

Документ подписан простой электронной подписью
 Информация о владельце:
 ФИО: Гаранин Максим Александрович
 Должность: Ректор
 Дата подписания: 02.10.2025 11:09:52
 Уникальный программный ключ:
 7708e3a47e66a8ee02711b298d7c78bd1e40bf88

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ПРИВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ»

МОДУЛЬ "ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ПРОГРАММИРОВАНИЕ"

Объектно-ориентированное программирование рабочая программа дисциплины (модуля)

Направление подготовки 09.03.03 Прикладная информатика

Направленность (профиль) Управление цифровой инфраструктурой организации

Квалификация **бакалавр**

Форма обучения **очная**

Общая трудоемкость **20 ЗЕТ**

Виды контроля в семестрах:

экзамены 5, 4

зачеты 3

курсовые работы 5

Распределение часов дисциплины по семестрам

Семестр (<Курс>.<Семестр на курсе>)	3 (2.1)		4 (2.2)		5 (3.1)		Итого	
Неделя	16 4/6		16		16 3/6			
Вид занятий	УП	РП	УП	РП	УП	РП	УП	РП
Лекции	16	16	32	32	32	32	80	80
Лабораторные	48	48	48	48	48	48	144	144
Конт. ч. на аттест.					1	1	1	1
Конт. ч. на аттест. в период ЭС	0,15	0,15	2,3	2,3	2,3	2,3	4,75	4,75
В том числе в форме практ.подготовки	48	48	48	48	82	82	178	178
Итого ауд.	64	64	80	80	80	80	224	224
Контактная работа	64,15	64,15	82,3	82,3	83,3	83,3	229,75	229,75
Сам. работа	71	71	181	181	180	180	432	432
Часы на контроль	8,85	8,85	24,7	24,7	24,7	24,7	58,25	58,25
Итого	144	144	288	288	288	288	720	720

Программу составил(и):

к.п.н., доцент, Горбатов С.В.

Рабочая программа дисциплины

Объектно-ориентированное программирование

разработана в соответствии с ФГОС ВО:

Федеральный государственный образовательный стандарт высшего образования - бакалавриат по направлению подготовки 09.03.03 Прикладная информатика (приказ Минобрнауки России от 19.09.2017 г. № 922)

составлена на основании учебного плана: 09.03.03-25-2-ПИБ.plm.plx

Направление подготовки 09.03.03 Прикладная информатика Направленность (профиль) Управление цифровой инфраструктурой организации

Рабочая программа одобрена на заседании кафедры

Цифровые технологии

Зав. кафедрой Ефимова Т.Б.

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)

1.1	изучение методов и приемов современного программирования и разработки приложений, формирование навыков разработки прототипов и законченных приложений с использованием современных языков объектно-ориентированного программирования
-----	--

2. МЕСТО ДИСЦИПЛИНЫ (МОДУЛЯ) В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Цикл (раздел) ОП:	Б1.О.13.02
-------------------	------------

3. КОМПЕТЕНЦИИ ОБУЧАЮЩЕГОСЯ, ФОРМИРУЕМЫЕ В РЕЗУЛЬТАТЕ ОСВОЕНИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)

ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения

ОПК-7.1 Разрабатывает алгоритмы и программы на языке программирования высокого уровня

ПК-4 Способен проводить тестирование компонентов программного обеспечения ИС

ПК-4.1 Проводит модульное тестирование программного обеспечения ИС, интеграционное тестирование

В результате освоения дисциплины (модуля) обучающийся должен

3.1	Знать:
3.1.1	основные принципы объектно-ориентированного программирования; понятия класса и объекта; принципы построения классов; создание объектов на основе
3.1.2	существующих классов, определение отдельных атрибутов;
3.1.3	понятия сплошного и выборочного ручного тестирования, средства автоматизированного тестирования Java (JUnit и др)
3.2	Уметь:
3.2.1	создавать собственные классы и объекты на языке программирования высокого уровня;
3.2.2	использовать средства автоматизированного тестирования Java (JUnit и др);
3.3	Владеть:
3.3.1	навыками алгоритмизации и программной реализации практических задач на языке высокого уровня

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Код занятия	Наименование разделов и тем /вид занятия/	Семестр / Курс	Часов	Примечание
	Раздел 1. Объектно-ориентированное программирование на Java			
1.1	Объекты и классы: реализация в языке Java /Лек/	3	1	
1.2	Объекты и классы: реализация в языке Java /Лаб/	3	8	Практическая подготовка
1.3	Объекты и классы: реализация в языке Java /Ср/	3	2	
1.4	Средства управления жизнью объекта /Лек/	3	1	
1.5	Средства управления жизнью объекта /Лаб/	3	4	Практическая подготовка
1.6	Средства управления жизнью объекта /Ср/	3	2	
1.7	Работа с массивами объектов /Лек/	3	1	
1.8	Работа с массивами объектов /Лаб/	3	2	Практическая подготовка
1.9	Работа с массивами объектов /Ср/	3	2	
1.10	Наследование в Java /Лек/	3	1	
1.11	Наследование в Java /Лек/	3	1	
1.12	Наследование в Java /Ср/	3	2	
1.13	Абстрактные классы и интерфейсы /Лек/	3	1	
1.14	Абстрактные классы и интерфейсы /Лаб/	3	2	Практическая подготовка
1.15	Абстрактные классы и интерфейсы /Ср/	3	2	
1.16	Пакеты /Лек/	3	1	

1.17	Пакеты /Лаб/	3	2	Практическая подготовка
1.18	Пакеты /Ср/	3	2	
1.19	Подсистема ввода вывода java.io /Лек/	3	1	
1.20	Подсистема ввода вывода java.io /Лаб/	3	2	Практическая подготовка
1.21	Подсистема ввода вывода java.io /Ср/	3	2	
1.22	Обработка исключений /Лек/	3	1	
1.23	Обработка исключений /Лаб/	3	2	Практическая подготовка
1.24	Обработка исключений /Ср/	3	2	
1.25	Классы Throwable и Exception: создание и использование /Лек/	3	1	
1.26	Классы Throwable и Exception: создание и использование /Лаб/	3	4	Практическая подготовка
1.27	Классы Throwable и Exception: создание и использование /Ср/	3	2	
1.28	Обобщенное программирование (Java Generics) /Лек/	3	1	
1.29	Обобщенное программирование (Java Generics) /Лаб/	3	4	Практическая подготовка
1.30	Обобщенное программирование (Java Generics) /Ср/	3	2	
1.31	Графическая подсистема Java. Классы AWT. /Лек/	3	2	
1.32	Графическая подсистема Java. Классы AWT. /Лаб/	3	4	Практическая подготовка
1.33	Графическая подсистема Java. Классы AWT. /Ср/	3	5	
1.34	Обработка событий в графической подсистеме (Listeners) /Лек/	3	1	
1.35	Обработка событий в графической подсистеме (Listeners) /Лаб/	3	4	Практическая подготовка
1.36	Обработка событий в графической подсистеме (Listeners) /Ср/	3	22	
1.37	Основные возможности графической системы Swing /Лек/	3	2	
1.38	Основные возможности графической системы Swing /Лаб/	3	10	Практическая подготовка
1.39	Основные возможности графической системы Swing /Ср/	3	24	
	Раздел 2. Программы на Java в архитектуре клиент-сервер			
2.1	Многопоточное программирование на Java /Лек/	4	2	
2.2	Многопоточное программирование на Java /Лаб/	4	4	Практическая подготовка
2.3	Многопоточное программирование на Java /Ср/	4	6	
2.4	Класс Thread и интерфейс Runnable /Лек/	4	2	
2.5	Класс Thread и интерфейс Runnable /Лаб/	4	4	Практическая подготовка
2.6	Класс Thread и интерфейс Runnable /Ср/	4	13	
2.7	Особенности написания многопоточных программ /Лек/	4	2	
2.8	Особенности написания многопоточных программ /Лаб/	4	4	Практическая подготовка
2.9	Особенности написания многопоточных программ /Ср/	4	14	
2.10	Организация пакетов стандартной библиотеки Java /Лек/	4	2	
2.11	Организация пакетов стандартной библиотеки Java /Лаб/	4	4	Практическая подготовка

2.12	Организация пакетов стандартной библиотеки Java /Ср/	4	14	
2.13	Класс String и производительность в Java приложениях /Лек/	4	2	
2.14	Класс String и производительность в Java приложениях /Лаб/	4	4	Практическая подготовка
2.15	Класс String и производительность в Java приложениях /Ср/	4	14	
2.16	Перечисления Java(java.lang.Enum) /Лек/	4	2	
2.17	Перечисления Java(java.lang.Enum) /Лаб/	4	4	Практическая подготовка
2.18	Перечисления Java(java.lang.Enum) /Ср/	4	14	
2.19	Контейнеры и коллекции в Java /Лек/	4	2	
2.20	Контейнеры и коллекции в Java /Лаб/	4	4	Практическая подготовка
2.21	Контейнеры и коллекции в Java /Ср/	4	14	
2.22	Итераторы /Лек/	4	2	
2.23	Итераторы /Лаб/	4	4	Практическая подготовка
2.24	Итераторы /Ср/	4	14	
2.25	Технология коллективной разработки Java приложений /Лек/	4	2	
2.26	Технология коллективной разработки Java приложений /Лек/	4	4	
2.27	Технология коллективной разработки Java приложений /Ср/	4	14	
2.28	Автоматизация сборки и размещения Java приложений /Лек/	4	2	
2.29	Автоматизация сборки и размещения Java приложений /Лаб/	4	4	Практическая подготовка
2.30	Автоматизация сборки и размещения Java приложений /Ср/	4	14	
2.31	Разработка прототипа и приложения (проект) /Лек/	4	2	
2.32	Разработка прототипа и приложения (проект) /Лаб/	4	4	Практическая подготовка
2.33	Разработка прототипа и приложения (проект) /Ср/	4	14	
2.34	Взаимодействие с базой данных /Лек/	4	2	
2.35	Взаимодействие с базой данных /Лаб/	4	4	Практическая подготовка
2.36	Взаимодействие с базой данных /Ср/	4	18	
2.37	Оптимизация приложений /Лек/	4	4	
2.38	Оптимизация приложений /Лаб/	4	4	Практическая подготовка
2.39	Оптимизация приложений /Ср/	4	18	
	Раздел 3. Специальные инструменты и методы современного программирования			
3.1	Системное программирование и язык C/C++. Сравнительный анализ с Java. Вопросы эффективности и безопасности /Лек/	5	8	
3.2	Системное программирование и язык C/C++. Сравнительный анализ с Java. Вопросы эффективности и безопасности /Лаб/	5	6	Практическая подготовка
3.3	Системное программирование и язык C/C++. Сравнительный анализ с Java. Вопросы эффективности и безопасности /Ср/	5	35	
3.4	Основные конструкции и типы данных в C++. Массивы. Синтаксические особенности, стандарты, совместимость. Часто используемые библиотеки и функции /Лек/	5	6	

3.5	Основные конструкции и типы данных в C++. Массивы. Синтаксические особенности, стандарты, совместимость. Часто используемые библиотеки и функции /Лаб/	5	10	Практическая подготовка
3.6	Основные конструкции и типы данных в C++. Массивы. Синтаксические особенности, стандарты, совместимость. Часто используемые библиотеки и функции /Ср/	5	40	
3.7	Объектно-ориентированное программирование в C++. Коллекции. Поток ввода-вывода. Работа с сетью. Низкоуровневое программирование /Лек/	5	10	
3.8	Объектно-ориентированное программирование в C++. Коллекции. Поток ввода-вывода. Работа с сетью. Низкоуровневое программирование /Лаб/	5	20	Практическая подготовка
3.9	Объектно-ориентированное программирование в C++. Коллекции. Поток ввода-вывода. Работа с сетью. Низкоуровневое программирование /Ср/	5	40	
3.10	Python как специализированный язык. Основной синтаксис. Области применения. Библиотеки для анализа данных, распознавания речи и образов /Лек/	5	8	
3.11	Python как специализированный язык. Основной синтаксис. Области применения. Библиотеки для анализа данных, распознавания речи и образов /Лаб/	5	12	Практическая подготовка
3.12	Python как специализированный язык. Основной синтаксис. Области применения. Библиотеки для анализа данных, распознавания речи и образов /Ср/	5	30	
3.13	Выполнение курсовой работы /Ср/	5	35	Практическая подготовка
Раздел 4. Контактные часы на аттестацию				
4.1	Зачет /КЭ/	3	0,15	
4.2	Экзамен /КЭ/	4	2,3	
4.3	Экзамен /КЭ/	5	2,3	
4.4	Курсовая работа /КА/	5	1	

5. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

Оценочные материалы для проведения промежуточной аттестации обучающихся приведены в приложении к рабочей программе дисциплины.

Формы и виды текущего контроля по дисциплине (модулю), виды заданий, критерии их оценивания, распределение баллов по видам текущего контроля разрабатываются преподавателем дисциплины с учетом ее специфики и доводятся до сведения обучающихся на первом учебном занятии.

Текущий контроль успеваемости осуществляется преподавателем дисциплины (модуля) в рамках контактной работы и самостоятельной работы обучающихся. Для фиксирования результатов текущего контроля может использоваться ЭИОС.

6. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

6.1. Рекомендуемая литература

6.1.2. Дополнительная литература

	Авторы, составители	Заглавие	Издательство, год	Эл. адрес
Л2.1	Долгинцев А. П.	Объектно-ориентированное программирование: конспект лекций	Самара: СамГУП С, 2001	http://e.lanbook.com/book/13
Л2.2	Гуськова О. И.	Объектно-ориентированное программирование в Java: учебное пособие	Москва: МПГУ, 2018	http://e.lanbook.com/book/12

6.2 Информационные технологии, используемые при осуществлении образовательного процесса по дисциплине (модулю)	
6.2.1 Перечень лицензионного и свободно распространяемого программного обеспечения	
6.2.1.1	Microsoft Windows10 Pro Договор №034210000481700004
6.2.1.2	Microsoft office 2013 (Лицензия № 61887848) Договор на поставку № 0342100004813000011
6.2.1.3	7-zip (http://www.7-zip.org/ (GNU LGPL license)Microsoft Windows10 Pro Договор №034210000481700004
6.2.1.4	Microsoft office 2013 (Лицензия № 61887848) Договор на поставку № 03421
6.2.2 Перечень профессиональных баз данных и информационных справочных систем	
6.2.2.1	Крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки- https://github.com/
6.2.2.2	База книг и публикаций Электронной библиотеки "Наука и Техника" - http://www.n-t.ru
6.2.2.3	Портал для разработчиков электронной техники: http://www.espec.ws/
6.2.2.4	База данных «Библиотека программиста» https://proglib.io/
6.2.2.5	База данных «Отраслевой портал специалистов» http://www.connect-wit.ru/
6.2.2.6	Гарант.ру https://www.garant.ru/
6.2.2.7	КонсультантПлюс http://www.consultant.ru/
7. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)	
7.1	мультимедийное оборудование для предоставления учебной информации большой аудитории и/или звукоусиливающее оборудование (стационарное или переносное).
7.2	Учебные аудитории для проведения занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, укомплектованные специализированной мебелью и техническими средствами обучения: мультимедийное оборудование и/или звукоусиливающее оборудование (стационарное или переносное)
7.3	Помещения для самостоятельной работы, оснащенные компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду университета.
7.4	Помещения для хранения и профилактического обслуживания учебного оборудования

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ
ПО ДИСЦИПЛИНЕ (МОДУЛЮ)**

**ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ
ПРОГРАММИРОВАНИЕ**

(наименование дисциплины(модуля))

Направление подготовки / специальность

09.03.03 Прикладная информатика

(код и наименование)

Направленность (профиль)/специализация

Управление цифровой инфраструктурой организации

(наименование)

Содержание

- Пояснительная записка.
- Типовые контрольные задания или иные материалы для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих уровень сформированности компетенций.
- Методические материалы, определяющие процедуру и критерии оценивания сформированности компетенций при проведении промежуточной аттестации.

1. Пояснительная записка

Цель промежуточной аттестации – оценивание промежуточных и окончательных результатов обучения по дисциплине, обеспечивающих достижение планируемых результатов освоения образовательной программы.

2. Типовые контрольные задания или иные материалы для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих уровень сформированности компетенций

ОПК-7: Способен разрабатывать алгоритмы и программы, пригодные для практического применения; ОПК-7.1: Разрабатывает алгоритмы и программы на языке программирования высокого уровня
Знать: основные принципы объектно-ориентированного программирования; понятия класса и объекта; принципы построения классов; создание объектов на основе существующих классов, определение отдельных атрибутов;

(Эссе)

Как проверить к какому классу принадлежит объект Java?

(Шаблон ответа)

Ключевое слово instanceof - Проверка принадлежности к классу

Это булев оператор, и выражение `foo instanceof Foo` истинно, если объект `foo` принадлежит классу `Foo` или его наследнику, или реализует интерфейс `Foo` (или, в общем виде, наследует класс, который реализует интерфейс, который наследует `Foo`).

(Эссе)

Опишите для чего нужен модификатор `protected`?

(Шаблон ответа)

Метод или поле, отмеченные модификатором `protected` доступны из любого места в текущем классе или пакете или в производных классах, даже если они находятся в других пакетах.

(Эссе)

Опишите какую функцию в языке Java выполняет служебное слово extends?

(Шаблон ответа)

Чтобы объявить один класс наследником от другого, надо использовать после имени класса наследника ключевое слово extends, после которого идет имя базового класса.

Запись class Employee extends Person означает что класс Employee наследуется от класса Person.

(Эссе)

Что делает instanceof Java?

(Шаблон ответа)

Оператор instanceof проверяет, принадлежит ли объект к определённому классу. Другими словами, object instanceof constructor проверяет, присутствует ли объект constructor.prototype в цепочке прототипов object .

(Эссе)

Что делает метод void?

(Шаблон ответа)

Метод void можно использовать в качестве возвращаемого типа метода (или локальной функции) для определения того, что метод не возвращает значение. Вы также можете использовать void как ссылочный тип для [объявления](#) того, что тип указателя неизвестен.

(Эссе)

Что такое перегрузка метода в Java?

(Шаблон ответа)

Java разрешает определение внутри одного класса двух или более методов с одним именем, если только [объявления](#) их параметров различны. В этом случае методы называют перегруженными, а процесс — перегрузкой методов. Перегрузка методов — один из способов поддержки полиморфизма в Java.

(Эссе)

Что такое переопределение метода в Java?

(Шаблон ответа)

Переопределение метода это возможность реализовать метод так, чтобы он имел идентичную сигнатуру с методом класса-предка, но предоставлял иное поведение, не вызывая коллизий при его использовании. Так же это один из инструментов реализации полиморфизма.

(Эссе)

Что такое суперкласс?

(Шаблон ответа)

В программировании суперклассом или родительским классом называют класс, на основе которого создаются другие классы. Классы, полученные на основе суперкласса, называются дочерними классами, производными классами или подклассами.

(Множественный выбор / Только один ответ)

Для чего используется оператор NEW?

a. (0%)

это антагонист оператора OLD

b. (0%)

для объявления нового класса

c. (100%)

для создания экземпляра класса

d. (0%)

для создания новой переменной

(Множественный выбор / Только один ответ)

Как вызвать static-метод внутри обычного?

a. (100%)

можно, ничего дополнительно делать не надо

b. (0%)

можно, надо перед этим перегрузить обычный метод класса

c. (0%)

никак, static-метод можно вызвать только от объекта класса

d. (0%)

можно, надо перед этим переопределить обычный метод класса

(Множественный выбор / Только один ответ)

Как называется шаблонная конструкция, которая позволяет описать в программе объект, его свойства и поведение?

a. (0%)

родительский объект

b. (0%)

декоратор

c. (0%)

шаблон

d. (100%)

класс

(Множественный выбор / Только один ответ)

Как объявить класс в коде?

a. (100%)

```
class MyClass {}
```

b. (0%)

```
select * from class MyClass {}
```

c. (0%)

```
new class MyClass {}
```

d. (0%)

MyClass extends class {}

(Множественный выбор / Только один ответ)

Отметьте базовый принцип ООП, заключающийся в способности программы идентично использовать объекты с одинаковым интерфейсом без информации о конкретном типе этого объекта? а. (0%)

абстракция

b. (100%)

полиморфизм

c. (0%)

наследование

d. (0%)

инкапсуляция

(Множественный выбор)

Отметьте верные утверждения.

a. (-100%)

в одном классе может быть только один конструктор

b. (50%)

конструктор по умолчанию исчезает после создания авторского конструктора с аргументами с. (50%)

в Java у всех классов есть конструктор по умолчанию

d. (-100%)

в функции-конструкторе обязательно должен быть хотя бы один аргумент

(Множественный выбор)

Отметьте строчки, которые могут быть идентификаторами (именами) переменных в Java? а. (-100%)

name

b. (33.33333%)

name2

c. (33.33333%)

\$cash

d. (33.33333%)

_count e.

(-100%)

true

(Множественный выбор)

Укажите, что из перечисленного является атрибутом класса

```
public class Dog {  
  
    String name;  
    int breed;  
  
}
```

a. (50%)

name

b. (50%)

breed

c. (-100%)

Dog

d. (-100%)

(Множественный выбор / Только один ответ)

Что из перечисленного является экземпляром класса?

a. (0%)

атрибут

b. (100%)

объект

c. (0%)

свойство

d. (0%)

константа

(Множественный выбор / Только один ответ)

Что означает ключевое слово «static» перед переменной?

a. (100%)

переменная принадлежит классу, а не экземпляру этого класса

b. (0%)

словом «static» обозначаются константы

c. (0%)

переменная принадлежит строго одному объекту

d. (0%)

переменная переменная не может быть унаследована

(Короткий ответ)

Как в ООП называются функции, объявление которых размещено внутри определения класса или структуры?

Ответ: метод

(Короткий ответ)

Как называется базовый принцип ООП, характеризующийся способностью объекта или класса базироваться на другом объекте или классе?

Ответ: наследование

(Короткий ответ)

Какое служебное слово, использующееся только в составе методов либо конструкторов экземпляра, представляет ссылку на текущий экземпляр класса?

Ответ: this

(Короткий ответ)

Какой модификатор делает доступными поля и методы класса только внутри данного класса?

Ответ: private

(Короткий ответ)

Отметьте верные способы обратиться к атрибуту name объекта cat.

Ответ: cat.name

(Короткий ответ)

Укажите модификатор доступа, делающий поле или метод открытым.

Ответ: public

ПК-4: Способен проводить тестирование компонентов программного обеспечения ИС ПК-4.1: Проводит модульное тестирование программного обеспечения ИС, интеграционное тестирование Знать: понятия сплошного и выборочного ручного тестирования, средства автоматизированного тестирования Java (JUnit и др)

(Множественный выбор / Только один ответ)

В системе багтрекинга приоритет задачи, созданной по сигнатуре, может определяться: а. (100%)

Комбинацией числа ошибок с данной сигнатурой и числа уникальных IP-адресов b. (0%) IP-адресом

источника

c. (0%)

Давностью возникновения ошибки

(Множественный выбор / Только один ответ)

Дайте определение верификации как вида деятельности

a. (100%)

это процесс доказательства того, что программная реализация системы и требования на систему соответствуют

друг другу и проектным стандартам b. (0%)

это процесс поиска и документирования дефектов программной реализации разрабатываемой системы

c. (0%)

это процесс поиска и исправления ошибок в проектной документации и программной реализации системы

d. (0%)

это процесс доказательства того, что программная система соответствует ожиданиям пользователя или заказчика

(Множественный выбор)

Какие цели и задачи достигаются и решаются в процессе управления конфигурациями?

a. (33.33333%)

обеспечение совместной работы коллектива разработчиков

b. (33.33333%)

управление изменениями данных

c. (33.33333%)

обеспечение целостности данных

d. (-100%)

обеспечение целостности технологических процессов

(Множественный выбор)

Какой из перечисленных методов будет успешно компилироваться? (Отметьте все варианты) а. (50%)

public void methodD() {}

b. (-100%)

public int methodE() { return 9.0;}

c. (50%)

public int methodD() { return 9;}

d. (-100%)

public int methodF() { return;}

e. (-100%)

public int methodG() { return null;}

f. (-100%)

public void methodA() { return;}

g. (-100%)

public void methodB() { return null;}

(Множественный выбор)

Модульное тестирование предназначено для

a. (-100%)

для тестирования модуля в условиях отсутствия воздействия со стороны пользователя b. (50%)

для максимальной изоляции побочного влияния на функционирование модуля со стороны остальных частей системы

c. (50%)

проверки функционирования каждого независимого программного модуля d. (-100%)

проверки функционирования одного замкнутого участка программного кода

(Множественный выбор / Только один ответ)

Тест, который проверяет, что функция деления при запросе 2/0 вернет ArithmeticException, это: a. (0%)

нагрузочный тест

b. (0%)

функциональный тест

c. (100%)

Unit тест

(Множественный выбор)

Укажите основные свойства роли «Тестировщик»

a. (33.33333%)

выявляет дефекты в системе

b. (33.33333%)

участвует в разработке функциональных требований

c. (33.33333%)

создает отчеты о найденных дефектах

d. (-100%)

устраняет обнаруженные дефекты в системе

(Множественный выбор)

Что из перечисленных конструкций будет скомпилировано без ошибок? a. (-100%)

`void public method() { }`

b. (50%)

`static final void method3() { }`

c. (-100%)

`void final method() {}`

d. (-100%)

```
private void int method() { }
```

e. (-100%)

```
public final int void method() { }
```

f. (50%)

```
final static void method4() { }
```

(Множественный выбор)

Что из перечисленных конструкций будет скомпилировано без ошибок? (Отметьте все варианты) а. (-100%)

```
public void moreC(int... nums, String values) {}
```

b. (-100%)

```
public void moreF(String... values, int[] nums) {}
```

c. (33.33333%)

```
public void moreG(String[] values, int[] nums) {}
```

d. (-100%)

```
public void moreE(String[] values, ...int nums) {}
```

e. (33.33333%)

```
public void moreA(int... nums) {}
```

f. (33.33333%)

```
public void moreB(String values, int... nums) {}
```

g. (-100%)

```
public void moreD(String... values, int... nums) {}
```

(Множественный выбор)

Что такое управление версиями?

a. (-100%)

автоматизированный процесс трансформации исходных текстов ПО в пакет исполняемых модулей

b. (50%)

одна из задач конфигурационного управления

c. (50%)

управление версиями файлов

Уметь: создавать собственные классы и объекты на языке программирования высокого уровня;

(Эссе)

Напишите класс Pet с методом walk, выводящим на экран фразу "Я буду гулять" и дочерний по отношению к классу Pet класс Cat, с методом mur, выводящим на экран фразу "Я буду мурчать", и дочерний по отношению к классу Cat класс Kitty, с методом sleep, выводящим на экран фразу "Я буду спать"

(Шаблон ответа)

```
class Pet{ public void walk (){System.out.println("Я буду гулять");} class
Cat extends Pet{ public void mur (){System.out.println("Я буду
мурчать");} class Kitty extends Cat{

public void sleep (){System.out.println("Я буду спать");}
}
```

(Эссе)

Напишите класс Pet с методом walk, выводящим на экран фразу "Я буду гулять" и дочерний по отношению к классу Pet класс Cat, с методом mur, выводящим на экран фразу "Я буду мурчать"

(Шаблон ответа)

```
class Pet{ public void walk (){System.out.println("Я буду гулять");} class
Cat extends Pet{ public void mur (){System.out.println("Я буду
мурчать");}
}
```

(Эссе)

Реализуйте класс Counter, в котором будет счётчик count, ведущий учет вызовов метода.

(Шаблон ответа)

```
public class Counter { static int
count;

public static void invokeCounter() {
    count++;
    System.out.println("Текущее значение счётчика - " + count); }
}
```

(Эссе)

Создай класс Artifact, описывающий хранящиеся в музее древности, и напиши необходимое количество конструкторов для него. Артефакты, которые хранятся в музее, бывают трех видов. Первый — о которых неизвестно ничего, кроме порядкового номера, присвоенному музеем (например:

212121). Второй — о которых известен порядковый номер и культура, которой он был создан (например: 212121, "Ацтеки"). Третий вид — о которых известен порядковый номер, культура, которой он был создан, и точный век его создания (например: 212121, "Ацтеки", 12). В методе main() создай по одному артефакту каждого вида.

(Эссе)

Спроектируйте класс Cat, у которого будут поля — name (String) age (int). Разрешите добавлять объекты как с указанием имени и возраста, так и без (необходимо наличие двух конструкторов).

(Шаблон ответа)

```
public class Cat {

    String name; int
age;

    //для домашних котов
    public Cat(String name, int age) {
        this.name = name;
        this.age = age; }

    //для уличных котов
    public Cat() { }

    public static void main(String[] args) {

        Cat barsik = new Cat("Barsik", 5);
        Cat streetCat = new Cat();
    }
}
```

(Эссе)

Спроектируйте класс User, у которого будут поля — имя (String) возраст (short) и рост (int). Создайте для него необходимое количество конструкторов, чтобы имя, возраст и рост можно было указывать в любом порядке.

Уметь: использовать средства автоматизированного тестирования Java (JUnit и др);

(Эссе)

Создайте программу, осуществляющую чтение текстового файла и загрузку всех его слов (разделители – пробелы, знаки препинания удаляются) в соответствующую коллекцию. Организовать подсчет чаще всего встречающегося слова. Интерфейс программы – нативное десктоп-приложение. В программе должны присутствовать методы для чтения и загрузки слов файла в коллекцию, для преобразования коллекции в структуру «ключ»-«значение», где «значение» – частота встречаемости слова, для поиска чаще всего встречающихся слов.

Предусмотреть автоматизированное тестирование средствами JUnit всех указанных методов. Окно диалога программы должно включать поле ввода для имени файла (с вызовом диалога открытия файла ОС), а также текстовое поле для вывода чаще всего встречающихся слов. Провести ручное тестирование для случаев а) корректного и некорректного имени файла б) пустого файла в) файла, в которых слова встречаются по одному разу. Оформить результаты эргономического тестирования приложения.

(Шаблон ответа)

Приложение создано, отлажено, работает, проходит тесты при корректных входных данных

Подготовлены тесты JUnit и успешно пройдены

Выполнено (и пройдено) ручное тестирование

Корректно оформлены результаты эргономического тестирования приложения

Владеть навыками: алгоритмизации и программной реализации практических задач на языке высокого уровня

(Эссе)

Напишите класс BaseConverter для конвертации из градусов по Цельсию в Кельвины, Фаренгейты, и так далее. У класса должен быть метод convert, который и делает конвертацию.

(Эссе)

Напишите метод fill, который принимает массив объектов, и реализацию интерфейса Function (или своего). Метод fill должен заполнить массив, получая новое значение по индексу с помощью реализации интерфейса Function. То есть, использовать его хочется так:

```
public static void main(String[] args) {  
    Integer[] squares = new Integer[100];  
    fill(squares, integer -> integer * integer); // 0, 1, 4, 9, 16 .....  
}
```

(Шаблон ответа)

```
public static <T> void fill(T[] objects, Function<Integer, ? extends T> function) { for(int i = 0; i < objects.length; i++){ objects[i] = function.apply(i); } }
```

(Эссе)

Напишите метод filter, который принимает на вход массив (любого типа) и реализацию интерфейса Filter с методом apply(Object o), чтобы убрать из массива лишнее.

Проверьте как он работает на строках или других объектах.

(Шаблон ответа)

```
public class Main {

    public static <T> T[] filter(T[] array, Function<? super T, Boolean> filter) {        int offset = 0;

        for (int i = 0; i < array.length; i++) {
            if (!filter.apply(array[i])) {
                offset++;
            } else {
                array[i - offset] = array[i];
            }
        }

        // Arrays.copyOf копирует значение из массива array в новый массив
        // с длиной array.length - offset
        return Arrays.copyOf(array, array.length - offset);    }

    public static void main(String[] args) {
        String array[] =
            new String[]{"1rewf ", "feefewf", "a", null, "1"};

        String[] newArray = filter(array, s -> s != null);    }
}
```

(Эссе)

Напишите свой класс StringBuilder, с возможностью оповещения других объектов об изменении своего состояния. Для этого делегируйте все методы стандартному StringBuilder, а в собственном классе реализуйте шаблон проектирования «Наблюдатель».

(Эссе)

1. Создайте пример наследования, реализуйте класс Student и класс Aspirant, аспирант отличается от студента наличием некой научной работы.

2. Класс Student содержит переменные: String firstName, lastName, group. А также, doubleaverageMark, содержащую среднюю оценку.
3. Создать метод getScholarship() для класса Student, который возвращает сумму стипендии. Если средняя оценка студента равна 5, то сумма 2000 грн, иначе 1900. Переопределить этот метод в классе Aspirant. Если средняя оценка аспиранта равна 5, то сумма 2500 грн, иначе 2200.
4. Создать массив типа Student, содержащий объекты класса Student и Aspirant. Вызвать метод getScholarship() для каждого элемента массива.

3. Методические материалы, определяющие процедуру и критерии оценивания сформированности компетенций при проведении промежуточной аттестации

Критерии формирования оценок по ответам на вопросы, выполнению тестовых заданий

- оценка **«отлично»** выставляется обучающемуся, если количество правильных ответов на вопросы составляет **100 – 90%** от общего объема заданных вопросов;
- оценка **«хорошо»** выставляется обучающемуся, если количество правильных ответов на вопросы – **89 – 76%** от общего объема заданных вопросов;
- оценка **«удовлетворительно»** выставляется обучающемуся, если количество правильных ответов на тестовые вопросы – **75–60 %** от общего объема заданных вопросов;
- оценка **«неудовлетворительно»** выставляется обучающемуся, если количество правильных ответов – менее **60%** от общего объема заданных вопросов.

Критерии формирования оценок по результатам выполнения заданий

«Отлично/зачтено» – ставится за работу, выполненную полностью без ошибок и недочетов.

«Хорошо/зачтено» – ставится за работу, выполненную полностью, но при наличии в ней не более одной негрубой ошибки и одного недочета, не более трех недочетов.

«Удовлетворительно/зачтено» – ставится за работу, если обучающийся правильно выполнил не менее 2/3 всей работы или допустил не более одной грубой ошибки и двух недочетов, не более одной грубой и одной негрубой ошибки, не более трех негрубых ошибок, одной негрубой ошибки и двух недочетов.

«Неудовлетворительно/не зачтено» – ставится за работу, если число ошибок и недочетов превысило норму для оценки «удовлетворительно» или правильно выполнено менее 2/3 всей работы.

Виды ошибок:

- *грубые ошибки: незнание основных понятий, правил, норм; незнание приемов решения задач; ошибки, показывающие неправильное понимание условия предложенного задания.*
- *негрубые ошибки: неточности формулировок, определений; нерациональный выбор хода решения.*
- *недочеты: нерациональные приемы выполнения задания; отдельные погрешности в формулировке выводов; небрежное выполнение задания.*