|  |
| --- |
| Домашнее задание №5 |
| Выполнила Дьякова Дарья |
| Вариант 6 |

|  |
| --- |
| БПИ207  13.12.2021 |

**Описание задания**

Задача о курильщиках:

Необходимо разработать многопоточное консольное приложение с использованием библиотеки POSIX Threads, моделирующее поведение курильщиков и посредника (в программе представлены три потока-курильщика и один поток-посредник). У одного из процессов-курильщиков есть безграничный запас бумаги, у другого – табака, а у третьего – спичек. Посредник выдает им остальные ингредиенты для сигарет по следующему принципу: посредник кладет на стол два рандомных ингредиента (табак и бумагу/бумагу и спички/табак и спички), тот курильщик, у которого есть третий ингредиент для изготовления сигареты, берет эти ингредиенты и скручивает сигарету. Пока курильщик скручивает сигарету и скуривает ее, посредник и остальные курильщики ждут его. После того как курильщик скурил сигарету, он сообщает об этом посреднику, и тот выкладывает на стол новые ингредиенты. При решении задачи необходимо использовать семафоры.

**Описание реализации задания**

Для реализации задания была выбрана парадигма «Клиенты и серверы», так как в данном случае один сервер-посредник обслуживает три клиента-курильщика. В нашей ситуации клиенты исполняют роль сервера, поскольку посредник получает ответ от курильщиков ответ о том, что они покурили, а курильщики в свою очередь получают запросы-ингредиенты от посредника. Сервер-посредник реализован как процесс, который не может обрабатывать несколько запросов сразу, то есть он дожидается, пока один курильщик скурит сигарету, а затем продолжает выкладывать ингредиенты.

Чтобы избежать проблем с дедлоками, в программе нет отдельного семафора для каждого ингредиента. Вместо этого есть три семафора, обозначающих комбинации ингредиентов.

Алгоритм действий любого из курильщиков (выполняется в бесконечном цикле):

1. Заблокировать семафор своих ингредиентов, чтобы обозначить, что он взял эти ингредиенты со стола
2. Вывести сообщение о том, что он взял ингредиенты со стола
3. Вызвать метод smoke(), имитирующий процесс курения (усыпление потока на заданное количество секунд и вывод времени курения)
4. Вывести сообщение о том, что курильщик скурил сигарету
5. Разблокировать семафор посредника, чтобы сообщить ему, что сигарета скурена

Алгоритм действий посредника:

1. Создать потоки трех курильщиков (посредник начинает взаимодействие с курильщиками)

Следующие действия происходят в цикле, пока количество ингредиентов не равно нулю:

1. Заблокировать семафор, отвечающий за то, покурил курильщик или нет, тем самым обозначая, что теперь посредник не выдает ингредиенты, а ждет, пока курильщик скурит сигарету
2. Генерация комбинации ингредиентов
3. В зависимости от выброшенных на стол ингредиентов выводится текст о выброшенных ингредиентах и разблокируется соответствующий семафор, что означает, что курильщик может брать ингредиенты

Когда все ингредиенты закончились, происходит завершение потока.

**Организация входных данных**

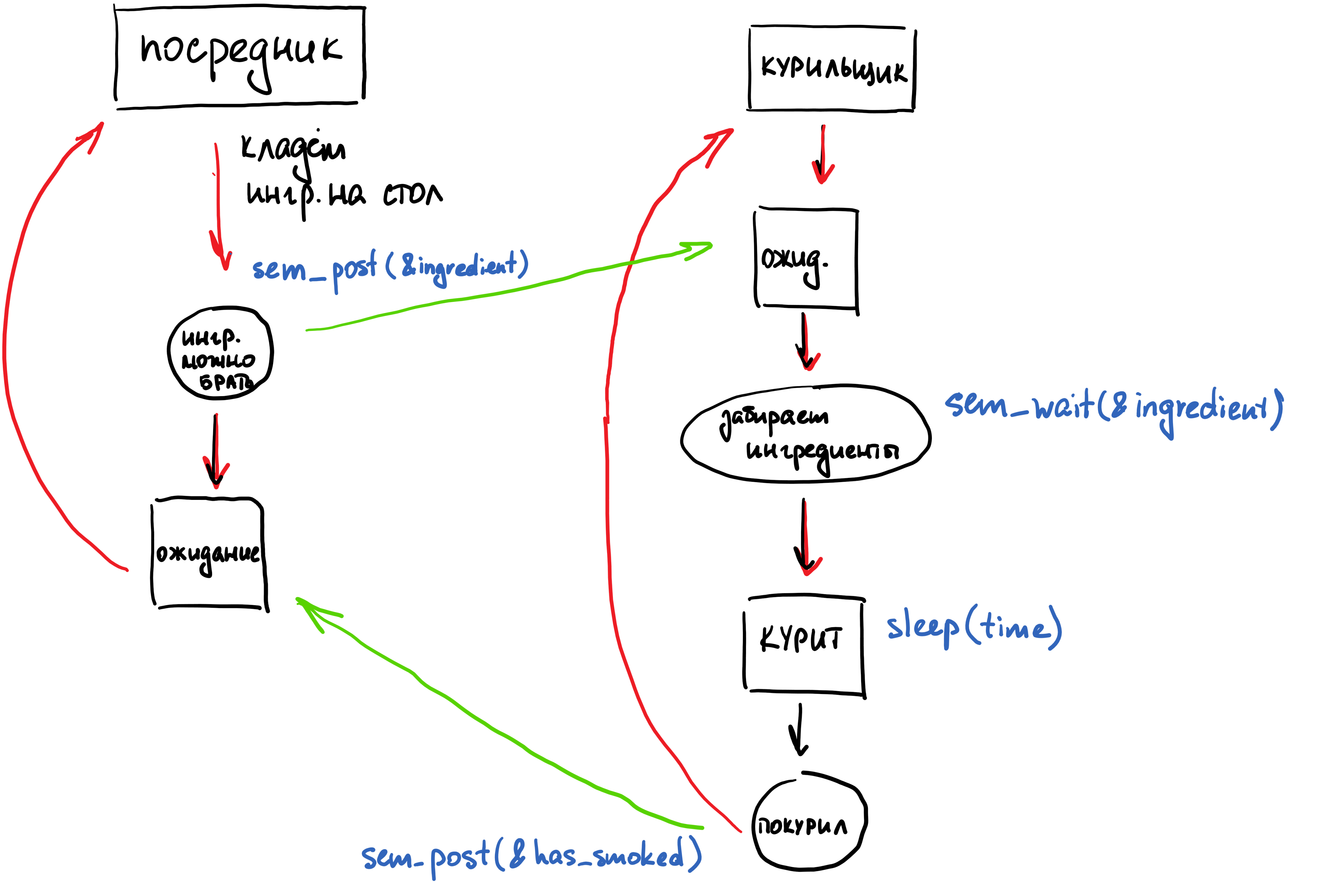
В данной программе пользователь может вводить данные в командную строку, при этом есть три сценария:

1. Пользователь вводит 4 целых числа, разделенных пробелами: первое – число комбинаций ингредиентов, которые есть в запасе у посредника, второе – количество секунд, которое будет курить первый курильщик (курильщик с бумагой), третье – количество секунд, которое будет курить второй курильщик (курильщик с табаком), четвертое – количество секунд, которое будет курить третий курильщик (курильщик со спичками)
2. Пользователь вводит одно целое число: число комбинаций ингредиентов, которые есть в запасе у посредника, остальные параметры генерируются рандомно в диапазоне от 1 до 5
3. Пользователь оставляет командную строку пустой. При этом все параметры задаются с помощью функции rand().

Пример входных данных для первого сценария: 5 1 2 3

У посредника есть 5 наборов ингредиентов, первый курильщик будет курить 1 секунду, второй – две, третий – три.

**Схема работы алгоритма   
(иллюстрация применения парадигмы «клиенты и серверы» в программе)**



**Список источников:**

1. 1)
2. [https://pro-prof.com/forums/topic/parallel-programming-paradigms](https://pro-prof.com/forums/topic/parallel-programming-paradigms )
3. <http://staff.mmcs.sfedu.ru/~dubrov/files/sl_parallel_05_paradigm.pdf>
4. <https://www.opennet.ru/base/dev/linux_threads.txt.html>
5. <https://unetway.com/tutorial/c-mnogopotocnost>
6. <http://ru.manpages.org/sem_overview/7>
7. <https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D1%8C>
8. <https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%BE%D0%BA_%D0%B2%D1%8B%D0%BF%D0%BE%D0%BB%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F>
9. <https://ru.wikipedia.org/wiki/POSIX_Threads>