Государственное образовательное учреждение

«Приднестровский государственный университет им. Т.Г. Шевченко»

Физико-технический институт

Факультет информатики и вычислительной техники

Кафедра информационных технологий

УТВЕРЖДАЮ

Заведующий кафедрой ИТ

Ю.А. Столяренко

«29» августа 2024 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине

ПАРАЛЛЕЛЬНОЕ ПРОГРАММИРОВАНИЕ

Направление подготовки
2.09.03.02 Информационные системы и технологии

Профиль подготовки
Безопасность информационных систем

Квалификация (степень) выпускника: **бакалавр**

Форма обучения: **очная, заочная**

Год набора: **2021 г.**

Разработал:
ст. преподаватель
кафедры ИТ

/О.В. Сылка

«29» августа 2024 г.

Тирасполь, 2024

Паспорт фонда оценочных средств по учебной дисциплине

1. В результате изучения дисциплины «Параллельное программирование» у обучающегося должны быть сформированы следующие компетенции:

Категория (группа) компетенций	Код и наименование	Код и наименование индикатора достижения универсальной компетенции
	ОПК-6. Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий	ИД-1 опк-6 Знать: методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий.
		ИД-2 опк-6 Уметь: применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий.
		ИД-3 опк-6 Иметь навыки: программирования, отладки и тестирования прототипов программно-технических комплексов задач

2. Программа оценивания контролируемой компетенции:

Текущая аттестация	Контролируемые модули, разделы (темы) дисциплины их название	Код контролируемой компетенции (или ее части)	Наименование оценочного средства
РУБЕЖНЫЙ КОНТРОЛЬ	Раздел 1. Параллельные алгоритмы Раздел 2. Алгоритмизация параллельных вычислений Раздел 3. Средства разработки параллельных вычислений с использованием потоков (нитей) кода	ПК-10	Контрольная работа №1 Лабораторная работа №1 Лабораторная работа №2 Лабораторная работа №3 Лабораторная работа №4
РУБЕЖНАЯ АТТЕСТАЦИЯ	Раздел 4. Средства разработки распределенных вычислений Раздел 5. Протокол и библиотеки MPI Раздел 6. Средства создания и отладки параллельных задач.		Контрольная работа №2 Лабораторная работа №5 Лабораторная работа №6 Лабораторная работа №7 Лабораторная работа №8
Промежуточная аттестация		Код контролируемой компетенции (или ее части)	Наименование оценочного средства
№1		ПК-10	Зачет

3. Показатели и критерии оценивания компетенции по этапам формирования, описание шкал оценивания

Этапы оценивания компетенции	Показатели достижения заданного уровня освоения компетенции	Критерии оценивания результатов обучения			
		2	**3**	**4**	**5**
Первый этап	ИД-1$_{ОПК-6}$ Знать: методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий.	Не знает	Знает основные понятия, но не знает особенностей современных методов их применения	Знает основные понятия и основы, но не может применять знания в полной мере в реальных ситуациях	Знает методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий.
Второй этап	ИД-2$_{ОПК-6}$ Уметь: применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий.	Не умеет	Умеет определить, какие методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий использовать, но не умеет в полной мере использовать их возможности	Умеет применять современные средства и языки программирования, но не умеет выбирать оптимальные решения	Умеет применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий.
Третий этап	ИД-3$_{ОПК-6}$ Иметь навыки: программирования, отладки и тестирования прототипов программно-технических комплексов задач	Не владеет	Владеет навыками программирования, отладки и тестирования прототипов программно-технических комплексов задач, но не умеет в полной мере использовать их возможности	Имеет навыки использования операционных систем, но не умеет выбирать оптимальные решения	Имеет навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач в профессиональной деятельносьт

4. Шкала оценивания

Согласно Положению «О порядке организации аттестации в ИТИ ПГУ им. Т.Г. Шевченко, итоговая оценка представляет собой сумму баллов, полученных студентом по итогу освоения дисциплины (модуля):

Оценка в традиционной шкале	Оценка в 100-балльной шкале	Буквенные эквиваленты оценок в шкале ЗЕ (% успешно аттестованных)
5 (отлично)	88–100	A (отлично) – 88-100 баллов
4 (хорошо)	70–87	B (очень хорошо) – 80-87баллов
		C (хорошо) – 70-79 баллов
3 (удовлетворительно)	50–69	D (удовлетворительно) – 60-69 баллов
		E (посредственно) – 50-59 баллов
2 (неудовлетворительно)	0–49	Fx – неудовлетворительно, с возможной пересдачей – 21-49 баллов
		F – неудовлетворительно, с повторным изучением дисциплины – 0-20 баллов

Расшифровка уровня знаний, соответствующего полученным баллам, дается в таблице, указанной ниже

A	“Отлично” - теоретическое содержание курса освоено полностью, без пробелов, необходимые практические навыки работы с освоенным материалом сформированы, все предусмотренные программой обучения учебные задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному.
B	“Очень хорошо” - теоретическое содержание курса освоено полностью, без пробелов, необходимые практические навыки работы с освоенным материалом в основном сформированы, все предусмотренные программой обучения учебные задания выполнены, качество выполнения большинства из них оценено числом баллов, близким к максимальному.
C	“Хорошо” - теоретическое содержание курса освоено полностью, без пробелов, некоторые практические навыки работы с освоенным материалом сформированы недостаточно, все предусмотренные программой обучения учебные задания выполнены, качество выполнения ни одного из них не оценено минимальным числом баллов, некоторые виды заданий выполнены с ошибками.
D	“Удовлетворительно” - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые практические навыки работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий, возможно, содержат ошибки.
E	“Посредственно” - теоретическое содержание курса освоено частично, некоторые практические навыки работы не сформированы, многие предусмотренные программой обучения учебные задания не выполнены, либо качество выполнения некоторых из них оценено числом баллов, близким к минимальному.
FX	“Условно неудовлетворительно” - теоретическое содержание курса освоено частично, необходимые практические навыки работы не сформированы, большинство предусмотренных программой обучения учебных заданий не выполнено, либо качество их выполнения оценено числом баллов, близким к минимальному; при дополнительной самостоятельной работе над материалом курса возможно повышение качества выполнения учебных заданий.

F	“Безусловно неудовлетворительно” - теоретическое содержание курса не освоено, необходимые практические навыки работы не сформированы, все выполненные учебные задания содержат грубые ошибки, дополнительная самостоятельная работа над материалом курса не приведет к какому-либо значимому повышению качества выполнения учебных заданий.

5. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций при изучении учебной дисциплины в процессе освоения образовательной программы

5.1 Типовой вариант К1 Контрольная работа 1.
Пример задания к модульному контролю №1
1. Описать работу класса *Thread*.
2. Способы управления объектом класса *Thread*.

5.2 Лабораторная работа №1-2. Примерный перечень тематик и методика выставления баллов, типовой вариант
Тема: Основы работы с потоками. Класс Thread.
Задание:
1. Создать метод со сложной для процессора обработкой информации
2. Запустить выполнение данного метода в цикле *N* раз и зафиксировать время выполнения
3. Используя класс *Thread* запустить выполнение этого метода параллельно в *N* потоках и зафиксировать время выполнения
4. Изменяя значение *N* составить таблицу занимаемого времени для выполнения операции и построить график зависимости времени выполнения от *N*

5.3Лабораторная работа №3-4. Примерный перечень тематик и методика выставления баллов, типовой вариант
Тема: Класс Task.
Задание:
1. Реализовать вызов метода (процедуры/функции) через объект класса *Task*
2. Вызываемый метод должен вызвать другой метод в главном потоке (*Main thread*)
3. За основу взять проект, созданный на лекции.
В предоставленном коде-заготовке код в *callback* должен выполниться в главном потоке

5.4 Контрольная работа №2. Примерный перечень заданий и методика выставления баллов
Пример задания к модульному контролю №2
Реализовать класс для параллельного преобразования данных из одного формата в другой.

5.5 Лабораторная работа №5-6. Примерный перечень тематик и методика выставления баллов, типовой вариант
Тема: Класс *Task*. Конструкция *Async/Await*.
Задание:

Реализовать тестовый пример, в котором обработка коллекции (массива) данных происходила бы параллельно
К примеру, имеется коллекция из чисел, нужно произвести одинаковые вычисления над каждым элементом коллекции и вернуть полученный результат
Внимание. Основная программа должна ждать, пока будут обработаны элементы коллекции, и вывести результат после завершения вычислений

5.6 Лабораторная работа №5-6. Примерный перечень тематик и методика выставления баллов, типовой вариант
Тема:
Задание:
Реализовать тестовый пример, в котором будет производиться следующие операции:
1. Создается и запускается задача (*Task*)
2. Основной поток ждет 1 секунду – *Thread.Sleep*(1000)
3. Происходит вызов завершения задачи (*Task*)
Создаваемый *Task* должен выполнять простую циклическую задачу, пока не будет прерван, например, проходит бесконечный цикл, в теле которого проверяется, не прервана ли задача
Вызов завершения задачи (*Task*) должен происходить через *CancellationTokenSource*

5.7 Типовой тест

- Что такое параллельное программирование?

а) Разработка программ без использования переменных
б) Создание графических интерфейсов
в) Написание кода с возможностью одновременного выполнения нескольких операций
г) Использование устаревших языков программирования

- Что является целью параллельного программирования?

а) Упрощение отладки
б) Увеличение объема памяти
в) Повышение производительности за счёт одновременного выполнения задач
г) Снижение температуры процессора

- Что такое поток (thread) в параллельном программировании?

а) Веб-страница
б) Единица выполнения внутри процесса
в) Имя переменной
г) Файл настройки

- Что такое процесс (process)?

а) Структура данных
б) Блок кода
в) Независимая единица выполнения с собственным адресным пространством
г) Сканер

- Какой механизм используется для синхронизации потоков?

а) Прокси
б) Компилятор

в) Мьютексы, семафоры и блокировки
г) Дескрипторы

- Что такое гонка данных (race condition)?

а) Метод сортировки
б) Ошибка, возникающая при конкурентном доступе к общим данным
в) Состояние перегрева CPU
г) Ошибка компиляции

- Что обеспечивает мьютекс?

а) Параллельное выполнение всех потоков
б) Очередь сообщений
в) Взаимоисключение при доступе к разделяемым ресурсам
г) Сжатие данных

- Какой тип памяти часто используется при параллельной обработке на GPU?

а) ROM
б) Кэш L1
в) Глобальная память
г) Виртуальная память

- Что такое OpenMP?

а) Протокол сетевой безопасности
б) Библиотека для работы с графикой
в) API для параллельного программирования с использованием директив компилятора
г) Система управления базами данных

- Какой ключ OpenMP используется для параллельного выполнения цикла?

а) #define
б) #pragma omp parallel for
в) #thread loop
г) #sync

- Что такое deadlock (взаимная блокировка)?

а) Ситуация, когда потоки ждут друг друга бесконечно
б) Ошибка компиляции
в) Завершение потока
г) Исключение в базе данных

- Что означает термин «спидап» (speedup) в контексте параллельного программирования?

а) Снижение частоты
б) Увеличение объема ОЗУ
в) Отношение времени выполнения последовательной и параллельной программы
г) Сравнение версий ОС

- Какой язык программирования часто используется для работы с CUDA?

а) Python
б) Java
в) C++
г) Ruby

- Что такое Amdahl’s Law?

а) Закон распределения трафика
б) Закон эффективности сетей
в) Ограничение ускорения параллельной программы из-за последовательной части
г) Правило кэширования

- Что такое семафор?

а) Вид маршрутизатора
б) Переменная-счетчик для синхронизации потоков
в) Компонент видеокарты
г) Драйвер

- Какой тип параллелизма характерен для многопроцессорных систем?

а) Векторный
б) Инструкционный
в) Многопоточный
г) Массивный параллелизм

- Что такое критическая секция?

а) Ошибка в логике программы
б) Блок кода, доступ к которому должен быть строго синхронизирован
в) Архив логов
г) Статическая переменная

- Что означает fork в Unix-системах?

а) Создание новой файловой системы
б) Копирование файла
в) Создание дочернего процесса
г) Откат изменений

- Что из перечисленного является фреймворком для параллельных вычислений на Python?

а) Pandas
б) Flask
в) Dask
г) Django

- Что такое MPI?

а) Протокол электронной почты
б) Система управления памятью
в) Интерфейс для обмена сообщениями в распределённых системах
г) Стандарт на файловую систему

- Что означает потокобезопасность (thread safety)?

а) Доступность библиотеки в интернете
б) Невозможность модификации кода
в) Возможность безопасного одновременного использования объекта несколькими потоками
г) Защита кода от вирусов

- Что такое map-reduce?

а) Метод кодирования

б) Модель параллельной обработки больших объёмов данных
в) Функция в Excel
г) Режим вывода

- Что означает понятие "granularity" (гранулярность) в параллельном программировании?

а) Количество открытых файлов
б) Объем графической памяти
в) Размер задач, выполняемых параллельно
г) Длина IP-адреса

- Что такое load balancing?

а) Установка Windows
б) Равномерное распределение задач между потоками или процессорами
в) Перезапуск скриптов
г) Диагностика ошибок

- Какой тип параллелизма реализуется на уровне инструкций?

а) Thread-level
б) Process-level
в) Instruction-level
г) Core-level

- Что делает директива `#pragma omp critical`?

а) Увеличивает поток данных
б) Обеспечивает последовательный доступ к критическому участку кода
в) Ускоряет цикл
г) Завершает процесс

- Что такое GPU?

а) Гибридный протокол
б) Универсальный язык
в) Графический процессор, способный к параллельной обработке
г) Устройство чтения данных

- Что такое поток исполнения (execution thread)?

а) Файл запуска
б) Поток логирования
в) Последовательность команд, выполняемая независимо
г) Сетевая передача данных

- Какой из языков поддерживает встроенные средства многопоточности?

а) C++
б) HTML
в) CSS
г) SQL

- Что означает термин "scalability" (масштабируемость)?

а) Возможность восстановления данных
б) Уменьшение ресурсов
в) Способность системы эффективно использовать увеличенные ресурсы
г) Защита от сбоев

5.8 Список вопросов к зачету по дисциплине

1. Преимущества и недостатки параллельной обработки данных.
2. Основные виды коллекций, встречающихся при параллельной обработке данных.
3. Основные виды коллизий.
4. Основные способы разрешения коллизий.
5. Способы передачи данных между потоками.
6. Способы передачи данных между несколькими приложениями.
7. Примитивы синхронизации. Мьютексы.
8. Примитивы синхронизации. Семафоры.
9. Параллельная обработка данных. Критическая секция.
10. Параллельная обработка данных. Состояние гонки.
11. Параллельная обработка данных. Разделяемая память.
12. Параллельная обработка данных. Условная переменная.
13. Транзакционная память. Сети Петри.