1. 什么是进程

进程是程序运行在操作系统上的实例，进程是操作系统分配资源的基本单位。

1. 进程和线程的区别是什么？

定义不同：进程是系统进行资源分配和调度的一个独立单位。 线程是进程的一个实体，是CPU调度和分配的基本单位 一个程序至少有一个进程，一个进程至少有一个线程。 线程的划分尺度小于进程(资源比进程少)，使得多线程程序并发性高。 进程在执行过程中拥有独立的内存单元，而多个线程共享内存，从而极大提高了程序的运行效率 线程不能独立运行，必须放在进程中。

1. 什么是主进程？

程序开始运行，就存在的进程就是主进程。

1. 写出创建进程的步骤

import multiprocessing

def process1():

print("这是一个子进程......")

def main():

process = multiprocessing.Process(target=process1)

process.start()

if name == 'main':

main()

1. 怎么获取进程的pid？

getpid()

1. 怎么给进程函数传递参数?

process = multiprocessing.Process(target=process1, args=(参数1, 参数2, ....))

1. 在主进程中，怎么阻塞等待子进程运行完成之后，才继续运行？

Join()

1. Process创建的实例对象的常用方法有什么？

1、start()：启动子进程实例（创建子进程）

2、is\_alive()：判断进程子进程是否还在活着

3、join([timeout])：是否等待子进程执行结束，或等待多少秒

4、terminate()：不管任务是否完成，立即终止子进程

编程

1. 编写一个程序，实现创建一个子进程，子进程每一秒打印一次：这是子进程，循环的第n次，打印 的次数是由主进程传递给子进程。

import multiprocessing

import time

def pro\_func(num):

for i in range(num):

time.sleep(1)

print("----> %d" % i)

if \_\_name\_\_ == '\_\_main\_\_':

num = int(input("请输入要循环的次数:"))

pro = multiprocessing.Process(target=pro\_func, args=(num,))

pro.start()