Приложение А.

Общая постановка задачи.

Требуется разработать программу в виде Linux-приложения, для выполнения различных частей которой создаются и запускается потоки управления, а для синхронизации доступа к требуемым ресурсам используются соответствующие объекты ОС. Выбор синхронизирующих объектов остается на усмотрение разработчика. Результат выполнения выводится на терминал/консоль. Программа должна быть устойчивой к некорректному пользовательскому вводу. Функционирование программы, если это не оговаривается особо, может быть завершено только путем принудительного снятия процесса с выполнения. В большинстве вариантов заданий решаются классические задачи синхронизации. Приостановка выполнения потока на заданный промежуток времени может выполняться вызовом функции *nanosleep()*.

```
#include <time.h>
int nanosleep(const struct timespec* req, struct timespec* rem);
```

Варианты заданий.

Вариант 1. «Обедающие философы 1[1]». В пансионе отдыхают и предаются размышлениям 5 философов (потоки), пронумерованные от 1 до 5. В столовой расположен круглый стол, вокруг которого расставлены 5 стульев, также пронумерованные от 1 до 5. На столе находится одна большая тарелка со спагетти, которая пополняется бесконечно, также там расставлены 5 тарелок, в которые накладывается спагетти, и 5 вилок (разделяемые ресурсы), назначение которых очевидно.

Для того чтобы пообедать, философ входит в столовую и *садится на стул со своим номером*. При этом есть философ сможет только в том случае, если свободны две вилки – справа и слева от его тарелки. При выполнении этого условия философ поднимает одновременно обе вилки и может поглощать пищу в течение какого-то заданного времени. В противном случае, философу приходится ждать освобождения обеих вилок.

Пообедав, философ кладет обе вилки на стол одновременно и уходит.

Величина временного промежутка для поглощения пищи устанавливается пользователем, а появление философа в столовой является случайной величиной с равномерным законом распределения.

Вариант 2. «Обедающие философы 2[1]». В пансионе отдыхают и предаются размышлениям 5 философов (потоки), пронумерованные от 1 до 5. В столовой расположен круглый стол, вокруг которого расставлены 5 стульев, также пронумерованные от 1 до 5. На столе находится одна большая тарелка со спагетти, которая пополняется бесконечно, также там расставлены 5 тарелок, в которые накладывается спагетти, и 5 вилок (разделяемые ресурсы), назначение которых очевидно.

1 http://ru.wikipedia.org/wiki/Проблема_обедающих_философов

Для того чтобы пообедать, философ входит в столовую и *садится на любой стул*. При этом есть философ сможет только в том случае, если свободны две вилки – справа и слева от его тарелки. При выполнении этого условия философ поднимает одновременно обе вилки и может поглощать пищу в течение какого-то заданного времени. В противном случае, философу приходится ждать освобождения обеих вилок.

Пообедав, философ кладет обе вилки на стол одновременно и уходит.

Величина временного промежутка для поглощения пищи устанавливается пользователем, а появление философа в столовой является случайной величиной с равномерным законом распределения.

Вариант 3. «Спящий парикмахер 1[1]». В парикмахерской расположено единственное кресло, на котором спит парикмахер, и несколько стульев (разделяемый ресурс) для клиентов.

Когда клиент (поток) приходит в парикмахерскую, он будит парикмахера, садится в кресло (разделяемый ресурс). Стрижка производится в течение заданного времени. Если же кресло занято другим клиентом, то вновь прибывший клиент занимает любой свободный стул и ожидает своей очереди (клиенты обслуживаются в порядке очередности, например, времени прибытия). Если все стулья заняты, то клиент поворачивается и уходит.

Когда обслужены все клиенты, парикмахер садится в кресло и снова засыпает.

Величина временного промежутка на стрижку и количество стульев устанавливаются пользователем, а появление клиента в парикмахерской является случайной величиной с равномерным законом распределения.

Вариант 4. «Спящий парикмахер 2[1]». В парикмахерской расположено единственное кресло (разделяемый ресурс), на котором спит парикмахер, и несколько стульев (разделяемые ресурсы) для *клиентов (потоки), которые делятся на два класса – обычные и «блатные»*. Сначала всегда обслуживаются «блатные» клиенты, и только после этого парикмахер может работать с обычными клиентами.

Когда клиент приходит в парикмахерскую, он будит парикмахера, садится в кресло. Стрижка производится в течение заданного времени. Если же кресло занято другим клиентом, то вновь прибывший клиент занимает любой свободный стул и ожидает своей очереди. Далее клиенты обслуживаются в порядке приоритета и очередности (времени прибытия). Если все стулья заняты, то клиент поворачивается и уходит.

Когда обслужены все клиенты, парикмахер садится в кресло и снова засыпает.

Величина временного промежутка на стрижку и количество стульев устанавливаются пользователем, а появление клиента и его статус («простой»

1 http://ru.wikipedia.org/wiki/Проблема_спящего_парикмахера

или «блатной») в парикмахерской являются случайными величинами с равномерным законом распределения.

Вариант 5. «Модель писателей/читателей». Рассмотрим взаимодействие двух потоков, один из которых пишет данные в буферный пул, а другой считывает их из пула. Буферный пул состоит из *N* буферов, каждый содержит одну запись. В общем случае поток-писатель и поток-читатель имеют разные скорости (длительности выполнения операция чтения и записи) и обращаются к пулу с переменной интенсивностью (начинают выполнять операции в случайные моменты времени с равномерным законом распределения). Для правильной работы поток-писатель приостанавливается, когда все буферы заняты, и переходит в активное состояние при наличии хотя бы одного свободного буфера. Поток-читатель приостанавливается, когда все буферы пусты, и активизируется, когда появляется, по крайней мере, одна запись.

Количество потоков-писателей и читателей, длительности операций чтения и записи, размер буферного пула устанавливаются пользователем.

Вариант 6. «Экзамен 1». В аудитории идет экзамен, на котором присутствуют *N* студентов (потоки) и преподаватель (поток). У студентов имеется две шпаргалки (разделяемые ресурсы), которые они передают друг другу в случайном порядке. Преподаватель читает газету и пытается уличить студентов в списывании, для чего через случайные промежутки времени убирает газету и осматривает аудиторию. Если во время очередного осмотра студенты будут передавать шпаргалку (разделяемый ресурс свободен), то экзамен заканчивается, и все получают оценку «неудовлетворительно». В случае, если у каждого студента шпаргалка побывала 3 раза, и преподаватель не смог поймать списывающих, то все студенты успевают списать ответ на вопросы своего билета, экзамен заканчивается, все получают отметку «отлично». Время, на которое студент задерживает шпаргалку у себя при каждом списывании, определяется случайно.

Значение *N* задается пользователем при старте приложения.

Вариант 7. «Железнодорожная станция». На железнодорожной станции *N* путей (разделяемые ресурсы), на каждом из которых может остановиться только один поезд (поток). Когда поезд прибывает на станцию, то он занимает тот путь, который на данный момент свободен, останавливается там на определённое время, а затем отправляется дальше. Если все пути заняты, то поезд останавливается на подъезде к станции и ожидает освобождения одного из путей.

Если в ожидании освобождения путей находятся несколько поездов, то они занимают освобождающиеся пути в порядке приезда на станцию. Поезда прибывают на станцию через произвольные промежутки времени.

Значение *N* задается пользователем при старте приложения.

Вариант 8. «Экзамен 2». Студенты (потоки) сдают экзамен. В аудитории *N* парт и один стол экзаменатора (разделяемые ресурсы). Студенты подходят к экзаменатору в произвольном порядке и садятся сдавать экзамен.

Далее экзаменатор либо ставит студенту оценку, либо выгоняет его, либо отправляет еще раз подумать за парту (это состояние является случайной величиной с равномерным законом). Когда аудитория освобождается заходит новая группа студентов.

Значение *N* задается пользователем при старте приложения.

Вариант 9. «Разложение функции». Реализовать программу вычисления значения функции *sin(x)*. Для вычисления использовать формулу разложения функции в ряд Тейлора[1].

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1}, \forall x$$

Разделить работу по вычислению каждого слагаемого суммы между потоками: одно слагаемое – один поток. Работу по вычислению суммы на основе полученных слагаемых организовать стартовом (основном) потоке.

Если результат проверки основным потоком буфера слагаемых на пустоту дает положительный результат, то основной поток выводит промежуточное значение суммы и приостанавливается. Возобновить работу основного потока может любой исполнительный поток при завершении вычисления своего слагаемого. Работа программы завершается, когда результат работы любого из исполнительных потоков меньше ε (ε – точность вычислений, задаваемая пользователем).

Все числа в программе представляются в вещественном формате двойной точности. При вычислении слагаемого по формуле $\frac{x^n}{n!}$ следует разложить его на множители, а **не выполнять** в порядке: а) возведение в степень; б) вычисление факториала; в) деление: $\frac{x}{1} \cdot \frac{x}{2} \cdot \ldots \cdot \frac{x}{n}$, где вначале выполняется деление.

Вариант 10. «Оборона крепости». В обороняющейся крепости имеется стрелок и заряжающий (потоки), а также 2 ружья (разделяемые ресурсы). На момент начала штурма оба ружья заряжены. Когда стрелок видит врага, он берет ружье и стреляет, это занимает около 5 единиц времени, заряжающий берет это ружье и заряжает его, это занимает около 15 единиц времени. Если оба ружья разряжены, стрелок ждет, аналогично, если оба ружья заряжены, заряжающий бездействует. Описанный процесс происходит бесконечно.

Размер единицы времени задается пользователем.

Вариант 11. «Аэропорт». В аэропорту *N* посадочных полос (разделяемые ресурсы), на каждую из которых может сесть только один самолет (поток). Когда самолет оказывается на подлете к аэропорту, он выбирает и садится на ту

1 Ряд Тейлора - http://ru.wikipedia.org/wiki/Ряд_Тейлора

полосу, которая на данный момент свободна. Определенное время полоса занята, пока самолет не доставят в зону прилета. Если на подлете все полосы заняты, то самолет кружит над аэропортом и ожидает освобождения одной из полос. Если в ожидании освобождения полос находятся несколько самолетов, то они занимают освобождающиеся полосы в порядке подлета. Самолеты прибывают в аэропорт через произвольные промежутки времени.

Описанный процесс происходит бесконечно. Значение *N* задается пользователем при старте процесса. Величина времени подлета и его интенсивность являются случайными величинами с равномерным законом распределения.

Вариант 12. «Защита лабораторных работ». Имеется группа из *N* студентов, в начале ленты они договариваются между собой в каком порядке будут защищать лабораторные работы преподавателю. Не сдавший студент идет в конец очереди. Каждый студент защищает лабораторную какое-то время, а также с какой-то вероятностью (равномерный закон распределения) положительного или отрицательного результата.

Описанный процесс происходит бесконечно. Значение *N* также задается пользователем при старте процесса. В конце вывести список студентов, с пометкой «защитил»/«не защитил», сколько было попыток.

Вариант 13. Инициативный проект студента.