МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

УТВЕРЖДАЮ
Декан факультета
информационных технологий
_____ А.Ю. Филиппович
"01" сентября 2020 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Программирование»

Направление подготовки
09.03.03 «Прикладная информатика»

Образовательная программа (профиль подготовки)
«Большие и открытые данные»

Квалификация (степень) выпускника
бакалавр

Форма обучения
Очная

Год приема - 2020

Москва 2020 г.

Программа составлена в соответствии с Федеральным государственным образовательным стандартом высшего образования по направлению подготовки бакалавров 09.03.03 «Прикладная информатика»

Программу составил:

старший преподаватель /В.М. Чернова/

Программа утверждена на заседании кафедры "Прикладная информатика" «28» августа 2020 г., протокол № 1

Заведующий кафедрой
профессор, к. э. н. /С.В. Суворов/

1. Цели освоения дисциплины

Основными целями изучения дисциплины «Программирование» в соответствии с ОПОП является формирование у студентов знаний о современных принципах, методах и средствах программирования на примере программирования прикладных задач на языке Java, подготовка студентов к деятельности в соответствии с квалификационной характеристикой бакалавра по направлению, в том числе формирование умений по выявлению необходимых совершенствований и разработке нового программного обеспечения.

К основным задачам освоения дисциплины «Программирование» следует отнести освоение методологии, анализа и выбора принципов и методов программирования прикладных задач на языке программирования высокого уровня.

2. Место дисциплины в структуре ООП бакалавриата

Дисциплина «Программирование» относится к числу учебных дисциплин базовой части базового цикла (Б.1.1) основной образовательной программы бакалавриата (Б.1.1.15).

Дисциплина «Программирование» взаимосвязана логически и содержательно-методически со следующими дисциплинами и практиками ООП:

В базовой части базового цикла (Б.1.1):

– Теоретические основы информатики;

– Вычислительные сети и системы;

– Математический анализ;

– Дискретная математика.

3. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенные с планируемыми результатами освоения образовательной программы.

В результате освоения дисциплины (модуля) у обучающихся формируются следующие компетенции и должны быть достигнуты следующие результаты обучения как этап формирования соответствующих компетенций:

Код компетенции	В результате освоения образовательной программы	Перечень планируемых результатов обучения по дисциплине

	обучающийся должен обладать	
ОПК-3	способностью к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям	**знать:** - общие принципы разработки и реализации программных алгоритмов, модернизации, модификации, инсталляции и сопровождения современного программного обеспечения - основные элементы языков программирования высокого уровня; - основные технологии программирования и приемы процедурного и объектно-ориентированного программирования; - методы объектно-ориентированного проектирования и программирования (на языке Java); **уметь:** - формализовать предметную область и разрабатывать структуру программы - работать в среде объектно-ориентированного программирования; - разрабатывать отдельные компоненты и дополнения программного обеспечения применительно к решаемым задачам, инсталлировать, сопровождать и администрировать эксплуатируемые программные средства - использовать полученные навыки для решения прикладных и системных задач; **владеть:** - технологиями программирования в объектно-ориентированных программных и операционных средах; - средствами языка программирования высокого уровня (Java) для реализации типовых алгоритмов обработки данных

4. Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 5 зачетных единиц – всего 180 академических часов, из которых 54 часа – аудиторные занятия и 126 часов самостоятельной работы студента. Аудиторные занятия включают 18 часов лекций и 36 часов лабораторных работ.

Дисциплина читается в 1 семестре. Лекции составляют 1 час в неделю (18 часов), лабораторные занятия – 2 часа в неделю (36 часов).

Форма контроля по результатам обучения – экзамен.

Структура и содержание разделов дисциплины «Программирование» по срокам и видам работы отражены в приложении 1.

Содержание разделов дисциплины

Введение

Предмет, задачи и содержание дисциплины. Структура курса, его место и роль в подготовке специалиста, связь с другими дисциплинами. Классификация программ по типу исполнения (компилируемые, интерпретируемые, исполняемые на виртуальных машинах). Особенности языка Java: интерпретируемость, независимость от платформы, мобильность. Виртуальная машина Java. JIT-компиляция.

Среда разработки

Средства разработки Java-приложений. Интегрированные среды разработки. Пример создания простейшей программы на языке Java в среде NetBeans, её компиляция в байт-код и запуск. Базовые пакеты и классы Java. Проекты. Пакеты. Уровни видимости классов. Импорт классов. Сборка мусора в языке Java. Структура проекта NetBeans. Создание в NetBeans простейшего приложения Java. Компиляция файлов проекта и запуск приложения. Jar-файлы. Отладка приложений в среде NetBeans. Документирование исходного кода в Java.

Типы данных языка программирования Java

Идентификаторы. Переменные и типы. Примитивные и ссылочные типы. Встроенные типы данных. Способы задания литералов различных типов. Управляющие последовательности. Символы Unicode. Специальные символы. Булевый (логический) тип. Основные операторы для работы с логическими величинами. Целые типы, переменные, константы. Основные операторы для работы с целочисленными величинами. Вещественные типы и класс Math. Работа со ссылочными переменными.

Операторы языка программирования Java

Оператор присваивания. Порядок действий (приоритет операторов). Приведение типов. Неявное приведение типов. Явное приведение, автоматическое расширение типов и instanceof. Строковое приведение. Правила явного и автоматического преобразования типа при работе с числовыми величинами. Метки в языке Java. Управляющие инструкции. Операторы и блоки.

Составной оператор. Условный оператор if. Оператор выбора switch. Условное выражение ...?... :

Операторы инкремента ++ и декремента -- . Оператор цикла for. Ошибки при использовании вещественного счетчика цикла. Эффективная организация циклов при вычислениях в формате с плавающей точкой. Особенности целочисленных вычислений. Организация циклов, приоритет операторов и арифметическое переполнение. Цикл с предусловием. Оператор цикла while. Цикл с постусловием. Оператор цикла do...while. Операторы прерывания continue, break, return, System.exit. Арифметические операторы. Логические операторы. Операторы сравнения. Побитовые операторы. Создание и работа с массивами. Одномерные массивы. Многомерные массивы. Присваивание и сравнение массивов. Коллекции, списки, итераторы. Перебор в цикле элементов коллекций. Оператор цикла for-each.

Основы объектно-ориентированного программирования в языке программирования Java

Сравнение парадигм процедурного программирования и объектно-ориентированного программирования. Основополагающие принципы объектно-ориентированного программирования: инкапсуляция, наследование и полиморфизм. Средства реализации инкапсуляции в языке Java. Классы языка программирования Java. Методы и поля. Конструкторы. Конструктор по умолчанию. Зарезервированные слова super и this. Блоки инициализации. Конструкторы в наследуемых классах. Порядок вызова конструкторов. Удаление неиспользуемых объектов и метод finalize. Проблема деструкторов для сложно устроенных объектов. Оболочечные классы. Упаковка (boxing) и распаковка (unboxing). Модификаторы уровня доступа default, public, protected, private. Управление наследованием в языке Java. Наследование. Суперклассы и подклассы. Переопределение методов. Наследование и правила видимости. Зарезервированное слово super. Проблемы множественного наследования классов. Интерфейсы. Отличия интерфейсов от классов. Проблемы наследования интерфейсов. Композиция как альтернатива множественному наследованию. Интерфейсы как средство реализации множественного наследования. Средства реализации полиморфизма в языке Java. Правила совместимости ссылочных типов как основа использования полиморфного кода. Приведение и проверка типов. Методы. Значения параметров. Применение методов для ограничения доступа. Сигнатура метода. Перегрузка методов. Статическое и динамическое связывание методов. Полиморфизм. Иерархия классов Java. Коренной класс Object и его методы. Функции. Модификаторы. Передача примитивных типов в

функции. Локальные и глобальные переменные. Модификаторы доступа и правила видимости. Ссылка this. Передача ссылочных типов в функции. Проблема изменения ссылки внутри подпрограммы. Статические (static) вложенные классы и интерфейсы. Внутренние (inner) классы. Локальные (local) классы. Анонимные (anonimous) классы и обработчики событий. Анонимные (anonimous) классы и слушатели событий (listeners). Статические члены класса. Абстрактные классы и методы.

Профилирование Java-приложений

Принцип «презумпции виновности» и тестирование классов языка Java. Профилирование Java-приложений. Настройка профилировщика «на лету». Точки профилирования. Профилирование использования памяти классами Java.

Исключительные ситуации языка программирования Java

Исключительные ситуации. Обработка исключительных ситуаций. Иерархия исключительных ситуаций. Контролируемые и неконтролируемые исключения. Объявление типа исключительной ситуации и оператор throw. Объявление метода, который может возбуждать исключительную ситуацию. Зарезервированное слово throws. Создание новых типов исключений. Оператор throw. Условие throws. Операторы try, catch и finally.

События языка программирования Java

Понятие события. Типы событий. Иерархия классов событий. События в Java. Модель делегирования событий. Интерфейсы блоков прослушивания событий. Способы реализации блока прослушивания. Использование внутренних классов, классов-адаптеров для упрощения обработки событий. Типы-перечисления (enum).

Работа с классами языка программирования Java

Работа с датами и временем. Класс GregorianCalendar. Работа со строками в Java. Строки как объекты. Классы String, StringBuffer и StringBuilder. Операции со строками. Сравнение строк. Поиск и извлечение строк. Посимвольная обработка строк. Пословное реверсирование строки. Преобразование всех символов строки к верхнему или нижнему регистру. Классы для работы с текстом. Поиск по образцу с помощью регулярных выражений. Специальные символы регулярных выражений. Классы Java Matcher и Pattern, используемые для работы с регулярными выражениями. Нахождение соответствий в тексте с

помощью регулярных выражений. Нахождение всех вхождений по образцу. Замена текста, отвечающего соответствию.

Основы ввода/вывода в языке программирования Java

Основные группы классов и интерфейсов пакета java.io. Стандартные потоки ввода-вывода. Организация ввода и вывода данных. Класс Scanner. Ввод/вывод в Java. Фильтрованные потоки. Ввод/вывод в Java. Буферизированные потоки. Ввод/вывод в Java. Канальные потоки. Ввод/вывод в Java. Синхронизация потоков данных. Работа с файлами и папками с помощью объектов типа File.

5. Образовательные технологии

Методика преподавания дисциплины «Программирование» и реализация компетентностного подхода в изложении и восприятии материала предусматривает использование следующих активных и интерактивных форм проведения групповых, индивидуальных, аудиторных занятий в сочетании с внеаудиторной работой с целью формирования и развития профессиональных навыков обучающихся:

– подготовка к выполнению лабораторных работ в лабораториях вуза;

– обсуждение и защита рефератов по дисциплине;

– подготовка, представление и обсуждение презентаций на семинарских занятиях;

– организация и проведение текущего контроля знаний студентов в форме бланкового тестирования;

– проведение интерактивных занятий по процедуре подготовки к интернет-тестированию;

– использование интерактивных форм текущего контроля в форме аудиторного и внеаудиторного интернет-тестирования;

– проведение мастер-классов экспертов и специалистов по программированию.

Удельный вес занятий, проводимых в интерактивных формах, определен главной целью образовательной программы, особенностью контингента обучающихся и содержанием дисциплины «Программирование» и в целом по дисциплине составляет 30% аудиторных занятий. Занятия лекционного типа составляют 33% от объема аудиторных занятий.

6. Оценочные средства для текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины и учебно-методическое обеспечение самостоятельной работы студентов

В процессе обучения используются следующие оценочные формы самостоятельной работы студентов, оценочные средства текущего контроля успеваемости и промежуточных аттестаций:

В первом семестре

- реферат по теме: «Особенности программирования на языке Java» (индивидуально для каждого обучающегося);
- подготовка к выполнению лабораторных работ и их защита.

Оценочные средства текущего контроля успеваемости включают контрольные вопросы и задания в форме бланкового и (или) компьютерного тестирования, для контроля освоения обучающимися разделов дисциплины, защита рефератов.

Образцы тестовых заданий, контрольных вопросов и заданий для проведения текущего контроля приведены в приложении 2.

6.1. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине (модулю)

6.1.1. Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы.

В результате освоения дисциплины (модуля) формируются следующие компетенции:

Код компетенции	В результате освоения образовательной программы обучающийся должен обладать
ОПК-3	способностью к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям

В процессе освоения образовательной программы данные компетенции, в том числе их отдельные компоненты, формируются поэтапно в ходе освоения

обучающимися дисциплин (модулей), практик в соответствии с учебным планом и календарным графиком учебного процесса.

6.1.2. Описание показателей и критериев оценивания компетенций, формируемых по итогам освоения дисциплины (модуля), описание шкал оценивания

Показателем оценивания компетенций на различных этапах их формирования является достижение обучающимися планируемых результатов обучения по дисциплине (модулю).

ОПК-3 способностью к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям				
Показатель	**Критерии оценивания**			
	2	**3**	**4**	**5**
знать: общие принципы разработки и реализации программных алгоритмов, модернизации, модификации, инсталляции и сопровождения современного программного обеспечения; основные элементы языков программирования высокого уровня; основные технологии программирования и приемы процедурного и объектно-ориентированного программирования; методы объектно-	Обучающийся демонстрирует полное отсутствие или недостаточное соответствие следующих знаний: общие принципы разработки, модернизации, модификации, инсталляции и сопровождения современного программного обеспечения, элементы языков высокого уровня и приемы и технологии ООП	Обучающийся демонстрирует неполное соответствие следующих знаний: общие принципы разработки, модернизации, модификации, инсталляции и сопровождения современного программного обеспечения, элементы языков высокого уровня и приемы и технологии ООП. Допускаются значительные ошибки, проявляется недостаточность знаний по ряду показателей, обучающийся испытывает значительные затруднения при	Обучающийся демонстрирует частичное соответствие следующих знаний: общие принципы разработки, модернизации, модификации, инсталляции и сопровождения современного программного обеспечения, элементы языков высокого уровня и приемы и технологии ООП, но допускаются незначительные ошибки, неточности, затруднения при операциях по работе с программным обеспечением.	Обучающийся демонстрирует полное соответствие следующих знаний: общие принципы разработки, модернизации, модификации, инсталляции и сопровождения современного программного обеспечения, элементы языков высокого уровня и приемы и технологии ООП, свободно оперирует приобретенными знаниями.

ориентированного проектирования и программирования (на языке Java)		операровании знаниями при их переносе на новые ситуации.		
уметь: формализовать предметную область и разрабатывать структуру программы работать в среде объектно-ориентированного программирования; разрабатывать отдельные компоненты и дополнения программного обеспечения применительно к решаемым задачам, инсталлировать, сопровождать и администрировать эксплуатируемые программные средства; использовать полученные навыки для решения прикладных и системных задач	Обучающийся не умеет или в недостаточной степени умеет разрабатывать отдельные компоненты и дополнения программного обеспечения применительно к решаемым задачам, инсталлировать, сопровождать и администрировать эксплуатируемые программные средства.	Обучающийся демонстрирует неполное соответствие следующих умений: разрабатывать отдельные компоненты и дополнения программного обеспечения применительно к решаемым задачам, инсталлировать, сопровождать и администрировать эксплуатируемые программные средства. Допускаются значительные ошибки, проявляется недостаточность умений, по ряду показателей, обучающийся испытывает значительные затруднения при операровании умениями при их переносе на новые ситуации.	Обучающийся демонстрирует частичное соответствие следующих умений: разрабатывать отдельные компоненты и дополнения программного обеспечения применительно к решаемым задачам, инсталлировать, сопровождать и администрировать эксплуатируемые программные средства. Умения освоены, но допускаются незначительные ошибки, неточности, затруднения при операциях по работе с программным обеспечением, переносе умений на новые, нестандартные ситуации.	Обучающийся демонстрирует полное соответствие следующих умений: разрабатывать отдельные компоненты и дополнения программного обеспечения применительно к решаемым задачам, инсталлировать, сопровождать и администрировать эксплуатируемые программные средства. Свободно оперирует приобретенными умениями, применяет их в ситуациях повышенной сложности.
владеть: технологиями программирования в объектно-ориентированных программных и операционных средах; средствами языка программирования высокого уровня	Обучающийся не владеет или в недостаточной степени владеет методами и средствами программирования прикладных задач на языке программирования высокого уровня.	Обучающийся владеет методами и средствами программирования прикладных задач на языке программирования высокого уровня в неполном объеме, допускаются значительные ошибки,	Обучающийся частично владеет методами и средствами программирования прикладных задач на языке программирования высокого уровня, навыки освоены, но допускаются незначительные	Обучающийся в полном объеме владеет методами и средствами программирования прикладных задач на языке программирования высокого уровня, свободно применяет полученные

(Java) для реализации типовых алгоритмов обработки данных.		проявляется недостаточность владения навыками по ряду показателей, Обучающийся испытывает значительные затруднения при применении навыков в новых ситуациях.	ошибки, неточности, затруднения при аналитических операциях, переносе умений на новые, нестандартные ситуации.	навыки в ситуациях повышенной сложности.

Шкалы оценивания результатов промежуточной аттестации и их описание:

Форма промежуточной аттестации: экзамен

Промежуточная аттестация обучающихся в форме экзамена проводится по результатам выполнения всех видов учебной работы, предусмотренных учебным планом по данной дисциплине (модулю), при этом учитываются результаты текущего контроля успеваемости в течение семестра. Оценка степени достижения обучающимися планируемых результатов обучения по дисциплине (модулю) проводится преподавателем, ведущим занятия по дисциплине (модулю) методом экспертной оценки. По итогам промежуточной аттестации по дисциплине (модулю) выставляется оценка «отлично», «хорошо», «удовлетворительно» или «неудовлетворительно».

К промежуточной аттестации допускаются только студенты, выполнившие все виды учебной работы, предусмотренные рабочей программой по дисциплине «Программирование»: выполнили лабораторные работы, выступили с докладом по теме реферата.

Шкала оценивания	**Описание**
Отлично	Выполнены все виды учебной работы, предусмотренные учебным планом. Студент демонстрирует соответствие знаний, умений, навыков приведенным в таблицах показателей, оперирует приобретенными знаниями, умениями, навыками, применяет их в ситуациях повышенной сложности. При этом могут быть допущены незначительные ошибки, неточности, затруднения при аналитических операциях, переносе знаний и умений на новые, нестандартные ситуации.

Хорошо	Выполнены все виды учебной работы, предусмотренные учебным планом. Студент демонстрирует неполное, правильное соответствие знаний, умений, навыков приведенным в таблицах показателей, либо если при этом были допущены 2-3 несущественные ошибки.
Удовлетворительно	Выполнены все виды учебной работы, предусмотренные учебным планом. Студент демонстрирует соответствие знаний, в котором освещена основная, наиболее важная часть материала, но при этом допущена одна значительная ошибка или неточность.
Неудовлетворительно	Не выполнен один или более видов учебной работы, предусмотренных учебным планом. Студент демонстрирует неполное соответствие знаний, умений, навыков приведенным в таблицах показателей, допускаются значительные ошибки, проявляется отсутствие знаний, умений, навыков по ряду показателей, студент испытывает значительные затруднения при оперировании знаниями и умениями при их переносе на новые ситуации.

Фонды оценочных средств представлены в приложении 2 к рабочей программе.

7. Учебно-методическое и информационное обеспечение дисциплины

а) основная литература:

1. Николаев Е. И. Объектно-ориентированное программирование: учебное пособие - Ставрополь: Изд-во СКФУ, 2015. - 225 с. — Режим доступа: http://www.knigafund.ru/books/200468 — Загл. с экрана.\

2. Вязовик, Н.А. Программирование на Java [Электронный ресурс] : учеб. пособие — Электрон. дан. — Москва : , 2016. — 603 с. — Режим доступа: https://e.lanbook.com/book/100405. — Загл. с экрана.

б) дополнительная литература:

2. Бурмистров, А.В. Программирования на языке JAVA. Методические указания к лабораторным работам [Электронный ресурс] : метод. указ. — Электрон. дан. — Пенза: ПензГТУ, 2014. — 150 с. — Режим доступа: https://e.lanbook.com/book/62752. — Загл. с экрана.

в) программное обеспечение и Интернет-ресурсы:

1. Язык программирования Java версии 8.1.

2. Интегрированная среда разработки IDE NetBeans 8.1.

8. Материально-техническое обеспечение дисциплины

Компьютерные классы кафедры «Прикладная информатика», оснащенные компьютерами с установленным программным обеспечением в соответствии с п. 7 рабочей программы

9. Методические рекомендации для самостоятельной работы студентов

Обучение по дисциплине «Программирование» предполагает изучение курса на аудиторных занятиях и самостоятельной работы обучающихся. Виды самостоятельной работы обучающегося, порядок и сроки ее выполнения:

– подготовка к лекциям, лабораторным работам и практическим занятиям с использованием конспекта лекций, материалов лабораторных занятий и учебно-методического и информационного обеспечения дисциплины (в течение 1-го семестра в соответствии с расписанием занятий);

– оформление отчетов по выполненным лабораторным работам и теоретическая подготовка к их сдаче (в течение 1-го семестра в соответствии с расписанием занятий).

Перечень вопросов для проведения текущего контроля и промежуточной аттестации – в соответствии с тематикой разделов дисциплины.

С целью обеспечения успешного обучения обучающийся должен готовиться к лекции, которая является важнейшей формой организации учебного процесса, поскольку:

– знакомит с новым учебным материалом;
– разъясняет учебные элементы, трудные для понимания;
– систематизирует учебный материал;
– ориентирует в учебном процессе.

Подготовка к занятиям лекционного типа заключается в следующем:

– внимательном чтении материала предыдущей лекции;
– предварительное ознакомление с темой предстоящей лекции (по тематическому плану, по информации лектора);
– ознакомление с учебным материалом по учебнику и учебным пособиям;
– уяснение места изучаемой темы в своей профессиональной подготовке;
– подготовке возможных вопросов, которые студент задаст лектору на

лекции.

Подготовка к написанию реферата исходит из того, что реферат – это самостоятельная учебно-исследовательская работа обучающегося, где автор раскрывает суть исследуемой проблемы, приводит различные точки зрения, а также собственные взгляды на нее. Содержание материала должно быть логичным, изложение материала носит проблемно-поисковый характер.

Этапы работы над рефератом.

1. Формулирование темы. Тема должна быть не только актуальной по своему значению, но и оригинальной, интересной по содержанию.

2. Подбор и изучение основных источников по теме.

3. Составление библиографии.

4. Обработка и систематизация информации.

5. Разработка плана реферата.

6. Оформление реферата в виде презентации.

7. Публичное выступление с результатами исследования на занятии.

Содержание работы должно отражать.

1. Знание современного состояния проблемы.

2. Обоснование выбранной темы.

3. Использование известных результатов и фактов.

4. Полноту цитируемой литературы, ссылки на работы ученых, занимающихся данной проблемой.

5. Актуальность поставленной проблемы.

6. Материал, подтверждающий научное, либо практическое значение в настоящее время.

Типовая структура реферата.

1. Титульный лист.

2. План (простой или развернутый).

3. Введение.

4. Основная часть.

5. Заключение.

6. Список литературы.

Реферат может рассматриваться как одна из форм контрольных работ. Объем реферата не должен превышать 10 страниц.

Представление реферата осуществляется в форме доклада с предъявлением презентации.

10. Методические рекомендации для преподавателя

Программа составлена в соответствии с Федеральным государственным образовательным стандартом высшего образования по направлению подготовки бакалавров 09.03.03 «Прикладная информатика»

Приложение 1 к
рабочей программе

Структура и содержание дисциплины «Программирование» по направлению подготовки 09.03.03 «Прикладная информатика» (бакалавр)

n/n	Раздел	Семестр	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов, и трудоемкость в часах					Виды самостоятельной работы студентов					Формы аттестации	
				Л	П/С	Лаб	СРС	КСР	К.Р.	К.П.	РГР	Реферат	К/р	Э	З
	Первый семестр														
1.1	**Введение. Особенности языка Java. Среда разработки NetBeans.**	1	1	2											
1.2	*Лабораторная работа «*Среда разработки NetBeans».	1	1			2	8								
1.3	**Типы данных языка программирования Java.**	1	2	2											
1.4	*Лабораторная работа «*Типы данных языка программирования Java и работа с ними».	1	2,3			4	12								
1.5	**Операторы языка программирования Java.**	1	3	2											
1.6	*Лабораторная работа* «Операторы языка программирования Java».	1	4,5			4	12								
1.7	**Основы объектно-**	1	4	2											

	ориентированного программирования в языке программирования Java. Выдача задания на реферат.						16					+			
1.8	*Лабораторная работа* «Классы и их структура в языке программирования Java».	1	6,7,8			8	16					+			
1.9	**Профилирование Java-приложений.**	1	5	2								+			
1.10	*Лабораторная работа* «Отладка объектно-ориентированных программ в языке программирования Java».	1	9,10			4	16					+			
1.11	**Исключительные ситуации и события языка программирования Java.**	1	6	2								+			
1.12	Лабораторная работа «Программирование исключительных ситуаций и событий языка программирования Java».	1	11,12			4	16					+			
1.13	**Работа с классами языка программирования Java.**	1	7	4								+			
1.14	*Лабораторная работа* «Применение встроенных классов в объектно-ориентированных программах в языке программирования Java».	1	13,14			6	16					+			
1.15	**Основы ввода/вывода в языке**	1	8	2								+			

	программирования Java.														
1.16	*Лабораторная работа* «Программирование ввода/вывода в языке программирования Java».	1	15-17			4	14					+			
	Форма аттестации											Один реферат		Э	
	Всего часов по дисциплине в первом семестре			18		36	126								

Приложение 2 к
рабочей программе

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
(МОСКОВСКИЙ ПОЛИТЕХ)

Направление подготовки: 09.03.03 «Прикладная информатика»

ОП (профиль): «Большие и открытые данные»

Форма обучения: очная

Вид профессиональной деятельности: (В соответствии с ФГОС ВО)

Форма обучения
Очная

Кафедра: «Прикладная информатика»

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

ПО ДИСЦИПЛИНЕ

«ПРОГРАММИРОВАНИЕ»

Состав: 1. Паспорт фонда оценочных средств

2. Описание оценочных средств:

-Показатели уровня сформированности компетенций.
-Пример тем рефератов.
-Пример комплекта разноуровневых задач и заданий.
-Пример вопросов устного опроса.

Составители:

ст.преподаватель Шульга М.В.

Москва, 2020 год

1. Паспорт фонда оценочных средств

Фонд оценочных средств включает в себя:

- Показатели уровня сформированности компетенций.
- Пример тем рефератов.
- Пример комплекта разноуровневых задач и заданий.
- Пример вопросов устного опроса.

ПОКАЗАТЕЛЬ УРОВНЯ СФОРМИРОВАННОСТИ КОМПЕТЕНЦИЙ

Программирование					
ФГОС ВО **09.03.03 «Прикладная информатика»**					
В процессе освоения данной дисциплины студент формирует и демонстрирует следующие **общепрофессиональные компетенции**:					
КОМПЕТЕНЦИИ		**Перечень компонентов**	**Технология формирования компетенций**	**Форма оценочного средства**	**Степени уровней освоения компетенций**
ИН-ДЕКС	**ФОРМУЛИРОВ-КА**				
ОПК-3	способностью к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз	**знать:** - общие принципы разработки и реализации программных алгоритмов, модернизации, модификации, инсталляции и сопровождения современного программного обеспечения - основные элементы языков программирования высокого уровня; - основные технологии программирования и приемы процедурного и объектно-ориентированного программирования; - методы объектно-ориентированного проектирования и	лекции, самостоятельная работа, лабораторные работы.	УО, РЗЗ, Р	**Базовый уровень** - студент способен разрабатывать и применять математические методы, системное и прикладное программное обеспечение для решения задач научной и проектно-технологической деятельности в стандартных учебных ситуациях. **Повышенный уровень** - студент способен разрабатывать и применять математические методы, системное и прикладное программное обеспечение для решения задач научной и проектно-технологической деятельности в практической деятельности.

	данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям	программирования (на языке Java); **уметь:** - формализовать предметную область и разрабатывать структуру программы - работать в среде объектно-ориентированного программирования; - разрабатывать отдельные компоненты и дополнения программного обеспечения применительно к решаемым задачам, инсталлировать, сопровождать и администрировать эксплуатируемые программные средства.- использовать полученные навыки для решения прикладных и системных задач; **владеть:** - технологиями программирования в объектно-ориентированных программных и операционных средах; - средствами языка программирования высокого уровня (Java) для реализации типовых алгоритмов обработки данных			

Приложение 3
к рабочей программе

Перечень оценочных средств по дисциплине «Программирование»

№ ОС	Наименование оценочного средства	Краткая характеристика оценочного средства	Представление оценочного средства в ФОС
1	Доклад (Д)	Продукт самостоятельной работы студента, представляющий собой краткое изложение в виде презентации полученных результатов теоретического анализа определенной научной (учебно-исследовательской) темы, где автор раскрывает суть исследуемой проблемы, приводит различные точки зрения, а также собственные взгляды на нее.	Темы докладов (с предоставлением презентации)
2	Устный опрос собеседование, (УО)	Средство контроля, организованное как специальная беседа педагогического работника с обучающимся на темы, связанные с изучаемой дисциплиной, и рассчитанное на выяснение объема знаний обучающегося по определенному разделу, теме, проблеме и т.п.	Вопросы по темам/разделам дисциплины
3	Доклад, сообщение (ДС)	Продукт самостоятельной работы студента, представляющий собой публичное выступление по представлению полученных результатов решения определенной учебно-практической, учебно-исследовательской или научной темы	Темы докладов, сообщений

Пример тем рефератов (ОПК-3)

1. Средства разработки Java-приложений. Интегрированные среды разработки.
2. Базовые пакеты и классы Java.
3. Правила явного и автоматического преобразования типа при работе с числовыми величинами.
4. Создание и работа с массивами.
5. Сравнение парадигм процедурного программирования и объектно-ориентированного программирования.
6. Основополагающие принципы объектно-ориентированного программирования.
7. Иерархия классов Java.
8. События в Java.
9. Поиск по образцу с помощью регулярных выражений.
10. Работа с файлами и папками с помощью объектов типа *File*.
11. Особенности программирования на языке Java (индивидуально для каждого обучающегося).

Пример комплекта разноуровневых задач и заданий для лабораторных работ

Напишите на языке Java программы расчета по двум формулам с вводом исходных данных и выводом результатов. Исходные данные представляют последовательность из пяти чисел. В программе использовать все виды циклов. Одну программу написать как консольное приложение, другую – с использованием оконного интерфейса. Результаты расчета по двум формулам должны совпасть (показать почему).

Вариант 1

$$z_1 = 2\sin^2(3\pi - 2\alpha)\cos^2(5\pi + 2\alpha)$$

$$z_2 = \frac{1}{4} - \frac{1}{4}\sin\left(\frac{5}{2}\pi - 8\alpha\right)$$

Вариант 2

$$z_1 = \cos\alpha + \sin\alpha + \cos 3\alpha + \sin 3\alpha$$

$$z_2 = 2\sqrt{2}\cos\alpha \cdot \sin\left(\frac{\pi}{4} + 2\alpha\right)$$

Вариант 3

$$z_1 = \frac{\sin 2\alpha + \sin 5\alpha - \sin 3\alpha}{\cos\alpha + 1 - 2\sin^2 2\alpha}$$

$$z_2 = 2\sin\alpha$$

Пример вопросов для экзамена

1. Особенности языка Java: интерпретируемость, независимость от платформы, мобильность. Примеры.
2. Классификация программ по типу исполнения (компилируемые, интерпретируемые, исполняемые на виртуальных машинах). Виртуальная машина Java. JIT-компиляция. Примеры.
3. Управляющие последовательности. Символы Unicode. Специальные символы. Примеры.
4. Идентификаторы. Переменные и типы. Примитивные и ссылочные типы. Примеры.
5. Работа со ссылочными переменными. Сборка мусора в языке Java. Примеры.
6. Проекты. Пакеты. Уровни видимости классов. Импорт классов. Примеры.
7. Структура проекта NetBeans. Примеры.
8. Документирование исходного кода в Java. Примеры.
9. Отладка приложений в среде NetBeans. Примеры.
10. Встроенные типы данных. Способы задания литералов различных типов.
11. Булевый (логический) тип. Основные операторы для работы с логическими величинами. Примеры.
12. Целые типы, переменные, константы. Основные операторы для работы с целочисленными величинами. Примеры.
13. Вещественные типы и класс *Math*. Примеры.
14. Оператор присваивания. Порядок действий (приоритет операторов). Примеры.
15. Приведение типов. Неявное приведение типов. Явное приведение, автоматическое расширение типов и instanceof. Строковое приведение. Примеры.
16. Правила явного и автоматического преобразования типа при работе с числовыми величинами. Примеры.
17. Метки в языке Java. Примеры.
18. Управляющие инструкции. Операторы и блоки. Составной оператор. Примеры.
19. Условный оператор *if* . Примеры.
20. Оператор выбора *switch*. Примеры.

21. Условное выражение ...*?*... *:* ... Примеры.
22. Операторы инкремента ++ и декремента -- Примеры.
23. Оператор цикла *for*. Примеры.
24. Цикл с предусловием. Оператор цикла *while*. Примеры.
25. Цикл с постусловием. Оператор цикла *do...while*. Примеры.
26. Операторы прерывания *continue, break, return, System.exit*. Примеры.
27. Арифметические операторы. Примеры.
28. Логические операторы. Операторы сравнения. Побитовые операторы. Примеры.
29. Оболочечные классы. Упаковка (*boxing*) и распаковка (*unboxing*). Примеры.
30. Массивы. Многомерные массивы. Примеры.
31. Присваивание и сравнение массивов. Примеры.
32. Коллекции, списки, итераторы. Примеры.
33. Перебор в цикле элементов коллекций. Оператор цикла *for-each*. Примеры.
34. Классы языка программирования Java. Методы и поля. Примеры.
35. Конструкторы. Конструктор по умолчанию. Зарезервированные слова *super* и *this*. Блоки инициализации. Примеры.
36. Конструкторы в наследуемых классах. Порядок вызова конструкторов. Примеры.
37. Удаление неиспользуемых объектов и метод *finalize*. Проблема деструкторов для сложно устроенных объектов. Примеры.
38. Модификаторы уровня доступа default, public, protected, private. Примеры.
39. Наследование. Суперклассы и подклассы. Переопределение методов. Примеры.
40. Наследование и правила видимости. Зарезервированное слово *super*. Примеры.
41. Проблемы множественного наследования классов. Интерфейсы. Примеры.
42. Отличия интерфейсов от классов. Проблемы наследования интерфейсов. Примеры.
43. Композиция как альтернатива множественному наследованию. Примеры.

44. Интерфейсы как средство реализации множественного наследования. Примеры.

45. Правила совместимости ссылочных типов как основа использования полиморфного кода. Приведение и проверка типов. Примеры.

46. Методы. Значения параметров. Применение методов для ограничения доступа. Примеры.

47. Сигнатура метода. Перегрузка методов. Примеры.

48. Статическое и динамическое связывание методов. Полиморфизм. Примеры.

49. Функции. Модификаторы. Передача примитивных типов в функции. Примеры.

50. Локальные и глобальные переменные. Модификаторы доступа и правила видимости. Ссылка *this*. Примеры.

51. Передача ссылочных типов в функции. Проблема изменения ссылки внутри подпрограммы. Примеры.

52. Статические (static) вложенные классы и интерфейсы. Примеры.

53. Внутренние (inner) классы. Примеры.

54. Локальные (local) классы. Примеры.

55. Анонимные (anonimous) классы и обработчики событий. Примеры.

56. Анонимные (anonimous) классы и слушатели событий (listeners). Примеры.

57. Статические члены класса. Примеры.

58. Абстрактные классы и методы. Примеры.

59. Исключительные ситуации. Обработка исключительных ситуаций. Иерархия исключительных ситуаций. Контролируемые и неконтролируемые исключения. Примеры.

60. Объявление типа исключительной ситуации и оператор *throw*. Примеры.

61. Объявление метода, который может возбуждать исключительную ситуацию. Зарезервированное слово *throws*. Примеры.

62. Создание новых типов исключений. Оператор throw. Условие throws. Примеры.

63. Операторы try, catch и finally. Примеры.

64. События в Java. Понятие события. Типы событий. Иерархия классов событий. Примеры.

65. События в Java. Модель делегирования событий. Примеры.

66. События в Java. Интерфейсы блоков прослушивания событий. Способы реализации блока прослушивания. Примеры.

67. События в Java. Использование внутренних классов, классов-адаптеров для упрощения обработки событий. Примеры.

68. Типы-перечисления (enum). Примеры.

69. Работа с датами и временем. Класс *GregorianCalendar*. Примеры.

70. Работа со строками в Java. Строки как объекты. Классы *String*, *StringBuffer* и *StringBuilder*. Примеры.

71. Операции со строками. Сравнение строк. Поиск и извлечение строк. Посимвольная обработка строк. Пословное реверсирование строки. Преобразование всех символов строки к верхнему или нижнему регистру. Примеры.

72. Классы для работы с текстом. Примеры.

73. Специальные символы регулярных выражений. Примеры.

74. Классы Java Matcher и Pattern, используемые для работы с регулярными выражениями. Примеры.

75. Нахождение соответствий в тексте с помощью регулярных выражений. Нахождение всех вхождений по образцу. Замена текста, отвечающего соответствию. Примеры.

76. Ввод/вывод в Java. Основные группы классов и интерфейсов пакета java.io. Примеры.

77. Стандартные потоки ввода-вывода. Организация ввода и вывода данных. Класс Scanner. Примеры.

78. Ввод/вывод в Java. Фильтрованные потоки. Примеры.

79. Ввод/вывод в Java. Буферизированные потоки. Примеры.

80. Ввод/вывод в Java. Канальные потоки. Примеры.

Пример вопросов устного опроса

1. История возникновения Java
2. Процедурное программирование.
3. Объектно-ориентированное программирование.
4. Основные понятия ООП.
5. Объекты и классы.
6. Абстракция данных.
7. Сценарий построения объектно-ориентированной программы
8. Основополагающие принципы ООП. Инкапсуляция.
9. Основополагающие принципы ООП. Наследование.
10. Управление наследованием.
11. Интерфейсы как средство реализации множественного наследования.
12. Основополагающие принципы ООП. Полиморфизм. Средства реализации полиморфизма.
13. Классификация программ по типу исполнения (компилируемые, интерпретируемые, исполняемые на виртуальных машинах).
14. Особенности языка и платформы Java.
15. Виртуальная машина Java. JIT-компиляция.
16. Создание простейшей программы на Java, её компиляция в байт-код и запуск
17. Базовые типы данных и литералы Java.
18. Операторы. Классы-оболочки. Операторы управления.
19. Средства разработки Java-приложений. Интегрированные среды разработки.
20. Встроенные типы данных. Способы задания литералов различных типов.
21. Приведение типов (явное и автоматическое). Константы и переменные.
22. Оператор присваивания. Порядок действий (приоритет операторов).
23. Арифметические операторы. Операторы инкремента и декремента.
24. Встроенный класс Math. Псевдослучайные числа.
25. Операторы сравнения и логические операторы.
26. Операторы ветвления. Условный оператор. Минимизация количества проверок
27. Операторы ветвления. Оператор множественного выбора. Его сравнение с условным оператором
28. Встроенный класс String. Строковые операции.

29. Стандартные потоки ввода-вывода. Организация ввода и вывода данных. Класс Scanner.

30. Операторы организации циклов. Цикл типа «n раз».

31. Операторы организации циклов. Цикл типа «пока» (с пред- и постпроверкой условия).

32. Массивы. Способы объявления и инициализации массивов. Индексация и размер массива.

33. Массивы. Алгоритмы сортировки.

34. Массивы. Многомерные массивы.

35. Статические методы классов. Методы функционального и процедурного типа.

36. Сигнатура метода. Перегрузка методов.

37. Сравнение парадигм.

38. Члены классов. Методы и поля

39. Специальные методы классов (конструкторы). Конструктор по умолчанию.

40. Модификаторы уровня доступа (default, public, protected, private).

41. Иерархия классов Java. Коренной класс Object и его методы.

42. Исключительные ситуации. Обработка исключительных ситуаций.

43. Приложения с графическим интерфейсом с использованием GUI-пакетов и апплеты.

44. Класс Math.

45. Переменные класса и константы. Ограничение доступа. Конструкторы. Методы.

46. Статические методы и поля. Модификатор native. Модификатор synchronized. Логические блоки.

47. Параметризованные классы. Параметризованные методы. Методы с переменным числом параметров.

48. Перечисления. Аннотации.

49. Наследование. Использование final.

50. Использование super и this.

51. Переопределение методов и полиморфизм. Полиморфизм и расширяемость.

52. Статические методы и полиморфизм.

53. Абстракция и абстрактные классы.

54. Клонирование объектов. “Сборка мусора” и освобождение ресурсов.

55. Интерфейсы.
56. Статический импорт.
57. Внутренние (inner) классы. Вложенные (nested) классы. Анонимные (anonymous).
58. Класс String.
59. Классы StringBuilder и StringBuffer.
60. Форматирование строк.
61. Регулярные выражения.
62. Интернационализация текста.
63. Иерархия и способы обработки. Оператор throw.Ключевое слово finally
64. Собственные исключения. Наследование и исключения. Отладочный механизм assertion.
65. Класс File. Байтовые и символьные потоки ввода/вывода. Предопределенные потоки.
66. Класс Scanner.
67. Коллекции. Общие определения.
68. Списки.
69. Множества.
70. Карты отображений.