

**Федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»**

УТВЕРЖДАЮ
Директор ИКНК
_____ Д.П. Зегжда
«17» июня 2025 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

«Программирование на языке Java»

| | |
|--|--|
| Разработчик | Высшая школа компьютерных технологий и информационных систем |
| Направление (специальность) подготовки | 09.03.01 Информатика и вычислительная техника |
| Наименование ООП | 09.03.01_01 Разработка компьютерных систем |
| Квалификация (степень) выпускника | бакалавр |
| Образовательный стандарт | СУОС |
| Форма обучения | Очная |

СОГЛАСОВАНО
Руководитель ОП
_____ Р.В. Цветков
«19» мая 2025 г.

Соответствует СУОС
Утверждена протоколом заседания
высшей школы "ВШКТиИС"
от «19» мая 2025 г. № 4

РПД разработал:
Профессор, д.т.н., проф. Г.Ф. Малыхина

1. Цели и планируемые результаты изучения дисциплины

Цели освоения дисциплины

Целью курса является изучение технологии программирования приложений, нативных и кроссплатформенных приложений, локальных и распределенных.

Результаты обучения выпускника

| Код | Результат обучения (компетенция) выпускника ООП |
|--------------|---|
| ПК-4 | Способен решать стандартные задачи в области информационных, информационно-управляющих и управляющих систем |
| ИД-1 ПК-4 | Решает стандартные задачи профессиональной деятельности, связанные с разработкой систем управления, информационных и информационно-управляющих систем |

Планируемые результаты изучения дисциплины

знания:

- современные информационные технологии и программные средства для проектирования информационно-управляющих систем

умения:

- определять структуру информационно-управляющих систем в соответствии с поставленной задачей

навыки:

- владение современными информационными технологиями и программными средствами автоматизированного проектирования компонентов информационно-управляющих систем

2. Место дисциплины в структуре ООП

В учебном плане дисциплина «Программирование на языке Java» относится к модулю «Дисциплины (модули) по выбору 2 (ДВ.2)».

Изучение дисциплины базируется на результатах освоения следующих дисциплин:

- Введение в профессиональную деятельность

3. Распределение трудоёмкости освоения дисциплины по видам учебной работы и формы текущего контроля и промежуточной аттестации

3.1. Виды учебной работы

| Виды учебной работы | Трудоёмкость по семестрам |
|--|---------------------------|
| | Очная форма |
| Лекционные занятия | 30 |
| Лабораторные занятия | 30 |
| Самостоятельная работа | 78 |
| Промежуточная аттестация (зачет с оценкой) | 6 |
| Общая трудоёмкость освоения дисциплины | 144, ач |
| | 4, зет |

3.2. Формы текущего контроля и промежуточной аттестации

| Формы текущего контроля и промежуточной аттестации | Количество по семестрам |
|--|-------------------------|
| | Очная форма |
| Промежуточная аттестация | |
| Зачеты с оценкой, шт. | 1 |

4. Содержание и результаты обучения

4.1 Разделы дисциплины и виды учебной работы

| № раздела | Разделы дисциплины, мероприятия текущего контроля | Очная форма | | |
|-----------|---|-------------|---------|--------|
| | | Лек, ач | Лаб, ач | СР, ач |
| 1. | Введение. Цели и задачи. Универсальный язык программирования UML. | 2 | 2 | 8 |
| 2. | Процедурное и объектно-ориентированное программирование. | 2 | 2 | 8 |
| 3. | Модификаторы данных, классов и методов | 2 | 2 | 7 |
| 4. | Процессы и потоки | 2 | 2 | 7 |

| | | | | |
|---|--|---------|----|----|
| 5. | Модели обработки событий. | 2 | 2 | 7 |
| 6. | Рисование. и анимация | 2 | 2 | 7 |
| 7. | Потоки ввода и вывода. | 2 | 2 | 7 |
| 8. | Технологии программирования клиент-серверных приложений | 4 | 4 | 7 |
| 9. | Программирование клиентских приложений баз данных. | 4 | 4 | 7 |
| 10. | Технологии обеспечение целостности данных и программ. | 4 | 4 | 6 |
| 11. | Технологии обеспечения конфиденциальности данных и программ. | 4 | 4 | 7 |
| Итого по видам учебной работы: | | 30 | 30 | 78 |
| Зачеты с оценкой, ач | | | | 0 |
| Часы на контроль, ач | | | | 0 |
| Промежуточная аттестация (зачет с оценкой) | | 6 | | |
| Общая трудоёмкость освоения: ач / зет | | 144 / 4 | | |

4.2. Содержание разделов и результаты изучения дисциплины

| Раздел дисциплины | Содержание |
|---|--|
| 1. Введение. Цели и задачи. Универсальный язык программирования UML. | <p> Диаграммы прецедентов для описания макро-требований к системе. Диаграммы деятельности для анализа и моделирования процессов. Диаграммы классов для описания классов и связей между ними. Диаграмма взаимодействий для описания последовательности и коопераций. Диаграмма состояний изображают объекты и сообщения, передаваемые между ними, а также изменение состояния объектов. Диаграмма компонентов определяет различные виды моделей: анализа, проектирования и реализации. </p> |
| 2. Процедурное и объектно-ориентированное программирование. | <p> Модификаторы данных, классов и методов. Модификаторы доступа и доступность классов, методов, полей. Финальные данные методы и классы. Абстрактные классы и методы. Статические классы, методы и поля, перегрузка статических методов, статические инициализаторы. Спецификаторы «native», «transient» и «synchronized». </p> |
| 3. Модификаторы данных, классов и методов | <p> Модификаторы доступа и доступность классов, методов, полей. Финальные данные методы и классы. Абстрактные классы и методы. Статические классы, методы и поля, перегрузка статических методов, статические инициализаторы. Спецификаторы «native», «transient» и «synchronized». </p> |
| 4. Процессы и потоки | <p> Потоки имеют некоторые состояния Приоритеты потоков. Планирование задач операционной системой. Класс Thread и интерфейс Runnable Расширение класса Thread: главный и дочерние потоки. Схема состояний потока Отдавать управление Приостановка потока Состояние засыпания Состояние блокирования Использование не синхронизированных потоков Состояние ожидания, использование мониторов. </p> |
| 5. Модели обработки событий. | <p> Иерархия классов событий и методы анализа событий. Две модели обработки событий. Модель делегирования событий, создание слушателей и функции обработки событий. Модель явной обработки событий, классы, методы и порядок вызова методов. Адаптеры для модели делегирования событий, использование внутренних классов и анонимных классов внутри метода для обработки событий. </p> |

| | |
|---|--|
| 6. Рисование. и анимация | Робастный и масштабируемый механизм рисования. Графический контекст рисования, выбор цвета фона, цвета пера и шрифта контекста. Рисование и заполнение графических объектов, вырезание ограниченной области. Спонтанное рисование и вызов метода paint(), update() и repaint(). Рисование в заэкраных областях и перерисовка изображений вне экрана. |
| 7. Потоки ввода и вывода. | Представление текста и кодирование символов. Класс File и навигация в локальной файловой системе, получение рекурсивного списка директорий Произвольный доступ к данным, хранящимся в файле, методы чтения и записи байтов. Модель потоков чтения и записи данных. Низкоуровневые потоки ввода, основные методы чтения и записи. Потоки - фильтры высокого уровня и потоки для буферизации. Потоки чтения и записи символов Unicode, чтение строк в RTF формате, ввод последовательности строк текста. Преобразование объектов в последовательную форму методом сериализации. Особенности работы с временными переменными. |
| 8. Технологии программирования клиент-серверных приложений | Клиент-серверные приложения, основанные на установлении соединения. Слушающие Сокеты для сервера и Сокеты для клиента. Потоки, связанные с сокетами, цепочка потоков для сервера и для клиента, методы ввода и вывода данных через сокет. Создание параллельных серверов, обслуживающих множество клиентов. Серверные приложения, сервлеты, серверные страницы. |
| 9. Программирование клиентских приложений баз данных. | Реляционные и объектно-ориентированные базы данных. Интерфейсы ODBC как открытый интерфейс взаимодействия с базами данных, установка драйвера. Статический и динамический SQL. Программирование транзакций. |
| 10. Технологии обеспечение целостности данных и программ. | Программирование методов обеспечения целостности конфиденциальности, целостности, доступности информации и невозможности отказа. Программирование дайджестов и цифровых подписей. Программирование приложений, защищенных симметричным шифрованием. |
| 11. Технологии обеспечения конфиденциальности данных и программ. | Программирование методов обеспечения конфиденциальности, целостности, доступности информации и невозможности отказа. Программирование дайджестов и цифровых подписей. Программирование приложений, защищенных симметричным и асимметричным шифрованием. Безопасные сокеты. |

5. Образовательные технологии

В преподавании курса используются преимущественно традиционные образовательные технологии: – лекции, – лабораторные занятия. Вместе с тем, нетрадиционным для курса является: – лабораторный практикум с использованием IntelliJ IDEA – интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. Учащиеся и преподаватели могут пользоваться продуктами JetBrains бесплатно в целях обучения. Предназначена для фулстек-разработки и создания корпоративных приложений. Поддерживает широкий набор фреймворков и технологий для бэкенда и фронтенда и включает инструменты для профилирования и работы с базами данных, HTTP-клиент и много других функций.

6. Лабораторный практикум

| № раздела | Наименование лабораторных работ | Трудоемкость, ач |
|--------------|---|---------------------|
| | | Очная форма |
| 1. | Составление Диаграммы прецедентов для описания макро-требований к системе. Диаграммы деятельности для анализа и моделирования процессов. Диаграммы классов для описания классов и связей между ними. Диаграмма взаимодействий для описания последовательности и коопераций. Диаграмма состояний | 2 |
| 2. | Написание и отладка программы, содержащей следующие элесенты. Модификаторы данных, классов и методов. Модификаторы доступа и доступности классов, методов, полей. Финальные данные методы и классы. Абстрактные классы и методы. | 4 |
| 3. | Написание и отладка программы, содержащей следующие элесенты. Статические классы, методы и поля, перегрузка статических методов, статические инициализаторы. Спецификаторы «native», «transient» и «synchronized». | 2 |
| 4. | Написание и отладка программы, содержащей следующие элесенты. Потоки имеющие некоторые состояния. Приоритеты потоков. Планирование задач операционной системой. Класс Thread и интерфейс Runnable Расширение класса Thread: главный и дочерние потоки. Схема состояний потока Отдавать управление Приостановка потока Состояние засыпания Состояние блокирования | 4 |
| 5. | Написание и отладка программы, содержащей следующие элесенты. Иерархия классов событий и методы анализа событий. Две модели обработки событий. Модель делегирования событий, создание слушателей и функции обработки событий. Модель явной обработки событий, классы, методы и порядок вызова методов. Адаптеры для модели делегирования событий, использование внутренних классов и анонимных классов внутри метода для обработки событий. | 2 |
| 6. | Написание и отладка программы, содержащей следующие элесенты. Рисование и заполнение графических объектов, вырезание ограниченной области. Спонтанное рисование и вызов метода paint(), update() и repaint(). | 4 |

| | | |
|--------------------|--|-----------|
| 7. | Написание и отладка программы, содержащей следующие элементы. Класс File и навигация в локальной файловой системе, получение рекурсивного списка директорий Произвольный доступ к данным, хранящимся в файле, методы чтения и записи байтов. Модель потоков чтения и записи данных. Низкоуровневые потоки ввода, основные методы чтения и записи. Потоки - фильтры высокого уровня и потоки для буферизации. Потоки чтения и записи символов Unicode, чтение строк в UTF формате, ввод последовательности строк текста. Преобразование объектов в последовательную форму методом сериализации. | 2 |
| 8. | Написание и отладка программы, содержащей следующие элементы. Слушающие Сокеты для сервера и Сокеты для клиента. Потоки, связанные с сокетами, цепочка потоков для сервера и для клиента, методы ввода и вывода данных через сокет | 4 |
| 9. | Написание и отладка программы, содержащей следующие элементы. Интерфейсы ODBC как открытый интерфейс взаимодействия с базами данных, установка драйвера. Статический и динамический SQL. Программирование транзакций. | 2 |
| 10. | Программирование методов обеспечения конфиденциальности и целостности. | 2 |
| 11. | Программирование безопасных сокетов. | 2 |
| Итого часов | | 30 |

7. Практические занятия

Не предусмотрено

8. Организация и учебно-методическое обеспечение самостоятельной работы

– Самостоятельная работа студентов, предполагающая их работу как в стенах университета, так и вне его. Включает в себя работу с источниками, в том числе отражающими современное состояние изучаемой области и последние научно-технические достижения (для этого необходимо использовать ресурсы Фундаментальной библиотеки СПбПУ, включающие в себя доступ с компьютеров локальной сети университета в базы данных издательств научной литературы Springer и других, баз данных профессиональных сообществ IEEE и др.). Предполагает обращение к учебным материалам, расположенным в открытом доступе в сети Интернет (в том числе дистанционного обучения, в том числе и на английском языке). Специфика дисциплины предполагает применения специальных программных средств, некоторые из которых (Java) являются коммерческими. Поэтому если самостоятельная работа студентов (например, ее творческий элемент) предполагает практическую программную

реализацию, математическое моделирование или исследование с целью приобретения навыков и более глубокого изучения вопросов дисциплины, то самостоятельная работа студентов должна проходить в специализированном компьютерном классе.

Для обеспечения работы студентов в помещениях учебного фонда института составляется график присутствия студентов (согласуется с часами работы учебно-вспомогательного персонала), ведется журнал посещений. Оформление отчетных материалов для промежуточного контроля преподавателем итогов работы студентов в семестре желательно производить вне стен университета, поскольку для этого не требуется ни специальное оборудование, ни специальные программы. Если самостоятельная работа студентов требует участия в мероприятиях вне стен университета (конференции, симпозиумы, выставки), то по итогам составляется краткий отчет о мероприятии, который сдается на кафедру сотруднику, ответственному за научно-исследовательскую работу студентов. Если самостоятельная работа студентов требует отдельных консультаций (для обсуждения промежуточных итогов творческой самостоятельной работы, результатов научно-исследовательской работы), то они производятся по согласованию с преподавателем, осуществляющим учебный процесс в рамках дисциплины. – Контроль самостоятельной работы студентов, предусматривающий проверку отчетных материалов и их обсуждение. При изучении студентами основ изучаемой дисциплины возможно проведение промежуточного тестирования с целью текущего контроля успешного их освоения. Тестирование может проводиться также в виде письменной работы в аудиториях учебного фонда института. В таком случае преподаватель назначает часы проведения промежуточной аттестации. Итоговая проверка теоретической подготовки студентов может быть проведена с помощью единого тестирования. Для проверки результатов творческой работы студентов преподавателю, отвечающему за проведения занятий в рамках учебной дисциплины, рекомендуется созвать комиссию из профессоров и доцентов кафедры для объективного рассмотрения результатов труда студентов и высокого качества ее оценки. Формальные задания (имеющие целью освоение студентами основ изучаемой дисциплины) следует проверять автоматическим образом, используя программные модули (дистанционные или стационарные, установленные на компьютеры локальной сети университета) проверки.

Работа с литературой подразумевает: • изучение основ изучаемого вопроса по учебной литературе, рекомендованной в пункте и образовательным ресурсам сети Интернет, • поиск учебных материалов сверх рекомендованных, ознакомление по научным 20 публикациям с современным состоянием изучаемого вопроса, составление собственного мнения о наилучших методах решения для заданной преподавателем учебной задачи, изучение патентов на методы решения и алгоритмы. • составление краткого или полного обзора при использовании материалов самостоятельной работы для подготовки выступления на кафедральном

аспирантском семинаре, на конференции или при подготовке статьи в журнал. Самостоятельное изучение разделов дисциплины предполагает более глубокий анализ тематики, предполагаемой к изучению в рамках дисциплины (выявление структурных недостатков методов и подходов, предложение путей их устранения, идеи улучшения характеристик, получение лучших результатов и т. д.), выполнение по заданию преподавателя имитационного моделирования (математического), анализ полученных результатов (как элементов творческой самостоятельной работы).

Примерное распределение времени самостоятельной работы студентов

| Вид самостоятельной работы | Примерная трудоемкость, ач |
|--|----------------------------------|
| | Очная форма |
| Текущая СР | |
| работа с лекционным материалом, с учебной литературой | 6 |
| опережающая самостоятельная работа (изучение нового материала до его изложения на занятиях) | 6 |
| самостоятельное изучение разделов дисциплины | 8 |
| выполнение домашних заданий, домашних контрольных работ | 8 |
| подготовка к лабораторным работам, к практическим и семинарским занятиям | 8 |
| подготовка к контрольным работам, коллоквиумам | 6 |
| Итого текущей СР: | 42 |
| Творческая проблемно-ориентированная СР | |
| выполнение расчётно-графических работ | 0 |
| выполнение курсового проекта или курсовой работы | 0 |
| поиск, изучение и презентация информации по заданной проблеме, анализ научных публикаций по заданной теме | 26 |
| работа над междисциплинарным проектом | 0 |
| исследовательская работа, участие в конференциях, семинарах, олимпиадах | 10 |
| анализ данных по заданной теме, выполнение расчётов, составление схем и моделей на основе собранных данных | 0 |
| Итого творческой СР: | 36 |
| Общая трудоемкость СР: | 78 |

9. Учебно-методическое обеспечение дисциплины

9.1. Адрес сайта курса

<https://open.spbstu.ru/eios/>

9.2. Рекомендуемая литература

Основная литература

| № | Автор, название, место издания, издательство, год (годы) издания | Год изд. | Источник |
|---|---|----------|----------|
| 1 | Давыдов В.Г. Язык и технологии Java, 2012. URL: http://elib.spbstu.ru/dl/pwd/2441.pdf | 2012 | ЭБ СПбПУ |

Дополнительная литература

| № | Автор, название, место издания, издательство, год (годы) издания | Год изд. | Источник |
|---|---|----------|----------|
| 1 | Станкевич Л.А. Когнитивные системы и роботы: Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2019. URL: http://elib.spbstu.ru/dl/2/i19-63.pdf | 2019 | ЭБ СПбПУ |

Ресурсы Интернета

1. <https://www.vmware.com/explore/us/springone>: <https://www.vmware.com/explore/us/springone>

9.3. Технические средства обеспечения дисциплины

Цикл работ выполняется в компьютерном классе Windows

в локальной сети TCP/IP.

Мультимедийный проектор.

10. Материально-техническое обеспечение дисциплины

Цикл работ выполняется в компьютерном классе Windows в локальной сети TCP/IP.

Мультимедийный проектор.

11. Критерии оценивания и оценочные средства

11.1. Критерии оценивания

Для дисциплины «Программирование на языке Java» формой аттестации является зачёт с оценкой. Дисциплина реализуется с применением системы индивидуальных достижений.

Текущий контроль успеваемости

Максимальное значение персонального суммарного результата обучения (ПСРО) по приведенной шкале - 100 баллов

Максимальное количество баллов приведенной шкалы по результатам прохождения двух точек контроля - 80 баллов.

Подробное описание правил проведения текущего контроля с указанием баллов по каждому контрольному мероприятию и критериев выставления оценки размещается в СДО в навигационном курсе дисциплины.

Промежуточная аттестация по дисциплине

Максимальное количество баллов по результатам проведения аттестационного испытания в период промежуточной аттестации – 20 баллов приведенной шкалы.

Промежуточная аттестация по дисциплине проводится в соответствии с расписанием.

Дисциплина реализуется с применением системы индивидуальных достижений.

Текущий контроль успеваемости

Максимальное значение персонального суммарного результата обучения (ПСРО) по приведенной шкале - 100 баллов

Максимальное количество баллов приведенной шкалы по результатам прохождения двух точек контроля - 80 баллов.

Подробное описание правил проведения текущего контроля с указанием баллов по каждому контрольному мероприятию и критериев выставления оценки размещается в СДО в навигационном курсе дисциплины.

Промежуточная аттестация по дисциплине

Максимальное количество баллов по результатам проведения аттестационного испытания в период промежуточной аттестации – 20 баллов приведенной шкалы.

Промежуточная аттестация по дисциплине проводится в соответствии с расписанием.

Зачет, предусматривающий комплексную оценку работы студента в семестре, оценку

качества предоставленных им для промежуточной аттестации материалов, индивидуальную подготовку студентами ответа на выбранные им вопросы из списка вопросов к зачету и ответ в виде беседы с преподавателем. Возможно проведение зачета в форме тестирования. В последнем случае критерии оценки (минимальное количество верных ответов, соответствующее оценкам «удовлетворительно», «хорошо» и «отлично») озвучиваются студентам заранее.

Результаты промежуточной аттестации, определяются на основе баллов, набранных в рамках применения, СИД

| Баллы по приведенной шкале в рамках применения СИД (ПСРО+ ПА) | Оценка по результатам промежуточной аттестации |
|---|--|
| | Экзамен/диф.зачет/зачет |
| 0 - 60 баллов | Неудовлетворительно/не зачтено |
| 61 - 75 баллов | Удовлетворительно/зачтено |
| 76 - 89 баллов | Хорошо/зачтено |
| 90 и более | Отлично/зачтено |

11.2. Оценочные средства

Код изменения размера апплета помещается в метод paint(). Какие утверждения верны ?

```
import java.applet.*;
import java.awt.*;
public class SecondApp extends Applet {
public void init() {
setBackground(Color.cyan);}
public void paint(Graphics g) {
Dimension d=getSize();
resize(d.width,d.height/2);}
}
```

Апплет голубого цвета имеет стандартную ширину и высоту в 2 раза меньше стандартной.

Если апплет свернуть/развернуть, то его высота уменьшится еще в 2 раза.

Если апплет свернуть/развернуть, то его высота не изменится.

Апплет отсутствует.

Апплет имеет стандартную ширину и нулевую высоту, апплет не имеет голубого цвета.

2. Какие методы следует ввести, чтобы после остановки апплета возобновить рисование со следующей по порядку буквы, а не с начала алфавита?

```

public class Fourth extends Applet implements Runnable {
    char msg;
    Thread t;

    public void init(){t=new Thread(this);t.start();}
    public void paint(Graphics g){ g.drawString(""+msg,50,100);}
    public void run() {
        char ch='A';
        try{while(true){
            repaint(); Thread.sleep(200);
            msg=(char)(ch++);
            if(ch=='Z')ch='A';}} catch(InterruptedException e){};}
    }
    public void start(){Thread.currentThread().resume();}
    public void stop() {Thread.currentThread().suspend();}
    public void start(){t.resume();}
    public void stop() {t.suspend();}
    public void start(){t.start();}
    public void stop() {t.stop();}
}

```

3. Каким будет результат модификации строки и что будет напечатано в строке 8?

```

public static void main (String[] args) {
    String s="Strings Strings";
    s.replace('g','n');
    s.trim();
    s.replace('i','a');
    s.substring(3);
    System.out.println(s);
}

```

annsStranns
 Str
 Strings Strings
 Stranns Stranns
 StrannsStranns

4. Какой результат компиляции и выполнения кода Вы ожидаете?

```

public static void main (String[] args) {

```

```

ArrayList aL1=new ArrayList();
aL1.add(new Integer(2));
aL1.add(new Double(5));
aL1.add(new String("ArrayList"));
aL1.remove(new Integer(2));
aL1.add(Math.E);
ArrayList aL2=new ArrayList(aL1);
if(aL1.equals(aL2))System.out.println("size="+aL2.size());
}

```

Ошибка компиляции в строке 7

Ошибка компиляции в строке 8

Ошибка компиляции в строке 5

Метод выполняется без ошибок. Результат: size=3

Метод выполняется без ошибок. Результат: size=4

5. Какой результат компиляции и выполнения фрагмента кода Вы ожидаете?

```

public static void main (String[] args) {
FirstFrame that=new FirstFrame ("Frame with TextField");
that.setSize(300,300);
that.setLayout(null);
TextField f=new TextField("Title");
f.setBounds(10,30,80,30);
that.add(f);
that.setVisible(true);}

```

Ошибка компиляции в строке 4. Из фрейма удаляется менеджер.

Компиляция проходит успешно. Текстовое поле имеет размер в точности 80*30

Компиляция проходит успешно. Появляется пустой фрейм размером 300*300

Компиляция проходит успешно. Появляется текстовое поле, размер которого соответствует надписи (шрифт по умолчанию)

Компиляция проходит успешно. Текстовое поле занимает весь фрейм

8. Какие утверждения относительно кода апплета справедливы, если все необходимые пакеты импортированы?

```

public class ASixth extends Applet {
public void init() {
requestFocus();

```

```

addKeyListener ( new KeyAdapter(){
public void keyTyped(KeyEvent ke){
showStatus("Pressed "+ke.getKeyChar());};
} ); }
}

```

Код компилируется без ошибок. События keyTyped обрабатываются, на остальные события апплет не реагирует

Код компилируется без ошибок, но апплет не реагирует на события keyTyped

Ошибка компиляции в строке 4, поскольку класс должен расширять класс KeyAdapter

Ошибка компиляции в строке 1 апплет должен выполнять интерфейс KeyListener

9. Какое из утверждений относительно апплета верно?

```

public void init() {
setBackground(Color.lightGray);
setForeground(Color.red);
TextField tf=new TextField("TextField is");
tf.setFont(new Font("Serif",Font.ITALIC,36));
add(tf);
}

public void paint(Graphics g) {

```

```

g.drawString("text component that allows for the editing of a single line of text",5,100); }

```

Серый цвет фона апплета и серый цвет текстового поля. Красная 36р строка “TextField”, красная 12р строка "text component ...”.

Серый цвет фона апплета и белый цвет текстового поля. Черная 36р строка “TextField”, красная 36р строка "text component ...”.

Серый цвет фона апплета и белый цвет текстового поля. Черная 36р строка “TextField”, черная 36р строка "text component ...”.

Серый цвет фона апплета и белый цвет текстового поля. Красная 36р строка “TextField”, красная 12р строка "text component ...”.

12. Методические рекомендации по организации изучения дисциплины

При изучении дисциплины рекомендуем активно пользоваться ресурсами Интернета:

- <https://www.jetbrains.com/ru-ru/idea/download/?section=windows>

- <https://www.jetbrains.com/ru-ru/idea/download/?section=windows>

- IntelliJ IDEA — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. - https://ru.wikipedia.org/wiki/IntelliJ_IDEA

Порекомендуем рассмотреть основные особенности работы с IntelliJ IDEA: создание кода на

основе информации о классах, редактор с мощными вспомогательными возможностями, встроенная поддержка рефакторинга кода, средства структурного анализа и проверки кода на наличие логических ошибок, потенциально опасных конструкций и возможных способов улучшения, поддержка J2EE-приложений, поддержка XML, визуальный редактор пользовательского интерфейса, встроенная поддержка unit-тестирования с использованием JUnit, поддержка основных систем контроля версий (SourceSafe, CVS, Starteam), полная поддержка Java 5, поддержка плагинов. Особое внимание уделено практическим приемам работы с IntelliJ IDEA.

13. Адаптация рабочей программы для лиц с ОВЗ

Адаптированная программа разрабатывается при наличии заявления со стороны обучающегося (родителей, законных представителей) и медицинских показаний (рекомендациями психолого-медико-педагогической комиссии). Для инвалидов адаптированная образовательная программа разрабатывается в соответствии с индивидуальной программой реабилитации.