓冲区里地内容用户可以随时修改
49. 字符的输入与输出：getchar()与putchar()用于读入字符与输出字符，但是注意遇到#号就会停止输入。同时在读取文件时遇到结尾也会输出EOF（用于文件读取的判断）
50. &运算符用于查找地址，后续应用会在指针中讲到。
51. 指针：值为内存地址的变量。在声明时，用数据类型+\*+指针名，例如：int \*p;在赋值时使用&符号赋予地址的值。同时，数组指针所指向的是数组的首元素的值。\*代表将该地址的内容取出（解引用）。指针本身也可以被取值。指针可以和整形相加减（代表向后移动指针位置）；同理，递增减指针就是把地址加减一。
52. 指针的兼容性低于数值类型
53. 可以使用const来保护数据的安全，放置被随意改动。
54. 字符串中的各种常用函数（这里不一一列举了）
55. 字符串排序：选择排序，使用指针来进行首字母的比较。
56. 文件的输入与输出：fopen(用于打开文件)，使用getc来进行文件的读取，读取到文件结尾时会输出EOF。最后一定要用fclose关闭函数。
57. fprintf与fscanf要求使用指向文件的指针，且作为参数时放在第一个位置。
58. 结构体函数：
59. ：用struct建立结构声明，用{}将内部结构变量括住，结构变量声明方式与普通变量一致。
60. ：初始化结构可以在花括号括起来的初始化列表进行初始化，个初始化项用逗号分隔。
61. 访问结构成员，在结构名之间与结构变量之间用.连接，之后可以看作是普通的变量。
62. 结构数组，将数据结构声明改为结构体即可。访问时先写下标再用.连接结构变量名。（结构体暂时学到这里，之后更多的还没看到。）

疑点：

1. 为什么浮点数运行速度更慢，以及浮点数存储是如何损失精度的（除了上溢和下溢有什么别的原因吗？）
2. 为什么stdint.h，即精确宽度整数类型如能转换使其类型与当前系统匹配；为什么inttype.h如何加速运算速度，仅仅依靠缩小int的长度吗？
3. 为什么作为函数参数传递时，char,short会被转成int，float会被转成double?
4. Float和double有什么区别？
5. 如何重定向输入与输出设备？
6. 为什么压缩文件时可以把每三个字符中的第一个给保留下来而不影响效果。（这里没看很懂。）