

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
**«Рязанский государственный радиотехнический университет  
имени В.Ф. Уткина»**

КАФЕДРА ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ**

по дисциплине  
**«Функциональное программирование»**

Направление подготовки  
02.03.03 «Математическое обеспечение и администрирование  
информационных систем»

ОПОП академического бакалавриата  
«Программное обеспечение компьютерных  
технологий и систем искусственного интеллекта»

Квалификация (степень) выпускника – бакалавр

Форма обучения – очная

Рязань

## **1 ОБЩИЕ ПОЛОЖЕНИЯ**

Оценочные материалы – это совокупность учебно-методических материалов (практических заданий, описаний форм и процедур проверки), предназначенных для оценки качества освоения обучающимися данной дисциплины как части ОПОП.

Цель – оценить соответствие знаний, умений и владений, приобретенных обучающимся в процессе изучения дисциплины, целям и требованиям ОПОП в ходе проведения промежуточной аттестации.

Основная задача – обеспечить оценку уровня сформированности общепрофессиональных и профессиональных компетенций.

Контроль знаний обучающихся проводится в форме промежуточной аттестации.

Промежуточная аттестация проводится в форме зачета. Форма проведения зачета – тестирование, письменный опрос по теоретическим вопросам и выполнение практических заданий.

## **2 ОПИСАНИЕ ПОКАЗАТЕЛЕЙ И КРИТЕРИЕВ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ**

Сформированность каждой компетенции (или ее части) в рамках освоения данной дисциплины оценивается по трехуровневой шкале:

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

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

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

**Уровень освоения компетенций, формируемых дисциплиной:**

**Описание критериев и шкалы оценивания тестирования:**

<b>Шкала оценивания</b>	<b>Критерий</b>
3 балла (эталонный уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 85 до 100%
2 балла (продвинутый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 70 до 84%
1 балл (пороговый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 50 до 69%
0 баллов	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 0 до 49%

**Описание критериев и шкалы оценивания теоретического вопроса:**

<b>Шкала оценивания</b>	<b>Критерий</b>
3 балла (эталонный уровень)	выставляется студенту, который дал полный ответ на вопрос, показал глубокие систематизированные знания, смог привести примеры, ответил на дополнительные вопросы преподавателя
2 балла (продвинутый уровень)	выставляется студенту, который дал полный ответ на вопрос, но на некоторые дополнительные вопросы преподавателя ответил только с помощью наводящих вопросов
1 балл (пороговый уровень)	выставляется студенту, который дал неполный ответ на вопрос в билете и смог ответить на дополнительные вопросы только с помощью преподавателя
0 баллов	выставляется студенту, который не смог ответить на вопрос

### **Описание критериев и шкалы оценивания практического задания:**

Шкала оценивания	Критерий
3 балла (эталонный уровень)	Задача решена верно
2 балла (продвинутый уровень)	Задача решена верно, но имеются неточности в логике решения
1 балл (пороговый уровень)	Задача решена верно, с дополнительными наводящими вопросами преподавателя
0 баллов	Задача не решена

На промежуточную аттестацию (зачет) выносится тест, два теоретических вопроса и 2 задачи. Максимально студент может набрать 15 баллов. Итоговый суммарный балл студента, полученный при прохождении промежуточной аттестации, переводится в традиционную форму по системе «зачтено» и «не зачтено».

**Оценки «зачтено»** заслуживает обучающийся, продемонстрировавший полное знание материала изученной дисциплины, усвоивший основную литературу, рекомендованную рабочей программой дисциплины; показавший систематический характер знаний по дисциплине, ответивший на все вопросы билета или допустивший погрешности в ответах на вопросы, но обладающий необходимыми знаниями для их устранения под руководством преподавателя.

Дополнительным условием получения оценки «зачтено» могут стать успехи при выполнении лабораторных работ, систематическая активная работа на лабораторных работах.

Оценка «зачтено» выставляется студенту, набравшему 8 и более баллов при промежуточной аттестации.

**Оценки «не зачтено»** заслуживает обучающийся, продемонстрировавший серьезные пробелы в знаниях основного материала изученной дисциплины, не ответивший на все вопросы билета и дополнительные вопросы. Как правило, оценка «не зачтено» ставится обучающимся, которые не могут продолжить обучение по образовательной программе без дополнительных занятий по соответствующей дисциплине (формирования и развития компетенций, закрепленных за данной дисциплиной).

Оценка «не зачтено» выставляется студенту, набравшему менее 8 баллов при промежуточной аттестации.

### **3 ПАСПОРТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ ПО ДИСЦИПЛИНЕ**

Контролируемые разделы (темы) дисциплины	Код контролируемой компетенции (или её части)	Вид, метод, форма оценочного мероприятия
Тема 1. Основы функционального программирования	ОПК-2.1, ОПК-2.2	Зачет
Тема 2. Функциональное программирование в сравнении с объектно-ориентированным программированием	ОПК-2.1, ОПК-2.2	Зачет
Тема 3. Функции, замыкания и области видимости	ОПК-2.1, ОПК-2.2	Зачет
Тема 4. Концепции функционального программирования	ОПК-2.1, ОПК-2.2	Зачет

Тема 5. JavaScript как язык для реализации	ОПК-2.1, ОПК-2.2	Зачет
--	------------------	-------

модели клиент-сервер		
Тема 6. Функции JavaScript	ОПК-2.1, ОПК-2.2	Зачет
Тема 7. Объекты JavaScript	ОПК-2.1, ОПК-2.2	Зачет
Тема 8. NodeJS и работа с файловой системой	ОПК-2.1, ОПК-2.2	Зачет

#### **4 ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ** **Промежуточная аттестация в форме экзамена**

<b>Код компетенции</b>	<b>Результаты освоения ОПОП</b> <b>Содержание компетенций</b>	
ОПК-2.1	Применяет современный математический аппарат, связанный с проектирование и оценкой качества программных продуктов и программных комплексов в различных областях деятельности	

**Типовые тестовые вопросы:**

1. Какой метод решения задач особенно характерен для функционального программирования?

- +символьная обработка информации;
- преобразование состояний памяти;
- обработка сигналов;
- симплекс-метод.

2. Какой из перечисленных механизмов реализации языка Лисп способствовал практическому успеху функционального программирования?

- +автоматизация повторного использования памяти — «сбор мусора»;
- математическая основа исходных понятий;
- отсутствие «синтаксического сахара» в представлении программ;
- бэктрекинг.

3. Какая особенность функционального подхода дает путь к достижению надежности программ?

- +доказательность основных построений при разработке универсальных функций;
- разнообразие встроенных функций и библиотек;
- высокий уровень языковых средств;
- использование рекурсии.

4. Кто впервые сформулировал идеи языка программирования, послужившие основой для функционального программирования?

- Николас Вирт;
- Тони Хоар;
- +Джон Мак-Карти.
- Норберт Винер.

5. Какой из перечисленных механизмов реализации языка Лисп способствовал практичности функционального программирования?

+традиционное включение в реализацию и интерпретатора, и компилятора одновременно;  
полнота средств управления вычислениями;  
отсутствие «синтаксического сахара» в представлении программ;  
бэктрекинг.

6. В каком языке программирования впервые поддержаны идеи функционального программирования?

- +Lisp;
- Python;
- Prolog;
- JavaScript.

7. Какова основная структура данных языка Lisp?

- +список;
- атом;
- таблица;
- массив.

8. Где содержатся пары «переменная-значение» в элементарном языке Lisp?

- в базе данных переменных;
- +в ассоциативном списке;
- в хэш-таблице;
- в стеке.

9. Какой подход в построении программ преимущественно используется в Lisp-системах программирования?

- объектно-ориентированный;
- +функциональный;
- императивный;
- рекурсивно-логический.

10. Что понимается под термином «форма» в языке Lisp?

- список из идентификатора функции, ее аргументов и ее всевозможных значений;
- +список из представления функции и перечня ее аргументов;
- программа на языке Lisp;
- отдельный модуль программы.

11. Что называют программой на языке Lisp?

- последовательность вычисляемых атомов;
- последовательность хорновских дизъюнктов;
- последовательность вызовов функций;
- +последовательность вычисляемых форм и выражений.

12. Какой подход в построении программ наиболее подходит для использования в Lisp-системах программирования?

- императивный;
- интерфейсный;
- +функциональный;
- декларативный.

13. Какой принцип является основным для функционального программирования?

- представление функции строится заранее;
- +представление функции строится и вычисляется также как и представление данных;
- при символьном представлении данных представление функции нельзя построить теми же средствами, что и обработку значений;
- представление функции строится на основе рекурсии.

14. Что называют элементарными функциями в Lisp-системе?  
любые неделимые элементы — атомы;  
встраиваемые S-выражения, имеющие определения на уровне исполнимого кода;  
+атомы, определенные как подпрограммы на уровне исполняемого кода;  
функции без параметров или с одним параметром.
15. Данные какого вида и типа могут выступать в качестве аргументов функции в
- справочные домены;  
+любая  
форма  
произвольно  
й длины и  
сложности;  
числа и  
строки;  
только атомы.

### **Типовые практические задания:**

#### **Задание 1**

При помощи цикла for выведите чётные числа от 2 до 10.

#### **Критерии выполнения задания 1**

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

#### **Задание 2**

Натуральное число, большее 1, называется простым, если оно ни на что не делится, кроме себя и 1. Другими словами,  $n > 1$  – простое, если при делении на любое число от 2 до  $n - 1$  есть остаток. Создайте код, который выводит все простые числа из интервала от 2 до 10. Результат должен быть: 2,3,5,7.

#### **Критерии выполнения задания 2**

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

#### **Задание 3**

Реализовать функцию square(array), возвращающую массив, состоящий из квадратов всех элементов исходного массива. Попробуйте использовать для этого метод массива map.

#### **Критерии выполнения задания 3**

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

#### **Задание 4**

Удалите из строки 'Я очень люблю программировать на Javascript' все гласные буквы.

***Критерии выполнения задания 4***

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

### **Задание 5**

Замените for на while

```
for (var i = 0; i < 3; i++) {
    console.log( "номер " + i + "!" );
}
```

### **Критерии выполнения задания 5**

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

### **Задание 6**

Найти максимальный и минимальный элементы в массиве var arr = [2,11,4,6,1,2,9,12,3].

### **Критерии выполнения задания 6**

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

#### **Типовые теоретические вопросы:**

- 1) Сущность функционального программирования.
- 2) Декларативный характер функционального программирования.
- 3) Чистые функции.
- 4) Ссылочная прозрачность.
- 5) Сохранение данных неизменяемыми.
- 6) Преимущества функционального программирования.
- 7) Функциональное программирование в сравнении с объектно-ориентированным программированием.
- 8) Функции, замыкания и область видимости.
- 9) Концепции функционального программирования.
- 10) Функциональное программирование на языке JavaScript.
- 11) Локальная область видимости.
- 12) Операции с объектом в JavaScript.
- 13) Асинхронная событийная модель работы в NodeJS.
- 14) Соединение пользователей с ресурсами.
- 15) Открытость.
- 16) Технологии масштабирования.
- 17) Гомогенные мультикомпьютерные системы.
- 18) Распределенные операционные системы.
- 19) Мультикомпьютерные операционные системы.
- 20) Программное обеспечение промежуточного уровня.

Код компетенции	Результаты освоения ОПОП Содержание компетенций
ОПК-2.2	Применяет современный математический аппарат, связанный с разработкой и реализацией программных продуктов и программных комплексов в различных областях деятельности

#### **Типовые тестовые вопросы:**

16. Каким языком программирования является JavaScript?

+мультипарадигменный язык программирования;

рекурсивно-логический язык  
программирования; компилируемый язык  
программирования; сильно типизированный  
язык.

17. Что получится в результате выполнения следующего кода на JavaScript?

```
(function(){
    return typeof arguments;
})();
+"object";
"array";
"arguments";
"undefined".
```

18. Что получится в результате выполнения следующего кода на JavaScript?

```
var f = function g(){ return 23; };
typeof g();
"number";
"undefined";
"function";
+Error.
```

19. Что получится в результате выполнения следующего кода на JavaScript?

```
(function(x){
    delete x;
    return x;
})(1);
+1;
null;
undefined;
Error.
```

20. Что получится в результате выполнения следующего кода на JavaScript?

```
var y = 1, x = y = typeof x;
x;
1;
"number";
undefined;
"undefined".
```

21. Что получится в результате выполнения следующего кода на JavaScript?

```
(function f(f){
    return typeof f();
})(function(){ return 1; });
+"number";
"undefined";
"function";
Error.
```

22. Что получится в результате выполнения следующего кода на JavaScript?

```
var foo = {
    bar: function() { return this.baz; },
```

baz: 1

```
};  
(function(){  
    return typeof arguments[0]();  
})(foo.bar);  
+"undefined";  
"object";  
"number";  
"function".
```

23. Что получится в результате выполнения следующего кода на JavaScript?

```
var foo = {  
    bar: function(){ return this.baz; },  
    baz: 1  
}  
typeof (f = foo.bar)();  
+"undefined";  
"object";  
"number";  
"function".
```

24. Что получится в результате выполнения следующего кода на JavaScript?

```
var f = (function f(){ return "1"; }, function g(){ return 2; })();  
typeof f;  
"string";  
+"number";  
"function";  
"undefined".
```

25. Что получится в результате выполнения следующего кода на JavaScript?

```
var x = 1;  
if (function f(){} ) {  
    x += typeof f;  
}  
x;  
1;  
"1function";  
+"1undefined";  
NaN.
```

26. Что получится в результате выполнения следующего кода на JavaScript?

```
var x = [typeof x, typeof y][1];  
typeof typeof x;  
"number";  
+"string";  
"undefined";  
"object".
```

27. Что получится в результате выполнения следующего кода на JavaScript?

```
(function(foo){  
    return typeof foo.bar;  
})({ foo: { bar: 1 } });
```

```
+ "undefined";
"object";
"number";
Error.
```

28. Что получится в результате выполнения следующего кода на JavaScript?

```
(function f(){
    function f(){ return 1; }
    return f();
    function f(){ return 2; }
})();
1;
+2;
Error (e.g. "Too much recursion");
undefined.
```

29. Что получится в результате выполнения следующего кода на JavaScript?

```
function f(){ return f; }
new f() instanceof f;
true;
+false;
null;
one.
```

30. Что получится в результате выполнения следующего кода на JavaScript?

```
with (function(x, undefined){ }) length;
1;
+2;
undefined;
Error.
```

#### **Типовые практические задания:**

##### **Задание 7**

Реализуйте функцию sum(array), возвращающую сумму всех элементов массива.

Попробуйте использовать для этого метод массива reduce.

##### **Критерии оценки задания 7**

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

##### **Задание 8**

Создайте функцию clone(), позволяющую создать копию объекта person1 так, чтобы при изменении одного объекта второй оставался прежним. Например:

```
var person1 = {
    hair: "brown",
    eyes: "green",
    age: 15
}
```

```
var person2 = clone(person1);
```

```
person2.hair      =      "red";
person1.hair === "brown"; // true
```

### ***Критерии оценки задания 8***

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

### ***Задание 9***

Найти первое число в массиве: var arr = [2,11,4,6,1,2,9,12,3]; больше 10 и прекратить выполнение цикла.

### ***Критерии выполнения задания 9***

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

### ***Задание 10***

С помощью методов массива split, join, push, pop, shift, unshift, splice превратите строку 'Не люблю программировать на Pascal' в строку 'Я очень люблю программировать на Javascript'.

Вы должны получить следующий код:

```
var string = 'Не люблю программировать на Pascal';
// ваш код
console.log(string); // выведет 'Я очень люблю программировать на Javascript'
```

### ***Критерии выполнения задания 10***

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

### ***Задание 11***

Реализовать рекурсивную факториал-функцию factorial(n).

### ***Критерии выполнения задания 11***

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

### ***Задание 12***

Создайте функцию merge(), объединяющую свойства двух объектов. Если попадаются свойства с одинаковым именем, отдавайте приоритет второму объекту:

```
var person1 = {
  hair: "brown",
  eyes: "green",
  age: 15
}
```

```
var person2 = {
  hair: "red",
  height: 180,
  isMale: true
```

```
}

var result = merge(person1, person2);

/*
result = {
  hair: "red",
  eyes: "green",
  age: 15,
  height: 180,
  isMale: true
}
*/
```

### **Критерии выполнения задания 12**

Задание считается выполненным, если: обучающийся разработал программу на языке JavaScript, корректно решающую поставленную задачу, и может объяснить принцип ее работы.

#### **Типовые теоретические вопросы:**

- 21) Варианты архитектуры клиент-сервер.
- 22) Уровень обработки.
- 23) Типы данных в JavaScript.
- 24) Глобальная область видимости.
- 25) Доступ к свойству объекта в JavaScript через переменную.
- 26) Модульная структура основных библиотек в NodeJS.
- 27) Прозрачность в распределенных системах.
- 28) Отделение правил от механизмов.
- 29) Общие концепции аппаратных решений.
- 30) Гетерогенные мультикомпьютерные системы.
- 31) Операционные системы для однопроцессорных компьютеров.
- 32) Системы с распределенной разделяемой памятью.
- 33) Клиенты и серверы.
- 34) Многозвездные архитектуры.
- 35) Современные варианты архитектуры.
- 36) Уровень данных.
- 37) Переменные и идентификаторы в JavaScript.
- 38) Рекурсия.
- 39) Встроенные объекты.
- 40) Работа с файловой системой в NodeJS.

Составил доцент кафедры ЭВМ,  
к.т.н., доцент,

В.А. Саблина

Заведующий кафедрой ЭВМ,  
д.т.н., профессор

Б.В. Костров

Оператор ЭДО ООО "Компания "Тензор"

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ

СОГЛАСОВАНО

ФГБОУ ВО "РГРТУ", РГРТУ, Костров Борис Васильевич,  
Заведующий кафедрой ЭВМ

27.11.25 12:54 (MSK)

Простая подпись