进程和线程

1. 大部分操作系统的任务调度采用时间片轮转的方式
    一个任务执行一小段时间(us)，会强制暂停，执行下一个任务，每个任务轮回执行
    时间片：一小段时间
    运行状态：任务执行的状态
    就绪状态：暂停的任务
2. CPU的执行效率高，时间片短，个个任务之间的切换就快，人感觉不到的---感觉并行
    宏观并行，微观串行

进程

1. 进程是一个独立功能的程序，在一个数据集合的一次动态执行过程，是操作系统进行资源分配和调度的以独立单位
    进程是正在执行的程序
    操作系统将时间片分配给了进程
    进程和进程之间相互独立，资源不共享
2. 进程是一个抽象的概念，没有统一的定义，由：程序，数据集合，进程块三部分构成
    程序：用于描述进程要完成的功能，是进程执行的指令
    数据集合：程序在执行的时候所需要的数据和工作区
    进程控制块：包含进程的描述信息，是进程存在的唯一标识
3. 进程的特性：
    1. 动态性：
        进程实质时候程序在多道程序中的执行过程，进程是动态产生，动态消亡
    2. 并发性：
        任何进程都可以同其他进程一起并发执行
    3. 独立性：
        进程是一个独立运行的基本单位，同时也是系统分配资源和调度的独立单
    4. 结构性

线程

1. 一个进程可以有很多线程,每个线程执行的任务不同
    同一个进程中有很线程，将共享进程中的全部资源
2. 一个线程由：线程ID，指令集，栈，寄存器组成
    线程ID：线程的唯一标识
    指令集：线程首先本质上就是一个指令集，通过这个指令集告诉CPU自己要做的事情
    栈：当创建一个线程时，就应该有一个专属的内存空间来存储线程的相关信息，这块内存空间就是栈
    寄存器：也是工作内存多线程共享一块主内存，工作内存

总结：

1. 线程是进程执行的最小单位，进程是操作系统分配资源的最小单位
2. 线程是一个进程中代码的不同执行路线
3. 线程和线程之间互相独立，但是资源共享（共享进程的资源）
4. 线程的调度和切换，线程的上下文切换要比进程的切换速度快的多

单线程和多线程

单线程：一个进程只有一个线程
多线程：一个进程中有多个线程

任何一个程序至少有一个线程====主线程

```
#单线程
def fun(music,count):
    for i in range(count):
        print('listen to the music %s遍'%(i+1))

def fun2(movie,count):
    for i in range(count):
        print('watch movie %s遍'%(i+1))


if __name__=='__main__':
    fun('凉凉',3)
    fun2('中国机长',3)
```

多线程模块：threading

Python提供了多种模块用来支持多线程编程
创建线程的方式
1. 继承Thread类并实现run方法
        threading.Thread类构造方法
        __init__(self,group=None,traget=None,name=None,args())
        group:预留参数，用于扩展功能
        target：可调对象（目标函数）
        name:线程的名字

```
import threading

# 继承thread类，实现run方法
class MyThead(threading.Thread):
    def __init__(self,name):
        super().__init__()
        self.name=name

    def run(self):
        #线程要做的事
        for i in range(10):
            print(self.name)
t1=MyThead('hello')
t2=MyThead('world')

t1.start()
t2.start()
```

2. 通过Thrad类的构造方法直接创建线程

```
def fun(music,count):
```

```
    for i in range(count):
        print('listen to the music %s遍'%(i+1))

def fun2(movie,count):
    for i in range(count):
        print('watch movie %s遍'%(i+1))


if __name__=='__main__':
    t1=threading.Thread(target=fun,args=('野狼disco！',3))
    t2=threading.Thread(target=fun2,args=('音乐僵尸',3))
    t1.start()
    t2.start()
    print(11111)
```

- Python的多线程创建形式

```
1. 创建线程（两种形式）
2. 开启线程
```