**操作系统实验 单处理器系统的进程调度**

1. 实验目的
2. 加深进程概念理解，明确进程与程序区别。
3. 理解操作系统中进程的组织、创建和调度等方法。
4. 实验内容

编写程序完成单处理器系统的进程调度，要求采用时间片轮转法调度策略。具体内容：

1. 确定PCB内容及其组织方式；
2. 要求模拟进程空闲（新）、就绪、运行、阻塞和完成5个状态；
3. 实现进程创建、进程调度、进程阻塞和进程唤醒4个功能（原语）；
4. 编写主函数对整个系统进程测试。

进一步扩展

采用多队列反馈式调度策略实现进程调度

1. 实现原理

带阻塞功能的多队列反馈式调度策略的原理：

1、由于进程有多种状态，每组队列不仅要按优先级分组，还要再按进程状态分组。

2、对按进程状态分组的队列使用时间片轮转法调度进程。

3、优先级越高，每次执行时间越短

1. 算法描述

使用中文的伪代码描述多队列反馈式调度策略算法。

创建：创建一个带有名称和指定优先级的进程并加入进程状态已初始化队列中。标识符从4开始分配。

''' <摘要>

''' 创建一个进程，并加入进程状态已初始化队列中。

''' </摘要>

''' <参数 名称="基础优先级">进程默认的优先级</参数>

公有 共享的 函数 创建进程(名称$, 基础优先级 为 优先级, 进程工作 为 进程工作时间线) 返回值为 进程模拟器

变量 进程 为 新建 进程模拟器

进程.控制块 = 新建 进程控制块 伴随 {

.名称 = 名称,

.基础优先级 = 基础优先级,

.标识符 = 调度器.当前.进程标识符种子

}

变量 新线程 为 新建 线程模拟器 伴随 {

.工作 = 进程工作,

.控制块 = 新建 线程控制块 伴随 {

.优先级 = 基础优先级,

.标识符 = 进程.控制块.线程标识符种子

}

}

进程.控制块.主线程 = 新线程

调度器.当前.核心.队列散列表(新线程.控制块.优先级)(进程状态.已初始化).入列(进程)

返回 进程

结束 函数

阻塞和恢复：这两个操作是通过修改进程状态和优先级完成的。修改优先级时进程在相应的队列间移动。

公有 过程 修改进程状态和优先级(状态 为 进程状态, 优先级 为 优先级?)

如果 是主线程 则

如果 优先级.有值 则

变量 队列散列表 = 调度器.当前.核心.队列散列表

变量 进程 = 队列散列表(控制块.优先级)(控制块.状态).出列()

队列错误检查(进程)

控制块.状态 = 状态

控制块.优先级 = 优先级.值

队列散列表(优先级.值)(状态).入列(进程)

否则

当前对象.状态 = 状态

结束 如果

结束 如果

结束 过程

''' <摘要>

''' 获取或切换线程优先级。主线程切换优先级会改变调度器中的队列。

''' </摘要>

公有 属性 优先级 为 线程优先级

取值

返回 控制块.优先级

结束 取值

赋值(新值 为 线程优先级)

如果 是主线程 则

变量 当前队列散列表 = 调度器.当前.核心.队列散列表

变量 进程 = 当前队列散列表(优先级)(状态).出列

队列错误检查(进程)

控制块.优先级 = 新值

当前队列散列表(新值)(状态).入列(进程)

引发事件 属性发生变化(当前对象, 新建 属性变化事件参数(取名称(优先级)))

结束 如果

结束 赋值

结束 属性

''' <摘要>

''' 获取或切换线程状态。主线程切换状态会改变调度器中的队列。使用限制：调节状态的对象必须位于队列顶端。

''' </摘要>

公有 属性 状态 为 线程状态

取值

返回 控制块.状态

结束 取值

赋值(新值 为 线程状态)

如果 是主线程 则

变量 当前队列散列表 = 调度器.当前.核心.队列散列表(控制块.优先级)

变量 进程 = 当前队列散列表(状态).出列()

队列错误检查(进程)

控制块.状态 = 新值

当前队列散列表(新值).入列(进程)

引发事件 属性发生变化(当前对象, 新建 属性变化事件参数(取名称(状态)))

结束 如果

结束 赋值

结束 属性

调度：此函数如果执行结果为真，则再次调用此函数调度接下来的进程。

大致的步骤是

* 查看进程创建命令列表，按需创建进程
* 捕获执行前快照用于显示
* 按优先级从高到低顺序找待启动的和就绪的进程执行相应的时间片执行，然后调整它们的队列，刚启动的进程是抢占式执行的。
* 更新每个阻塞的进程的等待信息。如果到了唤醒时间则让它进入下一段任务相关的队列
* 捕获执行后的快照用于显示

''' <摘要>

''' 单步执行。完成会返回 假

''' </摘要>

友元 函数 单步执行() 返回值为 布尔值

对 每个 当前进程 在 进程创建命令列表

如果 当前进程.起始时间 > 上次处理时间 并且 当前进程.起始时间 <= 现在处理时间 则

快照.标头.添加(当前进程.名称)

进程模拟器.创建(当前进程)

结束 如果

下一个

变量 进程快照表 为 新建 顺序列表(泛型参数 进程快照)

上次处理时间 = 现在处理时间

变量 是否有进程执行了 = 假

变量 切换了等待的进程 = 假

变量 指定运行的时间 为 新建 整数?

变量 运行过的进程标识符 = 0

'拍摄进程快照

变量 队列快照 为 新建 字符串构建器

对 每个 当前进程优先级 在 优先级表

变量 进程状态散列表 = 优先级散列表(当前进程优先级)

对 每个 当前进程状态 在 进程状态表

对 每个 当前进程 在 进程状态散列表(当前进程状态)

队列快照.添加行($"在优先级{当前进程优先级}, 进程状态{当前进程状态}, 有进程标识符为{当前进程.控制块.标识符}的进程")

下一个

下一个

下一个

'处理需要启动和执行的进程

枚举进程(过程(当前进程状态, 当前进程状态队列, 当前进程)

选择 情况 当前进程状态

情况 进程状态枚举.已初始化

当前进程.开始()

抢占任务

按照 情况 进程状态枚举.就绪 定义的行为执行任务

单步执行 = 真

情况 进程状态枚举.就绪

如果 非 是否有进程执行了 则

变量 线程快照 = 当前进程.运行或等待()

指定运行的时间 = 线程快照.消耗时间

现在处理时间 += 线程快照.消耗时间

进程快照表.添加(新建 进程快照(线程快照, 当前进程.控制块.标识符, 当前进程.控制块.名称))

变量 主线程 = 当前进程.控制块.主线程

变量 优先级 为 进程优先级枚举

选择 情况 主线程.优先级

情况 进程优先级枚举.最高

优先级 = 进程优先级枚举.高于普通

情况 进程优先级枚举.高于普通

优先级 = 进程优先级枚举.普通

情况 进程优先级枚举.普通

优先级 = 进程优先级枚举.低于普通

情况 进程优先级枚举.低于普通, 进程优先级枚举.最低

优先级 = 进程优先级枚举.最低

情况 否则

抛出 新建 模拟出错($"进程 {当前进程.控制块.名称} [{当前进程.控制块.标识符}] 的主线程 (标识符为{主线程.控制块.标识符}) 优先级数据损坏。")

结束 选择

主线程.换出(优先级)

是否有进程执行了 = 真

运行过的进程标识符 = 当前进程.控制块.标识符

结束 如果

单步执行 = 真

情况 进程状态枚举.阻塞

单步执行 = 真

情况 否则

抛出 新建 模拟出错("提供了错误的线程状态")

结束 选择

结束 过程)

'处理在等待的进程

枚举进程(过程(当前进程状态, 当前进程状态队列, 当前进程)

选择 情况 当前进程状态

情况 进程状态枚举.已初始化, 进程状态枚举.就绪

单步执行 = 真

情况 进程状态枚举.阻塞

'每一个线程都经历等待IO操作，并且未完成等待的应该留下快照。

对 每个 阻塞 在 当前进程状态队列

变量 阻塞控制块 = 阻塞.控制块

变量 任务列表 = 阻塞控制块.主线程.任务列表

如果 运行过的进程标识符 不是 阻塞控制块.标识符 并且 非 任务列表.应该切换 则

变量 线程快照 = 阻塞.运行或等待(指定运行的时间)

如果 线程快照.执行状态 不是 进程状态枚举.阻塞 则

抛出 新建 模拟出错("检测到阻塞队列中的非阻塞进程")

结束 如果

进程快照表.添加(新建 进程快照(线程快照, 阻塞控制块.标识符, 阻塞控制块.名称))

结束 如果

下一个

'每次只能切换一个等待的进程

如果 非 切换了等待的进程 并且 运行过的进程标识符 不是 当前进程.控制块.标识符 则

变量 主线程 = 当前进程.控制块.主线程

变量 顶部任务 = 主线程.任务列表

如果 顶部任务.应该切换 则

顶部任务.换出() '如果等待完了，切换任务到下一个

调试.写一行($"切换了进程：{当前进程.控制块.名称}")

主线程.换出() '移出阻塞队列, 然后看看应该移动到哪里

切换了等待的进程 = 真

结束 如果

结束 如果

单步执行 = 真

情况 否则

抛出 新建 模拟出错("提供了错误的线程状态")

结束 选择

结束 过程)

'写日志以便显示

如果 进程快照表.有任何一个 则 快照.写入(进程快照表, 队列快照.转换为字符串, 现在处理时间)

返回 单步执行

结束 函数

1. 执行测试

具体执行过程在课堂完成。这里只提供一张运行截图：

