Лабораторная работа 3

ДОПОЛНИТЕЛЬНЫЕ ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

Разработайте решение (алгоритм) указанной задачи. Для синхронизации процессов используйте указанный в задании механизм. Выполните реализацию решения задачи с помощью механизмов синхронизации учебной ОС pintos в виде теста ОС pintos. При запуске теста в стандартный поток вывода должны выводиться отладочные сообщения, демонстрирующие взаимодействия сущностей вашей задачи.

Опишите предложенный вами алгоритм в отчете. Приведите диаграмму состояний и взаимодействия процессов, а также несколько результатов тестовых прогонов вашего решения с разными входными данными.

Вариант	Задача	Механизм синхронизации	Файл реализации
1	Спящий парикмахер	Мониторы	tests/threads/hair.c
2	Медведь и пчелы	Мониторы	tests/threads/bear.c
3	Голодные птицы	Мониторы	tests/threads/birds.c
4	Молекула воды	Мониторы	tests/threads/molec.c
5	Общая душевая	Мониторы	tests/threads/shower.c
6	Американские горки	Мониторы	tests/threads/rolcoast.c
7	Спящий парикмахер	Семафоры	tests/threads/hair.c
8	Медведь и пчелы	Семафоры	tests/threads/bear.c
9	Голодные птицы	Семафоры	tests/threads/birds.c
10	Молекула воды	Семафоры	tests/threads/molec.c
11	Общая душевая	Семафоры	tests/threads/shower.c
12	Курильщики	Семафоры	tests/threads/smokers.c
13	Американские горки	Семафоры	tests/threads/rolcoast.c

Текст задач:

Спящий парикмахер:

У парикмахера есть одно рабочее место и приемная с несколькими стульями. Когда у парикмахера нет посетителей — он спит в своем кресле. Если приходит посетитель — он будит парикмахера, он усаживает клиента в кресло и подстригает его. Когда парикмахер заканчивает подстригать клиента, он отпускает клиента и затем идет в приёмную, чтобы посмотреть, есть ли там ожидающие клиенты. Если они есть, он приглашает одного из них и стрижет его. Если ждущих клиентов нет, он возвращается к своему креслу и спит в нем. Представьте парикмахера и посетителей процессами и реализуйте решение задачи. Для синхронизации используйте механизм, указанный в варианте. Во время стрижки процесс парикмахера вызывает timer_sleep на некоторое количество тиков (например, 10).

Тестирование решения:

```
pintos --qemu --  -q run "hair 5 100"
```

где первое число — количество посетителей в очереди (N), второе число — интервал (I) между двумя посетителями (в тиках), следующий посетитель приходит не раньше чем через I тиков.

Медведь и пчелы

Есть N пчел и 1 медведь. Они пользуются одним горшком меда, вмещающим K порций меда. Сначала горшок пустой. Пока горшок не наполнится, медведь спит. Потом съедает весь мед, и снова засыпает. Каждая пчела многократно собирает одну порцию меда и кладет ее в горшок. Пчела, которая приносит последнюю порцию меда и заполняет горшок, будит медведя. Представьте медведя и пчел процессами, разработайте код, моделирующий их действия. Для синхронизации используйте механизм, указанный в варианте. Во время поедания меда медведем пчелы могут собирать мед, но не

могут складывать его в горшок. Во время сбора меда процесс пчелы вызывает timer_sleep на некоторое количество тиков (например, 10).

Тестирование решения:

```
pintos --qemu --  -q run "bear 20 17"
```

где первое число — количество пчел (N), второе число — вместительность горшка (K).

Молекула воды

Атомы водорода и кислорода колеблются в пространстве, пытаясь собраться в молекулы воды. Для этого нужна взаимная синхронизация двух атомов водорода и одного атома кислорода. Пусть атомы кислорода и атомы водорода моделируются процессами (потоками). Каждый атом водорода вызывает процедуру *готовности*, когда он готов объединиться в молекулы. Аналогично обстоит дело и с атомами кислорода. Напишите реализацию этой процедуры, используя указанный в варианте механизм синхронизации. Обеспечьте отсутствие взаимной блокировки и зависания. Выполнение процесса атома на время вызова процедуры готовности приостанавливается (атом «засыпает»). После завершения процедуры готовности образуется молекула воды и процессы атомов, участвующих в создании молекулы, завершаются. На образование молекулы требуется определенное время: это моделируется вызовом timer_sleep (например, на 10 тиков) на время образования молекулы в процессе, создающем молекулу.

Тестирование решения:

```
pintos --qemu --  -q run "molec 10 25 50"
```

где первое число — количество атомов кислорода (O), второе — водорода (H), третье число — интервал появления атомов (I) в тиках, каждый следующий атом пребывает в систему не раньше чем через I тиков после предыдущего.

Голодные птицы

Есть N птенцов и их мать. Птенцы едят из общей миски, в которой сначала находятся F порций пищи. Мать спит. Каждый птенец съедает порцию еды, отдыхает некоторое время (timer_sleep), затем снова ест. Когда кончается еда, птенец, питавшийся последним, будит мать. Птица наполняет миску F порциями еды и снова спит, пока миска не опустеет и ее не позовет птенец. Эти действия повторяются бесконечно. Представьте птиц процессами и разработайте код, моделирующий их действия. Для синхронизации используйте указанный в варианте механизм.

Тестирование решения:

```
pintos --qemu --  -q run "birds 10 3"
```

где первое число — количество птенцов (N), второе число — вместимость тарелки для птенцов (F).

Американские горки

Есть N пассажиров и 1 вагончик. Пассажиры ждут очередь, чтобы проехать в вагончике, вмещающем C человек (C<N). Вагончик может ехать только заполненным. Вагончик ждет пока в него сядет C пассажиров, катает их, и снова ждет полного заполнения. Представьте пассажиров и вагончик процессами и разработайте коды процесса-пассажира и процесса-вагончика, для синхронизации используйте указанный в варианте механизм. Во время катания процесс вагончика вызывает timer_sleep на некоторое количество тиков (например, 10).

Тестирование решения:

```
pintos --qemu --  -q run "rolcoast 10 3 100"
```

где первое число — количество желающих покататься пассажиров (N), второе число — вместительность вагончика (C), третье число — интервал между подходящими пассажирами (I) в тиках, следующий желающий приходит не раньше чем через I тиков после предыдущего.

Общая душевая

В общежитии есть душевая, которой могут пользоваться и мужчины и женщины, но не одновременно. Представьте мужчин и женщин процессами и разработайте решение, используя механизм синхронизации, указанный в варианте. Учтите, что в душевой одновременно может находиться не более 3 человек. Решение должно обеспечивать необходимое исключение и отсутствие взаимоблокировок. Решение должно быть справедливым. На время помывки процесс вызывает timer_sleep на некоторое количество тиков (например, 10).

Тестирование решения:

```
pintos --qemu --  -q run "shower 10 15 100"
```

где первое число — количество мужчин в очереди, второе — количество женщин в очереди, третье число — интервал между подходящими людьми (в тиках), каждый следующий человек через указанное количество тиков после предыдущего.

Курильщики

Предположим, что есть три процесса-курильщика и один процесс-посредник. Курильщик непрерывно скручивает сигареты и курит их. Чтобы скрутить сигарету необходимо три компонента: табак, бумага, спички. У одного курильщика есть табак, у другого — бумага, у третьего — спички, и запасы этих компонент — бесконечны. Каждый курильщик ожидает появление на столе нужных ему компонентов (бездействует). Посредник кладет на стол два случайных компонента. Тот процесс, у которого есть третий компонент, забирает два компонента со стола, скручивает сигарету и курит ее. Во время "курения" процесс курильщика вызывает timer_sleep (например, на 10 тиков). Посредник дожидается, пока курильщик закончит. После этого цикл повторяется. Разработайте решение этой задачи с заданным механизмом синхронизации. Используйте разумное количество объектов синхронизации в решении.

Тестирование решения:

```
pintos --qemu --  -q run "smokers"
```