注意：

1. 如果子进程拷贝对象，对象和网络或者文件相关联，那么父进程会使用同一套对象属性，相互有一定的 关联性。
2. 如果子进程中单独创建对象，则和父进程完全没有关联

process进程对象属性

p.start()

p.join()

p.is\_alive()判断进程生命周期状态，处于生命周期得到True,否则返回FALSE

p.name进程名称默认为process-1

p.pid进程的pid号

p.daemon

默认状态为False主进程退出不会影响子进程执行 如果设置为TRUE则会随着主进程结束而结束

\*要在start前设置

\*一般不和join一起使用

创建自定义进程类

1. 继承Process

init

2、编写自己的\_\_init\_\_，同时加载父类init方法

spuer

3、重写run方法，可以通过生成的对象调用start自动运行

多进程

优点：可以使用计算机多核，进行任务的并发执行，提高执行效率

空间独立，数据安全

运行不受其他进程影响，创建方便

缺点：进程的创建和删除消耗的系统资源较多

进程池技术

产生原因：如果有大量的任务需要多进程完成。则可能则可能需要频繁的创建删除进程，给进计算带来较多而资源消耗。

原理：创建适当的进程放入进程池，用来处理待处理的事件，处理完毕后进程不销毁，仍然在进程池中等待处理其他事件。进程的复用降低了资源的消耗。

使用方法

1. 创建进程池，在池内放入适当的进程
2. 将事件加入到进程池等待队列
3. 不断取进程执行事件，直到所有事件执行完毕
4. 关闭进程池，回收进程

from multipeocessing import Pool

Pool(processes)

功能：创建进程池对象

参数：表示进程池中有多少进程

pool.apply\_async(func,args,kwds) 异步

功能： 将事件放入到进程池队列

参数：func事件函数

args以元组形式给func传参

kwds以字典形式给func传参

返回值：返回一个代表进程池事件的对象

pool.apply (func,args,kwds)

功能： 将事件放入到进程池队列

参数：func事件函数

args以元组形式给func传参

kwds以字典形式给func传参

pool.close（）

功能：关闭进程池

pool.join()

功能:回收进程池

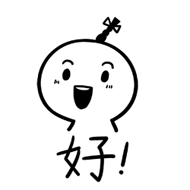
pool.map(fun，iter)

功能：将要坐的事件放入进程池

参数：func要执行的函数

iter迭代对象

返回值：返回事件函数的返回值列表



进程间通信（IPC）

原因：进程空间相对独立，资源无法相互获取，此时在不同进程间通信需要专门方法

进程间通信方法：管道 消息队列 共享内存 信号 信号量 套接字

管道通信：Pipe

通信原理：在内存中开辟管道空间，生成管道操作对象，多进程使用“同一个”管道对象进行操作即可实现通信。

multiprocessing----》Pipe

fd1,fd2 = Pipe(suolex = True)

功能：创建管道

参数：默认表示双向管道

如果设置为False则为单向管道

返回值： 表示管道的两端

如果是双向管道 都可以读写

如果是单向管道，则fd1只读，fd2只写

fd.recv()

功能：从管道读取信息

返回值：读取到的内容

\*如果管道为空则为阻塞

fd.send(data)

功能：向管道写入内容

参数：要写入的内容

\*可以发送Python数据类型

消息队列：

队列：线性的数据结构，先进先出

通信原理：在内存中建立队列数据结构模型，多个进程都可以通过队列存入内容，取出内容的顺序和存入顺序保持一致

创建队列：：

q = Queue(maxsize=0)

功能 ： 创建消息队列

参数：表示最多存放多少消息，默认表示根据内存分配存储

返回值： 队列对象

q.put(data,[block,timeout])

功能： 向队列存储消息

参数：data要存的内容

block默认队列满时会阻塞设置为false则非阻塞

timeout超时时间

data = q.get([block,timeout])

功能：获取队列信息

参数：block默认队列空时会阻塞设置为false则非阻塞

timeout超时时间

返回值：返回取出的内容

q.full()判断队列是否为满

q.empty()判断队列是否为空

q.qsize()判断队列消息数量

q.close()关闭队列

共享内存

通信原理：在内存中开辟一块空间，对多个进程课件，进程可以写入输入，但是每次写入的内容会覆盖之前的内容。

obj = Value(ctype,obj)

功能： 开辟共享内存的空间

参数：ctype要存储的数据类型

obj共享内存的初始化数据

返回： 共享内存对象



obj.value()即为共享内存值，对其修改即修改共享内存、

、

obj = Array(ctype,obj)

功能：开辟共享内存空间

参数：ctype 要存储的数据格式

obj 初始化存入的内容，比如列表，字符串，如果是整数则表示开辟空间的个数

返回值：返回共享内存的对象



\*可以通过遍历过每一个元素的值

e.g 【1，2，3】----》obj[1]==2

\*如果存入的是字符串

obj.value表示字符串的 首地址

管道 消息队列 共享内存

开辟方式 内存 内存 内存

读写方式 两端读写 先进先出 覆盖之前的内容

双向 单向

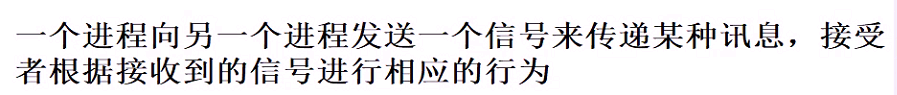
效率 一般 一般 较高

应用 多用于 广泛灵活 需要注意进行

父子进程 互斥操作

信号通信

一个进程向另一个进程法师那个一个信号来传递某种讯息，接受者根据接收到的信号进行相应的行为：



kill –l查看系统信号

kill –sig PID 向一个进程发送信号

关于信号：

信号名称 信号含义 默认处理方法

SIGHUP

SIGINT CTRU-C

SIGQUIT CTRU -\

SIGKILL 终止一个进程

SIGTSTP Ctrl –z

SIGSTOP 暂停一个进程

SIGALRM 时钟信号

SIGCHLD 子进程状态改变时给父进程发出

Python发送信号

signal

os.kill(pid,sig)

功能：发送信号

参数：pid 目标进程

sig 要发送的信号