在 64位 和 32 位操作系统上 sizeof 的区别

主要受到影响的是 long 和 指针 的字节

不受影响 都一样：

char : 1

Unsinged char : 1

Singed char : 1

Int: 4

Unsinged int: 4

Singed int : 4

Short : 2

Float : 4

Double: 8

受影响 在 64 位操作系统下

long : 8 long int : 8 unsinged long int: 8

long double : 16

(void\*) (char\*) (int\*) (long\*) (float\*)

(double\*) (long long\*) : 8

在 32 位操作系统下则为 一半

**Linux 命令**

find / -name “test.c”

Ctrl-z : 挂起一个进程 将一个正在前台执行的命令放到后台并暂停

bg : 将进程放到后台并唤醒

fg : 将进程放到前台并唤醒

/bin：存放最常用命令；  
/boot：启动Linux的核心文件；  
/dev：设备文件；  
/etc：存放各种配置文件；  
/home：用户主目录；  
/lib：系统最基本的动态链接共享库；  
/mnt：一般是空的，用来临时挂载别的文件系统；  
/proc：[虚拟目录](https://www.baidu.com/s?wd=%E8%99%9A%E6%8B%9F%E7%9B%AE%E5%BD%95&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://www.nowcoder.com/test/question/_blank)，是内存的映射；  
/sbin：系统管理员命令存放目录；  
/usr：最大的目录，存许应用程序和文件；  
/usr/X11R6：[X-Window](https://www.baidu.com/s?wd=X-Window&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://www.nowcoder.com/test/question/_blank)目录；  
/usr/src：Linux源代码；  
/usr/include：系统头文件；  
/usr/lib：存放常用动态链接共享库、静态档案库；

平均等待时间： 所有作业的运行时间/总的作业数

平均周转时间： 所有作业的运行时间+所有作业的等待时间/总的作业数

Eg. 现有4个同时到达的作业J1,J2,J3和J4,它们的执行时间分别是1小时，3小时，5小时，7小时，系统按单道方式运行且采用短作业优先算法，则平均周转时间是（7.5）小时

MySQL数据库中，变量存在两种：1. 系统变量 以“@@”开头；2.用户自定义变量

系统变量分为：1.全局系统变量 global

1. 会话系统变量 session

@@global 仅访问全局系统变量的值

@@session 仅访问会话系统变量的值

@@ 先访问会话 不存在则再去访问全局

Sql.mode 既是全局 又是会话...(三种方式都可以访问)

总时差TF( Total Float ):

指一项工作在不影响总工期的前提下所具有的机动时间，也就是 工作最早完成时间 和 最晚完成时间的差

自由时差FF( Free Float ):

指一项工作在不影响其紧后工作最早开始时间的条件下，本工作可以利用的机动时间。

通俗的说 总时差就是最多能拖多少天才开始启动工作；自由时差就是最早完成工作 到下次工作间可以休息多少天。

Eg. 在工程网络计划中，工作 M 的最早开始时间为第 16 天，其持续时间为 5 天。该工作有三项紧后工作，他们的最早开始时间分别为第 25 天、第 27 天和第 30 天，最迟开始时间分别为第 28 天、第 29 天和第 30 天。则工作 M 的总时差为（7 ）天。

类指针的声明是不会调用构造函数的...

但是指向一个类实例（new）会调用构造函数，类的声明也会调用构造函数.

下面例子的输出为：AXBB

1. P1类声明 调用默认构造函数 输出A
2. P2 实例化 调用带参构造函数 输出X
3. Delete p2 释放内存 调用析构函数 输出B
4. Return 0 程序结束 再次调动析构函数 输出B

**using** **namespace** std;

**class** TestClass{

**char** x;

**public**:

    TestClass() { cout << 'A'; }

    TestClass(**char** c) { cout << c; }

    ~TestClass() { cout << 'B'; }

};

**int** main() {

    TestClass p1, \*p2;

    p2 = **new** TestClass('X');

**delete** p2;

**return** 0;

}