

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ ВЛАДИМИРА ДАЛЯ»

Факультет компьютерных систем и информационных технологий
Кафедра информатики и программной инженерии

УТВЕРЖДАЮ

Декан факультета компьютерных
систем и информационных технологий

Кочевский А. А.

« 19 » 04 2023 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по учебной дисциплине

«Параллельные и распределенные вычисления»

09.03.03 Прикладная информатика

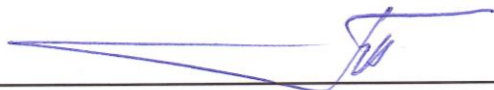
«Прикладная информатика в экономике»

Разработчик:

доцент  Письменский А.В.

ФОС рассмотрен и одобрен на заседании кафедры информатики и
программной инженерии
от 18 апреля 2023 г., протокол № 17

Заведующий кафедрой

 Кочевский А.А.

Луганск 2023 г.

**Паспорт
фонда оценочных средств по учебной дисциплине
«Параллельные и распределенные вычисления»**

**Перечень компетенций (элементов компетенций),
формируемых в результате освоения учебной дисциплины**

№ п/п	Код контроли руемой компетен ции	Формулировка контролируемой компетенции	Контролируемые темы учебной дисциплины	Этапы формирования (семестр изучения)
1	ПК-2.1	Способен использовать современные языки и системы программирования, технологии проектирования программного обеспечения	Тема 1. Тема 2. Тема 3. Тема 4. Тема 5.	начальный (5)
2	ПК-2.2	Способен сформулировать требования к разрабатываемому программному обеспечению, выполнить его реализацию и оформить техническую документацию на его компоненты	Тема 6. Тема 7. Тема 8. Тема 9. Тема 10.	начальный (5)
3	ПК-2.3	Способен осуществлять проектирование программного обеспечения конкретной ИС и разработку технической документации на ее компоненты документов; приемами оценки и согласования сроков выполнения поставленных задач	Тема 11. Тема 12. Тема 13. Тема 14. Тема 15. Тема 16.	начальный (5)

**Показатели и критерии оценивания компетенций,
описание шкал оценивания**

№ п/п	Код контролируемо й компетенции	Показатель оценивания (знания, умения, навыки)	Контролируемые темы учебной дисциплины	Наименование оценочного средства
1.	ПК-2.1	знать: - основные причины для использования параллелизма в приложении; - особенности использования платформенно-зависимых средств; - организация параллелизма за счет нескольких процессов; - организация параллелизма за счет нескольких потоков; - основные причины для использования параллелизма в приложении; - использование библиотеки многопоточности для C++; - особенности использования платформенно-зависимых средств; - базовые функции и операции управления потоками; уметь: - проектировать параллельные и распределенные системы;	Тема 1. Введение в Параллельные и распределенные вычисления. Тема 2. Зачем нужен параллелизм Тема 3. Параллелизм и многопоточность в C++. Тема 4. Управление потоками. Тема 5. Управление данными в потоке.	Вопросы для обсуждения (в виде докладов и сообщений), лабораторные работы, контрольные работы,
2.	ПК-2.2	знать: - методы управления данными в потоке; - способы передачи владения потока; - методы идентификация потоков; - проблемы разделения данных между потоками; - способы защиты разделяемых данных с помощью мьютексов; уметь: - проектировать параллельные и распределенные системы; - использовать библиотеку C++ Thread Library для реализации многопоточности; - выполнять защиту разделяемых данных с помощью мьютексов; - выполнять одновременный доступ к ресурсам.	Тема 6. Владение потоком. Тема 7. Разделение данных между потоками Тема 8. Защита разделяемых данных с помощью мьютексов. Тема 9. Взаимоблокировка: проблема и решение. Тема 10. Другие средства защиты разделяемых данных.	Вопросы для обсуждения (в виде докладов и сообщений), лабораторные работы, контрольные работы,
3.	ПК-2.3	Знать: методы организации потокобезопасной очереди на базе условных переменных,	Тема 11. Синхронизация параллельных	Вопросы для обсуждения (в виде докладов и

		методы доступа к результатам асинхронных операций Уметь: проектировать параллельные и распределенные системы. Знать: основные концепции библиотек работы со временем. Уметь: выполнять защиту разделяемых данных с помощью условных переменных. Знать: атомарные типы и операции; методы синхронизация операций и принудительного упорядочения; методы упорядочения, захвата-освобождения. Уметь: выполнять одновременный доступ к ресурсам; выполнять защиту разделяемых данных. Знать: атомарные типы и операции; методы синхронизация операций и принудительного упорядочения; методы упорядочения. Уметь: выполнять защиту разделяемых данных с помощью условных переменных; выполнять одновременный доступ к ресурсам; выполнять защиту разделяемых данных. Знать: методы организации потокобезопасной очереди на базе условных переменных, методы доступа к результатам асинхронных операций; методы захвата-освобождения Уметь: проектировать параллельные и распределенные системы; выполнять одновременный доступ к ресурсам; выполнять защиту разделяемых данных	операций Тема 12. Ожидание одноразовых событий с помощью механизма будущих результатов Тема 13. Ожидание с ограничением по времени Тема 14. Применение синхронизации операций для упрощения кода Тема 15. Модель памяти C++ и атомарные операции Тема 16. Синхронизация операций и принудительное упорядочение	сообщений), лабораторные работы, контрольные работы,
--	--	--	--	---

**Фонды оценочных средств по дисциплине
«Параллельные и распределенные вычисления»**

Вопросы для обсуждения (в виде докладов и сообщений):

1. Ссылки на г-значения и шаблоны функций;
2. Идентификация потоков;

3. Устранение проблематичных состояний гонки;
4. Структурирование кода для защиты разделяемых данных;
5. Гибкая блокировка с помощью `std::unique_lock`;
6. Защита редко обновляемых структур данных.
7. Синхронизация параллельных операций с помощью условных переменных;
8. Организация исключительного доступа к данным с помощью объекта `future`.
9. Организация потокобезопасной очереди на базе условных переменных;
10. Реализация многопоточного алгоритма быстрой сортировки;
11. Устранение проблематичных состояний гонки;
12. Структурирование кода для защиты разделяемых данных;
13. Гибкая блокировка с помощью `std::unique_lock`;
14. Защита редко обновляемых структур данных;
15. Реализация доступа к данным с помощью атомарных объектов;
16. Упорядочивание данных с помощью атомарных объектов `std::atomic<T>` и `std::atomic_flag`;
17. Упорядочение захват-освобождение и семантика `memory_order_consume`.
18. Ожидание события или иного условия
19. Ожидание условия с помощью условных переменных
20. Потокобезопасная очередь на базе условных переменных
21. Возврат значения из фоновой задачи
22. Ассоциирование задачи с будущим результатом
23. Использование `std::promise`
24. Сохранение исключения в будущем результате
25. Временные интервалы
26. Функции, принимающие таймаут
27. Функциональное программирование с применением будущих результатов
28. Быстрая сортировка в духе ФП
29. Синхронизация операций с помощью передачи сообщений
30. Объекты и ячейки памяти
31. Порядок модификации
32. Операции над `std::atomic_flag`
33. Последовательно согласованное упорядочение
34. Ослабленное упорядочение
35. Механизм ослабленного упорядочения
36. Упорядочение захват-освобождение
37. Транзитивная синхронизация с помощью упорядочения захват-освобождение
38. Операции над `std::atomic<bool>`
39. Операции над `std::atomic<T*>`: арифметика указателей

40. Операции над стандартными атомарными целочисленными типами
41. Основной шаблон класса `std::atomic<>`
42. Свободные функции для атомарных операций

Критерии и шкала оценивания по оценочному средству доклад, сообщение

Шкала оценивания (интервал баллов)	Критерий оценивания
5	Доклад (сообщение) представлен(о) на высоком уровне (студент в полном объеме осветил рассматриваемую проблематику, привел аргументы в пользу своих суждений, владеет профильным понятийным (категориальным) аппаратом и т.п.)
4	Доклад (сообщение) представлен(о) на среднем уровне (студент в целом осветил рассматриваемую проблематику, привел аргументы в пользу своих суждений, допустив некоторые неточности и т.п.)
3	Доклад (сообщение) представлен(о) на низком уровне (студент допустил существенные неточности, изложил материал с ошибками, не владеет в достаточной степени профильным категориальным аппаратом и т.п.)
2	Доклад (сообщение) представлен(о) на неудовлетворительном уровне или не представлен (студент не готов, не выполнил задание и т.п.)

Лабораторные работы

Лабораторная работа №1

Тема: Введение в параллельное программирование

Цель: закрепить полученные знания по параллелизму и многопоточности, научиться использовать их в своих приложениях.

Лабораторная работа №2

Тема: Управление потоками

Цель: изучить запуск потоков и различные способы задания кода, исполняемого в новом потоке, научиться присваивать уникальный идентификатор потока и управлять им.

Лабораторная работа №3

Тема: Управление данными в потоке

Цель: научиться передавать аргументы между потоками.

Лабораторная работа №4

Тема: Ссылки на *r*-значения

Цель: научиться передавать аргументы между потоками.

Лабораторная работа №5

Тема: Передача владения потоком

Цель: научиться передавать владение потоками между функциями, назначать количество потоков и время их выполнения, получать их идентификаторы.

Лабораторная работа №6

Тема: Задание количества потоков во время выполнения

Цель: научиться передавать владение потоками между функциями, назначать количество потоков и время их выполнения, получать их идентификаторы.

Лабораторная работа №7

Тема: Идентификация потоков

Цель: научиться передавать владение потоками между функциями, назначать количество потоков и время их выполнения, получать их идентификаторы.

Лабораторная работа №8

Тема: Защита разделяемых данных с помощью мьютексов

Цель: научиться защищать структуры данных от гонок с помощью мьютексов.

Лабораторная работа №9

Тема: Одновременный доступ к ресурсам

Цель: научиться защищать структуры данных от гонок с помощью мьютексов.

Лабораторная работа №10

Тема: Синхронизация действий, выполняемых в разных потоках

Цель работы: освоить механизмы синхронизации (условные переменные и будущие результаты (future))

Отчет должен включать в себя следующие разделы:

- формулировку задания;
- описание основных методов, используемых в лабораторной работе;
- результаты работы программы (в виде файла или в виде скриншота);
- анализ результатов.

Лабораторная работа №11

Тема: Моделирование одноразовых событий с помощью будущего результата

Цель работы: освоить механизмы шаблонного класса `std::future`, обеспечивающего механизм доступа к результатам асинхронных операций и `std::shared_future<>`

Отчет должен включать в себя следующие разделы:

- формулировку задания;
- описание основных методов, используемых в лабораторной работе;
- результаты работы программы (в виде файла или в виде скриншота);
- анализ результатов.

Лабораторная работа №12

Тема: Основные концепции библиотеки Chrono (C++)

Цель работы: изучение методов библиотеки chrono (namespace std::chrono)

Отчет должен включать в себя следующие разделы:

- формулировку задания;
- описание основных методов, используемых в лабораторной работе;
- результаты работы программы (в виде файла или в виде скриншота);
- анализ результатов.

Лабораторная работа №13

Тема: Алгоритм быстрой сортировки Quicksort

Цель работы: выполнить параллельную реализацию алгоритма Quicksort

Отчет должен включать в себя следующие разделы:

- формулировку задания;
- описание основных методов, используемых в лабораторной работе;
- результаты работы программы (в виде файла или в виде скриншота);
- анализ результатов.

Лабораторная работа №14

Тема: Атомарные типы и операции

Цель работы: освоить механизмы использования атомарных объектов и операций

Отчет должен включать в себя следующие разделы:

- формулировку задания;
- описание основных методов, используемых в лабораторной работе;
- результаты работы программы (в виде файла или в виде скриншота);
- анализ результатов.

Лабораторная работа №15

Тема: Синхронизация операций и принудительное упорядочение modification order

Цель работы: освоить механизмы упорядочивания атомарных объектов типов std::atomic<T> и std::atomic_flag.

Отчет должен включать в себя следующие разделы:

- формулировку задания;
- описание основных методов, используемых в лабораторной работе;
- результаты работы программы (в виде файла или в виде скриншота);
- анализ результатов.

Лабораторная работа №16

Тема: Зависимости по данным, упорядочение захват-освобождение и семантика `memory_order_consume`

Цель работы: освоить механизмы упорядочения захват-освобождения и семантика `memory_order_consume`

Отчет должен включать в себя следующие разделы:

- формулировку задания;
- описание основных методов, используемых в лабораторной работе;
- результаты работы программы (в виде файла или в виде скриншота);
- анализ результатов.

Критерии и шкала оценивания по оценочному средству лабораторная работа

Шкала оценивания (интервал баллов)	Критерий оценивания
5	Лабораторная работа выполнена на высоком уровне (правильность выполнения 90-100%)
4	Лабораторная работа выполнена на среднем уровне (правильность выполнения 75-89%)
3	Лабораторная работа выполнена на низком уровне (правильность выполнения 50-74%)
2	Лабораторная работа выполнена на неудовлетворительном уровне (правильность выполнения менее чем на 50%)

Вопросы к контрольным работам:

1. Подходы к организации параллелизма.
2. Параллельный код: состояние гонки (race condition)..
3. Использование параллелизма в приложении.
4. Устранение проблематичных состояний гонки.
5. История многопоточности в C++.
6. Защита разделяемых данных с помощью мьютексов.
7. Поддержка параллелизма в новом стандарте C++.
8. Использование мьютексов в C++.
9. Эффективность библиотеки многопоточности для C++.
10. Структурирование кода для защиты разделяемых данных.
11. Платформенно-зависимые средства.
12. Интерфейс адаптера контейнера `std::stack`.
13. Запуск потока в C++.

Критерии и шкала оценивания по оценочному средству контрольная работа

Шкала оценивания (интервал баллов)	Критерий оценивания
5	Контрольная работа выполнена на высоком уровне (правильные ответы даны на 90-100% вопросов/задач)
4	Контрольная работа выполнена на среднем уровне (правильные ответы даны на 75-89% вопросов/задач)
3	Контрольная работа выполнена на низком уровне (правильные ответы даны на 50-74% вопросов/задач)
2	Контрольная работа выполнена на неудовлетворительном уровне (правильные ответы даны менее чем на 50%)

Курсовые работы

Темы курсовых работ:

- Гибкая блокировка с помощью `std::unique_lock`;
- Защита редко обновляемых структур данных;
- Операции над стандартными атомарными целочисленными типами.
- Организация исключительного доступа к данным с помощью объекта `future`.
- Организация порядка данных и `std::memory_order_seq_cst`.
- Организация потокобезопасной очереди на базе условных переменных;
- Основные концепции библиотеки `Chrono`.
- Передача аргументов функции, заданной в `std::async`.
- Передача владения мьютексом между контекстами.
- Передача владения потоком.
- Платформенно-зависимые средства.
- Полное определение класса потокобезопасной очереди на базе условных переменных.
- Порядок модификации.
- Потокобезопасная очередь на базе условных переменных.
- Применение `std::lock()` и `std::lock_guard` для реализации операции обмена.
- Применение синхронизации операций для упрощения кода.
- Использование `std::async` вместо `std::thread`
- Использование `std::atomic_flag`.
- Использование `std::atomic<bool>`.
- Использование использования функции `wait_for(lock, duration, predicate)`.
- Использование `std::atomic<T*>`: арифметика указателей.
- Использование операции чтения-изменения-записи.
- Использование функции `try_lock_for(duration)`.
- Использование шаблонного класса `std::chrono::duration`.
- Использование шаблонного класса `std::ratio`.

- Использование шаблонного класса `time_point`.
- Реализация методов `std::promise::set_exception` и
- Реализация шаблона класса `std::chrono::time_point<>`.
- Реализация доступа к данным с помощью атомарных объектов;
- Реализация многопоточного алгоритма быстрой сортировки;
- Синхронизация параллельных операций с помощью условных переменных;
- Структурирование кода для защиты разделяемых данных;
- Упорядочение захват-освобождение и семантика
- Упорядочивание данных с помощью атомарных объектов
- Устранение проблематичных состояний гонки;

Методические указания к выполнению курсовой работы

Общие положения

Сегодня повсеместное использование информационных технологий стало объективной необходимостью. Спектр областей, в которых применяются информационные технологии, чрезвычайно широк.

Основными целями написания курсовой работы являются:

- закрепление знаний, полученных в процессе изучения дисциплины;
- выработка навыков творческого мышления;
- формирование профессиональных навыков, связанных с самостоятельной деятельностью будущего специалиста;
- оформление материалов (четкое, ясное, технически грамотное и качественное литературное изложение пояснительной записки и оформление графического материала работы);
- воспитание ответственности за качество принятых решений.

Цели и тематика курсовых работ

Целью курсовой работы по дисциплине является формирование у студентов углубленных профессиональных компетенций, связанных с использованием методов, алгоритмов, программных и технических средств реализации и использования прикладных интеллектуальных технологий обработки и анализа данных и процессов. Тематика курсовых работ разрабатывается студентом совместно с руководителем.

Каждая курсовая работа строго индивидуальна и ориентирована на развитие у студента определенной части профессиональных навыков и умения творчески решать практические задачи. По результатам выполнения курсовой работы оформляется пояснительная записка.

Вид, структура, содержание и объем курсовой работы

Курсовая работа студента – заключительный этап изучения дисциплины и, как правило, основывается на обобщении выполненных студентом лабораторных работ или представляет собой индивидуальное задание по изучаемой дисциплине и подготавливается к защите в завершающий период теоретического обучения.

Пояснительная записка представляет собой текстовый документ, содержащий описания проблем, решаемых в курсовой работе, технические расчеты (расчет параметров, характеристик и экономических показателей объекта проектирования, а также взаимодействия его функциональных частей, элементов конструкций и дополнительных данных), описание проектируемого объекта, принцип его действия, обоснование принятых технических, технологических и технико-экономических решений.

Рекомендуется следующий **состав и порядок расположения материала** в пояснительной записке к курсовому проекту:

- титульный лист;
- задание на курсовую работу;
- содержание;
- введение (содержит описание состояния проблемы, актуальность, цели и задачи проекта);
- основные главы (устанавливаются руководителем с учетом специфики темы проекта);
- заключение (включает основные результаты, полученные в ходе выполнения курсовой работы, выводы и рекомендации);
- список использованных источников, в т. ч. нормативных, проектных и справочных материалов;
- приложение (при необходимости).

Задание на курсовую работу должно содержать наименование темы проекта и предусматривать по возможности комплексное решение исследовательских задач. Вместе с тем, один из частных вопросов задания того или иного характера выделяется в качестве специальной части проекта и подлежит более глубокой разработке на основе общего решения.

В задании на курсовую работу указывается:

- наименование работы;
- содержание курсовой работы и рекомендуемый объем отдельных частей;
- специальная часть проекта;
- исходные данные;

- рекомендуемая литература;
- календарный график работы студента.

Задание на курсовую работу должно содержать элемент новизны, активизирующий инициативу студента. Каждое задание должно быть индивидуальным, а его тематика, по возможности, комплексной, охватывающей несколько взаимосвязанных задач.

Возможны «сквозные» задания, отдельные аспекты которых студент выполняет в течение нескольких семестров по нескольким, следующим друг за другом, дисциплинам, и которые могут входить в состав задания на выпускную квалификационную работу.

Основные главы пояснительной записки включает следующие разделы:

1. Анализ прикладной области, приводящий к конкретной постановке задачи с выделением совокупности понятий (интенционалы и экстенционалы понятий), подлежащих анализу методами интеллектуального анализа.

Постановка задачи включает:

- цель разработки в конкретной прикладной области для выбранных данных;
- перечень и методы решаемых задач для конкретных данных из выбранной прикладной области;
- общие требования к данным и результатам на уровне вход/выход;
- техническое задание с выбором конкретных методов решаемых задач.

2. Проектирование:

- анализ аналогов решения обозначенных в предыдущем разделе задач;
- выбор, системное и математическое описание конкретного метода (методов) анализа для решения поставленных задач;
- описание мер точности и адекватности применяемых методов;
- проектирование и описание системы хранения данных и систем знаний;
- описание онтологии, если имеется;
- проектирование архитектуры программной системы.

4. Реализация:

- выбор инструментальных средств;
- описание характеристик программной системы;
- разработка и описание пользовательского интерфейса;
- описание используемых компонент системы;
- инструкция пользователю с копиями экранных форм.

Правила оформления пояснительной записки

Пояснительная записка к курсовой работе оформляется в соответствии с требованиями действующих государственных стандартов.

Пояснительная записка может быть оформлена в текстовом процессоре.

Гарнитура TimesNewRoman, 14 пт, оглавление, разбивка на страницы, поля: справа – 2,5 см; сверху снизу– 2см, сверху – 2см, слева – 2см.

Материал пояснительной записки излагается грамотно, четко, сжато. Пояснительная записка должна быть сброшюрована.

Каждая курсовая работа должна проходить нормоконтроль на соответствие оформления общим требованиям к текстовым документам согласно стандартам. Нормоконтроль может проводиться как руководителем курсового проекта, так и утвержденным кафедрой нормоконтролером.

Критерии и шкала оценивания по оценочному средству курсовая работа

Шкала оценивания (интервал баллов)	Критерий оценивания
5	Курсовая работа выполнена на высоком уровне (правильность выполнения 90-100%)
4	Курсовая работа выполнена на среднем уровне (правильность выполнения 75-89%)
3	Курсовая работа выполнена на низком уровне (правильность выполнения 50-74%)
2	Курсовая работа выполнена на неудовлетворительном уровне (правильность выполнения менее чем на 50%)

Оценочные средства для промежуточной аттестации (экзамен)

1. `std::promise::set_exception_at_thread_exit`
2. `std::this_thread::sleep_for()`.
3. Ассоциирование задачи с будущим результатом.
4. Атомарные операции и типы в C++.
5. Взаимоблокировка: проблема и решение.
6. Возврат значения из фоновой задачи.
7. Временные интервалы.
8. Выполнить асинхронные вызовы, функция `std::async`.
9. Выполнить параллельную реализацию алгоритма Quicksort.
10. Выполнить синхронизацию действий с использованием
11. Выполнить синхронизацию действий с помощью разделяемого флага (защищенный мьютексом).
12. Выполнить синхронизацию действий с помощью функции

13. Выполнить синхронизацию действий средствами из стандартной библиотеки C++
14. Выполнить синхронизацию действий, используя условные переменные.
15. Гибкая блокировка с помощью `std::unique_lock`.
16. Дополнительные рекомендации, как избежать взаимоблокировок.
17. Задание количества потоков во время выполнения.
18. Запуск потока в C++.
19. Запуск потоков в фоновом режиме.
20. Защита разделяемых данных во время инициализации.
21. Защита разделяемых данных с помощью мьютексов.
22. Идентификация потоков.
23. Интерфейс адаптера контейнера `std::stack`.
24. Интерфейс класса `std::queue`.
25. Интерфейс класса `threadsafe_queue`.
26. Использование `std::future` для получения результата асинхронной задачи.
27. Использование `std::promise`.
28. Использование иерархии блокировок для предотвращения взаимоблокировок.
29. Использование иерархии блокировок для предотвращения взаимоблокировок.
30. Механизм ослабленного упорядочения.
31. Механизмы шаблонного класса `std::future`.
32. Модель простого конечного автомата для банкомата.
33. Моменты времени.
34. Обработка нескольких соединений в одном потоке с помощью объектов-обещаний.
35. Объекты и ячейки памяти в параллельных задачах.
36. Ожидание завершения потока в случае исключения.
37. Ожидание события или иного условия.
38. Ожидание условия с помощью условных переменных.
39. Ожидание условной переменной с таймаутом.
40. Операции над стандартными атомарными целочисленными типами.
41. Организация порядка данных и `std::memory_order_seq_cst`.
42. Основные концепции библиотеки Chrono.
43. Передача аргументов функции потока.
44. Передача аргументов функции, заданной в `std::async`.
45. Передача владения мьютексом между контекстами.
46. Передача владения потоком.
47. Платформенно-зависимые средства.
48. Полное определение класса потокобезопасной очереди на базе условных переменных.
49. Порядок модификации.

50. Потокобезопасная очередь на базе условных переменных.
51. Применение `std::lock()` и `std::lock_guard` для реализации операции обмена.
52. Применение синхронизации операций для упрощения кода.
53. Пример использования `std::async` вместо `std::thread`
54. Пример использования `std::atomic_flag`.
55. Пример использования `std::atomic<bool>`.
56. Пример использования функции `wait_for(lock, duration, predicate)`.
57. Пример использования `std::atomic<T*>`: арифметика указателей.
58. Пример использования операции чтения-изменения-записи.
59. Пример использования функции `try_lock_for(duration)`.
60. Пример использования шаблонного класса `std::chrono::duration`.
61. Пример использования шаблонного класса `std::ratio`.
62. Пример использования шаблонного класса `time_point`.
63. Пример реализации методов `std::promise::set_exception` и
64. Пример реализации шаблона класса `std::chrono::time_point<>`.
65. Пример упорядочивания ослабленных операций с помощью барьеров.
66. Пример чтения из очереди с применением атомарных операций.
67. Принудительное задание упорядочения неатомарных операций с помощью атомарных.
68. Проблемы разделения данных между потоками.
69. Реализация алгоритма `std::partition()`.
70. Реализация иерархического мьютекса.
71. Реализация интерфейса с применением `std::packaged_task`.
72. Реализация локального, по отношению к потоку, хранилища.
73. Реализация модели захвата-освобождения.
74. Реализация модели последовательной согласованности (`sequential consistency`).
75. Реализация рекурсивной сортировки в духе ФП.
76. Реализация транзитивной синхронизации с помощью упорядочения захват-освобождение.
77. Свободные функции для атомарных операций, примеры.
78. Семантика перемещения.
79. Синхронизация операций с помощью передачи сообщений.
80. Синхронизация параллельных операций.
81. Состояние гонки: возврат указатель на вытолкнутый элемент.
82. Состояние гонки: комбинированный вариант.
83. Состояние гонки: наличия копирующего или перемещающего конструктора.
84. Состояние гонки: передача функции `pop()` ссылки на переменную.
85. Сохранение исключения в будущем результате.
86. Ссылки на `r`-значения и шаблоны функций.
87. Ссылки на `r`-значения.

- 88. Упорядочение доступа к памяти для атомарных операций.
- 89. Функции, принимающие таймаут.
- 90. Функциональное программирование с применением будущих результатов.
- 91. Шаблон класса `std::atomic<>`.
- 92. Ячейки памяти(memory location).

Критерии и шкала оценивания по оценочному средству промежуточный контроль (экзамен)

Шкала оценивания (интервал баллов)	Критерий оценивания
отлично (5)	Студент глубоко и в полном объёме владеет программным материалом. Грамотно, исчерпывающе и логично его излагает в устной или письменной форме. При этом знает рекомендованную литературу, проявляет творческий подход в ответах на вопросы и правильно обосновывает принятые решения, хорошо владеет умениями и навыками при выполнении практических задач.
хорошо (4)	Студент знает программный материал, грамотно и по сути излагает его в устной или письменной форме, допуская незначительные неточности в утверждениях, трактовках, определениях и категориях или незначительное количество ошибок. При этом владеет необходимыми умениями и навыками при выполнении практических задач.
удовлетворительно (3)	Студент знает только основной программный материал, допускает неточности, недостаточно чёткие формулировки, непоследовательность в ответах, излагаемых в устной или письменной форме. При этом недостаточно владеет умениями и навыками при выполнении практических задач. Допускает до 30% ошибок в излагаемых ответах.
неудовлетворительно (2)	Студент не знает значительной части программного материала. При этом допускает принципиальные ошибки в доказательствах, в трактовке понятий и категорий, проявляет низкую культуру знаний, не владеет основными умениями и навыками при выполнении практических задач. Студент отказывается от ответов на дополнительные вопросы

Лист изменений и дополнений

№ п/п	Виды дополнений и изменений	Дата и номер протокола заседания кафедры (кафедр), на котором были рассмотрены и одобрены изменения и дополнения	Подпись (с расшифровкой) заведующего кафедрой (заведующих кафедрами)

Экспертное заключение

Представленный фонд оценочных средств (далее – ФОС) по дисциплине «Параллельные и распределенные вычисления» соответствует требованиям ФГОС ВО.

Предлагаемые формы и средства текущего и промежуточного контроля адекватны целям и задачам реализации основной профессиональной образовательной программы по направлению подготовки 09.03.03 Прикладная информатика.

Оценочные средства для текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины представлены в полном объеме.

Виды оценочных средств, включенные в представленный фонд, отвечают основным принципам формирования ФОС.

Разработанный и представленный для экспертизы фонд оценочных средств рекомендуется к использованию в процессе подготовки обучающихся по указанному направлению.

Председатель учебно-методической
комиссии факультета компьютерных
систем и информационных
технологий



Ветрова Н. Н.