

Задание № 4 Разработка многопоточных приложений

Цели и задачи: Изучить работу с потоками. Научиться разбивать задачу на части, для последующего их выполнения различными потоками.

Общие сведения

Потоки предоставляют возможность проведения параллельных или псевдопараллельных (в случае одного ядра) вычислений. Они могут порождаться во время работы программы, процесса или другого потока. Основное отличие потоков от процессов заключается в том, что различные потоки имеют различные пути выполнения, но при этом пользуются общей памятью. Таким образом, несколько потоков, могут пользоваться глобальными переменными, и любое изменение данных одним потоком, будет доступно и для всех остальных. Существует несколько моделей построения многопоточных приложений. Среди них можно отметить следующие.

Итеративный параллелизм. Используется для реализации нескольких потоков (часто идентичных), каждый из которых содержит циклы. Потоки программы, описываются итеративными функциями и работают совместно над решением одной задачи.

Рекурсивный параллелизм. Используется в программах с одной или несколькими рекурсивными процедурами, вызов которых независим. Это технологии «разделяй и властвуй» или «перебор с возвратами».

Производители и потребители. Парадигма взаимодействующих неравноправных потоков. Одни потоки «производят» данные, другие их «потребляют». Часто такие потоки организуются в конвейер, через который проходит информация. Каждый поток конвейера потребляет выход своего предшественника и производит входные данные для своего последователя.

Метод дихотомии. Потоки сформированы во взаимодействующую древовидную структуру, в которой основная обработка связана с внутренними вычислениями. Взаимодействие между потоками осуществляется неинтенсивно.

Клиенты и серверы. Еще один способ взаимодействия неравноправных потоков. Клиентский поток запрашивает сервер и ждет ответа. Серверный поток ожидает запроса от клиента, затем действует в соответствии с поступившим запросом.

Управляющий и рабочие. Модель организации вычислений, при которой существует поток, координирующий работу всех остальных по-

токов. Как правило, управляющий поток распределяет данные, собирает и анализирует результаты.

Взаимодействующие равные. Модель, в которой исключен управляющий поток, не занимающийся непосредственными вычислениями. Распределение работ в таком приложении либо фиксировано заранее, либо динамически определяется во время выполнения.

Портфель задач. Один из распространенных способов динамического распределения работ. Как правило, реализуется с помощью разделяемой переменной, доступ к которой в один момент времени имеет только один поток или процесс.

Варианты заданий

1. **Задача о парикмахере.** В тихом городке есть парикмахерская. Салон парикмахерской мал, работает в нем нем может только один парикмахер, обслуживающий одного посетителя. Есть несколько стульев для ожидания в очереди. Парикмахер всю жизнь обслуживает посетителей. Когда в салоне никого нет, он спит в кресле. Когда посетитель приходит и видит спящего парикмахера, он будет его, сядется в кресло, и сидит в нем, пока парикмахер обслуживает его. Если посетитель приходит, а парикмахер занят, то он встает в очередь, садится на свободный стул и «засыпает». После стрижки парикмахер сам провожает посетителя. Если есть ожидающие посетители, то парикмахер будит одного из них, ждет пока тот сядет в кресло парикмахера и начинает стрижку. Если никого нет, он снова сядется в свое кресло и засыпает до прихода посетителя. Количество стульев для ожидания в очереди ограничено числом N . Если стульев не хватает, то пришедший посетитель уходит. Создать многопоточное приложение, моделирующее рабочий день парикмахерской. *Парикмахер и каждый из посетителей являются отдельными потоками. Допускается управляющий поток, порождающий новых посетителей.*



2. **Задача о Винни-Пухе и правильных пчелах.** В одном лесу живут N пчел и один медведь, которые используют один горшок меда, вместимостью H глотков. Сначала горшок пустой. Пока горшок не наполнится, медведь спит. Как только горшок заполняется, медведь просыпается и съедает весь мед, после чего снова засыпает. Каждая пчела многократно собирает по одному глотку меда и кладет его в горшок. Пчела, которая приносит последнюю порцию меда, будит медведя. Создать многопоточное приложение, моделирующее поведение пчел и медведя. *Медведь и каждая из пчел моделируются отдельными потоками.*



3. **Задача о Винни-Пухе и неправильных пчелах.** $N > 3$ пчел живет в улье, каждая пчела может собирать мед и сторожить улей. Пчела не покинет улей, если кроме нее в нем нет других пчел. Каждая пчела приносит за раз одну порцию меда. Всего в улей может войти **тридцать** порций меда. Винни-Пух спит пока меда в улье меньше половины, но как только его становится достаточно, он просыпается и пытается достать весь мед из улья. Если в улье находится менее чем **три** пчелы, Винни-Пух забирает мед, убегает, съедает мед и снова засыпает. Если в улье пчел больше, они кусают Винни-Пуха, он убегает, лечит укус, и снова бежит за медом. Создать многопоточное приложение, моделирующее поведение пчел и медведя. Осуществить балансировку, обеспечивающую циклическое освобождение улья от меда. *Медведь и каждая из пчел моделируются отдельными потоками.*
4. **Задача о Винни-Пухе и мстительных пчелах.** Неправильные пчелы, подсчитав в конце месяца убытки от наличия в лесу Винни-Пуха, решили разыскать его и наказать в назидание всем другим любителям сладкого. Для поисков медведя они поделили лес на участки, каждый из которых прочесывает одна стая неправильных



пчел. В случае нахождения медведя на своем участке стая проводит показательное наказание и возвращается в улей. Если участок прочесан, а Винни-Пух на нем не обнаружен, стая также возвращается в улей. Там она получает информацию об еще неисследованных участках и снова улетает. Требуется создать многопоточное приложение, моделирующее действия пчел. При решении использовать парадигму «портфель задач». Каждая стая пчел — отдельный поток.

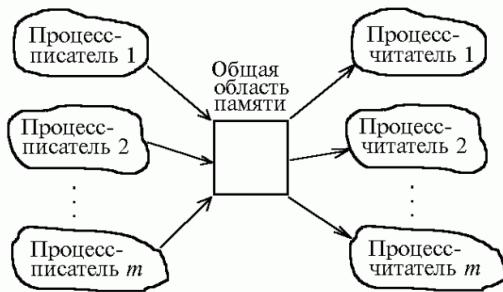


5. **Задача о читателях и писателях.** Базу данных, состоящую из целых чисел, разделяют два типа потоков: **N** читателей и **K** писателей. Читатели периодически просматривают случайные записи базы данных и выводят номер своей записи, индекс записи и ее значение. Писатели изменяют случайные записи на случайное число и также выводят информацию о своем номере, индексе записи, старом значении и новом значении. Предполагается, что в начале БД находится в непротиворечивом состоянии (все числа отсортированы). Каждая отдельная новая запись переводит БД из одного непротиворечивого состояния в другое (то есть, новая сортировка

может поменять индексы записей). Для предотвращения взаимного влияния транзакций процесс–писатель должен иметь исключительный доступ к БД. Если к БД не обращается ни один из процессов–писателей, то выполнять транзакции могут одновременно сколько угодно читателей. Создать многопоточное приложение с потоками–писателями и потоками–читателями. *Каждый читатель и писатель моделируется отдельным потоком.*

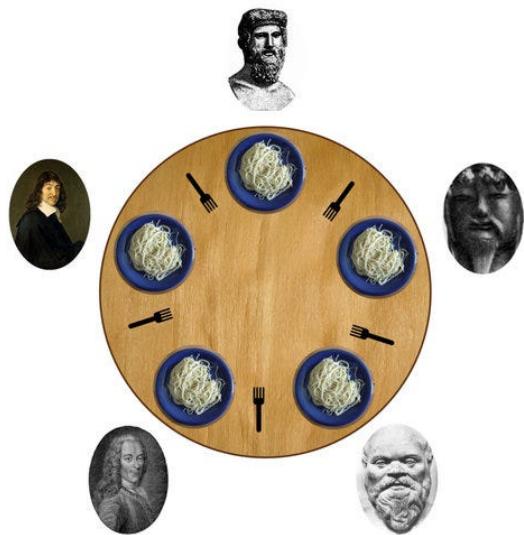


6. **Задача о читателях и писателях («грязное чтение»).** Базу данных разделяют два типа потоков — читатели и писатели. Читатели периодически просматривают случайные записи базы данных и выводя номер свой номер, индекс записи и ее значение. Писатели изменяют случайные записи на случайное число и также выводят информацию о своем номере, индексе записи, старом значении и новом значении. Предполагается, что в начале БД находится в непротиворечивом состоянии (все числа отсортированы). Каждая отдельная новая запись переводит БД из одного непротиворечивого состояния в другое (то есть, новая сортировка может поменять индексы записей). Транзакции выполняются в режиме «грязного чтения». То есть, процесс–писатель не может получить доступ к БД только в том случае, если ее уже занял другой процесс–писатель, а процессы–читатели ему не мешают обратиться к БД. Поэтому он может изменять базу данных, когда в ней находятся читатели. Создать многопоточное приложение с потоками–писателями и потоками–читателями.
7. **Задача о читателях и писателях («подтвержденное чтение»).** Базу данных разделяют два типа процессов — читатели и писатели. Читатели периодически просматривают случайные записи базы данных и выводя номер свой номер, индекс записи и ее значение. Писатели изменяют случайные записи на случайное число и также выводят информацию о своем номере, индексе записи, старом значении



и новом значении. Предполагается, что в начале БД находится в непротиворечивом состоянии (все числа отсортированы). Каждая отдельная новая запись переводит БД из одного непротиворечивого состояния в другое (то есть, новая сортировка может поменять индексы записей). Транзакции выполняются в режиме «подтвержденного чтения», то есть процесс-писатель не может получить доступ к БД в том случае, если ее занял другой процесс-писатель или процесс-читатель. К БД может обратиться одновременно сколько угодно процессов-читателей. Процесс-читатель получает доступ к БД, даже если ее уже занял процесс-писатель. Создать многопоточное приложение с потоками-писателями и потоками-читателями.

8. **Задача об обедающих философах.** Это классическая задача на взаимодействие параллельных процессов. **Пять** философов сидят возле круглого стола. Они проводят жизнь, чередуя приемы пищи и размышления. В центре стола находится большое блюдо спагетти. Спагетти длинные и запутанные, философам тяжело управляться с ними, поэтому каждый из них, что бы поесть, должен пользоваться двумя вилками. К несчастью, философам дали только **пять** вилок. Между каждой парой философов лежит одна вилка. Поэтому эти высококультурные и предельно вежливые люди договорились, что каждый будет пользоваться только теми вилками, которые лежат рядом с ним (слева и справа). Написать многопоточную программу, моделирующую поведение философов с помощью семафоров. Программа должна избегать фатальной ситуации, в которой все философы голодны, но ни один из них не может взять обе вилки (например, каждый из философов держит по одной вилке и не хочет отдавать ее). Время, которое отводится на прием пищи и размышление задается случайно в некотором разумном для наблюдения из вне диапазоне. **Решение должно быть симметричным, то есть все потоки-философы должны выполнять один и тот же код (являться равноправными потоками).**



9. **Задача о каннибалах.** Племя из N дикарей ест вместе из большого горшка, который вмещает M кусков тушеного миссионера. Когда дикарь хочет обедать, он ест из горшка один кусок, если только горшок не пуст, иначе дикарь будит повара и ждет, пока тот не наполнит горшок. Повар, сварив обед, засыпает. Создать многопоточное приложение, моделирующее обед дикарей. **Повар и каждый из дикарей задаются отдельными потоками.**



10. **Задача о курильщиках.** Есть три потока—курильщика и один поток—посредник. Курильщик непрерывно скручивает сигареты и курит их. Чтобы скрутить сигарету, нужны табак, бумага и спички. У одного курильщика есть табак, у второго — бумага, а у третьего — спички. Посредник через **некоторое случайное время** кладет на стол по два разных случайных компонента. Тот Курильщик, у которого есть третий компонент, забирает компоненты со стола, скручивает сигарету и курит **некоторое случайное время**. По-

средник через отведенный ему интервал времени, который может быть больше или меньше времени курения, выкладывает следующий набор. Если курильщик, которому нужен этот набор, свободен, то он начинает курить. Если курильщик еще курит, то процесс передачи дожидается окончания курения. Затем процесс повторяется. В принципе возможна ситуация, когда все три курильщика могут курить одновременно или все могут ждать, когда посредник соизволит выложить очередной набор. **Создать многопоточное приложение, моделирующее поведение курильщиков и посредника.**



11. **Военная задача.** Анчуария и Тарантерия — два крохотных латиноамериканских государства, затерянных в южных Андах. Диктатор Анчуарии, дон Федерико, объявил войну диктатуре Тарантерии, дону Эрнандо. У обоих диктаторов очень мало солдат, но очень много снарядов для минометов, привезенных с последней американской гуманитарной помощью. Поэтому армии обеих сторон просто обстреливают наугад территорию противника, надеясь поразить что-нибудь ценное. Стрельба ведется по очереди до тех пор, пока либо не будут уничтожены все **K** цели (каждая стоимостью C_i), либо стоимость **N** потраченных снарядов не превысит суммарную стоимость всего того, что ими можно уничтожить. Создать многопоточное приложение, моделирующее военные действия. *Обратить особое внимание на отображение состояния государств в процессе моделирования.* Размер каждого государства $X \times Y$ клеток.
12. **Задача о супермаркете.** В супермаркете работают **два** кассира, покупатели заходят в супермаркет, делают покупки и становятся в



очередь к случайному кассиру. Пока очередь пуста, кассир «спит», как только появляется покупатель, кассир «просыпается». Покупатель ожидает в очереди, пока не подойдет к кассиру. Очереди имеют ограниченную длину N . Если она длиннее, то покупатель не встает ни в одну из очередей и уходит. Если одна из очередей заполнена, то покупатель встает в другую. Создать многопоточное приложение, моделирующее рабочий день супермаркета. **Каждый покупатель и кассиры задаются отдельными потоками.**



13. **Первая задача о магазине.** В магазине работают **три** отдела, каждый отдел обслуживает **один** продавец. Покупатель, зайдя в магазин, делает покупки в одном или нескольких произвольных отделах, обходя их в **произвольном (случайном)** порядке. Если в выбранном отделе продавец не свободен, покупатель становится в очередь и ожидает, пока продавец не освободится. Создать многопоточное приложение, моделирующее рабочий день магазина. **Каждого покупателя и продавцов моделировать отдельными потоками.** Размер очереди не оговаривается. Считается, что для данной задачи она не ограничена (но моделирование должно быть в разумных пределах).

14. **Вторая Задача о магазине (забывчивые покупатели).** В ма-



газине работают **два** отдела, каждый отдел обладает уникальным ассортиментом. В каждом отделе работает один продавец. В магазин ходят исключительно забывчивые покупатели, поэтому каждый покупатель носит с собой список из **K** товаров, которые желает купить. Покупатель приобретает товары точно в том порядке, в каком они записаны в его списке. При этом товары в списке расположены в **случайном** порядке, что заставляет покупателя многократно переходить от отдела к отделу, если это требуется для совершения покупок. Продавец может обслужить только одного покупателя за раз. Покупатель, вставший в очередь, засыпает пока не дойдет до продавца. Продавец засыпает, если в его отделе нет покупателей, и просыпается, если появится хотя бы один. Создать многопоточное приложение, моделирующее работу магазина в течение рабочего дня. **Каждый покупатель и продавец задаются отдельным потоком.**

15. **Задача о больнице.** В больнице **два** дежурных врача принимают пациентов, выслушивают их жалобы и отправляют или к стоматологу, или к хирургу, или к терапевту. Стоматолог, хирург и терапевт лечат пациентов. Каждый врач может принять только одного пациента за раз. Пациенты стоят в очереди к врачам и никогда их не покидают. Создать многопоточное приложение, моделирующее рабочий день клиники. **Каждого из врачей и пациента моделировать отдельным потоком.**
16. **Задача о социалистической гостинице.** В гостинице **30** одноместных номеров. Клиенты гостиницы снимают номер на одни или несколько суток (задается при создании клиента). Если в гостинице нет свободных номеров, клиенты не уходят, а устраиваются на рядом с гостиницей на скамейках и ждут в порядке очереди, по-



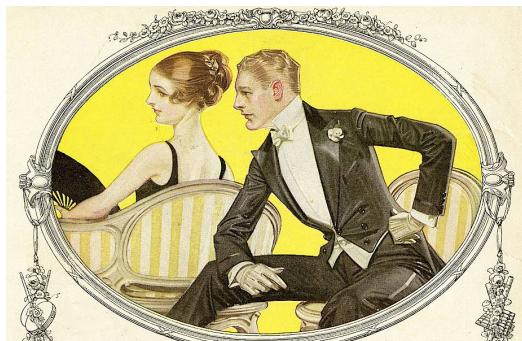
ка любой номеров не освободится (других гостиниц в городе нет). Создать многопоточное приложение, моделирующее работу гостиницы. **Каждого клиента и гостиницу (точнее ее администратора) моделировать отдельным потоком.**



17. **Задача о капиталистической гостинице.** В гостинице 10 номеров с ценой 2000 УЕ, 10 номеров с ценой 4000 УЕ и 5 номеров с ценой 6000 УЕ. Клиент, зашедший в гостиницу, обладает некоторой (случайной) суммой и получает номер по своим финансовым возможностям, если тот свободен. При наличии денег и разных по стоимости номеров он выбирает **случайный** номер. Если доступных по цене свободных номеров нет, клиент уходит искать ночлег в другое место. Клиенты порождаются динамически и уничтожаются при освобождении номера или уходе из гостиницы при невозможности оплаты. Создать многопоточное приложение, моделирующее работу гостиницы. **Каждого клиента и гостиницу (точнее ее администратора) моделировать отдельным потоком.**
18. **Задача о гендерной гостинице (леди и джентльмены).** В гостинице 10 номеров рассчитаны на одного человека и 15 номеров рассчитаны на двух человек. В гостиницу случайно приходят клиенты леди и клиенты джентльмены, и конечно они могут провести



ночь в номере только с представителем своего пола. Если для клиента не находится подходящего номера, он уходит искать ночлег в другое место. Клиенты порождаются динамически и уничтожаются при освобождении номера или уходе из гостиницы при невозможности поселиться. Создать многопоточное приложение, моделирующее работу гостиницы. **Каждого клиента и гостиницу (точнее ее администратора) моделировать отдельным потоком.**



19. **Задача об умной клумбе.** На клумбе растет **40** цветов, за ними непрерывно следят два садовника и поливают увядшие цветы, при этом оба садовника очень боятся полить один и тот же цветок, который еще не начал вянуть. Создать многопоточное приложение, моделирующее состояния цветков на клумбе и действия садовников. **Клумба — отдельный поток который следит за состоянием всех цветков и информирует, который из них полит, увяда-ет (нужно полить), засох или перелил и сгнил (бесполезно поливать).** Каждый садовник задается отдельным потоком.
20. **Задача об умных цветах.** На клумбе растет **40** цветов, за ними непрерывно следят два садовника и поливают увядшие цветы, при



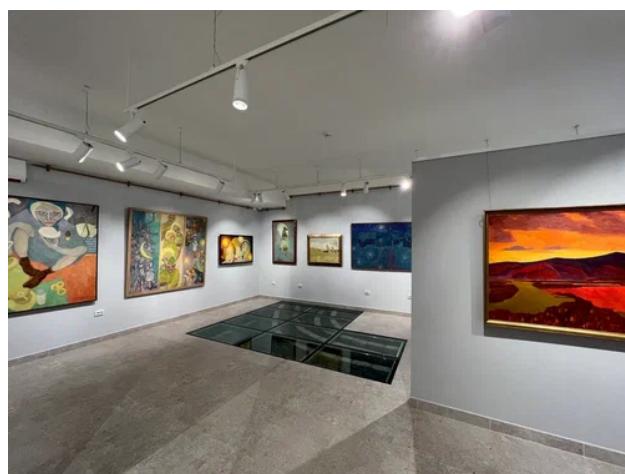
этом оба садовника очень боятся полить один и тот же цветок, который еще не начал вянуть. Создать многопоточное приложение, моделирующее состояния цветков на клумбе и действия садовников. **Каждый цветок — отдельный поток который следит за своим состоянием и информирует, что он полит, увядает (нужно полить), засох или перелил и сгнил (бесполезно поливать)**. Каждый садовник задается отдельным потоком.

21. **Задача о нелюдимых садовниках.** Имеется пустой участок земли (двумерный массив размером $M \times N$) и план сада, разбитого на отдельные квадраты. От 10 до 30 процентов (задается случайно) площади сада заняты прудами или камнями. То есть недоступны для ухаживания. Эти квадраты располагаются на плане произвольным (случайным) образом. Ухаживание за садом выполняют два садовника, которые не хотят встречаться друг другом (то есть, одновременно появляться в одном и том же квадрате). Первый садовник начинает работу с верхнего левого угла сада и перемещается слева направо, сделав ряд, он спускается вниз и идет в обратном направлении, пропуская обработанные участки. Второй садовник начинает работу с нижнего правого угла сада и перемещается снизу вверх, сделав ряд, он перемещается влево и также идет в обратную сторону. Если садовник видит, что участок сада уже обработан другим садовником или является необрабатываемым, он идет дальше. Если по пути какой-то участок занят другим садовником, то садовник ожидает когда участок освободится, чтобы пройти дальше на доступный ему необрабатываемый участок. Садовники должны работать одновременно со скоростями, **определенными как параметры задачи**. Прохождение через любой квадрат занимает некоторое время, которое задается **константой**, меньшей чем времена обработки и принимается за единицу времени. Создать многопоточное приложение, моделирующее работу садовников. **Каждый**

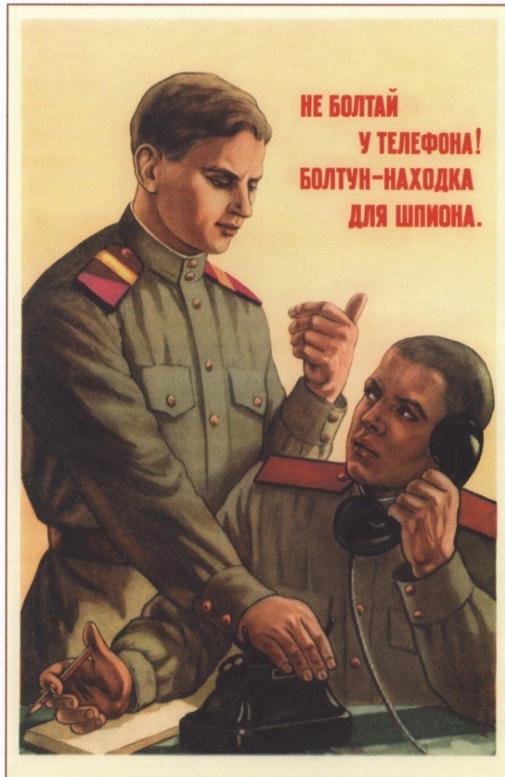
садовник — это отдельный поток.



22. **Задача о картинной галерее.** Вахтер следит за тем, чтобы в картинной галерее одновременно было не более **50** посетителей. Для обозрения представлены **5** картин. Каждый посетитель случайно переходит от картины к картине, но если на желаемую картину любуются более десяти посетителей, он стоит в стороне и ждет, пока число желающих увидеть эту картину не станет меньше. Посетитель покидает галерею по завершении осмотра всех картин. Каждый посетитель уникальный (имеет свой номер). В галерее также пытаются постоянно зайти новые посетители, которые ожидают своей очереди и разрешения от вахтера, если та заполнена. Создать многопоточное приложение, моделирующее однодневную работу картинной галереи (например, можно ограничить числом посетителей от 100 до 300). *Вахтер и посетители — отдельные потоки.*



23. Задача о болтунах. **Н** болтунов имеют телефоны. Они либо некоторое (**случайное**) время ждут звонков, либо звонят друг другу, чтобы побеседовать. Если телефон **случайного** абонента занят, болтун будет звонить другому **случайному** абоненту, пока ему кто-нибудь не ответит. Побеседовав **некоторое** время, болтун или ждет звонка, или звонит на другой случайный номер. Создать многопоточное приложение, моделирующее поведение болтунов. Каждый болтун моделируется отдельным потоком.



24. Задача о программистах. В отделе работают три программиста. Каждый программист пишет свою программу и отдает ее на проверку одному из двух оставшихся программистов, выбирая его случайно. Программист проверяет чужую программу, когда его собственная уже написана и передана на проверку. По завершении проверки, программист возвращает программу с результатом (формируемым случайно по любому из выбранных Вами законов): программа написана правильно или неправильно. Программист «спит», если отправил свою программу и не проверяет чужую программу. Программист «просыпается», когда получает заключение от другого

го программиста. Если программа признана правильной, программист пишет другую программу, если программа признана неправильной, программист исправляет ее и отправляет на проверку тому же программисту, который ее проверял. К исправлению своей программы он приступает, завершив проверку чужой программы. При наличии в очереди проверяемых программ и приходе заключения о неправильной своей программы программист может выбирать любую из возможных работ. Необходимо учесть, что кто-то из программистов может получать несколько программ на проверку. Создать многопоточное приложение, моделирующее работу программистов. **Каждый программист — это отдельный поток.**



25. **Задача об инвентаризации по рядам.** После нового года в библиотеке университета обнаружилась пропажа каталога. После поиска и наказания виноватых, ректор дал указание восстановить каталог силами студентов. Фонд библиотека представляет собой прямоугольное помещение, в котором находится **M** рядов по **N** шкафов по **K** книг в каждом шкафу. Требуется создать многопоточное приложение, составляющее каталог. При решении задачи использовать метод «портфель задач». В качестве отдельного потока задается составление каталога одним студентом для одного ряда. **Следует помнить, что в данном случае при занесении данных в уже заполняемый каталог придется производить сортировку своих данных с уже внесенными.**

Примечание. Каталог — это список книг, упорядоченный (отсортированный) по названию книги (в данном случае в качестве названия можно взять и целое число — не обязательно использовать ASCII строку символов). Каждая строка каталога содержит идентифицирующее значение (номер или строку), местоположение книги, включающее номер ряда, номер шкафа, номер книги в шкафу. Перед запуском потоков по составлению каталогов необходимо случайным образом расположить книги в шкафах, используя генератор случайных чисел. Так же нужно преварительно вывести список замещения книг по шкафам, чтобы в конце сопоставить его с тем, как они оказались описанными в каталоге.

26. **Задача об инвентаризации по книгам.** После нового года в библиотеке университета обнаружилась пропажа каталога. После поиска и наказания, виноватых ректор дал указание восстановить каталог силами студентов. Фонд библиотека представляет собой прямоугольное помещение, в котором находится M рядов по N шкафов по K книг в каждом шкафу. Требуется создать многопоточное приложение, составляющее каталог. При решении использовать метод «портфель задач», причем в качестве отдельного потока задается внесение в каталог записи об отдельной книге.



Примечание. Каталог — это список книг, упорядоченный (отсортированный) по названию книги (в данном случае в качестве названия можно взять и целое число — не обязательно использовать ASCII строку символов). Каждая строка каталога содержит идентифицирующее значение (номер или строку), местоположение книги, включающее номер ряда, номер шкафа, номер книги в шкафу. Перед запуском потоков по составлению каталогов необходимо случайным образом расположить книги в шкафах, используя генератор случайных чисел. Так же нужно преварительно вывести список замещения книг по шкафам, чтобы в конце сопоставить его с тем, как они оказались описанными в каталоге.

вывести список замещения книг по шкафам, чтобы в конце сопоставить его с тем, как они оказались описанными в каталоге.

27. **Задача о привлекательной студентке.** У одной очень привлекательной студентки есть N поклонников. Традиционно в день св. Валентина очень привлекательная студентка проводит романтический вечер с одним из поклонников. Счастливый избранник заранее не известен. С утра очень привлекательная студентка получает N «валентинок» с различными вариантами романтического вечера. Выбрав наиболее заманчивое предложение, студентка извещает счастливчика о своем согласии, а остальных — об отказе. Требуется создать многопоточное приложение, моделирующее поведение студентки. При решении использовать парадигму «клиент–сервер» с активным ожиданием (предполагается, что поклонник не сразу получает ответ, а только после того, как все они сделают запросы, после чего студентка выберет лучшего и разошлет все соответствующие уведомления).



28. **Задача о производстве булавок.** В цехе по заточке булавок все необходимые операции осуществляются рабочими на трех производственных участках. На первом участке $1 \leq K \leq 3$ рабочих получают тупые булавки и в течение некоторого случайного времени каждый из них проверяет ее на предмет кривизны. Если булавка не кривая, то рабочий передает ее на второй участок, на котором $1 \leq L \leq 5$ работников осуществляют заточку. Случайно выбранный из них свободный работник осуществляет заточку и передает заточенную булавку на третий участок, на котором $1 \leq M \leq 2$ осуществляют контроль качества операции. Требуется создать многопоточное приложение, моделирующее работу цеха. При решении использовать парадигму «производитель–потребитель». Каждый работник — это отдельный поток. Следует учесть, что каждая

из операций выполняется за случайное время которое не связано с конкретным рабочим. Возможны различные способы передачи (на выбор). Либо непосредственно по одной булавке, либо через ящики, в которых осуществляется буферизация некоторого конечное количества булавок. Метод передачи булавок между участками описать.



29. **Задача про экзамен.** Преподаватель проводит экзамен у группы студентов. Каждый студент получает свой билет, сообщает его номер и готовит письменный ответ за некоторое случайное время. Подготовив ответ, он передает его преподавателю. Преподаватель некоторое случайное время просматривает ответ и сообщает студенту оценку. Требуется создать многопоточное приложение, моделирующее действия преподавателя и студентов. При решении использовать парадигму «клиент–сервер».



30. **Военная операция.** Темной-темной ночью прапорщики Иванов, Петров и Нечепорук занимаются хищением военного имущества со склада родной военной части. Будучи умными людьми и отличниками боевой и строевой подготовки, прапорщики ввели разделение труда. Иванов выносит имущество со склада и передает его в руки Петрову, который грузит его в грузовик. Нечепорук стоит на шухере и заодно подсчитывает рыночную стоимость добычи поле погрузки в грузовик очередной партии похищенного. Требуется составить многопоточное приложение, моделирующее деятельность прапорщиков-потоков. **Необходимо учесть случайное время выполнения каждым прапорщиком своей боевой задачи и организовать в программе корректную их синхронизацию.**



31. **Задача о Пути Кулака.** На седых склонах Гималаев стоит древний буддистский монастырь: Гуань-Инь-Янь. Каждый год в день сошествия на землю боддисатвы монахи монастыря собираются на совместное празднество и показывают свое совершенствование на Пути Кулака. Всех соревнующихся монахов первоначально разбивают на пары. Бои продолжаются до выявления победителя. Монах который победил в финальном бою, забирает себе на хранение статую боддисатвы. Реализовать многопоточное приложение, определяющее победителя. В качестве входных данных используется массив, в котором хранится количество энергии Ци каждого монаха. При победе монах забирает энергию Ци своего противника. Новые пары образуются среди победителей других пар в порядке завершения поединков. То есть, возможна ситуация, когда бойцы, участвующие в поединке могут быстро победить и начать биться с

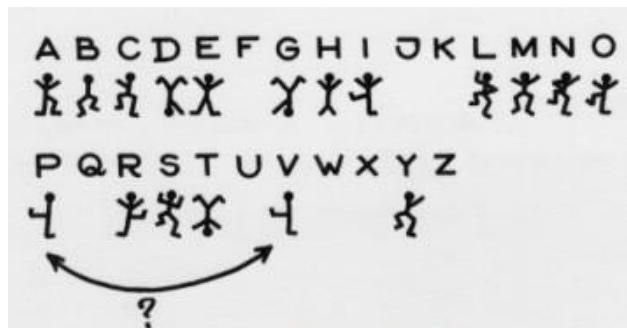
другими, в то время как поединки начавшиеся ранее, могут продолжаться. Причем длительное время. Каждый поединок протекает некоторое случайное время, которое пропорционально отношению энергии Ци победенного к энергии Ци победителя, умноженному на поправочный коэффициент, позволяющий отслеживать протекание поединка на экране дисплея (например, путем умножения этого отношения на 1000 миллисекунд или другое более удобное значение).



32. **Задача об Острове Сокровищ.** Шайка пиратов под предводительством Джона Сильвера высадилась на берег Острова Сокровищ. Не смотря на добытую карту острова старого Флинта, местоположение сокровищ по-прежнему остается загадкой, поэтому искать клад приходится практически на ощупь. Так как Сильвер ходит на деревянной ноге, то самому бродить по джунглям ему не с руки. Джон Сильвер поделил остров на участки, а пиратов на небольшие группы. Каждой группе поручается искать клад на одном из участков, а сам Сильвер ждет на берегу. Пираты, обшарив свою часть острова, возвращаются к Сильверу и докладывают о результатах. **Требуется создать многопоточное приложение с управляемым потоком**, моделирующее действия Сильвера и пиратов. Примечание. Количество участков превышает число поисковых групп. *Потоки должны задавать каждую группу и Сильвера (управляющий поток)*.
33. **Пляшущие человечки.** На тайном собрании глав преступного мира города Лондона председатель собрания профессор Мориарти постановил: отныне вся переписка между преступниками должна



вестись тайнописью. В качестве стандарта были выбраны «пляшущие человечки», шифр, в котором каждой букве латинского алфавита соответствует хитроумный значок. Реализовать многопоточное приложение, шифрующее исходный текст (в качестве ключа используется кодовая таблица, устанавливающая однозначное соответствие между каждой буквой и каким-нибудь числом). Каждый поток-шифровальщик кодирует свои кусочки общего текста. При решении использовать парадигму портфеля задач. Потоки работают асинхронно, формируя свой закодированный фрагмент в случайное время. Следовательно, при занесении в портфель необходимо проводить упорядочение фрагментов. В программе необходимо вывести исходный текст, закодированные фрагменты по мере их формирования, окончательный закодированный текст.



34. **Задача о сельской библиотеке.** В библиотеке имеется N книг, Каждая из книг в одном экземпляре. M читателей регулярно заглядывают в библиотеку, выбирая для чтения от одной до трех книг и читая их некоторое количество дней. Если желаемой книги нет, то читатель, взяв существующие, дожидается от библиотекаря

информации об ее появлении и приходит в библиотеку, чтобы специально забрать ее. Возможна ситуация, когда несколько читателей конкурируют из-за этой популярной книги. Создать многопоточное приложение, моделирующее заданный процесс.



35. **Список чисел.** Вывести список всех натуральных чисел, содержащих от 4 до 9 значащих цифр, которые после умножения на N , будут содержать все те же самые цифры в произвольной последовательности и в произвольном количестве. Входные данные: целое положительное число N , от 2 до 9. Количество используемых потоков K также является входным параметром.

