C/C++知识要点总结

- C基础知识
  - 一、数据类型及运算
    - 1 补码
      - 原码转补码不考虑符号位
      - 补码转原码，符号位不参与运算
      - 取反编 + 1 == 取反减 - 1
    - 2 科学计数法表示
      - 1.3 * 10^11 --> 1.3E11
      - 9.34 * 10^-3 --> 9.34E-3
    - 3 相关概念
      - sizeof()是一个运算，英语动词
      - ++结果不能再在其前上
      - 判断一个整数是否是2^n (2,4,6,8,16...) —— !(x & (x - 1))
      - 三目条件运算符代码意优 —— 编译器能产生比if...else...更优的代码
    - 4 运算符优先级、结合方向规则
      - 单目 > 双目
      - 算术 > 关系 > 位 > 逻辑 > 条件（三目）> 赋值 > 逗号
        - 算术： + - * /
        - 关系： > < >= <=
        - 位： & | ^ —— 单目： ~
        - 逻辑： && || —— 单目： !
      - 两足单目 > 双目的规度下
      - 自右向左的三种运算符
        - 单目
        - 赋值
        - 条件
      - 其余都是自左向右
    - 5 数据输入与输出
      - 1、printf()语句从右向左计算输出表达式的值
        - i = 1;
        - printf("%d, %d\n", i++, i--);
        - res= 0,1
        - 先执行i--，再执行i++
      - 2、常用输出函数
        - printf()
        - putchar()
          - 输出一个字符
          - 必须是字符型变量或常量
        - puts() —— 输出一个字符串 —— 必须是字符类或常量
      - 3、常用输入函数
        - scanf()
        - gets()
          - 每次读取一个字符串
        - getche()
          - conio.h中
          - 读取字符不用按回车
        - getchar()
          - stdio.h中
          - 完成后须按回车
        - 区别 —— scanf不能输入含空格字符串，gets可以
        - 每次读取一个字符
  - 二、选择语句和循环语句
    - 1 switch
      - switch(表达式)
      - case 常量表达式
        - 常量表达式可能为整型、字符型
        - 不允许浮点型
  - 三、数组
    - 1 定义
      - 定义数组单独初值
        - Turbo C会给数组置0
        - VC则取随机值
      - 定义静态数组，则系统自动赋0
    - 2 比较字符串数组中的值
      - C! —— strcmp(str1, str2)
      - C++: —— str1 == str2
      - JAVA! —— str1.equals(str2) —— java中，str1 == str2 比较的是地址
  - 四、指针
    - 1 指针运算
      - 指针相减 —— 表示两指针所指地址之间的数据个数
      - 指针相加 —— 没有意义，错误
    - 2 数组与指针
      - 1、一维数组首地址
        - int a[10], *p;
        - p = &a[0];
        - p = a
        - 等价，将数组首元素的首地址赋给指针p
        - 表示
          - a&, a —— 数组首元素的首地址
          - a —— 数组首地址
        - 对比
          - a == a&[0]
          - a != a& —— 地址值相同，含义不同
      - 2、二维数组首地址
        - int a[10][10];
        - a
          - 二维数组首元素首地址
          - 代表一维数组元素的首地址
        - a& —— 数组首地址
        - a&[0] —— 二维数组首元素首地址
        - a&[0][0]
          - a&[0][0] != a
          - a[0] == a&[0][0]
        - 地址值相同，含义不同
      - 3、二维数组指针
        - int (*p)[3]
        - int *p[3]
        - int *(p[3])
        - 指向含3个元素的二维数组的行指针
        - 数组每列有3个元素
        - 指针数组，每个元素均是一个指针
    - 3 指针与引用
      - 区别
        - 本质区别
          - 引用必须总是指向某经对象
            - 不能使用指向空值的引用
            - 任何情况下
          - 指针可以指向空值
            - 不存在指向空值的引用
          - 效率比使用指针高
        - 合法性区别
          - 使用引用前，无需测试其合法性
            - 判空
          - 使用指针总是需要判空
        - 可修改区别
          - 指针可修重赋值值，引指向另一对象
          - 引用
            - 总指向初始化时被指定的对象
            - 以后都不能改变
            - 但指定对象的内容可以改变
        - 应用区别
          - 指针场景
            - 存在不指向任何对象的情况
            - 不同的时刻指向不同对象的情况
          - 引用场景 —— 指向一个对象总就不会改变指向的情况
          - 场景区别
      - ps: 声明引用 / const常量 的同时，必须初始化
    - 4 函数指针
      - (1) float(**def)[10];
        - 二级指针
        - 指向一个一维数组的指针
        - 数组元素都是float
      - (2) double*(*gh)[10];
        - 指针gh，指向一个一维数组
        - 该数组元素的类型均为double *
      - (3) double(*f[10])();
        - f是一个数组，含10个元素
        - 元素都是函数指针
        - 指向的函数
          - 没有参数
          - 返回double类型的值
      - (4) int *((*b)[10]);
        - 和int *(*b)[10]一样
      - (5) long (* fun)(int)
        - 函数指针
  - 五、类型转换
    - 1 (int a) 模式
      - float a = 1.0f;
      - (int)a实际上是以浮点数a为参数构造了一个整型数，该整数的值是1。
      - (int)a隐藏在该编译器将a当作整数看（并没有做任何实质上的转换）。
    - 2 unsigned int
      - unsigned int a = 0xFFFFFFF7;
      - unsigned char i = (unsigned char)a; —— ii 00000007
      - char *b = (char *)&a; —— *b: fffffff7
    - 3 隐式类型转换
      - 算术运算式中，低类型数据转换为高类型
      - 赋值语句式 —— 右边表达式的最后隐式转换为左边变量的类型，并赋值给他
      - 函数调用中参数传递时，系统按式地将实参转换为形参的类型后，赋给形参
      - 函数有返回值时，系统将隐式地将返回表达式类型转换为返回值类型，赋值给调用函数
  - 六、位运算相关
    - 1 取两数的平均值
      - (a & y) + ((a ^ y) >> 1)
    - 2 只比较两数较大值
      - max = ((a + b) + abs(a - b)) / 2
    - 3 三数取中间数
      - t1 = max(a, b)
      - t2 = max(b, c)
      - t3 = max(a, c)
      - min(t1, min(t2, t3))
  - 七、函数
    - 静态函数
      - 不可被其他文件调用的函数
    - 函数重载
      - 参数类型不同
      - 参数个数不同
      - 对应的类型导致要求
  - 八、#define & const & sizeof
    - 1 #define实例
      - #define SEC (60 * 60 * 24 * 365)UL
      - #define MIN(A, B) ((A) <= (B) ? (A) : (B))
    - 2 const,#define的区别
      - const
        - 有数据类型
        - 可进行类型安全检查
        - 可对其进行调试
      - #define
        - 没有数据类型
        - 仅进行字符替换，没有类型安全检查
        - 无法调试
      - 对const
        - 被当做一个不能被改变的普通变量
        - extern
          - const bufsize = 100;
          - char buf[bufsize];
    - 3 字节对齐
      - (1) 数据对齐规则
        - 结构的首地址必须是结构体最大类型的整数倍地址
        - 结构体的每一个成员起始地址必须是自身类型大小的整数倍
      - (2) 结构体的整体大小必须可被对齐规整除
      - (3) 结构体的整体大小必须可被本结构内的最亮类型整除
    - 4 sizeof
      - 结构体或类内的静态变量
        - struct s{
          - int a;
          - static int b;
        - };
        - s ss;
        - sizeof(ss)
        - 结果：4
          - 静态变量存放在全局数据区
          - sizeof计算栈中分配的大小
      - 任何类型指针大小相同：4（32位）
      - 对函数使用sizeof —— 在编译时将会被函数返回值的类型取代
      - 空类大小 —— 空类大小
        - 单继承：1
        - 多继承：1
        - 虚继承：4 —— 涉及虚表（虚指针）
    - 5 内联函数 va、宏
      - 相比普通函数 —— 加快程序运行速度
        - 不需中断调用
      - 内联
        - 直接嵌入目标代码
        - 要保数据类型检查
      - 宏
        - 简单的替换
        - 不象参数类型检查