МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ИМЕНИ В.Ф. УТКИНА»

Кафедра вычислительной и прикладной математики (ВПМ)

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ**

Современные технологии разработки программного обеспечения

Направление подготовки

09.03.03 Прикладная информатика

Направленность (профиль) подготовки

Прикладная информатика

Квалификация выпускника – бакалавр

Формы обучения – очная

Рязань

# ОБЩИЕ ПОЛОЖЕНИЯ

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

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

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

Контроль знаний обучающихся проводится в форме промежуточной аттестации – экзамен в семестре, указанном на титульном листе рабочей программе дисциплины (РПД).

# ОПИСАНИЕ ПОКАЗАТЕЛЕЙ И КРИТЕРИЕВ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ

Сформированность каждой компетенции в рамках освоения данной дисциплины оценивается по следующим шкалам.

## Для сдачи практических работ

| **Шкала оценивания** | **Критерии оценивания** |
| --- | --- |
| «зачтено» | 1. Практическая работа выполнена правильно. Студент ответил на все вопросы преподавателя по теме практической работы
 |
| «незачтено» | 1. В иных случаях
 |

## Для тестовых заданий

| **Шкала оценивания** | **Критерии оценивания** |
| --- | --- |
| «отлично» | 1. Студент ответил правильно на 90%-100% тестовых заданий
 |
| «хорошо» | 1. Студент ответил правильно на 75%-89,(9)% тестовых заданий
 |
| «удовлетворительно» | 1. Студент ответил правильно на 60%-74,(9)% тестовых заданий
 |
| «неудовлетворительно» | 1. В иных случаях
 |

## Для промежуточной аттестации

В качестве критериев оценивания промежуточной аттестации используются критерии оценивания из п. 2.11 Положения о промежуточной аттестации обучающихся по образовательным программам высшего образования РГРТУ. Выпуск 3.

# ПАСПОРТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ ПО ДИСЦИПЛИНЕ

Разделы и темы (виды занятия), соответствующие им компетенции и формы контроля приведены в разделе 4 РПД.

# ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ

## Промежуточная аттестация (вид указан в п. 1)

|  |
| --- |
| **ПК-1: Способен разрабатывать требования, проектировать и выполнять программную реализацию программного обеспечения** |
| **ПК-1.1. Анализирует требования к программному обеспечению** |

### Вопросы закрытого типа

Как объявить класс в коде?

**class MyClass {}**

new class MyClass {}

select \* from class MyClass {}

MyClass extends class {}

Что выведет следующий фрагмент программы?

int a = 5; int b = 5 + a;

if (a > b) {

System.out.println("a > b");

} else if (a < b) {

System.out.println("a < b");

} else {

}

System.out.println("a = b");

Код написан с ошибкой, ничего не выведется.

a > b

a = b

**a < b**

Для чего используется оператор NEW?

Для создания новой переменной.

Для объявления нового класса.

**Для создания экземпляра класса.**

Это антагонист оператора OLD.

Что означает ключевое слово extends?

**Что данный класс наследуется от другого.**

Что это дополнительный модуль класса, который расширяет его свойства.

Что два класса делают одно и то же.

Что это самый большой класс в программе.

Сколько параметров может принимать функция?

Не более 10

Не более 20

Не более 5

Не более 3

**Неограниченное количество**

### Вопросы открытого типа

Для чего необходимо ключевое слово this

**Это указатель на текущий объект класса внутри самого класса. Его можно опускать при вызове метода класса, но лучше этого не делать.**

Как создать массив 5 чисел типа int и заполнить его значениями 1, 2, 3, 4, 5?

int[] a = new int[] {1, 2, 3, 4, 5};

Что будет выведено на экран после запуска этого фрагмента?

int i = 1;

{

 int i = 2;

 System.out.println("1. i=" + i);

 i = 3;

 System.out.println("2. i=" + i);

}

System.out.println("3. i=" + i);

**Ничего. Будет ошибка компиляции из-за дублирования переменной.**

|  |
| --- |
| **ПК-1: Способен разрабатывать требования, проектировать и выполнять программную реализацию программного обеспечения** |
| **ПК-1.3. Проектирует программное обеспечение и выполняет его программную реализацию** |

### Вопросы закрытого типа

Что выведет следующий фрагмент программы?

int[] a = new int[] {4,5,6,7,8};

int[] b = new int[a.length];

for (int i = 0; i < a.length; ++i) {

b[i] = a[i] \* 2;

}

Удваивает все значения массива a

Удваивает все значения массива b

**Заполняет массив b удвоенными значениями массива a**

Заполняет массив a удвоенными значениями массива b

Будет ли выведено сообщение об ошибке после запуска следующего фрагмента и по какой причине?

int[] a = new int[] {1, 2, 3};

int[] b = new int[a.length];

b[4] = 4;

Ошибка означает переполнение памяти.

**Ошибка означает выход за рамки границ массива.**

Ошибка означает попытку вставить в массив некорректное значение.

На самом деле этот код не вызовет ошибку.

Для чего можно использовать Java?

Для создания сайтов

**Для всего перечисленного**

Для создания программ под ПК

Для создания игр

Только для создания игр и программ

Где правильно создана простая переменная?

char str = 'ab';

**bool isDone = true;**

int[] a;

**float x = 23.3f;**

byte x = 100000;

### Вопросы открытого типа

Что означает перегрузка метода в Java (overload).

**Несколько методов с одинаковым названием, но разным набором параметров.**

Что означает переопределение метода в Java (override)?

**Изменение поведения метода класса относительно родительского.**

Зачем нужны спецификаторы доступа?

**Ограничивать доступ к элементам класса.**

Фрагмент вызовет ошибку компиляции. В чем эта ошибка состоит?

double doubleNumber;

int integerNumber1, integerNumber2;

char aChar;

real realNumber;

Double anotherDouble;

float f1, f2;

final int intValue;

**В Java нет типа real.**

В чем особенность переменной, объявленной так final int intValue;?

**Объявлена константа.**

|  |
| --- |
| **ПК-4: Способен управлять проектами в области информационных технологий на основе полученных планов проектов** |
| **ПК-4.1. Инициирует, планирует и организует исполнение работ проекта** |

### Вопросы закрытого типа

Чем отличаются static-метод класса от обычного метода класса?

Поведение обычного метода класса можно изменить в классе-наследнике, а поведение static-метода нельзя.

Обычный метод класса можно переопределить, а static-метод нельзя.

**Обычный метод класса работает от объекта класса, а static-метод от всего класса.**

Static-метод класса можно вызывать только внутри класса, а обычный - в любой части кода.

Как вызвать static-метод внутри обычного?

Никак, static-метод можно вызвать только от объекта класса.

Можно, надо перед этим перегрузить обычный метод класса.

Можно, надо перед этим переопределить обычный метод класса.

**Можно, ничего дополнительно делать не надо.**

Как вызвать обычный метод класса внутри static-метода?

**Никак, static-метод не работает с объектом класса.**

Можно, надо перед этим перегрузить обычный метод класса.

Можно, надо перед этим переопределить обычный метод класса.

Можно, ничего дополнительно делать не надо.

В чем здесь ошибка?

int a, b;

System.out.print("Введите первое число: ");

Scanner num = new Scanner(System.in);

a = num.nextFloat();

Ошибок нет

Вместо System.in надо использовать System.out

int a, b – необходимо записывать по отдельности

**Вместо nextFloat надо использовать nextInt**

Каждый файл должен называться...

как вам захочется

**по имени класса в нём**

по имени названия пакета

по имени первой библиотеки в нём

по имени основного метода в нем

### Вопросы открытого типа

Что выведет следующий фрагмент программы?

int a = 9;

boolean isDone = false;

if (a % 3 != 0 || !isDone)

 System.out.print("Готово");

**Надпись «Готово»**

Сколько деструкторов у классов на языке Java?

**В языке Java у классов нет деструкторов.**

Почему нет деструкторов в языке Java? Как удаляются объекты?

**Объекты удаляются автоматически после того, как на них не будет ни одной ссылки.**

Зачем нужен конструктор в классе?

**Для инициализации полей.**

Почему не стоит в конструкторе выполнять другие действия, помимо инициализации полей?

**Эти действия могут привести к появлению исключения. Поэтому объект может быть не создан, что влечет необходимость написания дополнительного кода для проверки, создался ли объект.**

## Типовые вопросы к экзамену

1. Структура класса. Объекты класса. Класс Object. Спецификаторы доступа.
2. Методы. Назначение методов. Методы для получения и установки значений полей. Ключевое слово this.
3. Пакеты. Понятие и назначение пакета. Короткие и полные имена. Область видимости. Правила именования пакета. Директива import. Структура программы на языке Java. Метод main.
4. Java Code Conventions. Правило именования элементов программы на языке Java. Комментарии в программе.
5. Типы переменных Переменные и константы. Преобразования типов.
6. Базовые и ссылочные типы. Объявление и инициализация переменных. Объявление и инициализация констант. Преобразования базовых типов. Классы-оболочки базовых типов. Константный тип
7. Операции. Операция присваивания. Арифметические операции. Инкремент и декремент. Операции сравнения. Логические операции. Тернарная операция «?:».
8. Ввод и вывод данных на консоль. Стандартные потоки языка Java. Вывод данных на консоль. Ввод данных с консоли.
9. Строки. Класс String. Особенности строк класса String. Инициализация строк. Операции со строками. Сравнение строк. Метод toString. Классы StringBuilder и StringBuffer.
10. Управляющие конструкции. Операторы ветвления. Условный оператор. Оператор выбора. Операторы цикла. Программный блок.
11. Массивы. Объявление массивов и инициализация их элементов. Цикл перебора элементов. Класс Arrays. Интерфейсы Comparable и Comparator. Многомерные массивы.
12. Основные принципы объектно-ориентированного программирования. Ключевое слово super.
13. Методы и конструкторы с неизвестным количеством параметров. Абстрактные классы и методы. Статические методы и поля. Классы, методы и поля со спецификатором final.
14. Интерфейсы. Перечисления Enum. Методы класса Enum.
15. Класс Locale. Файлы ресурсов.
16. Исключения. Назначение исключений. Конструкция try-catch-finally. Оператор throw. Директива throws.
17. Работа с файлами. Символьные и байтовые потоки. Класс File. Буферизированные символьные потоки.
18. Коллекции. Параметризация. Виды коллекций в Java. Класс Collections и его методы.