МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ В.Ф. УТКИНА»

Кафедра «Вычислительная и прикладная математика»

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ
«Анализ алгоритмов»

Направление подготовки
09.03.04 «Программная инженерия»

Направленность (профиль) подготовки
«Программное обеспечение систем искусственного интеллекта»

Уровень подготовки – бакалавриат

Квалификация выпускника – бакалавр

Форма обучения – очная

Срок обучения – 4 года

Рязань

1. ОБЩИЕ ПОЛОЖЕНИЯ

Оценочные материалы – это совокупность учебно-методических материалов и процедур, предназначенных для оценки качества освоения обучающимися данной дисциплины как части основной образовательной программы.

Цель – оценить соответствие знаний, умений и уровня приобретенных компетенций, обучающихся целям и требованиям основной образовательной программы в ходе проведения текущего контроля и промежуточной аттестации.

Основная задача – обеспечить оценку уровня сформированности компетенций и индикаторов их достижения, приобретаемых обучающимся в соответствии с этими требованиями.

Контроль знаний обучающихся проводится в форме текущего контроля и промежуточной аттестации.

Текущий контроль успеваемости и промежуточная аттестация проводятся с целью определения степени усвоения учебного материала, своевременного выявления и устранения недостатков в подготовке обучающихся, организации работы обучающихся в ходе учебных занятий и оказания им индивидуальной помощи.

К контролю текущей успеваемости относятся проверка знаний, умений и навыков обучающихся на практических занятиях по результатам выполнения и защиты обучающимися индивидуальных заданий, по результатам выполнения контрольных работ и тестов, по результатам проверки качества конспектов лекций и иных материалов.

В качестве оценочных средств на протяжении семестра используется устные и письменные ответы студентов на индивидуальные вопросы, письменное тестирование по теоретическим разделам курса, реферат. Дополнительным средством оценки знаний и умений студентов является отчет о выполнении практических заданий и его защита.

По итогам курса обучающиеся сдают зачет.

1. Перечень компетенций с указанием этапов их формирования

При освоении дисциплины формируются следующие компетенции: ПК-3 (индикатор ПК-3.2).

Указанные компетенции формируются в соответствии со следующими этапами:

– формирование и развитие теоретических знаний, предусмотренных указанными компетенциями (лекционные занятия, самостоятельная работа студентов);

– приобретение и развитие практических умений предусмотренных компетенциями (практические занятия, самостоятельная работа студентов);

– закрепление теоретических знаний, умений и практических навыков, предусмотренных компетенциями, в ходе решения конкретных задач на занятиях, выполнения индивидуальных заданий на практических занятиях и их защиты, а так же в процессе сдачи зачета.

2. Показатели и критерии оценивания компетенций (*результатов*) на различных этапах их формирования, описание шкал оценивания

Сформированность каждой компетенции в рамках освоения данной дисциплины оценивается по трехуровневой шкале:

– пороговый уровень является обязательным для всех обучающихся по завершении освоения дисциплины;

– продвинутый уровень характеризуется превышением минимальных характеристик сформированности компетенций по завершении освоения дисциплины;

– эталонный уровень характеризуется максимально возможной выраженностью компетенций и является важным качественным ориентиром для самосовершенствования.

При достаточном качестве освоения более 80% приведенных знаний, умений и навыков

преподаватель оценивает освоение данной компетенции в рамках настоящей дисциплины на эталонном уровне, при освоении более 60% приведенных знаний, умений и навыков – на продвинутом, при освоении более 40% приведенных знаний умений и навыков – на пороговом уровне. При освоении менее 40% приведенных знаний, умений и навыков компетенция в рамках настоящей дисциплины считается неосвоенной.

Уровень сформированности каждой компетенции на различных этапах ее формирования в процессе освоения данной дисциплины оценивается в ходе текущего контроля успеваемости и представлено различными видами оценочных средств.

Оценке сформированности в рамках данной дисциплины подлежат компетенции/индикаторы:

Показатели достижения планируемых результатов обучения и критерии их оценивания на разных уровнях формирования компетенций приведены в таблице 1.

Таблица 1. Показатели достижения индикаторов компетенции

1	2	3	4
Компетенция: код по ФГОС 3++, формулировка	**Индикаторы**	**Этап**	**Наименование оценочного средства**
ПК-3 (09.03.04/02 Программное обеспечение систем искусственного интеллекта) Способен разрабатывать и тестировать программные компоненты решения задач в системах ИИ	**ПК-3.2 Разрабатывает приложения систем искусственного интеллекта** ЗНАТЬ - знает современные языки программирования, библиотеки и программные платформы для функционального, логического, объектно-ориентированного программирования приложений систем искусственного интеллекта (Python. R, C++, C#) УМЕТЬ - разрабатывать программные приложения систем искусственного интеллекта, с использованием современных языков программирования, библиотек и программных платформ функционального, логического, объектно-ориентированного программирования (Python. R, C++, C#) ВЛАДЕТЬ - основными принципами разработки приложений систем искусственного интеллекта	1	Зачет

Преподавателем оценивается содержательная сторона и качество материалов, приведенных в отчетах студента по практическим занятиям. Кроме того, преподавателем учитываются ответы студента на вопросы по соответствующим видам занятий при текущем контроле:

– контрольные опросы;

– задания для практических занятий.

Принимается во внимание **знания** обучающимися:

- основных программных платформ и компонентов систем искусственного интеллекта: механизмы логического вывода (рассуждений) объяснений, приобретения знаний, интеллектуальных интерфейсов, принципы Data Ops иDey Ops;

- современных языков программирования, библиотек и программных платформ для функционального, логического, объектно-ориентированного программирования приложений систем искусственного интеллекта (Python. R, C++, C#);

- основных критериев качества систем искусственного интеллекта, методов и инструментальных средств тестирования работоспособности и качества функционирования систем искусственного интеллекта;

наличие **умений**:

- настраивать основные программные платформы и компоненты систем искусственного интеллекта: механизмов логического вывода (рассуждений), объяснений, приобретений знаний, интеллектуальных интерфейсов на особенности проблемной области, участвует в их разработке;

- разрабатывать программные приложения систем искусственного интеллекта, с использованием современных языков программирования, библиотек и программных платформ функционального, логического, объектно-ориентированного программирования (Python. R, C++, C#);

- проводить тестирование работоспособности и качества функционирования систем искусственного интеллекта и проверять выполнение требований к системам искусственного интеллекта со стороны пользователя;

обладание навыками**:**

- настройки программного обеспечения и компонентов систем искусственного интеллекта;

- основными принципами разработки приложений систем искусственного интеллекта;

- методологией тестирования систем искусственного интеллекта.

Критерии оценивания уровня сформированности компетенции в процессе выполнения практических работ:

41%-60% правильных ответов соответствует пороговому уровню сформированности компетенции на данном этапе ее формирования;

61%-80% правильных ответов соответствует продвинутому уровню сформированности компетенции на данном этапе ее формирования;

81%-100% правильных ответов соответствует эталонному уровню сформированности компетенции на данном этапе ее формирования.

Сформированность уровня компетенций не ниже порогового является основанием для допуска обучающегося к промежуточной аттестации по данной дисциплине.

Формой промежуточной аттестации по данной дисциплине является зачет, оцениваемый по принятой в ФГБОУ ВО «РГРТУ» системе: «зачтено» и «не зачтено».

Критерии оценивания промежуточной аттестации представлены в таблице.

Шкала оценивания	**Критерии оценивания**
«зачтено»	**оценки «зачтено»** заслуживает обучающийся,

	продемонстрировавший полное знание материала изученной дисциплины, усвоивший основную литературу, рекомендованную рабочей программой дисциплины; выполнивший все практические задания; показавший систематический характер знаний по дисциплине, ответивший на все вопросы билета или допустивший погрешность в ответе вопросы, но обладающий необходимыми знаниями для их устранения под руководством преподавателя;
«не зачтено»	**оценки «не зачтено»** заслуживает обучающийся, не выполнивший практические задания, продемонстрировавший серьезные пробелы в знаниях основного материала изученной дисциплины, не ответивший на все вопросы билета и дополнительные вопросы. Оценка «не зачтено» ставится обучающимся, которые не могут продолжить обучение по образовательной программе без дополнительных занятий по соответствующей дисциплине (формирования и развития компетенций, закрепленных за данной дисциплиной).

3. Типовые контрольные задания или иные материалы

ФОС по дисциплине содержит следующие оценочные средства, позволяющие оценить знания, умения и уровень приобретенных компетенций при текущем контроле и промежуточной аттестации, разбитые по модулям дисциплины:

- перечни вопросов на зачет;
- макеты билетов к зачету.

Средства для оценки различных уровней формирования компетенций по категориям «знать», «уметь», «владеть» обеспечивают реализацию основных принципов контроля, таких, как объективность и независимость, практико-ориентированность, междисциплинарность.

С учетом этого, контрольные вопросы (задания, задачи,) входящие в ФОС, для различных категорий и уровней освоения компетенций имеют следующий вид:

Уровень ЗНАТЬ

Дескрипторы	Пример задания из оценочного средства
современные языки программирования, библиотеки и программные платформы для функционального, логического, объектно-ориентированного программирования приложений систем ИИ (Python, R, C++, C#)	1. Описать различия между рекурсивной и не рекурсивной реализациями алгоритма поиска расстояния Левенштейна. Указать на особенности емкостной и временной сложности. 2. Дать классификацию алгоритмов по трудоемкости. Указать, что требуется для задания модели вычислений. 3. Дать классификацию видов параллелизма. Сформулировать закон Амдала. Перечислить факторы, ограничивающие выигрыш от распараллеливания. 4. Дать определение графовых моделей алгоритма: информационного графа, графа управления, информационной и операционной истории. Указать, для которой модели невозможно ветвление.

Уровень УМЕТЬ

Дескрипторы	Пример задания из оценочного средства
разрабатывать программные	1. Провести сравнение по трудоёмкости алгоритмов

приложения СИИ с использованием современных языков программирования, библиотек и программных платформ для функционального, логического, объектно-ориентированного программирования (Python, R, C++, C#)	сортировки пузырьком и на основании бинарного дерева. 2. Описать стандартный алгоритм умножения матриц четырьмя графовыми моделями. 3. Разработать схему параллельного алгоритма нахождения расстояния Левенштейна. Указать ограничения, проистекающие из зависимости фрагментов алгоритма по данным. 4. Разработать алгоритм нахождения слова в словаре, предполагающий сегментированное хранение словаря и учитывающий частоту употребления первой буквы слова. 5. Написать регулярное выражение, позволяющее выделять из текста электронные адреса, и составить схему состояний конечного автомата, допускающего тот же поиск. 6. Оценить согласно закону Амдала максимально достижимое ускорение на 4 процессорах при 20 % кода, не допускающего параллельного выполнения. Приняв за константы затраты за диспетчеризацию, время выполнения линейного участка кода и время последовательного выполнения распараллеливаемого участка кода, дать оценку времени выполнения кода в параллельном режиме на 4 процессорах.

Перечни вопросов к зачету

1. Написать регулярное выражение, позволяющее выделять из текста подстроку согласно варианту. Составить схему состояний конечного автомата, допускающего тот же поиск. Разработать программу, составляющую конечный автомат для поиска введённой подстроки в строке. Разработать программу, осуществляющую поиск подстроки в строке на основании сформированного конечного автомата.

Примеры вариантов:

*Вариант №1.*Поиск электронных адресов.

*Вариант №2.*Поиск телефонных номеров.

2. Реализовать рекурсивно и не рекурсивно алгоритм, согласно варианту. Провести оценку реализаций алгоритма по трудоёмкости и затратам по памяти, учитывая затраты на рекурсию. Разработать программное обеспечение, реализующее обе версии алгоритма и позволяющее провести замеры времени выполнения каждой реализации и их затраты по памяти. Провести эксперименты, на основании замеров и сравнительного анализа двух реализаций сделать вывод об их эффективности.

При решении задачи использовать элементы программного кода для замеров времени работы реализаций алгоритмов, созданные в лабораторной работе 2.

Примеры вариантов:

*Вариант №1.*Умножение матрицы на основании метода декомпозиции задачи, с делением матрицы на части, шаг останова рекурсии – тривиальные размеры текущих участков матриц.

*Вариант №2.*Нахождение факториала от целого числа N. При переполнении разрядной сетки выдавать ответ 0.

Перечень лабораторных работ

ЛР1.1 Метод динамического программирования. Линейная и рекурсивная реализации алгоритмов определения расстояния Левенштейна и расстояния Дамерау-Левенштейна между строками и их анализ.

Цель работы: изучение и применение метода динамического программирования на материале алгоритмов Левенштейна и Дамерау-Левенштейна; получение практических навыков реализации указанных алгоритмов.

Задачи работы: изучение алгоритмов Левенштейна и Дамерау-Левенштейна нахождения расстояния между строками; реализация двух алгоритмов в матричной версии и одного из алгоритмов в рекурсивной версии; сравнение рекурсивной и не рекурсивных реализаций по затрачиваемым ресурсам (времени и памяти); экспериментальное подтверждение различий в эффективности рекурсивной и не рекурсивных реализаций при помощи разработанного программного обеспечения на материале замеров времени выполнения каждого алгоритма на варьирующихся длинах строк; описание и обоснование полученных результатов в отчете о выполненной лабораторной работе.

ЛР 1.2 Реализация и анализ традиционного алгоритма умножения матриц, эффективного алгоритма Винограда, оптимизированного алгоритма Винограда. Теоретические и экспериментальные оценки трудоёмкости и ёмкости по памяти.

Цель работы: получение практических навыков оценки трудоёмкости алгоритмов умножения матриц, получение практических навыков реализации алгоритмов умножения матриц с замером времени выполнения.

Задачи работы: изучение трудоёмкостей алгоритмов умножения матриц и получение их теоретических оценок на основании анализа количества выполняемых операций в выбранной модели оценки трудоемкости; получение навыка оптимизации алгоритма с целью снижения трудоёмкости его выполнения на примере решения задачи умножения матриц; программная реализация стандартного алгоритма умножения, алгоритма Винограда и оптимизированного по трудоемкости студентом алгоритма Винограда (должны быть применены минимум 3 типа улучшений; эффект от оптимизации должен быть описан в терминах трудоемкости) с функциональностью замера времени выполнения реализации каждого алгоритма; экспериментальное подтверждение при помощи разработанного программного обеспечения оценок трудоёмкости на материале замеров времени выполнения реализаций трех алгоритмов на варьирующихся размерах матриц (в т.ч. для лучшего и худшего случаев для алгоритма Винограда); описание и обоснование полученных результатов в отчете о выполненной лабораторной работе.

ЛР 1.3 Реализация и анализ нерекурсивных алгоритмов сортировки. Теоретические и экспериментальные оценки трудоёмкости.

Цель работы: получение практических навыков оценки трудоёмкости нерекурсивных алгоритмов на материале алгоритмов сортировки; получение практических навыков реализации не рекурсивных алгоритмов сортировки с подсчётом количества выполненных операций и с замером времени выполнения.

Задачи работы: изучение трудоёмкостей не рекурсивных алгоритмов сортировки массива и получение теоретических оценок трех алгоритмов на основании анализа количества выполняемых операций в выбранной модели оценки трудоемкости; разработка программного обеспечения, реализующего не рекурсивные алгоритмы сортировкис замером времени выполнения каждого алгоритма; экспериментальное подтверждение при помощи разработанного программного обеспечения оценок трудоёмкости на материале замеров времени выполнения трех алгоритмов на варьирующихся размерах массивов для лучшего, худшего и произвольного случаев (которые могут разниться для различных алгоритмов); описание и обоснование полученных результатов в отчете о выполненной лабораторной работе.

ЛР 1.4 Реализация алгоритмов поиска подстроки в строке.

Цель работы: получение практических навыков реализации алгоритмов поиска подстроки в строке.

Задачи работы: изучение алгоритмов поиска подстроки в строке (стандартного, Кнута-Морриса-Пратта, Бойера-Мура) и их трудоёмкостей в терминах затраченных сравнений в лучшем и худшем случаях; разработка программного обеспечения, реализующего указанные алгоритмы; описание алгоритмов поиска подстроки в строке, включая процедуры заполнения вспомогательных массивов, с примерами и тестами, а также описание реализованного программного обеспечения и полученных результатов в отчете о выполненной лабораторной работе.

ЛР 2.1 Графовые модели алгоритмов. Потоковые алгоритмы.

Цель работы: получение практических навыков описания алгоритма графовыми

моделями и анализа алгоритма на материале графовых моделей на предмет выделения зависимости по данным и выделения ресурса параллелизма, получение практических навыков реализации потоковых алгоритмов.

Задачи работы: описание алгоритма четырьмя графовыми моделями: графом управления, информационным графом, операционной историей и информационной историей; анализ алгоритма на предмет выделения зависимостей по данным, информационно независимых участков, ресурса параллелизма (конечного, массового); разработка и реализация потокового алгоритма; описание и обоснование полученных результатов в отчете о выполненной лабораторной работе.

ЛР 2.2 Параллельная реализация алгоритмов умножения матриц (алгоритмы традиционный, Винограда), её анализ, экспериментальные оценки времени выполнения.

Цель работы: получение практических навыков оценки ресурса параллелизма алгоритмов умножения матриц; получение практических навыков оценки трудоёмкости параллельных алгоритмов умножения матриц; получение практических навыков реализации параллельных алгоритмов умножения матриц с замером времени выполнения для параллельных и последовательных реализаций алгоритмов.

Задачи работы: изучение подходов к выделению ресурса параллелизма и к распараллеливанию алгоритмов на материале алгоритмов умножения матриц (стандартного и Винограда); изучение трудоёмкости для фрагментов алгоритмов умножения матриц (подвергающихся и не подвергающихся распараллеливанию) и получение её оценок для алгоритма в целом; получение навыка анализа информационной зависимости фрагментов алгоритма и корректной адаптации разработанного параллельного алгоритма к этим зависимостям;получение навыка параллельной реализации алгоритма с целью снижения трудоёмкости выполнения в сравнении с последовательным алгоритмом на примере решения задачи умножения матриц; экспериментальное подтверждение при помощи разработанного программного обеспечения оценок трудоёмкости на материале замеров времени выполнения алгоритмов на варьирующихся размерах матриц (для лучшего и худшего случаев) при выполнении параллельной реализации с различным количеством потоков и при выполнении последовательной версии алгоритмов; описание и обоснование полученных результатов в отчете о выполненной лабораторной работе.

ЛР 2.3 Моделирование конвейера в параллельном режиме: одна лента конвейера симулируется одним потоком

Цель работы: получение практических навыков оценки ресурса параллелизма алгоритмов при переносе на конвейерную схему вычислений; получение практических навыков оценки трудоёмкости потоковых параллельных алгоритмов; получение практических навыков реализации параллельных алгоритмов.

Задачи работы: изучение подходов к выделению ресурса параллелизма и к распараллеливанию алгоритмов на материале выбранной задачи (по вариантам; например, блочное шифрование); получение навыка анализа информационной зависимости фрагментов алгоритма и корректной адаптации разработанного параллельного алгоритма к этим зависимостям; разработка алгоритма, моделирующего работу конвейера, с учетом

особенностей синхронизации между потоками; получение навыка параллельной реализации поточного алгоритма как конвейера с целью снижения трудоёмкости выполнения в сравнении с последовательным алгоритмом; экспериментальное подтверждение при помощи разработанного программного обеспечения эффективности параллельной реализации по сравнению с однопоточной; описание и обоснование полученных результатов в отчете о выполненной лабораторной работе.

ЛР 3.1 Реализация и параметризация муравьиного алгоритма.

Цель работы: получение практических навыков реализации эвристических алгоритмов, позволяющих решить задачу оптимизации с приемлемым качеством и за меньшее время; получение практических навыков параметризации алгоритма на примере муравьиного алгоритма и изучения влияния настроечных параметров алгоритма на результирующие значения (в задаче коммивояжера – на длину пути).

Задачи работы: изучение эволюционных алгоритмов, в частности, муравьиного алгоритма; применение муравьиного алгоритма к решению задачи коммивояжера; разработка программного обеспечения, реализующего метод перебора и муравьиный алгоритм применительно к решению задачи коммивояжера; экспериментальное подтверждение различия в трудоемкостях реализованных алгоритмов при помощи разработанного программного обеспечения на материале замеров времени выполнения алгоритмов; параметризация муравьиного алгоритма и изучение влияния настроечных параметров алгоритма на результирующие значения (в задаче коммивояжера – на длину пути) при помощи разработанного программного обеспечения; описание и обоснование полученных результатов в отчете о выполненной лабораторной работе.