МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Луганский государственный университет имени Владимира Даля»
(ФГБОУ ВО «ЛГУ им. В. Даля»)

Северодонецкий технологический институт
Кафедра информационных технологий, приборостроения и электротехники

УТВЕРЖДАЮ:
Врио. директора СТИ (филиал)
ФГБОУ ВО «ЛГУ им. В. Даля»
_____ Ю.В. Бородач
(подпись)
«26» 09 2024 года

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

«Программирование для параллельных вычислений»

По направлению подготовки: 09.03.04 Программная инженерия

Профиль: Разработка программно-информационных систем

Северодонецк – 2024

Лист согласования РПУД

Рабочая программа учебной дисциплины «Программирование для параллельных вычислений» по направлению подготовки 09.03.04 Программная инженерия (профиль «Разработка программно-информационных систем») – 24 с.

Рабочая программа учебной дисциплины «Программирование для параллельных вычислений» разработана в соответствии федеральным государственным образовательным стандартом высшего образования – бакалавриат по направлению подготовки 09.03.04 Программная инженерия, утвержденным приказом Министерства образования и науки Российской Федерации от 19 сентября 2017 г. № 920 (с изменениями и дополнениями в соответствии с приказами Министерства образования и науки Российской Федерации № 1456 от 26.11.2020 г., № 83 от 08.02.2021 г., № 662 от 19.07.2022 г. и № 208 от 27.02.2023 г.).

СОСТАВИТЕЛЬ:
ст. преподаватель Кузнецова Е.В.

Рабочая программа дисциплины утверждена на заседании кафедры информационных технологий, приборостроения и электротехники «_05_» _сентября_ 2024 г., протокол №_1_.

Заведующий кафедрой ИТПЭ _____ В.Г. Чебан

Переутверждена: «____» _____ 20___г., протокол №___.

Рекомендована на заседании учебно-методической комиссии Северодонецкого технологического института (филиал) федерального государственного бюджетного образовательного учреждения высшего образования «Луганский государственный университет имени Владимира Даля» «_16_» _сентября_ 2024 г., протокол №_1_.

Председатель учебно-методической комиссии
СТИ (филиал) ФГБОУ ВО «ЛГУ им. В.Даля» _____ Ю.В. Бородач

© Кузнецова Е.В., 2024 г.
© ФГБОУ ВО «ЛГУ им. В. Даля» СТИ (филиал), 2024 г.

Структура и содержание дисциплины

1. Цели и задачи дисциплины, ее место в учебном процессе

Цель изучения дисциплины – разработка параллельных и распределенных систем, а также их программного или аппаратного воплощения. Использование многоядерных процессоров или многопроцессорных систем для осуществления многопоточности и выполнения нескольких процессов параллельно.

Задачи: изучение основных принципов разработки параллельных и распределенных систем для различных применений: автоматизация процессов распознавания образов, адаптивное управление, аппроксимация функционалов, прогнозирование, создание экспертных систем, организация ассоциативной памяти и многие другие приложения. Изучение стандартной библиотека C++ Thread Library для реализации высокоуровневых абстракций.

При освоении программы у обучающихся формируется информационно-коммуникационная компетентность – знания, умения и навыки по объектно-ориентированному программированию, необходимые для изучения других общеобразовательных предметов, для их использования в ходе изучения специальных дисциплин профессионального цикла, в практической деятельности и повседневной жизни.

2. Место дисциплины в структуре ОПОП ВО

Дисциплина «Программирование для параллельных вычислений» относится к части учебного плана, формируемой участниками образовательных отношений.

Необходимыми условиями для освоения дисциплины являются: знания и принципы построения современных кроссплатформенных приложений и знаний ООП, языка программирования C++. Изложены основные принципы построения распределенных систем, основные стандарты проектирования программных систем, пакеты программ для объектно-ориентированного программирования.

Дисциплина основывается на базе дисциплин предыдущего уровня образования и является логическим продолжением содержания дисциплин естественно-научного цикла.

Содержание дисциплины является логическим продолжением содержания дисциплин: программирование, функциональное и логическое программирование, численные методы и служит основой для освоения дисциплин: преддипломная практика, выполнения и защиты ВКР.

3. Требования к результатам освоения содержания дисциплины

Студенты, завершившие изучение дисциплины «Программирование для параллельных вычислений», должны

знать:

- 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;
- методы управления данными в потоке;
- способы передачи владения потока;
- методы идентификация потоков;
- проблемы разделения данных между потоками;
- способы защиты разделяемых данных с помощью мьютексов;

– методы организации потокобезопасной очереди на базе условных переменных

– методы доступа к результатам асинхронных операций;

– основные концепции библиотек работы со временем;

– основы функционального программирования с применением будущих результатов;

– атомарные типы и операции;

– методы синхронизация операций и принудительного упорядочения;

– методы упорядочения, захвата-освобождения.

уметь:

– проектировать параллельные и распределенные системы; о использовать библиотеку C++ Thread Library для реализации многопоточности;

– выполнять защиту разделяемых данных с помощью условных переменных;

– выполнять одновременный доступ к ресурсам;

– выполнять защиту разделяемых данных.

Перечисленные результаты образования являются основой для формирования следующих компетенций (в соответствии с ФГОС ВО и требованиями к результатам освоения основной профессиональной образовательной программы (ОПОП ВО):

Код и наименование ПК	**Код и наименование индикатора достижения ПК**
ПК-3. Владеет навыками моделирования, анализа и использования формальных методов конструирования программного обеспечения	ПК-3.1. Знать: методы и приемы формализации задач; языки формализации функциональных спецификаций; методы и приемы алгоритмизации поставленных задач; нотации и программные продукты для графического отображения алгоритмов; алгоритмы решения типовых задач, области и способы их применения ПК-3.2. Уметь: использовать и применять: методы и приемы формализации задач; методы и приемы алгоритмизации поставленных задач; программные продукты для графического отображения алгоритмов; стандартные алгоритмы в соответствующих предметных областях ПК-3.3. Владеть: приемами составления формализованных описаний решений поставленных задач в соответствии с требованиями технического задания или других нормативных документов; приемами разработки алгоритмов решения поставленных задач в соответствии с требованиями технического задания или других нормативных документов; приемами оценки и согласования сроков выполнения поставленных задач

4. Структура и содержание дисциплины

4.1. Объем учебной дисциплины и виды учебной работы

Вид учебной работы	Объем часов (з.е.)		
	Очная форма	Очно-заочная форма	Заочная форма
Объем учебной дисциплины (всего)	**324 (9 з.е.)**		**324 (9 з.е.)**
Обязательная аудиторная учебная нагрузка дисциплины (всего) в том числе:	140		24
Лекции	56		12
Семинарские занятия	-		-
Практические занятия			-
Лабораторные работы	84		12
Курсовая работа (курсовой проект)	-		-
Индивидуальное задание	36		36
Самостоятельная работа студента (всего)	184		300
Форма аттестации	экзамен		экзамен

4.2. Содержание разделов дисциплины

Семестр 7

Тема 1. Введение в параллельное программирование.

Параллелизм в вычислительных системах. Подходы к организации параллелизма.

Параллелизм за счет нескольких процессов. Параллелизм за счет нескольких потоков.

Тема 2. Зачем нужен параллелизм.

Применение параллелизма для разделения обязанностей. Применение параллелизма для повышения производительности. Когда параллелизм вреден.

Тема 3. Параллелизм и многопоточность в C++.

История многопоточности в C++. Поддержка параллелизма в новом стандарте. Эффективность библиотеки многопоточности для C++. Платформенно-зависимые средства.

Тема 4. Управление потоками.

Базовые операции управления потоками. Запуск потока. Ожидание завершения потока.

Ожидание в случае исключения. Запуск потоков в фоновом режиме.

Тема 5. Управление данными в потоке.

Передача аргументов функции потока. Ссылки на r-значения. Семантика перемещения.

Ссылки на r-значения и шаблоны функций.

Тема 6. Владения потоком.

Передача владения потоком. Задание количества потоков во время выполнения.

Идентификация потоков.

Тема 7. Разделение данных между потоками.

Проблемы разделения данных между потоками. Гонки. Устранение проблематичных состояний гонки.

Тема 8. Защита разделяемых данных с помощью мьютексов.

Использование мьютексов в C++. Структурирование кода для защиты разделяемых данных. Выявление состояний гонки, внутренне присущих интерфейсам.

Тема 9. Взаимоблокировка: проблема и решение.

Дополнительные рекомендации, как избежать взаимоблокировок. Гибкая блокировка с помощью std::unique_lock. Передача владения мьютексом между контекстами. Выбор правильной гранулярности блокировки.

Тема 10. Другие средства защиты разделяемых данных.

Защита разделяемых данных во время инициализации. Защита редко обновляемых структур данных. Рекурсивная блокировка.

Семестр 8

Тема 11. Синхронизация параллельных операций

Ожидание события или иного условия. Ожидание условия с помощью условных переменных. Потокобезопасная очередь на базе условных переменных.

Тема 12. Ожидание одноразовых событий с помощью механизма будущих результатов

Возврат значения из фоновой задачи. Ассоциирование задачи с будущим результатом. Использование std::promise. Сохранение исключения в будущем результате. Ожидание в нескольких потоках.

Тема 13. Ожидание с ограничением по времени

Часы. Временные интервалы. Моменты времени. Функции, принимающие таймаут.

Тема 14. Применение синхронизации операций для упрощения кода

Функциональное программирование с применением будущих результатов. Синхронизация операций с помощью передачи сообщений.

Тема 15. Модель памяти C++ и атомарные операции

Объекты и ячейки памяти. Объекты, ячейки памяти и параллелизм.

Порядок модификации. Атомарные операции и типы в C++.

Тема 16. Синхронизация операций и принудительное упорядочение

Отношение синхронизируется. Отношение происходит раньше. Упорядочение доступа к памяти для атомарных операций. Последовательности освобождений. Барьеры. Упорядочение неатомарных операций с помощью атомарных

4.3. Лекции

№ п/п	Название темы	Объем часов		
		Очная форма	Очно-заочная форма	Заочная форма
1.	Введение в параллельное программирование	3		2
2.	Зачем нужен параллелизм	3		
3.	Параллелизм и многопоточность в C++	3		2
4.	Управление потоками	3		
5.	Управление данными в потоке	3		2
6.	Владения потоком	3		
7.	Разделение данных между потоками	3		
8.	Защита разделяемых данных с помощью мьютексов	3		2
9.	Выявление состояний гонки	3		
10.	Взаимоблокировка: проблема и решение	3		
11.	Передача владения мьютексом между контекстами	3		2
12.	Другие средства защиты разделяемых данных	3		
13.	Защита редко обновляемых структур данных	3		
14.	Рекурсивная блокировка	3		

15.	Синхронизация параллельных операций	3		
16.	Ожидание одноразовых событий с помощью механизма будущих результатов	3		
17.	Ожидание с ограничением по времени	2		
18.	Применение синхронизации операций для упрощения кода	2		
19.	Модель памяти C++ и атомарные операции	2		2
20.	Синхронизация операций и принудительное упорядочение	2		
Итого:		**56**		**12**

4.4. Практические (семинарские) занятия

Не предусмотрены.

4.5. Лабораторные работы

№ п/п	Название темы	Объем часов		
		Очная форма	Очно-заочная форма	Заочная форма
1.	Введение в параллельное программирование	5		2
2.	Управление потоками	5		
3.	Управление данными в потоке	5		
4.	Ссылки на r-значения	5		2
5.	Передача владения потоком	5		
6.	Задание количества потоков во время выполнения	5		
7.	Идентификация потоков	5		
8.	Защита разделяемых данных с помощью мьютексов	5		2
9.	Одновременный доступ к ресурсам	5		
10.	Синхронизация действия, выполняемые в разных потоках	5		
11.	Моделирование одноразовых событий с помощью будущего результата	5		2
12.	Основные концепции библиотеки Chrono (C++)	5		
13.	Алгоритма быстрой сортировки Quicksort	6		
14.	Атомарные типы и операции	6		2
15.	Синхронизация операций и принудительное упорядочение modification order	6		
16.	Зависимости по данным, упорядочение захват-освобождение и семантика memory_order_consume	6		2
Итого:		**84**		**12**

4.6. Самостоятельная работа студентов

№ п/п	Название темы	Вид СРС	Объем часов	
			Очная форма	Заочная форма
1.	Подходы к организации параллелизма	Написание реферата, создание презентации по теме	7	12
2.	Когда параллелизм вреден	Написание реферата, создание презентации по теме	7	12
3.	Платформенно-зависимые средства	Написание реферата, создание презентации по теме	7	12
4.	Запуск потоков в фоновом режиме	Подготовка к лабораторным работам	7	12
5.	Ссылки на r-значения и шаблоны функций	Подготовка к лабораторным работам	7	12
6.	Идентификация потоков	Подготовка к лабораторным работам	7	12
7.	Устранение проблематичных состояний гонки	Подготовка к лабораторным работам	7	12
8.	Структурирование кода для защиты разделяемых данных	Подготовка к лабораторным работам	7	12
9.	Гибкая блокировка с помощью std::unique_lock	Подготовка к лабораторным работам	7	12
10.	Защита редко обновляемых структур данных	Подготовка к лабораторным работам	7	12
11.	Быстрая сортировка в духе ФП	Написание реферата, создание презентации по теме	7	12
12.	Параллельная реализация Quicksort в духе ФП	Написание реферата, создание презентации по теме	7	12
13.	Стандартные атомарные типы	Написание реферата, создание презентации по теме	7	12
14.	Операции над std::atomic_flag	Подготовка к лабораторным работам	7	12
15.	Операции над std::atomic<bool>	Подготовка к лабораторным работам	7	12
16.	Операции над std::atomic<T*>: арифметика указателей	Подготовка к лабораторным работам	7	12
17.	Операции над стандартными атомарными целочисленными типами	Подготовка к лабораторным работам	8	12
18.	Основной шаблон класса std::atomic<>	Подготовка к лабораторным работам	8	12
19.	Свободные функции для атомарных операций	Подготовка к лабораторным работам	8	12
20.	Последовательно согласованное упорядочение	Подготовка к лабораторным работам	8	12
21.	Непоследовательно согласованное упорядочение доступа к памяти	Подготовка к лабораторным работам	8	12
22.	Ослабленное упорядочение	Подготовка к лабораторным работам	8	12

23.	Механизм ослабленного упорядочения	Подготовка к лабораторным работам	8	12
24.	Упорядочение захват-освобождение	Подготовка к лабораторным работам	8	12
25.	Транзитивная синхронизация с помощью упорядочения захват-освобождение	Написание реферата, создание презентации по теме	8	12
Итого:			**184**	**300**

4.7. Курсовые работы/проекты

Курсовые работы (проекты) не предусмотрены.

5. Образовательные технологии

Преподавание дисциплины ведется с применением следующих видов образовательных технологий:

– традиционные объяснительно-иллюстративные технологии, которые обеспечивают доступность учебного материала для большинства студентов, системность, отработанность организационных форм и привычных методов, относительно малые затраты времени;

– технологии проблемного обучения, направленные на развитие познавательной активности, творческой самостоятельности студентов и предполагающие последовательное и целенаправленное выдвижение перед студентом познавательных задач, разрешение которых позволяет студентам активно усваивать знания (используются поисковые методы; постановка познавательных задач);

– технологии развивающего обучения, позволяющие ориентировать учебный процесс на потенциальные возможности студентов, их реализацию и развитие;

– технологии концентрированного обучения, суть которых состоит в создании максимально близкой к естественным психологическим особенностям человеческого восприятия структуры учебного процесса и которые дают возможность глубокого и системного изучения содержания учебных дисциплин за счет объединения занятий в тематические блоки;

– технологии модульного обучения, дающие возможность обеспечения гибкости процесса обучения, адаптации его к индивидуальным потребностям и особенностям обучающихся (применяются, как правило, при самостоятельном обучении студентов по индивидуальному учебному плану);

– технологии дифференцированного обучения, обеспечивающие возможность создания оптимальных условий для развития интересов и способностей студентов, в том числе и студентов с особыми образовательными потребностями, что позволяет реализовать в культурно-образовательном пространстве университета идею создания равных возможностей для получения образования;

– технологии активного (контекстного) обучения, с помощью которых осуществляется моделирование предметного, проблемного и социального содержания будущей профессиональной деятельности студентов (используются активные и интерактивные методы обучения) и т.д.

Максимальная эффективность педагогического процесса достигается путем конструирования оптимального комплекса педагогических технологий и (или) их элементов на личностно-ориентированной, деятельностной, диалогической основе и использования необходимых современных средств обучения.

6. Учебно-методическое и программно-информационное обеспечение дисциплины:

а) основная литература:

1. Энтони Уильямс. Параллельное программирование на C++ в действии. Практика разработки многопоточных программ. Пер. с англ. Слинкин А. А. – М.: ДМК Пресс, 2018. – 672с.: ил.

б) дополнительная литература:

1. Богачёв К. Ю. Б73 Основы параллельного программирования [Электронный ресурс]: учебное пособие / К. Ю. Богачёв. – 2-е изд. (эл.). – М.: БИНОМ. Л

2. Антонов А.С. Параллельное программирование с использованием технологии OpenMP: Учебное пособие. – М.: Изд-во МГУ, 2009. – 77 с.

4. OpenMP Architecture Review Board (http://www.openmp.org/).

5. The Community of OpenMP Users, Researchers, Tool Developers and Providers (http://www.compunity.org/).

6. OpenMP Application Program Interface Version 3.0 May 2018 (http://www.openmp.org/mp-documents/spec30.pdf).

7. Что такое OpenMP? (http://parallel.ru/tech/tech_dev/openmp.html).

8. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. СПб.: БХВ-Петербург, 2002. – 608 с.

9. Barbara Chapman, Gabriele Jost, Ruud van der Pas. Using OpenMP: portable shared memory parallel programming (Scientific and Engineering Computation). Cambridge, Massachusetts: The MIT Press., 2008. – 353 pp.

10. Антонов А.С. Введение в параллельные вычисления. Методическое пособие.-М.: Изд-во Физического факультета МГУ, 2002. – 70 с.

11. Антонов А.С. Параллельное программирование с использованием технологии MPI: Учебное пособие. – М.: Изд-во МГУ, 2014. – 71 с.

Методические указания:

1. Методические указания к лабораторным работам по дисциплине «Параллельные и распределенные вычисления» для студентов всех направлений подготовки. Луганск: Изд-во ЛНУ им. В.Даля, 2018. – 31 с.

2. Конспект лекций по дисциплине “Параллельные и распределенные вычисления” (для студентов по направлению «Прикладная информатика» 09.03.03) Часть I / Сост. Письменский А.В. – Луганск: изд-во ЛГУ им. В.И.Даля, 2018.– 116 с.

3. Конспект лекций по дисциплине «Параллельное программирование» (для студентов по направлению «Программная инженерия» 09.04.04) / Сост. Письменский А.В. – Луганск: ГОУ ВО ЛНР ЛГУ В.И.Даля, 2022.– 116 с.

в) интернет-ресурсы:

1. Министерство образования и науки Российской Федерации – http://минобрнауки.рф

2. Министерства природных ресурсов и экологии Российской Федерации – http://www.mnr.gov.ru

3. Федеральная служба по надзору в сфере образования и науки – http://obrnadzor.gov.ru

4. Министерство образования и науки Луганской Народной Республики – https://minobr.su

5. Министерство природных ресурсов и экологической безопасности ЛНР – https://www.mprlnr.su

6. Народный совет Луганской Народной Республики – https://nslnr.su

7. Портал Федеральных государственных образовательных стандартов высшего образования – http://fgosvo.ru

8. Федеральный портал «Российское образование» – http://www.edu.ru
9. Информационная система «Единое окно доступа к образовательным ресурсам» – http://window.edu.ru
10. Федеральный центр информационно-образовательных ресурсов – http://fcior.edu.ru

Электронные библиотечные системы и ресурсы:
1. Электронно-библиотечная система «Консультант студента» – http://www.studentlibrary.ru/cgi-bin/mb4x
2. Электронно-библиотечная система «StudMed.ru» – https://www.studmed.ru
3. Научная электронная библиотека eLIBRARI.RU» – http://elibrary.ru
4. ЭБС Издательства «ЛАНЬ» – https://e.lanbook.com

Информационный ресурс библиотеки образовательной организации
1. Научная библиотека имени А. Н. Коняева – http://biblio.dahluniver.ru

7. Материально-техническое обеспечение дисциплины

Лекционные занятия: комплект электронных презентаций/слайдов; аудитория, оснащенная презентационной техникой (проектор, экран, компьютер/ноутбук).

Лабораторные работы: компьютерный класс, презентационная техника (проектор, экран, компьютер/ноутбук), пакеты ПО общего назначения. Рабочее место преподавателя, оснащенное компьютером с доступом в Интернет, рабочие места студентов, оснащенные компьютерами с доступом в Интернет, предназначенные для работы в электронной образовательной среде.

Программное обеспечение:

Функциональное назначение	**Бесплатное программное обеспечение**	**Ссылки**
Офисный пакет	Libre Office 6.3.1	https://www.libreoffice.org/ https://ru.wikipedia.org/wiki/LibreOffice
Операционная система	UBUNTU 19.04	https://ubuntu.com/ https://ru.wikipedia.org/wiki/Ubuntu
Браузер	Firefox Mozilla	http://www.mozilla.org/ru/firefox/fx
Браузер	Opera	http://www.opera.com
Архиватор	7Zip	http://www.7-zip.org/
Редактор PDF	PDFCreator	http://www.pdfforge.org/pdfcreator
Математический редактор	Mathcad Prime 6.0	https://www.mathcad.com/ru/
Интегрированная среда разработки	Visual Studio 2019	https://visualstudio.microsoft.com/ru/free-developer-offers/
Серверная платформа и программная среда	Open Server Panel	https://ospanel.io/download/
Интегрированная среда разработки	CODESYS V3	https://owen.ru/product/codesys_v3
Графическая среда имитационного моделирования	Simulink	https://matlab.ru/products/Simulink

8. Оценочные средства по учебной дисциплине

Паспорт
фонда оценочных средств по учебной дисциплине «Программирование для параллельных вычислений»

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

№ п/п	Код контролируемой компетенции	Формулировка контролируемой компетенции	Контролируемые темы учебной дисциплины	Этапы формирования (семестр изучения)
1	ПК-3.1	Знать: методы и приемы формализации задач; языки формализации функциональных спецификаций; методы и приемы алгоритмизации поставленных задач; нотации и программные продукты для графического отображения алгоритмов; алгоритмы решения типовых задач, области и способы их применения	Тема 1. Тема 2. Тема 3. Тема 4. Тема 5.	начальный (5)
2	ПК-3.2	Уметь: использовать и применять: методы и приемы формализации задач; методы и приемы алгоритмизации поставленных задач; программные продукты для графического отображения алгоритмов; стандартные алгоритмы в соответствующих предметных областях	Тема 6. Тема 7. Тема 8. Тема 9. Тема 10.	начальный (5)
3	ПК-3.3	Владеть: приемами составления формализованных описаний решений поставленных задач в соответствии с требованиями технического задания или других нормативных документов; приемами разработки алгоритмов решения поставленных задач в соответствии с требованиями технического задания или других нормативных 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 контроли-руемой компетенции	Показатель оценивания (знания, умения, навыки)	Контролируемые темы учебной дисциплины	Наименование оценочного средства
1.	ПК-3.1	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-3.2	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	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

		разделяемых данных с помощью мьютексов; - выполнять одновременный доступ к ресурсам		
3.	ПК-3.3	Знать: - методы организации потокобезопасной очереди на базе условных переменных; - методы доступа к результатам асинхронных операций; - основные концепции библиотек работы со временем; - атомарные типы и операции; -методы синхронизации операций и принудительного упорядочения; - методы упорядочения, захвата-освобождения; - атомарные типы и операции; - методы синхронизация операций и принудительного упорядочения; - методы упорядочения; - методы организации потокобезопасной очереди на базе условных переменных, методы доступа к результатам. Уметь: - проектировать параллельные и распределенные системы; - выполнять защиту разделяемых данных с помощью условных переменных; - выполнять одновременный доступ к ресурсам; - выполнять защиту разделяемых данных; - выполнять защиту разделяемых данных с помощью условных переменных; - выполнять одновременный доступ к ресурсам; - выполнять защиту разделяемых данных; - проектировать параллельные и распределенные системы; - выполнять одновременный доступ к ресурсам; - выполнять защиту разделяемых данных	Тема 11. Синхронизация параллельных 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	Вопросы для обсуждения (в виде докладов и сообщений), лабораторные работы, контрольные работы

Фонды оценочных средств по дисциплине
«Программирование для параллельных вычислений»

Вопросы для обсуждения (в виде докладов и сообщений):

1. Ссылки на r-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 упорядочение захват-освобождение 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. Использования 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%)

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 использование 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

9. Особенности организации обучения для лиц с ограниченными возможностями здоровья и инвалидов

При необходимости рабочая программа учебной дисциплины может быть адаптирована для обеспечения образовательного процесса инвалидов и лиц с ограниченными возможностями здоровья, в том числе с применением электронного обучения и дистанционных образовательных технологий.

Для этого требуется заявление студента (его законного представителя) и заключение психолого-медико-педагогической комиссии (ПМПК). В случае необходимости обучающимся из числа лиц с ограниченными возможностями здоровья (по заявлению обучающегося), а для инвалидов также в соответствии с индивидуальной программой реабилитации инвалида могут предлагаться следующие варианты восприятия учебной информации с учетом их индивидуальных психофизических особенностей:

– создание текстовой версии любого нетекстового контента для его возможного преобразования в альтернативные формы, удобные для различных пользователей;

– создание контента, который можно представить в различных видах без потери данных или структуры, предусмотреть возможность масштабирования текста и изображений без потери качества, предусмотреть доступность управления контентом с клавиатуры;

– создание возможностей для обучающихся воспринимать одну и ту же информацию из разных источников, например, так, чтобы лица с нарушениями слуха получали информацию визуально, с нарушениями зрения – аудиально;

– применение программных средств, обеспечивающих возможность освоения навыков и умений, формируемых дисциплиной (модулем), за счёт альтернативных способов, в том числе виртуальных лабораторий и симуляционных технологий;

– применение электронного обучения, дистанционных образовательных технологий для передачи информации, организации различных форм интерактивной контактной работы обучающегося с преподавателем, в том числе вебинаров, которые могут быть использованы для проведения виртуальных лекций с возможностью взаимодействия всех участников дистанционного обучения, проведения семинаров, выступления с докладами и защиты выполненных работ, проведения тренингов, организации коллективной работы;

– применение электронного обучения, дистанционных образовательных технологий для организации форм текущего и промежуточного контроля;

– увеличение продолжительности сдачи обучающимся инвалидом или лицом с ограниченными возможностями здоровья форм промежуточной аттестации по отношению к установленной продолжительности их сдачи;

– продолжительность сдачи зачёта или экзамена, проводимого в письменной форме, – не более чем на 90 минут;

– продолжительность подготовки обучающегося к ответу на зачёте или экзамене, проводимом в устной форме, – не более чем на 20 минут;

– продолжительность выступления обучающегося при защите курсовой работы – не более чем на 15 минут.

Лист изменений и дополнений

№ п/п	Виды дополнений и изменений с указанием страниц	Дата и номер протокола заседания кафедры (кафедр), на котором были рассмотрены и одобрены изменения и дополнения	Подпись (с расшифровкой) заведующего кафедрой (заведующих кафедрами)
1.			
2.			
3.			
4.			