Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Максимов Алексей Борисович
Должность: директор департамента по образовательной политике
Дата подписания: 30.10.2023 16:04:17
Уникальный программный ключ:
8db180d1a3f02ac9e60521a5672742735c18b1d6

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
Факультет информационных технологий

УТВЕРЖДЕНО
Декан факультета
Информационных технологий
_____ / Д.Г. Демидов /
«16» ____02_____ 2023 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Функциональное программирование»

Направление подготовки/специальность
09.03.02 Информационные системы и технологии

Профиль/специализация
Информационные технологии в медиаиндустрии и дизайне

Квалификация
Бакалавр

Формы обучения
Очная

Москва, 2023 г.

Разработчик(и):

ст. преподаватель кафедры
«Информатика и информационные технологии»  / И. К. Новичков /

Согласовано:
Заведующий кафедрой
«Информатика и информационные технологии»,
к.т.н.  / Е.В. Булатников /

Содержание

1. Цели, задачи и планируемые результаты обучения по дисциплине 4
2. Место дисциплины в структуре образовательной программы 4
3. Структура и содержание дисциплины 5
3.1. Виды учебной работы и трудоемкость 5
3.2. Тематический план изучения дисциплины 5
3.3. Содержание дисциплины 6
3.4. Тематика семинарских/практических и лабораторных занятий 9
3.5. Тематика курсовых проектов (курсовых работ) 9
4. Учебно-методическое и информационное обеспечение 9
4.1. Нормативные документы и ГОСТы 9
4.2. Основная литература 10
4.3. Дополнительная литература 10
4.4. Электронные образовательные ресурсы 10
4.5. Лицензионное и свободно распространяемое программное обеспечение 10
4.6. Современные профессиональные базы данных и информационные справочные системы 11
5. Материально-техническое обеспечение 11
6. Методические рекомендации 11
6.1. Методические рекомендации для преподавателя по организации обучения 11
6.2. Методические указания для обучающихся по освоению дисциплины 11
7. Фонд оценочных средств 12
7.1. Методы контроля и оценивания результатов обучения 12
7.2. Шкала и критерии оценивания результатов обучения 12
7.3. Оценочные средства 13

1. Цели, задачи и планируемые результаты обучения по дисциплине

Целью дисциплины “Функциональное программирование” является обучение студентов основам функционального подхода к программированию, в частности, с использованием языков JavaScript и TypeScript. Студенты изучают принципы чистых функций, побочных эффектов, рекурсии и итерации, функций высшего порядка, каррирования и частичного применения, а также монад и функционального состояния. Они также учатся основам функционального тестирования и отладки. Практическая направленность дисциплины заключается в том, чтобы студенты могли применять полученные знания в реальных проектах по разработке программного обеспечения, использующего функциональный подход.

К основным **задачам** освоения дисциплины следует отнести:

– Изучение основ функционального программирования, включая принципы чистых функций, побочных эффектов, рекурсии и итерации, функций высшего порядка, каррирования и частичного применения, а также монад и функционального состояния.

– Приобретение навыков функционального тестирования и отладки.

– Изучение и применение языков JavaScript, TypeScript и F# для функционального программирования.

– Практическое применение полученных знаний в реальных проектах по разработке программного обеспечения, использующего функциональный подход.

Обучение по дисциплине «Функциональное программирование» направлено на формирование у обучающихся следующих компетенций:

Код и наименование компетенций	Индикаторы достижения компетенции
ПК-2. Способен выполнять работы и управлять работами по созданию (модификации) и сопровождению медийных информационных ресурсов	ИПК-2.1. Знает способы управления работами по созданию и обслуживанию ИС в медиаиндустрии и методов дизайна применяемых в этой отрасли: принципы проектирования логической структуры веб–страниц; типовые решения, библиотеки программных модулей, шаблоны, классы объектов, используемые при разработке веб–проекта; методы и средства проектирования веб–ресурсов, основные приемы построения 3Dмоделей; способы и приемы редактирования моделей; основные понятия визуализации сцен; основы анимации.
	ИПК-2.2. Умеет управлять работами по разработке и обслуживанию ИС в медиаиндустрии: продумывать наиболее удобные решения подачи информации; использовать существующие типовые решения и шаблоны веб–ресурсов; создавать и редактировать 3D-модели; подбирать материалы и текстуру поверхности моделей; выполнять визуализацию сцен; выполнять анимацию 3D модели.
	ИПК-2.3. Владеет навыками применения программного обеспечения для управления работами по разработке ИС в медиаиндустрии: методами проектирования медийных веб–ресурсов; навыками разработки и изменения типовыми приемами работы в пакетах

	трехмерной графики.
ПК-4. Способен проводить работы по интеграции программных модулей и компонент и проверку работоспособности выпусков программных продуктов применительно к объектам медиаиндустрии	ИПК-4.1. Знает методы и способы интеграции программных модулей ИС в медиаиндустрии; базовые технологии веб-программирования; подходы к определению понятия алгоритма; основные свойства алгоритмов и структур данных; способы представления алгоритмов;
	ИПК-4.2. Умеет проводить верификацию выпусков ИС в медиаиндустрии; использовать клиентские и серверные языки web-программирования для создания интернет-приложений: разрабатывать эффективные алгоритмы с точки зрения пространственных и временных характеристик; определять оптимальные структуры данных при разработке алгоритмов; определять сложность алгоритмов.
	ИПК-4.3. Имеет навыки применения программного обеспечения для верификации версий ИТ в медиаиндустрии; навыками решения типовых задач клиентской и серверной веб-разработки: анализа и трассировки алгоритмов; современными методами разработки алгоритмов; способами представления алгоритмов.

2. Место дисциплины в структуре образовательной программы

Дисциплина относится к части, формируемой участниками образовательных отношений Блока 1. Дисциплины (модули) учебного плана программы бакалавриата.

Дисциплина взаимосвязана логически и содержательно-методически со следующими дисциплинами и практиками ОПОП:

- BackEnd-разработка;
- Шаблоны проектирования;
- Производственная практика (проектно-технологическая);Производственная практика (преддипломная);
- Выполнение и защита выпускной квалификационной работы.

3. Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 3 зачетные единицы (108 часа).

3.1. Виды учебной работы и трудоемкость

(по формам обучения)

3.1.1. Очная форма обучения

№ п/п	Вид учебной работы	Количество часов	Семестры
			5
1	**Аудиторные занятия**	**54**	**54**
	В том числе:		
1.1	Лекции	18	18
1.2	Семинарские/практические занятия		

1.3	Лабораторные занятия	36	36
2	**Самостоятельная работа**	**54**	**54**
	В том числе:		
2.1	Подготовка и выполнение лабораторных работ	54	54
3	**Промежуточная аттестация**		
	Экзамен/зачет		зачет
	Итого:	**108**	**108**

3.2. Тематический план изучения дисциплины

(по формам обучения)

3.2.1. Очная форма обучения

№ п/п	Разделы/темы дисциплины	Трудоемкость, час					
		Всего	Аудиторная работа				Самостоятельная работа
			Лекции	Семинарские/ практические занятия	Лабораторные занятия	Практическая подготовка	
1.1	Тема 1. «Введение в функциональное программирование»	1	1				
2.1	Тема 2. «Чистые функции и побочные эффекты»	1	1				
3.1	Тема 3. «Основы JS для функционального программирования»	1	1				
3.2	Лабораторная работа 1. «Основы JS для функционального программирования»	15			6		9
4.1	Тема 4. «Рекурсия и итерация»	1	1				
5.1	Тема 5. «Применение функционального программирования в JS»	2	2				
5.2	Лабораторная работа 2. «Применение функционального программирования в JS»	15			6		9
6.1	Тема 6. «Функции высшего порядка»	1	1				
7.1	Тема 7. «Основы TypeScript для функционального программирования»	2	2				
7.2	Лабораторная работа 3. «Основы TypeScript для функционального программирования»	15			6		9
8.1	Тема 8. «Каррирование и частичное применение»	1	1				
9.1	Тема 9. «Применение функционального программирования в TypeScript»	2	2				

9.2	Лабораторная работа 4. «Применение функционального программирования в TypeScript»	15			6		9
10.1	Тема 10. «Монады и функциональное состояние»	1	1				
11.1	Тема 11. «Основы F# для функционального программирования»	2	2				
11.2	Лабораторная работа 5. «Основы F# для функционального программирования»	15			6		9
12.1	Тема 12. «Функциональное тестирование и отладка»	1	1				
13.1	Тема 13. «Применение функционального программирования в F#»	2	2				
13.2	Лабораторная работа 6. «Применение функционального программирования в F#»	15			6		9
Итого		**108**	**18**		**36**		**54**

3.3. Содержание дисциплины

- **Тема 1. Введение в функциональное программирование:**
  - Определение функционального программирования: Что такое функциональное программирование и какие его основные принципы.
  - История и развитие: Как развивалось функциональное программирование, его корни и влияние на современные языки программирования.
  - Преимущества и недостатки: Обсуждение преимуществ функционального программирования, таких как предсказуемость и тестируемость, а также его недостатков, таких как сложность понимания для новичков.
  - Основные концепции: Обсуждение основных концепций, таких как чистые функции, неизменяемость, рекурсия и т.д.
  - Практические примеры: Примеры кода, демонстрирующие принципы функционального программирования.
- **Чистые функции и побочные эффекты:**
  - Определение чистых функций: Что такое чистые функции и почему они важны в функциональном программировании.
  - Понятие побочных эффектов: Что такое побочные эффекты и как они влияют на поведение программы.
  - Влияние на надежность и предсказуемость кода: Как чистые функции и отсутствие побочных эффектов делают код более надежным и предсказуемым.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
- **Основы JS для функционального программирования:**
  - Особенности JavaScript для функционального программирования: Обзор особенностей JavaScript, которые делают его подходящим для функционального программирования.
  - Примеры функционального кода на JavaScript: Примеры кода, демонстрирующие использование функционального программирования на JavaScript.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления

материала.

- **Рекурсия и итерация:**
  - Определение рекурсии и итерации: Что такое рекурсия и итерация и как они используются в функциональном программировании.
  - Сравнение рекурсии и итерации: Различия между рекурсией и итерацией и когда использовать каждую из них.
  - Применение в функциональном программировании: Примеры использования рекурсии и итерации в функциональном коде.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
- **Применение функционального программирования в JS:**
  - Практические примеры использования принципов функционального программирования: Примеры задач, которые можно решить с помощью функционального программирования на JavaScript.
  - Разработка функционального проекта на JavaScript: Создание небольшого проекта, используя принципы функционального программирования.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
- **Функции высшего порядка:**
  - Определение функций высшего порядка: Что такое функции высшего порядка и как они используются в функциональном программировании.
  - Примеры и применение: Примеры функций высшего порядка и их применение в функциональном коде.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
- **Основы TypeScript для функционального программирования:**
  - Особенности TypeScript для функционального программирования: Обзор особенностей TypeScript, которые делают его подходящим для функционального программирования.
  - Примеры функционального кода на TypeScript: Примеры кода, демонстрирующие использование функционального программирования на TypeScript.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
- **Каррирование и частичное применение:**
  - Определение каррирования и частичного применения: Что такое каррирование и частичное применение и как они используются в функциональном программировании.
  - Примеры и применение: Примеры каррирования и частичного применения и их применение в функциональном коде.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
- **Применение функционального программирования в TypeScript:**
  - Практические примеры использования принципов функционального программирования: Примеры задач, которые можно решить с помощью функционального программирования на TypeScript.
  - Разработка функционального проекта на TypeScript: Создание небольшого проекта, используя принципы функционального программирования.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
- **Монады и функциональное состояние:**
  - Определение монад: Что такое монады и как они используются в функциональном программировании.

- Управление состоянием и побочными эффектами: Как монады помогают управлять состоянием и побочными эффектами в функциональном коде.
- Примеры и упражнения: Практические примеры и упражнения для закрепления материала.

– **Основы F# для функционального программирования:**
  - Особенности F# как функционального языка программирования: Обзор особенностей F#, которые делают его подходящим для функционального программирования.
  - Примеры функционального кода на F#: Примеры кода, демонстрирующие использование функционального программирования на F#.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
– **Функциональное тестирование и отладка:**
  - Подходы к тестированию функционального кода: Обзор методов тестирования, специфичных для функционального программирования.
  - Отладка функционального кода: Техники отладки, которые можно использовать при работе с функциональным кодом.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.
– **Применение функционального программирования в F#:**
  - Практические примеры использования принципов функционального программирования: Примеры задач, которые можно решить с помощью функционального программирования на F#.
  - Разработка функционального проекта на F#: Создание небольшого проекта, используя принципы функционального программирования.
  - Примеры и упражнения: Практические примеры и упражнения для закрепления материала.

3.4. Тематика семинарских/практических и лабораторных занятий

3.4.1. Семинарские/практические занятия

Семинарские и практические занятия не предусмотрены.

3.4.2. Лабораторные занятия

Лабораторная работа 1. «Основы JS для функционального программирования.»

Студенты разработают набор чистых функций для работы с массивами в JavaScript, создадут функцию высшего порядка и выполнют общие математические операции.

Лабораторная работа 2. «Применение функционального программирования в JS.»

Студенты разработают небольшое веб-приложение, например, список задач, используя принципы функционального программирования.

Лабораторная работа 3. «Основы TypeScript для функционального программирования.»

Студенты разработают набор чистых функций для работы с массивами в TypeScript, создадут функцию, которая принимает другую функцию в качестве аргумента и/или возвращает функцию.

Лабораторная работа 4. «Применение функционального программирования в TypeScript.»

Студенты разработают небольшое веб-приложение, например, список задач, используя принципы функционального программирования.

Лабораторная работа 5. «Основы F# для функционального программирования.»

Студенты разработают набор чистых функций для выполнения общих математических операций, напишут рекурсивную функцию для решения известной задачи, например, вычисление факториала.

Лабораторная работа 6. «Применение функционального программирования в F#.»

Студенты разработают небольшое консольное приложение, например, калькулятор, используя принципы функционального программирования.

3.5. Тематика курсовых проектов (курсовых работ)

Курсовые проекты не предусмотрены.

4. Учебно-методическое и информационное обеспечение

4.1. Нормативные документы и ГОСТы

1. Федеральный закон от 29 декабря 2012 года No 273-ФЗ «Об образовании в Российской Федерации» (с изменениями и дополнениями);
2. Федеральный государственный образовательный стандарт высшего образования - бака-лавриат по направлению подготовки 09.03.02 Информационные системы и технологии, утвер-жденный Приказом Министерства образования и науки РФ от 19 сентября 2017 г. N 929 "Об утверждении федерального... Редакция с изменениями N 1456 от 26.11.2020;
3. Приказ Министерства образования и науки РФ от 05 апреля 2017 г. No 301 «Об утвер-ждении Порядка организации и осуществления образовательной деятельности по образователь-ным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры.

4.2. Основная литература

1. Конева, С. И. Функциональное программирование. Ч.1 : учебное пособие / С. И. Конева. — Ростов-на-Дону : Северо-Кавказский филиал Московского технического университета связи и информатики, 2018. — 53 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/89511.html (дата обращения: 01.10.2023)
2. Городняя, Л. В. Основы функционального программирования : учебное пособие / Л. В. Городняя. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. — 246 с. — ISBN 978-5-4497-0932-5. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/102042.html (дата обращения: 26.08.2023)
3. “JavaScript. Подробное руководство” автора Дэвида Флэнагана, издательство O’Reilly, 2011 год, 1136 страниц, ISBN: 978-5-8459-1797-4.
4. “TypeScript быстро”, издательский дом “Питер”, 2023 год.
5. “Функциональное программирование на F#” автора Дмитрия Сошникова, издательство ДМК Пресс, 2017 год, ISBN: 978-5-94074-689-8.

4.3. Дополнительная литература

1. Кудрявцев, К. Я. Функциональное программирование : конспект лекций / К. Я. Кудрявцев. — Москва : Национальный исследовательский ядерный университет «МИФИ», 2020. — 111 с. — ISBN 978-5-7262-2672-9. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/116429.html (дата обращения: 01.10.2023).
2. Чукич, И. Функциональное программирование на языке C++ / И. Чукич ; перевод

В. Ю. Винник, А. Н. Киселев. — Москва : ДМК Пресс, 2020. — 360 с. — ISBN 978-5-97060-781-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/124744.html (дата обращения: 14.10.2022).
3. “Выразительный JavaScript. Современное веб-программирование” автора Марейна Хавербеке, издательство “Питер”, 2019 год, 480 страниц.

4.4. Электронные образовательные ресурсы

ЭОР разрабатывается.

4.5. Лицензионное и свободно распространяемое программное обеспечение

1. Microsoft Visual Studio: интегрированная среда разработки (IDE), которая поддерживает различные языки программирования и позволяет разрабатывать кроссплатформенные приложения.
2. Visual Studio Code: легковесный и расширяемый текстовый редактор, позволяющий разрабатывать кроссплатформенные приложения на различных языках программирования.

4.6. Современные профессиональные базы данных и информационные справочные системы

1. ОП "Юрайт" https://urait.ru/
2. IPR Smart https://www.iprbookshop.ru/
3. ЭБС "Лань" https://e.lanbook.com/

5. Материально-техническое обеспечение

Методика преподавания дисциплины «Функциональное программирование» предусматривает использование онлайн-курса в системе дистанционного обучения Университета, групповых и индивидуальных консультаций обучающихся, аудиторных занятий в сочетании с внеаудиторной работой с целью формирования и развития профессиональных навыков обучающихся.

Лабораторные работы по дисциплине «Функциональное программирование» осуществляются в форме самостоятельной проработки теоретического материала обучающимися; выполнения практического задания; защиты преподавателю лабораторной работы (знание теоретического материала и выполнение практического задания по теме лабораторной работы).

6. Методические рекомендации

6.1. Методические рекомендации для преподавателя по организации обучения

Методика преподавания дисциплины «Функциональное программирование» предусматривает использование онлайн-курсов в системе дистанционного обучения, проведение групповых и индивидуальных консультаций, а также аудиторных занятий в сочетании с внеаудиторной работой для формирования и развития профессиональных навыков студентов.

Лабораторные работы по дисциплине «Функциональное программирование» включают самостоятельную проработку теоретического материала, выполнение практического задания и защиту лабораторной работы перед преподавателем, включая проверку знания теоретического материала и успешное выполнение задания по теме

работы.

6.2. Методические указания для обучающихся по освоению дисциплины

Изучение дисциплины осуществляется в соответствии с учебным планом.

На занятиях осуществляется закрепление полученных, в том числе и в процессе самостоятельной работы, знаний. Особое внимание обращается на умение применять полученные знания на практике, в том числе при решении реальных задач, отличающихся от проработанных.

В процессе самостоятельной работы студенты закрепляют и углубляют знания, полученные во время аудиторных занятий, самостоятельно знакомятся с теоретическим материалом, выполняют лабораторные работы, готовятся к текущему контролю и промежуточной аттестации.

Текущий контроль осуществляется на аудиторных занятиях в виде защиты лабораторных работ. Критериями оценки результатов являются:

– уровень освоения теоретического материала;

– уровень владения практическими навыками (в виде вопросов по процессу выполнения лабораторных работ);

– умения обучающегося использовать теоретические знания при выполнении практических задач (в виде дополнительных заданий);

– сформированность компетенций;

– оформление материала в соответствии с требованиями.

Промежуточный контроль осуществляется на зачете в форме тестирования в системе дистанционного обучения Университета, включающего вопросы на знание практической части.

7. Фонд оценочных средств

7.1. Методы контроля и оценивания результатов обучения

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

7.2. Шкала и критерии оценивания результатов обучения

К промежуточной аттестации допускаются только студенты, выполнившие все виды учебной работы, предусмотренные рабочей программой по дисциплине «Функциональное программирование».

7.2.1. Критерии оценки ответа на зачете

(формирование компетенций — ПК-2 и ПК-4)

«Зачтено»:

Выполнены все виды учебной работы, предусмотренные учебным планом. Обучающийся демонстрирует прочные теоретические знания, практические навыки, владеет терминами, делает аргументированные выводы и обобщения, приводит примеры, оперирует приобретенными знаниями, умениями, навыками, применяет их в ситуациях повышенной сложности. При этом могут быть допущены незначительные ошибки, неточности, затруднения при аналитических операциях, переносе знаний и умений на новые, нестандартные ситуации, которые обучающийся может исправить при коррекции преподавателем.

«Не зачтено»:

Не выполнен один или более видов учебной работы, предусмотренных учебным планом. Обучающийся демонстрирует незнание теоретических основ предмета, отсутствие

практических навыков, не умеет делать аргументированные выводы и приводить примеры, не владеет терминами, проявляет отсутствие логичности и последовательности изложения, делает ошибки, которые не может исправить даже при коррекции преподавателем,

отказывается отвечать на дополнительные вопросы, допускает значительные ошибки, испытывает значительные затруднения при оперировании знаниями и умениями при их переносе на новые ситуации.

7.2.2. Критерии оценки работы обучающегося на лабораторных занятиях:
(формирование компетенций — ПК-2 и ПК-4)

«5» (отлично): выполнены все практические задания, предусмотренные лабораторными работами, обучающийся четко и без ошибок ответил на все контрольные вопросы, проявил творческий подход при выполнении заданий, смог выполнить дополнительные задания.

«4» (хорошо): выполнены все практические задания, предусмотренные лабораторными работами, обучающийся с корректирующими замечаниями преподавателя ответил на все контрольные вопросы, проявил творческий подход при выполнении заданий, смог частично выполнить дополнительные задания.

«3» (удовлетворительно): выполнены все практические задания, предусмотренные лабораторными работами, с замечаниями преподавателя; обучающийся ответил на все контрольные вопросы с замечаниями, дополнительные задания выполнены с замечаниями.

«2» (неудовлетворительно): обучающийся не выполнил или выполнил неправильно практические задания, предусмотренные лабораторными работами, обучающийся ответил на контрольные вопросы с ошибками или не ответил на контрольные вопросы, дополнительные задания выполнены неверно или не выполнены.

7.3. Оценочные средства

7.3.1 Вопросы к зачету

1. Какой из следующих принципов НЕ относится к функциональному программированию?
    - “+” Использование чистых функций
    - “-” Изменение глобального состояния
    - “+” Избегание побочных эффектов
    - “+” Использование рекурсии вместо циклов
2. Что такое функции высшего порядка в контексте функционального программирования?
    - “-” Функции, которые могут быть вызваны только в определенных условиях
    - “+” Функции, которые принимают другие функции в качестве аргументов или возвращают их
    - “-” Функции, которые могут вызывать только другие функции
    - “-” Функции, которые могут быть вызваны только другими функциями
3. Что такое каррирование в функциональном программировании?
    - “-” Процесс преобразования функции с несколькими аргументами в функцию с одним аргументом
    - “+” Процесс преобразования функции с несколькими аргументами в последовательность функций, каждая из которых принимает один аргумент
    - “-” Процесс преобразования функции с одним аргументом в функцию с несколькими аргументами
    - “-” Процесс преобразования функции с одним аргументом в последовательность функций, каждая из которых принимает несколько аргументов

4. Что такое монады в контексте функционального программирования?
    - “-” Структуры данных, используемые для представления списков
    - “+” Структуры данных, используемые для обработки побочных эффектов в чистом функциональном коде
    - “-” Структуры данных, используемые для представления деревьев
    - “-” Структуры данных, используемые для представления графов
5. Какой из следующих языков программирования НЕ поддерживает функциональное программирование?
    - “-” JavaScript
    - “+” C
    - “-” F#
    - “-” Haskell