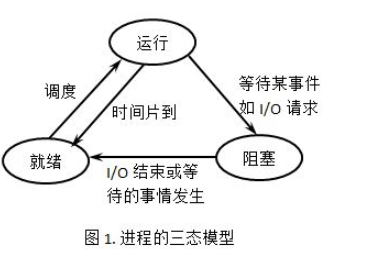
进程在运行中不断地改变其运行状态。通常，一个运行进程必须具有以下三种基本状态。  
就绪(Ready)状态  
当进程已分配到除CPU 以外的所有必要的资源，只要获得处理机便可立即执行，这时的进程状态称为就绪状态。  
执行（Running）状态  
当进程已获得处理机，其程序正在处理机上执行，此时的进程状态称为执行状态。  
阻塞(Blocked)状态  
正在执行的进程，由于等待某个事件发生而无法执行时，便放弃处理机而处于阻塞状态搜索。引起进程阻塞的事件可有多种，例如，等待I/O完成、申请缓冲区不能满足、等待信件(信号)等。



背景知识：

栈存放什么：栈存储运行时声明的变量——对象引用（或基础类型, primitive）内存空间， 栈的实现是先入后出的。

堆存放什么：堆分配每一个对象内容（实例）内存空间。

栈溢出：java.lang.StackOverflowError

堆溢出：java.lang.OutOfMemoryError: Java heap space

栈溢出实现，可以递归调用方法，这样随着栈深度的增加，JVM 维持着一条长长的方法调用轨迹。

堆溢出实现，可以循环创建对象或大的对象；

直到内存不够分配，产生栈溢出。

一、栈溢出

两种情况：

线程请求的栈深度大于虚拟机允许的最大深度 StackOverflowError

虚拟机在扩展栈深度时，无法申请到足够的内存空间 OutOfMemoryError

二、堆溢出

创建对象时如果没有可以分配的堆内存，就会出现堆溢出。

三、 两者实例如下：

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.Scanner;

public class testCode {

public void testHeap(){//堆溢出

List<byte[]> list = new ArrayList<>();

int i=0;

while (true){

list.add(new byte[5\*1024\*1024]);

System.out.println("count is: "+(++i));

}

}

int num = 1;

public void testStack(){ //栈溢出

num++;

this.testStack();

}

public static void main(String[] agrs){

testCode t = new testCode();

t.testHeap();

t.testStack();

}

}

/\*\*java堆溢出实例  
 \* 原理：java的堆是用来存放对象实例的，所以我们只要做到以下三点就可以使堆溢出：  
 \* 1、限制堆的大小，不可扩展  
 \* 2、不断新建对象  
 \* 3、保持对象存活不被回收  
 \* 对应的，我们需要：  
 \* 1、改变JVM的启动参数，将堆的最小值和最大值设成一样，这样就可以避免堆自动扩展（其实不一样也可以）  
 \* 2、不断产生对象  
 \* 3、使用一个List来保存对象，保持对象存活  
 \*