МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

«СОГЛАСОВАНО»
Руководитель ОП
_____ Добржинский Ю.В.
«01» сентября 2017 г.

«УТВЕРЖДАЮ»
Заведующий кафедрой
«Информационные системы управления»
_____ А.И. Сухомлинов
«01» сентября 2017 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Технология программирования

Направление подготовки 09.03.01 Информатика и вычислительная техника

профиль Автоматизированные системы обработки информации и управления

Форма подготовки очная

курс 2 семестр 4
лекции 36 час.
практические занятия ___час.
лабораторные работы 54 час.
в том числе с использованием МАО лек. 18 /пр.____/лаб.____ час.
в том числе в электронной форме лек. 18 /пр.____/лаб.____ час.
всего часов аудиторной нагрузки 90 час.
в том числе с использованием МАО ___ час.
в том числе в электронной форме ___ час.
самостоятельная работа 27 час.
в том числе на подготовку к экзамену 27 час.
курсовая работа / курсовой проект ___ семестр
зачет ___ семестр
экзамен 4 семестр

Рабочая программа составлена в соответствии с образовательным стандартом, самостоятельно установленный ДВФУ по направлению подготовки 09.03.01 Информатика и вычислительная техника, утвержденный приказом ректора ДВФУ от 04.04.2016 № 12-13-593

Рабочая программа обсуждена на заседании кафедры «Информационные системы управления», протокол № 1 от «1» сентября 2017 г.

Заведующий кафедрой ИСУ к.т.н., доцент А.И. Сухомлинов
Составитель: ст. преподаватель Березкина Г.Л.

I. Рабочая программа пересмотрена на заседании кафедры:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

II. Рабочая программа пересмотрена на заседании кафедры:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

ABSTRACT

Bachelor's degree in 09.03.01 Informatics and Computer Technology

Study profile: «Computer Aided Systems of Information Processing and Management»

Course title: Technology programming

Basic part of Block, 4 credits

Instructor: Galina Berezkina

At the beginning of the course a student should be able to:

computer science, information technology, the organization of computers and systems, decision theory, database, the theoretical foundations of computer-aided control, computer networks and telecommunications, management information systems, modeling of economic and technical systems.

Learning outcomes:

The development of this discipline is caused by the practical need for shaping the masters of professional competence in the use of modern methods and software design, use of technology, methods and means of production software.

Course description:

The study area of the course includes basic concept, programming methodology, programming technology, stages and phases of development, technological approaches, styles of programming.

Main course literature:

1. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=5115 Asarina I. V. Object-oriented programming in C++: lectures and exercises. - "Hot line - TV-commercial"Publisher: 978-5-9912-7001-4ISBN: 2012: 2nd ed. stereo-type. Edition: 320 p.
2. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=8781 Babushkin, I. A., Okulov S. M. Workshop on object - oriented programming.- "Binom. Laboratory of knowledge" Publishing house: 978-5-9963-0954-2ISBN: 2012:3rd ed. Edition:366 p.
3. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=11847 Tsukanov N. And. Dmitrieva T. A. Theory and practice of logic programming language Visual Prolog 7.Textbook for high schools. Moscow Publishing house:"Hot line-Telecom" : ISBN: 978-5-9912-0194-0 2013 - 232 p.
4. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=40771 Kubensky A. A. Functional programming. St. Petersburg: Publishing house: SPbSU ITMO (St. Petersburg national research University of information technologies, mechanics and optics)W ISBN: 2010:-.p. 251

Form of final knowledge control: pass-fail exam

Аннотация к рабочей программе дисциплины
«Технология программирования»

Дисциплина «Технология программирования» предназначена для обучения бакалаврами направления 09.03.01 – «Информатика и вычислительная техника» и входит в состав блока учебных дисциплин выбора по профилю подготовки. Трудоемкость дисциплины составляет 144 час. (4 ЗЕ). Аудиторные занятия составляют 72 час., включая 36 час., лекций, 54 час. лабораторных работ. По дисциплине предусмотрена самостоятельная работа в объеме 54 час. Дисциплина изучается в четвертом семестре на втором курсе. Изучению данной дисциплины предшествует освоение предмета «Введение в программирование», «Объектно-ориентированное программирование».

Цель дисциплины – обучение бакалавров теоретическим основам и практическим навыкам применения различных методологий и технологий программирования с целью достижения высокого качества разработки программного обеспечения.

Задачи дисциплины заключаются:

- в освоении бакалаврами фундаментальных теоретических положений современных методологий и технологий программирования,
- в формировании у бакалавров интегрированного восприятия стратегии деятельности, организации предприятия и его информационных технологий,
- в приобретении компетенций применения перспективных методологий, методов и средств технологии программирования,
- в развитии умений проведения анализа выбора существующих методологий и средств технологии программирования.

Для решения поставленных задач предусмотрены соответствующие виды учебной работы – аудиторные занятия (лекции, практические занятия) и самостоятельная работа.

Для успешного изучения дисциплины «Технология программирования» у обучающихся должны быть предварительно сформированы компетенции:

ОПК-1 способностью инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем

ОПК-5 способностью решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-

коммуникационных технологий и с учетом основных требований информационной безопасности

В результате изучения данной дисциплины у студентов формируются следующие компетенции:

Код и формулировка компетенции	**Этапы формирования компетенции**	
ОПК-2 - способностью осваивать методики использования программных средств для решения практических задач	Знает	принципы, методы формализации, алгоритмизации и реализации программного обеспечения с помощью языков программирования;
	Умеет	проводить анализ существующих средств разработки программного обеспечения систем, их выбор, внедрение и применение для решения поставленных задач;
	Владеет	методами и средствами реализации программного обеспечения;
ОПК-4 - способностью участвовать в настройке и наладке программно-аппаратных комплексов	Знает	современные технологии разработки программных комплексов с использованием CASE-средств
	Умеет	использовать на практике CASE-средства
	Владеет	инструментальными средами для различных языков программирования
ПК-3 способностью разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования	Знает	Принципы системного подхода, и связанные с ним технологии программирования
	Умеет	Основополагающие теоретические положения, определяющие процесс разработки программного обеспечения информационных систем;
	Владеет	Различными методологиями и технологиями разработки программного обеспечения информационных систем

Изучение дисциплины включает в себя освоение теоретического материала на лекциях и выполнение лабораторных работ.

Для формирования вышеуказанных компетенций в рамках дисциплины «Технология программирования» применяются следующие методы активного/интерактивного обучения: лекция беседа, лекция консультация, лекция пресс-конференция, работа в малой группе.

I. СТРУКТУРА И СОДЕРЖАНИЕ ТЕОРЕТИЧЕСКОЙ ЧАСТИ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины составляет 4 зачетных единиц, т.е. 144 академических часов, в т.ч. аудиторные занятия - 90 часов, самостоятельная работа студента (СРС) – 54 часов.

Раздел 1. Введение в технологию программирования. (2 часа)

Тема 1. Базовые понятия технологий программирования (2 часа)

Определение основных понятий, используемых в курсе "Технология программирования". Определение алгоритма. Свойства алгоритма. Понятия кибернетики, связанные с понятием алгоритма. Модель. Моделирование. Жизненный цикл программного обеспечения. Жизненного цикла программного обеспечения. Проектирование. Проект.

Раздел 2 Основные методологии (4часов)

Тема 1. Обзор методологий, методология императивного и объектно-ориентированного программирования (1 часа).

Обзор методологий императивного программирования, объектно-ориентированного программирования, функционального программирования, логического программирования, программирования в ограничениях и другие.

Методология императивного программирования. Методология объектно-ориентированного программирования.

Тема 2. Методология функционального программирования. . Методология логического программирования. (1 час)

Определение, происхождение методы и концепции, синтаксис и семантика. Класс задач.

Тема 3. Остальные методологии. (1 час)

Методология программирования в ограничениях. Методология структурного императивного программирования. Методология императивного параллельного программирования. Методологии программирования, управляемого потоками данных. Методологии доступ -

ориентированного программирования. Методологии нейросетевого программирования (2 часа))

Раздел 3. Технологии программирования (3 часа)

Тема 1 Технологии программирования (1 часа)

Технологии программирования – основные составляющие и понятия. Определение жизненного цикла программы. Определение технологии программирования. Определение процесса. Определение технологического подхода. Стадии жизненного цикла программного обеспечения. Классификация технологических подходов. Классификация технологических процессов. Процессы жизненного цикла, определяемые международным стандартом ISO 12207 [ISO/IEC 12207:1995]. Основные направления развития и совершенствования технологических процессов.

Тема 2. Классические технологические процессы. (1 часа)

Возникновение и исследование идеи. Управление.

Тема 3. Анализ требований и проектирование. (1 часа)

Проектирование в большом (проектирование архитектуры).

Тема 4. Стиль программирования (кодирования) программы. (1 часа).

Определения стиля программирования. Правило стандартизации стиля. Составляющие стиля программирования. Защитное программирование. Выбор языка программирования.

Тема 5. Тестирование и отладка. (1 часа)

Типы тестирования программ в процессе разработки. Эксплуатация и сопровождение.

Тема 6. Стандартные технологические процессы (1 часа)

Основные технологические процессы. Приобретение. Поставка. Разработка. Эксплуатация. Сопровождение. Управление.

Усовершенствование. Обучение. Управление. Вспомогательные процессы. Документирование. Управление конфигурацией. Обеспечение качества. Верификация. Аттестация. Совместная оценка. Аудит. Разрешение проблем. Организационные процессы. Управление. Создание инфраструктуры. Усовершенствование. Обучение. Взаимосвязь между процессами.

Раздел 3 Технологические подходы (6 часов)

Тема 1. Основные стадии технологических подходов (1 часа)

Стадии технологических подходов. Стадия возникновения и исследования идеи. Стадия планирования проекта. Стадия анализа и проектирования. Стадия реализации. Стадия версии разработчика. Стадия оценки жизнеспособности продукта. Стадия альфа - версии. Стадия бета-версии. Стадия версии первой поставки пользователю.

Тема 2. Основные технологические подходы. (1 часа)

Ранние технологические подходы. Каскадные технологические подходы. Спиральная модель. Каркасные технологические подходы. Рациональный унифицированный процесс. Синтезирующее программирование.

Тема 3. Генетические технологические подходы. (1 часа)

Сборочное (расширяемое) программирование. Конкретизирующее программирование. Подходы на основе формальных преобразований. Технология стерильного цеха.

Тема 4. Формальные генетические подходы. (1 часа)

Группа ранних подходов быстрой разработки. Эволюционное прототипирование. Итеративная разработка. Постадийная разработка. Адаптивные технологические подходы. Экстремальное программирование. Адаптивная разработка. Подходы исследовательского программирования. Компьютерный дарвинизм.

Тема 5. Технологии коллективной разработки. (1 часа)

Авторская разработка. Коллективная разработка. Бригада главного программиста. Общинная модель разработки.

Раздел 4. Качество программного обеспечения (1 часа)

Подходы к качеству программного обеспечения. Характеристики качества программного обеспечения. Функциональность. Надежность. Удобство. Эффективность. Сопровождаемость. Добротность. Оценка качества процесса разработки. Модель зрелости процесса разработки программного обеспечения. Стандартизация информационных технологий.

Использование активных форм обучения.

Для данного курса лекции проводятся в форме лекции беседы, лекция консультация, лекция пресс-конференция. Более 60 процентов лекционных занятий проводятся с использованием активных форм обучения

Во время лекции у бакалавров должен быть раздаточный материал, который они должны активно использовать.

II. СТРУКТУРА И СОДЕРЖАНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ ДИСЦИПЛИНЫ

Учебным планом предусмотрены лабораторные работы (54 часа). Цель занятий – закрепить знания, полученные при изучении теоретической части дисциплины и получить практические навыки управления проектами.

Содержание лабораторных работ

Лабораторная работа 1. Функциональное программирование. Лисп (10 часов).

Тестирование и отладка программ на языке Лисп.

Лабораторная работа 2. Логическое программирование. Пролог (16 часов).

Тестирование и отладка программ на языке ПРОЛОГ.

Лабораторная работа 3. Объектно-ориентированное программирование. C# (14 часов).

Тестирование и отладка программ на языке C# .

Лабораторная работа 4. Объектно-ориентированное программирование. JAVA (14 часов).

Тестирование и отладка программ на языке JAVA .

Программирование с использованием языка JAVA

Форма проведения практических занятий – коллективное занятие с постановкой и решением проблемного задания, закрепляющего знания, полученные на лекции, и навыки, полученные на практических занятиях.

В начале занятия один из обучаемых выполняет задание у доски совместно с преподавателем и другими обучаемыми, в дальнейшем все обучаемыми получают индивидуальные задания.

III/ УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

Учебно-методическое обеспечение самостоятельной работы обучающихся по дисциплине представлено в Приложении 1 и включает в себя:

план-график выполнения самостоятельной работы по дисциплине, в том числе примерные нормы времени на выполнение по каждому заданию;

характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению;

требования к представлению и оформлению результатов самостоятельной работы;

критерии оценки выполнения самостоятельной работы.

IV. КОНТРОЛЬ ДОСТИЖЕНИЯ ЦЕЛЕЙ КУРСА

№ п/п	Контролируемые разделы / темы дисциплины	Коды и этапы формирования компетенций	Оценочные средства	
			текущий контроль	промежуточная аттестация

1	Разделы 1-4	ОПК-2 - способностью осваивать методики использования программных средств для решения практических задач	принципы, методы формализации, алгоритмизации и реализации программного обеспечения с помощью языков программирования; проводить анализ существующих средств разработки программного обеспечения систем, их выбор, внедрение и применение для решения поставленных задач; методами и средствами реализации программного обеспечения;	Контрольная работа 1, собеседование (УО-1)	Вопросы к экзамену 1 - 67
2	Разделы 1-4	ОПК-4 - способностью участвовать в настройке и наладке программно-аппаратных комплексов	современные технологии разработки программных комплексов с использованием CASE-средств использовать на практике CASE-средства инструментальными средами для различных языков программирования	Контрольная работа 2 собеседование (УО-1)	Вопросы к экзамену 1 - 67
3	Раздел 1-4.	ПК-3 способностью разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования	Принципы системного подхода, и связанные с ним технологии программирования Основополагающие теоретические положения, определяющие процесс разработки программного обеспечения	Контрольная работа 3 собеседование (УО-1), разно уровневые задания (ПР-11)	Вопросы к экзамену 1 - 67

			информационных систем;		
			Различными методологиями и технологиями разработки программного обеспечения информационных систем		

Текущий контроль проводится в течение всего периода изучения курса.

В семестре контроль осуществляется на лекционных и практических занятиях. На лекционных занятиях проводятся контрольные работы по основным разделам курса. Цель проведения контрольных работ - определить уровень усвоения студентами и оценить качество их теоретических знаний по данным темам.

На практических занятиях контроль осуществляется при сдаче практических заданий. Магистрант должен свободно ориентироваться в представленном материале (отчет по практическому заданию и объяснить полученные результаты).

Этапы, формы и методы для текущего контроля успеваемости

В процессе обучения проводится текущий, промежуточный и итоговый контроль достижений

Текущий контроль проводится в течение всего периода изучения курса .

В семестре текущий контроль осуществляется на лекционных и практических занятиях. На лекционных занятиях проводятся контрольные работы по основным разделам курса. Цель проведения контрольных работ - определить уровень усвоения студентами и оценить качество их теоретических знаний по данным темам.

На практических занятиях контроль осуществляется при сдаче практических заданий. Магистрант должен свободно ориентироваться в представленном материале (отчет по практическому заданию и объяснить полученные результаты).

Разделы дисциплины, по которым осуществляется промежуточный контроль:

1. Основные методологии.
2. Технологии программирования.
3. Характеристики качества ПО.

Итоговый контроль

В соответствии с учебным планом дисциплины предусмотрены формы итоговой аттестации:

4 семестр –экзамен.

Вопросы для промежуточного и итогового контроля

1. Определение алгоритма.
2. Свойства алгоритма.
3. Понятия кибернетики, связанные с понятием алгоритма.
4. Что такое модель.
5. Что такое моделирование.
6. Что такое жизненный цикл программного обеспечения.
7. Перечислите этапы жизненного цикла программного обеспечения.
8. Что такое проектирование.
9. Что такое проект.
10. Перечислите основные методологии программирования.
11. Дайте определение методологии императивного программирования.
12. Методы и концепции методологии императивного программирования.
13. Вычислительная модель методологии императивного программирования.
14. Основные языки методологии императивного программирования.
15. Перечислите класс задач методологии императивного программирования.
16. Дайте определение методологии ОО программирования.

17. Методы и концепции методологии ОО программирования.
18. Вычислительная модель методологии ОО программирования.
19. Дайте определение методологии функционального программирования.
20. Основные языки методологии функционального программирования.
21. Класс задач методологии функционального программирования.
22. Методы и концепции методологии функционального программирования.
23. Вычислительная модель методологии функционального программирования.
24. Дайте определение методологии логического программирования.
25. Методы и концепции методологии логического программирования.
26. Вычислительная модель методологии логического программирования.
27. Основные языки методологии логического программирования.
28. Класс задач методологии логического программирования
29. Дайте определение методологии программирования в ограничениях.
30. Методы и концепции методологии программирования в ограничениях.
31. Вычислительная модель методологии программирования в ограничениях.
32. Основные языки методологии программирования в ограничениях.
33. Класс задач методологии программирования в ограничениях.
34. Дайте определение методологии структурного императивного программирования.
35. Методы и концепции методологии структурного императивного программирования.
36. Основные языки методологии структурного императивного программирования.

37. Класс задач методологии структурного императивного программирования.
38. Вычислительная модель методологии структурного императивного программирования.
39. Дайте определение методологии императивного параллельного программирования.
40. Методы и концепции методологии императивного параллельного программирования.
41. Вычислительная модель методологии императивного параллельного программирования.
42. Основные языки методологии императивного параллельного программирования.
43. Класс задач методологии императивного параллельного программирования
44. Определение методологии программирования, управляемого потоками данных.
45. Определение методологии доступ - ориентированного программирования.
46. Определение методологии нейросетевого программирования.
47. Определение жизненного цикла программы.
48. Определение технологии программирования.
49. Дать определение процесса.
50. Что такое технологический подход.
51. Что такое стадия жизненного цикла программного обеспечения.
52. Классификация технологических подходов.
53. Классификация технологических процессов.
54. Процессы жизненного цикла, определяемые международным стандартом ISO 12207 [ISO/IEC 12207:1995].
55. Назовите основные направления развития и совершенствования технологических процессов.

56. Составляющие классического процесса «Возникновение и исследование идеи».
57. Составляющие классического процесса «Управление»
58. Перечислите особенности европейского менеджмента.
59. Перечислите особенности американского менеджмента.
60. Перечислите особенности японского менеджмента.
61. Перечислите особенности российского менеджмента.
62. Перечислите методики оценок времени и затрат.
63. Перечислите методы управления проектами.
64. Современные подходы к управлению проектами (практики/навыки).
65. Составляющие классического процесса «Анализ требований и проектирование»
66. Что такое спецификация программы.
67. Средства спецификаций.
68. Составляющие спецификаций
69. Виды спецификаций.
70. Способы представления спецификаций.
71. Что такое архитектура программного продукта
72. Основные структуры, описывающие программную архитектуру.
73. Перечислите архитектурные стили.
74. Проектирование архитектуры (в большом). Структурная методология.
75. Проектирование архитектуры (в большом). Объектно-ориентированная методология.
76. Проектирование архитектуры (в малом). Структурная методология.
77. Проектирование архитектуры (в малом). Объектно- ориентированная методология.
78. Методы анализа и построения спецификаций. Структурная методология.

79. Методы анализа и построения спецификаций. Объектно-ориентированная методология.

80. Основные подходы к ведению анализа и проектирования. Структурная методология.

81. Основные подходы к ведению анализа и проектирования. Объектно-ориентированная методология.

82. Определения стиля программирования.

83. Правило стандартизации стиля.

84. Составляющие стиля программирования.

85. Защитное программирование.

86. Выбор языка программирования.

87. Определение тестирования.

88. Определение отладки.

89. Основные стратегии тестирования.

90. Типичные ошибки.

91. Основные группы тестов.

92. Типы тестирования программ в процессе разработки.

93. Алгоритм отладки программ.

94. Способы доставки программ до пользователя.

95. Основные классы задач, решаемые на этапе эксплуатации и сопровождения.

96. Типы сопровождения программ.

97. Основные технологические процессы. Приобретение (основные действия).

98. Основные технологические процессы. Поставка (основные действия).

99. Основные технологические процессы. Разработка (основные действия).

100. Основные технологические процессы. Эксплуатация (основные действия).

101. Основные технологические процессы. Сопровождение (основные действия).

102. Основные организационные процессы. Управление (основные действия).

103. Основные организационные процессы. Усовершенствование (основные действия).

104. Основные организационные процессы. Обучение (основные действия).

105. Основные организационные процессы. Управление (основные действия).

106. Вспомогательные процессы. Документирование

107. Вспомогательные процессы. Управление конфигурацией

108. Вспомогательные процессы. Обеспечение качества

109. Вспомогательные процессы. Верификация

110. Вспомогательные процессы. Аттестация

111. Вспомогательные процессы. Совместная оценка

112. Вспомогательные процессы. Аудит

113. Вспомогательные процессы Разрешение проблем

114. Организационные процессы. Управление

115. Организационные процессы. Создание инфраструктуры

116. Организационные процессы. Усовершенствование

117. Организационные процессы. Обучение

118. Организационные процессы. Взаимосвязь между процессами

119. Стадии технологических подходов

120. Стадия возникновения и исследования идеи.

121. Стадия планирования проекта.

122. Стадия анализа и проектирования.

123. Стадия реализации.

124. Стадия версии разработчика.

125. Стадия оценки жизнеспособности продукта.

126. Стадия альфа - версии.
127. Стадия бета-версии.
128. Стадия версии первой поставки пользователю.
129. Основные технологические подходы
130. Ранние технологические подходы
131. Каскадные технологические подходы
132. Спиральная модель
133. Каркасные технологические подходы. Рациональный унифицированный процесс
134. Генетические технологические подходы. Синтезирующее программирование
135. Генетические технологические подходы. Сборочное (расширяемое) программирование
136. Генетические технологические подходы. Конкретизирующее программирование
137. Подходы на основе формальных преобразований. Технология стерильного цеха
138. Подходы на основе формальных преобразований. Формальные генетические подходы
139. Подходы на основе формальных преобразований. Группа ранних подходов быстрой разработки
140. Подходы на основе формальных преобразований. Эволюционное прототипирование
141. Подходы на основе формальных преобразований. Итеративная разработка
142. Подходы на основе формальных преобразований. Постадийная разработка
143. Адаптивные технологические подходы. Экстремальное программирование
144. Адаптивные технологические подходы. Адаптивная разработка

145. Подходы исследовательского программирования. Компьютерный дарвинизм

146. Технологии коллективной разработки. Авторская разработка

147. Технологии коллективной разработки. Коллективная разработка

148. Технологии коллективной разработки. Бригада главного программиста

149. Технологии коллективной разработки. Общинная модель разработки

150. Подходы к качеству программного обеспечения

151. Характеристики качества программного обеспечения.

152. Характеристики качества программного обеспечения. Функциональность

153. Характеристики качества программного обеспечения. Надежность

154. Характеристики качества программного обеспечения. Удобство.

155. Характеристики качества программного обеспечения. Эффективность.

156. Характеристики качества программного обеспечения. Сопровождаемость .

157. Характеристики качества программного обеспечения. Добротность

158. Оценка качества процесса разработки

159. Модель зрелости процесса разработки программного обеспечения

160. Стандартизация информационных технологий

V. СПИСОК УЧЕБНОЙ ЛИТЕРАТУРЫ И ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ КОНТРОЛЬ ДОСТИЖЕНИЯ ЦЕЛЕЙ КУРСА

Основная литература

1. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=5115

Ашарина И.В. Объектно-ориентированное программирование в

C++: лекции и упражнения. - "Горячая линия - Телеком"Издательство: 978-5-9912-7001-4ISBN: 2012: 2-е изд., стереотип. Издание: 320 стр.

2. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=8781 Бабушкина И.А. Окулов С.М. Практикум по объектно - ориентированному программированию.- "Бином. Лаборатория знаний" Издательство: 978-5-9963-0954-2ISBN: 2012:3-е изд. (эл.)Издание:366 стр.
3. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=11847 Цуканова Н.И. Дмитриева Т.А. Теория и практика логического программирования на языке Visual Prolog 7.Учебное пособие для вузов. Москва Издательство:"Горячая линия-Телеком" : ISBN: 978-5-9912-0194-0 2013 - 232 с.
4. http://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=40771 Кубенский А.А. Функциональное программирование. С.Петербург: Издательство: СПбНИУ ИТМО (Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики)Ж ISBN: 2010Год: 251 стр.

Дополнительная литература

1. Вендров А.М. Практикум по проектированию программного обеспечения экономических информационных систем.- Учебное пособие.-.М: Финансы и статистика. 2004.-192с.
2. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учебник.- М.: Финансы и статистика. 2004.-352с
3. Калянов Г.Н. CASE- технологии. Консалтинг при автоматизации бизнес-процессов. 2-е изд. перераб. и доп. – М.: Горячая линия - Телеком, 2004.- 320с.

4. Леоненков А.В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose: Учебное пособие М.: Интернет - Университет информационных технологий; БИНОМ, Лаборатория знаний, 2006. – 320с.
5. Макконнелл С. Сколько стоит программный проект. – М.: «Русская редакция», СПб.:Питер , 2007 – 297с.
6. Л. Басс, П Клеменс, Р. Кацман Архитектура программного обеспечения на практике – С.Петербург: изд-во Питер, 2006. - 578с.
7. Герман О.В., Герман Ю.О. Программирование на Java и C# для студента + CD С.Петербург: Издательство BHV, ISBN: 5-94157-710-9: 2005 – 272 с.
8. Мартынов Н.Н. C# для начинающих. Москва: Издательство: Кудиц-Пресс: ISBN: 5-91136-023, 2007, - 704с.
9. Беллиназо М., Ватсон К. C#. С.Петербург: Издательство: Питер ISBN: 5-85582-228-1, 2006 - 862 с.
10. Монахов В.С. Язык программирования Java и среда NetBeans. С.Петербург: Издательство: БХВ-Петербург: ISBN 978-5-9775-0671-7; 2011 г. 862
11. Березкина Г.Л. Автоматизированная разработка АСОИУ: Учебно-методический комплекс. - Владивосток. Изд-во ДВГТУ, 2007. – 208с.
12. Березкина Г.Л. Технология программирования: Учебно-методический комплекс.- Владивосток. Изд-во ДВГТУ, 2008. – 112с.
13. Бобровский С.И. Технологии Delphi. Разработка приложений для бизнеса. Учебный курс. – СПб: Питер.- 2007 – 380с.
14. Бобровский С.И. Технологии Borland. Разработка приложений для бизнеса. Учебный курс. – СПб: Питер.- 2007 – 380с.
15. Леоньтьев Б.К. Microsoft Office Visio 2003 не для дилетантов: Построение проектов, диаграмм и бизнес – схем в операционной

системе Microsoft Windows XP.- М.: ЗАО «Новый издательский дом», 2005. – 384с.

16. Березкина Г.Л. Макетирование информационной системы на первом уровне. Разработка прототипа системы – Владивосток., электронный вариант. 2009 - 16с.

VI. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Результаты освоения дисциплины достигаются за счет использования в процессе обучения: лекций с применением мультимедийных технологий, активных методов обучения с использованием LMS Blackboard; лабораторных занятий на базе компьютерной сети на платформах Liux и Windows.

Все необходимые примеры выполнения практических заданий приведены в LMS Blackboard.

VII. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Для обеспечения учебного процесса по дисциплине «Управление проектами систем» используется следующее материально-техническое обеспечение: компьютеры, операционная система Windows, Интернет, текстовый редактор MS Word, табличный процессор MS Excel, компьютерный класс, LMS Blackboard, LMS Blackboard Collaborate, персональные компьютеры студентов, а также программное обеспечение, разработанное преподавателем.

Для успешного усвоения дисциплины необходимо следующее установленное программное обеспечение:

Транслятор xlisp

Visual Prolog;

Visual Studio 2010;

Eclipse

NetBeans

Операционные системы рабочих станций и серверов

Приложение 1

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

по дисциплине «Технология программирования»
Направление – 09.03.01 «Информатика и вычислительная техника»
Профиль – «Автоматизированные системы
обработки информации и управления»
Форма подготовки – очная

Владивосток
2017

Самостоятельная (внеаудиторная) работа является важной частью образовательной деятельности высшей школы, её роль значительно возрастает в бакалавриате. Количество часов на самостоятельную работу – 36.

Сведения о содержании самостоятельной работы бакалавров по курсу “Технология программирования” приведены ниже:

1. Проработка курса лекций (9 часов) - Работа на лекциях, семинарах. Перед лекцией обучаемый должен иметь распечатанный раздаточный материал по теме лекции, и знать материал предыдущей лекции, уметь ответить на поставленные вопросы.

2. Подготовка к лабораторным занятиям (18 часов) - Работа на лабораторных занятиях. Перед лабораторной работой обучаемый должен изучить необходимые методические указания, подготовить вопросы к преподавателю, оформить и защитить отчет по предыдущей лабораторной работе.

3. Изучение тем: (18 часов). – Стоимостные методы оценки затрат на разработку информационных систем, сравнение стандартов оценки качества информационных систем. Подготовка доклада или сообщения по заданной преподавателем теме, выступление на семинаре.

4. Подготовка к экзамену (18 часов)

Методические указания по самостоятельной работе студентов

Виды самостоятельной работы, их характеристика

Самостоятельная учебная работа представлена такими формами учебного процесса, как лекция, практические и лабораторные занятия, подготовка к ним. Студент должен уметь вести краткие записи лекций, составлять конспекты, планы и тезисы выступлений, подбирать литературу и т.д.

Научная самостоятельная работа студента заключается в его участии в индивидуальной работе с преподавателями кафедры, в научных конференциях разного уровня, а также в написании контрольных, курсовых проектов. Положительное значение научной работы проявляется в ряде обстоятельств:

- будущие специалисты участвуют в процессе добывания новых знаний;

- приобретаемые знания становятся прочными и целенаправленными;

- студенты видят практические плоды своего труда, что эффективно стимулирует их дальнейшую деятельность;

- приобретаются начальные навыки в научном исследовании.

В ходе научной работы студент овладевает приемами теоретического мышления. Выполнение исследования начинается с формулирования темы, разработки плана, подбора и изучения литературы, выбора методологии и средств анализа и проектирования, а также сбора и обработки материала. Самое важное в исследовании наступает после получения нового материала: его осмысливание, сравнение с ранее известными данными, анализ и синтез, изложение результатов, передача их обществу (доклад, сообщение, опубликование, реализация приложения и т.д.).

Организационно такая работа протекает:

- индивидуально под руководством преподавателя (научного руководителя);

- в сотрудничестве с преподавателями кафедры.

Тема может иметь чисто учебное значение (курсовой проект), быть ценной в научном и практическом отношении (выпускные квалификационные работы, технологические проекты и др.).

Важным является умение доложить результаты исследования и подготовить их к опубликованию. Такое умение само по себе не рождается. Ему надо целеустремленно и настойчиво учиться.

Учебная и научная работа имеет в основном образовательное назначение, формирует интеллектуальные качества будущего специалиста. Навыки работы в коллективе студент приобретает, как правило, через участие в коллективных проектах и в общественной жизни вуза.

В вузе все виды самостоятельной работы студента подчиняются целям учебного процесса. Организация самостоятельной работы студентов должна сочетаться со всеми применяемыми в вузе методами обучения и вместе с ними представлять единую систему средств по приобретению знаний и выработке навыков.

Основные формы самостоятельной учебной работы:

1. Работа над конспектом лекции: лекции – основной источник информации по многим предметам, позволяющий не только изучить материал, но и получить представление о наличии других источников, сопоставить разные взгляды на основные проблемы данного курса. Лекции предоставляют возможность «интерактивного» обучения, когда есть возможность задавать преподавателю вопросы и получать на них ответы. Поэтому имеет смысл находить время для хотя бы беглого просмотра информации по материалу лекций (учебники, справочники и пр.) и непонятные, а также дискуссионные моменты обсуждать с преподавателем, другими студентами;

2. Подготовка к практическому/лабораторному занятию: производится, как правило, с использованием методических пособий, состоит в теоретической подготовке (особенно для семинаров) и выполнении практических заданий (решение задач, ответы на вопросы и т.д.). В данном курсе используются следующие формы практических занятий:

- лабораторные занятия с использование вычислительной техники;

- практикум по освоению тех или иных навыков, методик.

3. Доработка конспекта лекции с применением учебника, методической литературы, дополнительной литературы: этот вид самостоятельной работы студентов особенно важен в том случае, когда изучаемый предмет содержит много неоднозначно трактуемых вопросов, проблем. Тогда преподаватель заведомо не может успеть изложить различные точки зрения, и студент должен самостоятельно ознакомиться с ними по имеющейся литературе. Кроме того, рабочая программа предметов предполагает рассмотрение некоторых относительно несложных тем только во время самостоятельных занятий, без чтения лектором;

4. Подбор, изучение, анализ и конспектирование рекомендованной литературы;

5. Самостоятельное изучение отдельных тем, параграфов;

6. Консультации по сложным, непонятным вопросам лекций, семинаров, зачетов;

7. Подготовка к экзамену: один из самых ответственных видов самостоятельной работы, и в то же время возможность сэкономить большое количество времени в период сессии, если эту подготовку начинать заблаговременно. Одно из главных правил – представлять себе общую логику предмета, что достигается проработкой планов лекций, составлением опорных конспектов, схем, таблиц. Фактически основной вид полготовки к экзамену – «свертывание» большого объема информации в компактный вид, а также тренировка в ее «развертывании» (примеры к теории, выведение одних закономерностей из других и т.д.). Владение этими технологиями обеспечивает, пожалуй, более половины успеха. Тем более что преподаватель обычно замечает в течение семестра целенаправленную подготовку такого студента и может поощрить его тем или иным способом. Надо также правильно распределить силы, не только готовясь к самому экзамену, но и позаботившись о допуске к нему (часто это хорошее посещение занятий,

выполнение в назначенный срок практических заданий, активность на семинарах). Наконец, необходимо выяснить условия проведения самого экзаменационного испытания, использовав для этой цели, прежде всего консультацию (хотя преподаватель обычно касается этой темы заранее), - количество и характер вопросов, форму проведения (устно или письменно), возможность использования при подготовке различных материалов и пособий (таблицы, схемы, тетради для практических занятий и т.д.).

8. Используется следующая форма научной самостоятельной работы (долговременная): подготовка доклада к конференции: часто студенты для выступлений на научных и научно-практических конференциях используют материалы курсовых работ. Это вполне оправдано, но тогда возникают два вопроса: как обеспечить этим материалам качество научного доклада, который должен решать определенную проблему, иметь новизну и актуальность: как быть первокурсникам, еще не защитивших ни одну курсовую работу. Видимо, каждый студент должен определиться с первой научной темой уже в первые месяцы учебы, что позволит расширить круг интересов, приобретать важные навыки педагога - исследователя, необходимые в дальнейшем совершенствовании в своей профессии. Отсюда следует полезность раннего начала знакомства с литературой, что является вторым этапом любой научной (и методической) работы (первый этап – определение проблемы, темы и гипотезы исследования). Следующий очень важный шаг – правильно спроектировать и осуществить практическую реализацию. Один из самых ответственных этапов – обобщение результатов реализации, что сопровождается анализом качества проекта и анализом затрат на его реализацию. Последнее – формулировка выводов, содержащих данные о решении проблемы предметной области или исследования, положительном или отрицательном (в чем нет ничего страшного) результате. В заключении часто намечают

основные пути расширения работы, ее продолжения. Обычно доклад иллюстрируется наглядными презентациями, которые необходимо заранее подготовить.

Таким образом, все виды самостоятельной работы взаимосвязаны и взаимообусловлены, ведущее место занимает учебная самостоятельная деятельность.

Все они направлены на повышение как личностных, так и компитентностных качеств будущего специалиста

Приложение 2

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине «Проектирование автоматизированных систем обработки информации и управления»
Направление – 09.03.01 «Информатика и вычислительная техника»
Профиль – «Автоматизированные системы обработки информации и управления»
Форма подготовки – очная

Владивосток
2017

Паспорт ФОС

Код и формулировка компетенции	Этапы формирования компетенции	
ОПК-2 - способностью осваивать методики использования программных средств для решения практических задач	Знает	принципы, методы формализации, алгоритмизации и реализации программного обеспечения с помощью языков программирования;
	Умеет	проводить анализ существующих средств разработки программного обеспечения систем, их выбор, внедрение и применение для решения поставленных задач;
	Владеет	методами и средствами реализации программного обеспечения;
ОПК-4 - способностью участвовать в настройке и наладке программно-аппаратных комплексов	Знает	современные технологии разработки программных комплексов с использованием CASE-средств
	Умеет	использовать на практике CASE-средства
	Владеет	инструментальными средами для различных языков программирования
ПК-3 способностью разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования	Знает	Принципы системного подхода, и связанные с ним технологии программирования
	Умеет	Основополагающие теоретические положения, определяющие процесс разработки программного обеспечения информационных систем;
	Владеет	Различными методологиями и технологиями разработки программного обеспечения информационных систем

N п/п	Контролируемые разделы дисциплины	Коды и этапы формирования компетенций		Оценочные средства - наименования	
				текущий контроль	промежуточная аттестация
1	Введение в технологию программирования.	ОПК-2, ОПК-4, ПК-1	Знает	собеседование УО-1	собеседование УО-1
			Умеет	собеседование УО-1	собеседование УО-1
			Владеет	собеседование УО-1	собеседование УО-1
2	Основные методологии	ОПК-2, ОПК-4, ПК-1	Знает	собеседование УО-1	собеседование УО-1
			Умеет	собеседование УО-1	собеседование УО-1
			Владеет	собеседование УО-1	собеседование УО-1
3	Технологии программирования	ОПК-2, ОПК-4, ПК-1	Знает	собеседование УО-1	собеседование УО-1
			Умеет	собеседование УО-1	собеседование УО-1
			Владеет	собеседование УО-1	собеседование УО-1
4	Технологические подходы	ОПК-2, ОПК-4, ПК-1	Знает	собеседование УО-1	собеседование УО-1
			Умеет	собеседование УО-1	собеседование УО-1
			Владеет	собеседование	собеседование

				УО-1	УО-1
	Качество программного обеспечения	ОПК-2, ОПК-4, ПК-1	Знает	собеседование УО-1	собеседование УО-1
			Умеет	собеседование УО-1	собеседование УО-1
			Владеет	собеседование УО-1	собеседование УО-1

Шкала оценивания уровня сформированности компетенций

Код и формулировка компетенции	**Этапы формирования компетенции**		**критерии**	**показатели**
ОПК-2 - способностью осваивать методики использования программных средств для решения практических задач	знает (пороговый уровень)	принципы, методы формализации, алгоритмизации и реализации программного обеспечения с помощью языков программирования;	знает методы формализации, алгоритмизации и реализации программного обеспечения	способность дать определение методам формализации, алгоритмизации и реализации программного обеспечени
	умеет (продвинутый)	проводить анализ существующих средств разработки программного обеспечения систем, их выбор, внедрение и применение для решения поставленных задач;	умеет проводить анализ существующих методологий/средств разработки систем, их выбор, внедрение и применение для решения поставленных задач	способность проводить анализ существующих методологий/средств разработки систем, их выбор, внедрение и применение для решения поставленных задач
	владеет (высокий)	методами и средствами реализации программного обеспечения;	владеет методами и средствами анализа, моделирования и оптимизации объектов профессиональной деятельности и их компонентов	способность использовать методы и средства анализа, моделирования и оптимизации объектов профессиональной деятельности и их компонентов
ОПК-4 - способностью участвовать в настройке и наладке программно-аппаратных комплексов	знает (пороговый уровень)	современные технологии разработки программных комплексов с использованием CASE-средств	знает современные технологии разработки программных комплексов с использованием CASE-средств	способность дать определение, определить современные технологии разработки программных комплексов с использованием CASE-средств
	умеет (продвинутый)	использовать на практике CASE-средства	умеет проводить анализ с помощью современных технологий разработки программных комплексов с использованием CASE-средств	способность проводить анализ с помощью современных технологий разработки программных комплексов с использованием CASE-средств
	владеет (высокий)	инструментальными средами для различных языков программирования	владеет методами анализа с помощью современных технологий разработки программных комплексов с использованием CASE-средств	способность практического применения результатов анализа на основе современных технологий разработки программных комплексов с использованием CASE-средств
ПК-3 - способностью разрабатывать модели компонентов информационных систем,	знает (пороговый уровень)	Принципы системного подхода, и связанные с ним технологии программирования	знает принципы объектного подхода, и связанные с ним объектно-ориентированные языки программирования	способность назвать, определить принципы объектного подхода, и связанные с ним объектно-ориентированные языки программирования
	умеет (продвин	Основополагающие теоретические	умеет свободно использовать основополагающие	способность свободно использовать

включая модели баз данных и модели интерфейсов «человек – электронно-вычислительная машина	утый)	положения, определяющие процесс разработки программного обеспечения информационных систем;	теоретические положения, определяющие процесс разработки программного обеспечения информационных систем	основополагающие теоретические положения, определяющие процесс разработки программного обеспечения информационных систем
	владеет (высокий)	Различными методологиями и технологиями разработки программного обеспечения информационных систем	владеет различными методологиями разработки программного обеспечения информационных систем	способность свободно реализовывать любую методологию разработки программного обеспечения информационных систем

Методические рекомендации, определяющие процедуры оценивания результатов освоения дисциплины «технология программирования»

Текущая аттестация студентов. Текущая аттестация студентов по дисциплине «проводится в соответствии с локальными нормативными актами ДВФУ и является обязательной.

Текущая аттестация по дисциплине проводится в форме контрольных мероприятий (выполнение практических заданий) по оцениванию фактических результатов обучения студентов и осуществляется ведущим преподавателем.

Объектами оценивания выступают:

- учебная дисциплина (активность на занятиях, своевременность выполнения различных видов заданий, посещаемость всех видов занятий по аттестуемой дисциплине);
- степень усвоения теоретических знаний;
- уровень овладения практическими умениями и навыками по всем видам учебной работы;
- результаты самостоятельной работы.

Краткая характеристика оценочных средств:

- УО-1 - Собеседование - средство контроля, организованное как специальная беседа преподавателя с обучающимся на темы, связанные с

изучаемой дисциплиной, и рассчитанное на выяснение объема знаний, обучающегося по определенному разделу, теме, проблеме и т.п.

- УО-3 - Доклад, сообщение - продукт самостоятельной работы обучающегося, представляющий собой публичное выступление по представлению полученных результатов решения определенной учебно-практической, учебно-исследовательской или научной темы
- ПР-1 – Тест – система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающихся.
- ПР-11 - Разноуровневые задачи - реконструктивного уровня, позволяющие оценивать и диагностировать умения синтезировать, анализировать, обобщать фактический и теоретический материал с формулированием конкретных выводов, установлением причинно-следственных связей; творческого уровня, позволяющие оценивать и диагностировать умения, интегрировать знания различных областей, аргументировать собственную точку зрения.

Критерии оценки устных ответов

- 100-85 баллов - если ответ показывает прочные знания основных процессов изучаемой предметной области, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять сущность, явлений, процессов, событий, делать выводы и обобщения, давать аргументированные ответы, приводить примеры; свободное владение монологической речью, логичность и последовательность ответа; умение приводить примеры современных проблем изучаемой области.
- 85-76 - баллов - ответ, обнаруживающий прочные знания основных процессов изучаемой предметной области, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять сущность, явлений, процессов, событий, делать выводы и обобщения, давать аргументированные ответы, приводить примеры; свободное владение

монологической речью, логичность и последовательность ответа. Однако допускается одна - две неточности в ответе.

- 75-61 - балл – оценивается ответ, свидетельствующий в основном о знании процессов изучаемой предметной области, отличающийся недостаточной глубиной и полнотой раскрытия темы; знанием основных вопросов теории; слабо сформированными навыками анализа явлений, процессов, недостаточным умением давать аргументированные ответы и приводить примеры; недостаточно свободным владением монологической речью, логичностью и последовательностью ответа. Допускается несколько ошибок в содержании ответа; неумение привести пример развития ситуации, провести связь с другими аспектами изучаемой области.
- 60-50 баллов – ответ, обнаруживающий незнание процессов изучаемой предметной области, отличающийся неглубоким раскрытием темы; незнанием основных вопросов теории, несформированными навыками анализа явлений, процессов; неумением давать аргументированные ответы, слабым владением монологической речью, отсутствием логичности и последовательности. Допускаются серьезные ошибки в содержании ответа; незнание современной проблематики изучаемой области.

Критерии оценки выполнения практических занятий

- 100-86 баллов - если ответ показывает глубокое и систематическое знание всего программного материала и структуры конкретного вопроса, а также основного содержания и новаций лекционного курса по сравнению с учебной литературой. Студент демонстрирует отчетливое и свободное владение концептуально-понятийным аппаратом, научным языком и терминологией соответствующей научной области. Знание основной литературы и знакомство с дополнительно рекомендованной литературой. Логически корректное и убедительное изложение ответа.
- 85-76 - баллов - знание узловых проблем программы и основного содержания лекционного курса; умение пользоваться концептуально-понятийным аппаратом в процессе анализа основных проблем в рамках

данной темы; знание важнейших работ из списка рекомендованной литературы. В целом логически корректное, но не всегда точное и аргументированное изложение ответа.

- 75-61 - балл – фрагментарные, поверхностные знания важнейших разделов программы и содержания лекционного курса; затруднения с использованием научно-понятийного аппарата и терминологии учебной дисциплины; неполное знакомство с рекомендованной литературой; частичные затруднения с выполнением предусмотренных программой заданий; стремление логически определенно и последовательно изложить ответ.
- 60-50 баллов – незнание, либо отрывочное представление о данной проблеме в рамках учебно-программного материала; неумение использовать понятийный аппарат; отсутствие логической связи в ответе.

Вопросы итогового и промежуточного контроля

В соответствии с учебным планом дисциплины предусмотрены формы промежуточной аттестации:

II семестр – экзамен.

1. Определение алгоритма.
2. Свойства алгоритма.
3. Понятия кибернетики, связанные с понятием алгоритма.
4. Что такое модель.
5. Что такое моделирование.
6. Что такое жизненный цикл программного обеспечения.
7. Перечислите этапы жизненного цикла программного обеспечения.
8. Что такое проектирование.
9. Что такое проект.
10. Перечислите основные методологии программирования.
11. Дайте определение методологии императивного программирования.
12. Методы и концепции методологии императивного программирования.

13. Вычислительная модель методологии императивного программирования.
14. Основные языки методологии императивного программирования.
15. Перечислите класс задач методологии императивного программирования.
16. Дайте определение методологии ОО программирования.
17. Методы и концепции методологии ОО программирования.
18. Вычислительная модель методологии ОО программирования.
19. Дайте определение методологии функционального программирования.
20. Основные языки методологии функционального программирования.
21. Класс задач методологии функционального программирования.
22. Методы и концепции методологии функционального программирования.
23. Вычислительная модель методологии функционального программирования.
24. Дайте определение методологии логического программирования.
25. Методы и концепции методологии логического программирования.
26. Вычислительная модель методологии логического программирования.
27. Основные языки методологии логического программирования.
28. Класс задач методологии логического программирования
29. Дайте определение методологии программирования в ограничениях.
30. Методы и концепции методологии программирования в ограничениях.
31. Вычислительная модель методологии программирования в ограничениях.
32. Основные языки методологии программирования в ограничениях.
33. Класс задач методологии программирования в ограничениях.
34. Дайте определение методологии структурного императивного программирования.
35. Методы и концепции методологии структурного императивного

программирования.

36. Основные языки методологии структурного императивного программирования.
37. Класс задач методологии структурного императивного программирования.
38. Вычислительная модель методологии структурного императивного программирования.
39. Дайте определение методологии императивного параллельного программирования.
40. Методы и концепции методологии императивного параллельного программирования.
41. Вычислительная модель методологии императивного параллельного программирования.
42. Основные языки методологии императивного параллельного программирования.
43. Класс задач методологии императивного параллельного программирования
44. Определение методологии программирования, управляемого потоками данных.
45. Определение методологии доступ - ориентированного программирования.
46. Определение методологии нейросетевого программирования.
47. Определение жизненного цикла программы.
48. Определение технологии программирования.
49. Дать определение процесса.
50. Что такое технологический подход.
51. Что такое стадия жизненного цикла программного обеспечения.
52. Классификация технологических подходов.
53. Классификация технологических процессов.
54. Процессы жизненного цикла, определяемые международным

стандартом ISO 12207 [ISO/IEC 12207:1995].

55. Назовите основные направления развития и совершенствования технологических процессов.
56. Составляющие классического процесса «Возникновение и исследование идеи».
57. Составляющие классического процесса «Управление»
58. Перечислите особенности европейского менеджмента.
59. Перечислите особенности американского менеджмента.
60. Перечислите особенности японского менеджмента.
61. Перечислите особенности российского менеджмента.
62. Перечислите методики оценок времени и затрат.
63. Перечислите методы управления проектами.
64. Современные подходы к управлению проектами (практики/навыки).
65. Составляющие классического процесса «Анализ требований и проектирование»
66. Что такое спецификация программы.
67. Средства спецификаций.
68. Составляющие спецификаций
69. Виды спецификаций.
70. Способы представления спецификаций.
71. Что такое архитектура программного продукта
72. Основные структуры, описывающие программную архитектуру.
73. Перечислите архитектурные стили.
74. Проектирование архитектуры (в большом). Структурная методология.
75. Проектирование архитектуры (в большом). Объектно-ориентированная методология.
76. Проектирование архитектуры (в малом). Структурная методология.
77. Проектирование архитектуры (в малом). Объектно- ориентированная методология.
78. Методы анализа и построения спецификаций. Структурная

методология.

79. Методы анализа и построения спецификаций. Объектно-ориентированная методология.
80. Основные подходы к ведению анализа и проектирования. Структурная методология.
81. Основные подходы к ведению анализа и проектирования. Объектно-ориентированная методология.
82. Определения стиля программирования.
83. Правило стандартизации стиля.
84. Составляющие стиля программирования.
85. Защитное программирование.
86. Выбор языка программирования.
87. Определение тестирования.
88. Определение отладки.
89. Основные стратегии тестирования.
90. Типичные ошибки.
91. Основные группы тестов.
92. Типы тестирования программ в процессе разработки.
93. Алгоритм отладки программ.
94. Способы доставки программ до пользователя.
95. Основные классы задач, решаемые на этапе эксплуатации и сопровождения.
96. Типы сопровождения программ.

    1.96.1 Основные технологические процессы. Приобретение (основные действия).

97. Основные технологические процессы. Поставка (основные действия).
98. Основные технологические процессы. Разработка (основные действия).
99. Основные технологические процессы. Эксплуатация (основные действия).

100. Основные технологические процессы. Сопровождение (основные действия).
101. Основные организационные процессы. Управление (основные действия).
102. Основные организационные процессы. Усовершенствование (основные действия).
103. Основные организационные процессы. Обучение (основные действия).
104. Основные организационные процессы. Управление (основные действия).
105. Вспомогательные процессы. Документирование
106. Вспомогательные процессы. Управление конфигурацией
107. Вспомогательные процессы. Обеспечение качества
108. Вспомогательные процессы. Верификация
109. Вспомогательные процессы. Аттестация
110. Вспомогательные процессы. Совместная оценка
111. Вспомогательные процессы. Аудит
112. Вспомогательные процессы Разрешение проблем
113. Организационные процессы. Управление
114. Организационные процессы. Создание инфраструктуры
115. Организационные процессы. Усовершенствование
116. Организационные процессы. Обучение
117. Организационные процессы. Взаимосвязь между процессами
118. Стадии технологических подходов
119. Стадия возникновения и исследования идеи.
120. Стадия планирования проекта.
121. Стадия анализа и проектирования.
122. Стадия реализации.
123. Стадия версии разработчика.
124. Стадия оценки жизнеспособности продукта.

125. Стадия альфа - версии.
126. Стадия бета-версии.
127. Стадия версии первой поставки пользователю.
128. Основные технологические подходы
129. Ранние технологические подходы
130. Каскадные технологические подходы
131. Спиральная модель
132. Каркасные технологические подходы. Рациональный унифицированный процесс
133. Генетические технологические подходы. Синтезирующее программирование
134. Генетические технологические подходы. Сборочное (расширяемое) программирование
135. Генетические технологические подходы. Конкретизирующее программирование
136. Подходы на основе формальных преобразований. Технология стерильного цеха
137. Подходы на основе формальных преобразований. Формальные генетические подходы
138. Подходы на основе формальных преобразований. Группа ранних подходов быстрой разработки
139. Подходы на основе формальных преобразований. Эволюционное прототипирование
140. Подходы на основе формальных преобразований. Итеративная разработка
141. Подходы на основе формальных преобразований. Постадийная разработка
142. Адаптивные технологические подходы. Экстремальное программирование
143. Адаптивные технологические подходы. Адаптивная разработка

144. Подходы исследовательского программирования. Компьютерный дарвинизм
145. Технологии коллективной разработки. Авторская разработка
146. Технологии коллективной разработки. Коллективная разработка
147. Технологии коллективной разработки. Бригада главного программиста
148. Технологии коллективной разработки. Общинная модель разработки
149. Подходы к качеству программного обеспечения
150. Характеристики качества программного обеспечения.
151. Характеристики качества программного обеспечения. Функциональность
152. Характеристики качества программного обеспечения. Надежность
153. Характеристики качества программного обеспечения. Удобство.
154. Характеристики качества программного обеспечения. Эффективность.
155. Характеристики качества программного обеспечения. Сопровождаемость .
156. Характеристики качества программного обеспечения. Добротность

1.156.1Оценка качества процесса разработки

157. Модель зрелости процесса разработки программного обеспечения
158. Стандартизация информационных технологий

Тестирование по теме

«Язык Lisp»

1. Основными типами данных в языке Lisp являются:

A. числовой и символьный типы;

B. атомы и списки;

C. строки и массивы;

D. процедуры и функции.

2. Для чего служит функция CONS?

A. строит новый список из переданных ей в качестве аргументов головы и хвоста;

B. создает список из элементов;

C. присваивает символу или связывает с ним некоторое значение;

D. возвращает значение свойства, связанного с символом.

3. Для чего служит функция SET?

A. строит новый список из переданных ей в качестве аргументов головы и хвоста;

B. создает список из элементов;

C. присваивает символу или связывает с ним некоторое значение;

D. проверяет идентичность записей.

4. Формальность параметров означает, что…

A. их можно заменить на любые другие символы, и это не отразится на вычислениях, определяемых функцией;

B. их можно заменить только на данные того же типа, во избежание возникновения ошибок;

C. их нельзя заменить на другие символы;

D. их можно заменить на любые другие символы при определенных дополнительно заданных условиях.

5. Как называется функция, которая сначала вычисляет значение аргумента, а затем выводит это значение, при этом перед выводом аргумента переходит на новую строку, а после него выводит пробел?

A. PRINTF;

B. PRINT 1;

C. PRINT;

D. PRINT C.

6. Каков конечный результат выполнения следующих функций:

_(SETQ a ‘(b c d))

_(RPLACA a ‘d)

_(RPLACD a ‘(o n m))

_a ?

A.(o n m);

B.(b c d o n m);

C.(o n m b c d);

D.(d o n m).

7. Каким будет результат работы программы:

_(LET(a 1) (b 46))

_(- b 3)

_(+ a b)

_(LIST a b) ?

A.1 46;

B.1 44;

C.1 43;

D.47.

8. Условное предложение _(COND (p1 a1) … (pn an)) реализуется следующим образом:

A.выражения pi вычисляются последовательно до тех пор, пока не встретится выражение, значением которого является T. Вычисляется результирующее выражение, соответствующее этому предикату, и полученное значение возвращается в качестве значения всего предложения COND. Если истинного предиката нет, то значением COND будет NIL;

B. выражения pi вычисляются последовательно до тех пор, пока не встретится выражение, значением которого является NIL. Вычисляется результирующее выражение, соответствующее этому предикату, и полученное значение возвращается в качестве значения всего предложения COND. Если ложного предиката нет, то значением COND будет NIL;

C. выражения pi вычисляются последовательно до тех пор, пока не встретится выражение, значением которого является T. Вычисляется результирующее выражение, соответствующее этому предикату, и полученное значение возвращается в качестве значения выражения pn . Если истинного предиката нет, то значением COND будет NIL;

D. выражения pi вычисляются последовательно до тех пор, пока не встретится выражение, значением которого является T. Вычисляется результирующее выражение, соответствующее этому предикату, и полученное значение возвращается в качестве значения всего предложения COND. Если истинного предиката нет, то значением COND будет значение выражения p1.

9. Возможно ли в Lisp осуществлять передачу управления?

A. да, возможно; это приветствуется и активно используется;

B. да, возможно, но во многих системах не рекомендуется использовать такие операторы, так как их можно заменить другими предложениями и, как правило, в более понятной форме;

C. да, возможно, но для этого необходимо подключить специальную библиотеку;

D. нет, нельзя.

10. Заполните пропуск: Рекурсию называют _____, если она встречается одновременно в нескольких аргументах функции.

A. простой;

B. параллельной;

C. перекрёстной;

D. взаимной.

11. Продолжите предложение: APPLY является функцией двух аргументов, из которых …:

A. первый аргумент представляет собой список, а второй – функцию, которая применяется к элементам этого списка;

B. первый аргумент представляет собой символ, а второй – список, в начало которого добавляется этот символ;

C. первый аргумент представляет собой функцию, а второй – список, к элементам которого применяется эта функция;

D. первый аргумент представляет собой символ, а второй – список, в конец которого добавляется этот символ.

12. Как выглядит синтаксис определения макроса в Lisp?

A. (DEFMACRO имя лямбда-список тело);

B. (DEFMACRO имя тело лямбда-список);

C. (MACRO имя лямбда-список тело);

D. (MACRO имя тело лямбда-список).

13. Каким образом можно преобразовать список (a b c (d e)) в точечную нотацию?

A. _(a b c (d e)) o (a . b . c . (d . e));

B. _(a b c (d e)) o (a . b . c . d . e);

C. _(a b c (d e)) o (a . (b . (c . ((d . (e))))));

D. _(a b c (d e)) o (a . (b . (c . ((d . (e . nil)) . nil)))).

14. Заполните пропуск: Строка состоит из последовательности знаков. В строке знаки записываются в последовательности друг за другом, для

ограничения которой с обеих сторон в качестве ограничителя используется знак _____.

A. { };

B. [ ];

C. « »;

D. // //.

15. Продолжите предложение: Одной из важнейших концепций формализма фреймов является …:

A. абстракция;

B. ветвление;

C. отображение;

D. наследование.

16. Какая функция возвращает в качестве значения первый элемент списка:

A. CDR

B. CAR

C. ATOM

17. Какие основные типы данных языка lisp:

A. Строки и символы

B. Атомы и списки

C. Указатели и ссылки

18. Выбирите верную запись сложения двух чисел

A. +(2 4)

B. (2 + 4)

C. (+ 2 4)

19. Какой символ означает, что мы имеем дело с определением функции:

A. defun

B. lambda

C. function

20. Какая форма записи циклического вычисления верная

A. (DO ((var 1 знач1 шаг1) (var 2 знач2 шаг2) ...)
(условие-окончания форма11 форма12 ...)
форма21 форма22 ...)

B. (DO (условие-окончания) ((var 1 знач1 шаг1) (var 2 знач2 шаг2) ...)
(форма11 форма12 ...) форма21 форма22 ...)

C. (DO ((var 1 знач1 шаг1) ((var 2 знач2 шаг2) ...)
(форма11 форма12 ...) форма21 форма22 ...) (условие-окончания))

21. Для каких целей используется функция Defun

A. Для создания новых функции

B. Для именования функции

C. Для вызова функции

22. Функция чтения Read обрабатывает выражение:

A. посимвольно

B. целиком

C. построчно

23. Остаются или нет связи между статичискими переменными и значениями поле выполнения предложения LET:

A. Да

B. Нет

24. В чем отличие предложений Prog 1 и ProgN

A. В количестве используемых аргументов

B. В номере возвращаемого аргумента

C. Нет отличий

25. В каком порядке вычисляются операторы предложения Prog

A. Сверху вниз

B. Снизу вверх

C. Согласно их приоритету

26. Какая из этих записей является точечной парой

A. (.a b)

B. .(a b)

C. (a . b)

27. Какая из этих функций не является функцией вывода

A. Princ

B. Print

C. Println

28. Какое значение по умолчанию присваивается по умолчанию в начале вычисления приложения DOI:

A. Nil

B. 1

C. Значение не присваивается

29. В чем различие между функциями MAPCAR и MAPLIST

A. MAPCAR работает с символами, а MAPLIST со списками

B. MAPLIST осуществляет действия не над элементами списка, а над последовательными CDR этого списка.

C. Нету отличий

30. С помощью какого кодового символа начинается написание макроса:

A. MACRO

B. DEFMACRO

C. MACRO 1

31.Функция CAR в языке LISP служит для:

А. получения длины списка

Б. получения первого элемента списка

В. получения хвоста списка

Г. построения списка

32.Функция CDR служит для:

А. получения длины списка

Б. получения первого элемента списка

В. получения хвоста списка

Г. построения списка

33.Результатом выполнения _(CONS ‘(a b) ‘(c d)) будет

А. (a b c d)

Б. ((a) b c d)

В. ((a b) c d)

Г. выражение некорректно

34.Для присваивания символу свойств служит специальная функция языка LISP:

А. SETPROPERTY

Б. SETVALUE

В. Symbol-value

Г. специальной функции нет, нужно использовать SETF

35.С помощью, какой функции можно определить новую функцию в языке LISP:

А. defun

Б. function

В. procedure

Г. в языке LISP нельзя определить функцию

36.Какого предложения не существую в языке LISP:

А. SWITCH

Б. COND

В. IF

Г. CASE

37.Функция eval выполняет следующее действие:

А. Сравнивает значения

Б. Присваивает значение

В. Возвращает результат выражения

38.С помощью какого предложения можно работать с несколькими вычисляемыми формами:

А. LET

Б. PROGN

В. LIST

39.Синтаксис определения макроса выглядит:

А. (имя DEFMACRO лямбда-список тело)

Б. (DEFMACRO имя тело лямбда-список)

В. (DEFMACRO имя лямбда-список тело)

40.Какого вида функционала не существует:

А. Отображающего

Б. Применяющего

В. Вычисляющего

41.Какой формы рекурсии не существует:

А. параллельной

Б. простой

В. взаимной

Г. последовательной

42.Преобразовав данный список _(a b c (d e)) в точечную нотацию, получим:

А. (a . (b . (c . ((d . e)))))

Б. (a . (b . (c . ((d . (e . nil)) . nil))))

В. (a . (b . (c . ((d . (e . nil)) . nil))) .nil)

Г. (a . b . c . ((d . (e . nil))))

43.Можно ли определить структурный тип в языке LISP:

А. Нет

Б. Да, с помощью макроса DEFMACRO

В. Да, с помощью STRUCT

Г. Да, с помощью DEFSTRUCT

44.Какой тип данных отсутствует в языке LISP:

А. INTEGER

Б. CHAR

В. FLOAT

Г. LISP - бестиповый язык. Отсутствуют все выше приведенные типы

45.К отображающим функционалам относится:

А. MAPCAR

Б. APPLY

В. FUNCALL

Тестирование по теме
«Язык Пролог»

1. Раздел информатики, целью которого является разработка компьютерных интеллектуальных систем, называется . .

- теоретической информатикой;
- естественным интеллектом;
- кибернетикой;
- искусственным интеллектом;
- практической информатикой.

2. Что такое экспертная система?

- нейрокомпьютер;
- определенная предметная область искусственного интеллекта;
- система искусственного интеллекта, заключающая в себе знания специалиста – эксперта в определенной предметной области;
- компьютерная система, моделирующая рассуждения человека;
- логическая модель знаний.

3. Экспертные системы используются для …

- автоматического принятия сложных решений;
- оказания помощи для хранения баз знаний
- оказания помощи при работе с базами данных;
- оказания помощи при работе с базами знаний;
- оказания помощи в принятии сложных решений.

4. Логическая модель знаний состоит из …

- фактов и правил;
- фактов;
- правил;
- предложений;
- заявлений

5. Что такое факт?

- это логическая модель знаний;
- это утверждение общего характера;
- это утверждение правила;
- это частное утверждение;
- нет правильного ответа.

6. Что такое правило?

- это утверждение факта;
- это частное утверждение;
- это утверждение общего характера;
- это логическая модель знаний;
- нет правильного ответа.

7. Что такое база знаний?

- это компьютерная модель знаний специалиста в определенной предметной области;
- это компьютерная модель логических рассуждений специалиста в определенной предметной области;
- это компьютерная модель фактов;
- это компьютерная модель правил;
- все ответы правильные.

8.Что такое механизм вывода?

- нет правильного ответа;

- это модель алгоритма вывода ответов на экран монитора;
- это вывод ответов на внешние запоминающие устройства компьютера;
- это модель алгоритма
- это модель логических рассуждений, на основе базы знаний.создания ответов ;

9. В основу языка логического программирования ПРОЛОГ положена …

- модель правил базы знаний;
- модель логических рассуждений на основе базы знаний;
- модель эксперта;
- логическая модель структуры базы знаний;
- нет правильного ответа.

10. "Земля – планета Солнечной системы." Это …

- нет правильного ответа;
- правило;
- цель;
- механизм вывода;
- факт.

11. Если планета движется вокруг Солнца, то это планета Солнечной системы. Это …

- факт;
- правило;
- цель;
- механизм вывода;
- нет правильного ответа;

12. Запись сын(A,B):–отец(B,A) является:

- правилом;

- фактом;
- целью;
- механизмом вывода
- ; нет правильного ответа.

13. В записи сын(A,B):–отец(B,A) – …

- A,B – результаты;
- A,B – голова правила;
- B,A – тело правила;
- A,B – аргументы;
- A,B – факты.

14. В записи сын(A,B):–отец(B,A) – …

- сын(A,B) – левая конечность правила;
- отец(B,A) – голова правила;
- сын(A,B) – голова правила;
- отец(B,A) – правая конечность правила;
- нет правильного ответа.

15. В записи сын(A,B):–отец(B,A) – …

- отец(B,A) – тело правила;
- сын(A,B) – тело правила;
- сын(A,B) – левая конечность правила;
- отец(B,A) – правая конечность правила;
- нет правильного ответа.

16. Переменная (в терминологии Пролога) служит для обозначения

- конкретного факта;
- различных фактов;
- конкретной цели;
- различных правил;

- различных объектов.

17. В записи сын(A,B):–отец(B,A) – A и B …

- переменные, являющиеся именами конкретных объектов;
- аргументы, являющиеся именами конкретных объектов;
- переменные, не являющиеся именами конкретных объектов;
- константы, являющиеся именами конкретных объектов;
- все ответы правильные.

18. Запись сын(A,B):–отец(B,A) означает:

- ЕСЛИ B – отец A, ТО A является сыном B;
- ЕСЛИ A – отец B, ТО B является сыном A;
- ЕСЛИ A – сын B, ТО B является отцом A;
- ЕСЛИ B – сын A, ТО A является отцом B;
- нет правильного ответа.

19. Цель - это …

- нет правильного ответа;
- ответ на запрос (вопрос) к базе знаний;
- запрос (вопрос) к пользователю от базы знаний;
- ответ экспертной системы на запрос;
- запрос (вопрос) к базе знаний.

20. Цель (запрос) первого типа позволяет …

- опровергнуть справедливость факта;
- подтвердить справедливость факта; подтвердить справедливость правила;
- опровергнуть справедливость правила;
- нет правильного ответа.

21. В терминологии Пролога ставится цель - подтвердить справедливость факта. Какой получится ответ в результате?

- "да";
- "нет";
- "да" или "нет";
- название одного подходящего объекта;
- название всех подходящих объектов.

22. Цель (запрос) второго типа позволяет …

- перечислить все значения переменных, присутствующих в запросе и удовлетворяющих фактам и правилам базы знаний.
- перечислить все значения переменных, присутствующих в запросе и не удовлетворяющих фактам и правилам базы знаний.
- перечислить все значения переменных, присутствующих в запросе и удовлетворяющих фактам базы знаний.
- перечислить все значения переменных, присутствующих в запросе и удовлетворяющих правилам базы знаний.
- нет правильного ответа.

23. Запись вида ? -ворует(птица_Синица,X) является:

- записью;
- полем;
- правилом;
- целью;
- вопросом.

Условие для вопросов 25-30.

БЗ содержит информацию о домашний животных (кошках, собаках) и их хозяевах:

Собака(тузик).

Собака(фантик).

Кот(кузя).

Возраст(тузик,3) .

Возраст(фантик,5).

Возраст(кузя,6).

Хозяин(андрей,тузик).

Хозяин(андрей,кузя).Хозяин(оля,фантик).

Утверждение: собака - друг человека может быть записано с помощью правила Друг(X):-Собака(X)

25. Каким будет ответ на цель: ?Друг(X)

- тузик, фантик;
- тузик;
- нет;
- кузя;
- да.

26. Вслед за правилом Друг(X):-Собака (X) записано правило Друг(X):-Кот(X). Каким будет ответ на цель: ?Друг(X)

- нет;
- тузик, фантик, кузя;
- кузя;
- фантик, кузя;
- да.

27. В БЗ записаны правила: Друг(X):-Собака(X). Друг(X):-Кот(X). Каким будет ответ на цель: ?Друг(X),Кот(X)

- тузик, фантик, кузя;
- да;

- нет решения;
- кузя;
- тузик, фантик.

28. Выбрать цель, позволяющую найти всех котов в возрасте 6 лет.

- ?Кот(X),Возраст(X,6);
- ?Кот(X),Возраст(Y,6);
- ?Кот(X),Возраст(6,X);
- ?Кот(X),Возраст(6,Y);
- ?Возраст(Кот,6).

29. Выбрать цель, позволяющую найти всех собак в возрасте 3 года и имеющих хозяина Андрея.

- Собака(Y), Возраст (3,X), Хозяин(андрей,X);
- Собака(X), Возраст(X,3), Хозяин(андрей,X);
- Собака(X), Возраст(Y,3), Хозяин(андрей,Y);
- Собака(Y), Возраст(X,3), Хозяин(андрей,X);
- Собака(X), Возраст(3,X), Хозяин(андрей,X).

30. Каким будет ответ на цель: ?Хозяин(X, кузя),Хозяин(X,тузик).

- оля;
- андрей, оля;
- да;
- андрей;
- нет.

Тестирование по теме

«Язык C#»

1. Что будет выведено в результате выполнения программы?

```
class Program
{     static void Main(string[] args)
   {         var a = null;
      a = 10;
      Console.WriteLine(a);
      Console.ReadLine();
   }
}
```

Варианты ответа:

1. 0
2. 10
3. Пустая строка
4. Возникнет ошибка на этапе компиляции

2. Что будет выведено в результате выполнения программы?

```
class Program
{     enum En
   {         First = 15,
      Second,
      Third = 54
    }
    static void Main(string[] args)
   {         Console.WriteLine((int)En.Second);
      Console.ReadLine();
   }
}
```

Варианты ответа:

1. 0

2. 1
3. 16
4. Возникнет ошибка на этапе компиляции

3. Когда и почему использовать StringBuilder предпочтительнее чем string?

Варианты ответа:

1. Если строка редко изменяется
2. Если строка часто изменяется
3. Если строка содержит спецсимволы
4. Если строка содержит исключительно цифры

4. Что произойдет в результате выполнения программы?

```
class Test
{     static void Main(string[] args)
   {          Test test = null;
      try
      {               test = new Test();
      }
      catch (Exception)
      {               Console.Write("1");
      }
      Console.Write(test.N);
      Console.ReadLine();
   }
    public Test()
   {          throw new Exception();
   }
    public int N = 5;
}
```

Варианты ответа:

1. Будет выведено 15
2. Будет выведено 1
3. Возникнет NullReferenceException
4. Будет выведено 1, после чего возникнет необработанный NullReferenceException

5. Что будет выведено в результате выполнения программы?

```
class Test
{     static void Main(string[] args)
   {         var test = new Test();
      test.Print();
      Console.ReadLine();
   }
    public void Print()
   {         var array = new int[] {3, 5, 7};
      try
      {              Console.Write(array[5]);
      }
      catch (IndexOutOfRangeException)
      {              Console.Write("3");
      }
      catch (Exception)
      {              Console.Write("9");
      }
      finally
      {              Console.Write("7");
      }
   }
```

```
}
```

Варианты ответа:

1. 7
2. 37
3. 397
4. 97

6. Что будет выведено в результате выполнения программы?

```
class Program
{     static void Main(string[] args)
   {         string s1 = "asd";
      string s2 = "qwe";
      Swap(ref s1, ref s2);
      Console.WriteLine("s1: {0}, s2: {1}", s1, s2);
      Console.ReadLine();
   }
    public static void Swap(ref string s1, ref string s2)
   {         string tmpString = s1;
      s1 = s2;
      s2 = tmpString;
   }
}
```

Варианты ответа:

1. s1: qwe, s2: asd
2. s1: asd, s2: qwe
3. s1: qwe, s2: qwe
4. Возникнет ошибка на этапе компиляции

7. Что будет выведено в результате выполнения программы?

```
class Program
```

```
{     static void Main(string[] args)
   {        Console.WriteLine(Average(5, 3, 7));
      Console.ReadLine();
   }
    public static double Average(params int[] values)
   {        double sum = 0;
      for (int i = 0; i < values.Length; i++)
      {           sum += values[i];
      }
      return sum/values.Length;
   }
}
```

Варианты ответа:

1. 5
2. 15
3. 0
4. Возникнет ошибка на этапе компиляции

8. Что будет выведено в результате выполнения программы?

```
class Program
{     static void Main(string[] args)
   {        int c = 3;
      Console.Write(Sum(5, 3, out c) + " ");
      Console.Write(c);
      Console.ReadLine();
   }

   static int Sum(int a, int b, out int c)
   {        c = a*b;
      return a + b;
```

```
    }
}
```

Варианты ответа:

1. 8
2. 8 3
3. 8 15
4. Возникнет ошибка на этапе компиляции

9. Что будет выведено в результате выполнения программы? Проверка на переполнение в масштабах всего проекта выключена.

```
class Program
{     static void Main(string[] args)
   {          byte a = 200;
      byte b1 = 250;
      byte b2 = 10;
        byte c;
        try
      {              c = (byte) Add(a, b1);
         Console.Write(c + " ");
           c = (byte) Add(a, b2);
         Console.Write(c + " ");
           c = checked((byte) Add(a, b2));
         Console.Write(c + " ");
           c = checked((byte) Add(a, b1));
         Console.Write(c + " ");
      }
      catch (Exception e)
      {              Console.WriteLine(e.Message);
      }
        Console.ReadLine();
```

```
    }

    static int Add(int a, int b)
    {       return a + b;
    }
}
```

Варианты ответа:

- Arithmetic operation resulted in an overflow. 210 210 Arithmetic operation resulted in an overflow.
- 194 210 210 Arithmetic operation resulted in an overflow.
- 194 210 210 194
- Возникнет ошибка на этапе компиляции

10. Что будет выведено в результате выполнения программы?

```
class Program
{   static void Main(string[] args)
    {       var s1 = new S();
        s1.A = new A() {N = 3};
          var s2 = s1;
          s1.A.N = 5;
          Console.Write(s1.A.N);
        Console.Write(s2.A.N);
          Console.ReadLine();
    }
}
 struct S
{   public A A;
}
  class A
{   public int N;
```

```
}
```

Варианты ответа:

1. 55
2. 35
3. 33
4. Возникнет ошибка на этапе компиляции

11. Что будет выведено в результате выполнения программы?

```
class Program
{     static void Main(string[] args)
   {         try
      {              var a1 = new A1();
         Console.Write("1");
         var a2 = new A2();
      }
      catch (Exception)
      {              Console.Write("3");
      }
      Console.ReadLine();
   }
}

class A1
{     public int B1;
}

class A2
{     public int B2;
     public A2(int b2)
   {         B2 = b2;
```

```
    }
}
```

Варианты ответа:

1. 1
2. 3
3. 13
4. Возникнет ошибка на этапе компиляции

12.Что будет выведено в результате выполнения программы?

```
class Program
{     static void Main(string[] args)
   {         Console.Write(A.B);
      var a1 = new A();
      Console.Write(A.B);
      a1.Write();
      Console.ReadLine();
   }
}
 public class A
{     public static int B;
     public A()
   {         B = 3;
   }
     public void Write()
   {         Console.Write(B);
   }
     static A()
   {         B = 5;
   }
}
```

Варианты ответа:

1. 033
2. 533
3. 553
4. Возникнет ошибка на этапе компиляции

13.Что будет выведено в результате выполнения программы?

```
class Program
{      static void Main(string[] args)
    {          var a = new A(2) {B = 3};
          Console.Write(a.B);
          Console.ReadLine();
    }
}
  public class A
{      public int B { get; set; }
       public A(int b)
    {          Console.Write("1");
          B = b;
    }
}
```

Варианты ответа:

1. 3
2. 12
3. 13
4. Возникнет ошибка на этапе компиляции

14.Какие утверждения относительно языка C# верны?

Варианты ответа:

1. Допустимо множественное наследование
2. Класс может реализовать несколько интерфейсов
3. Интерфейс может наследоваться от множества других интерфейсов
4. Нельзя наследовать от класса, помеченного ключевым словом sealed

15.Что будет выведено в результате выполнения программы?

```
class Program
{      static void Main(string[] args)
    {          var b = new B();
        var c = new C();
          Console.Write(b.Sum(2, 3));
        Console.Write(c.Sum(2, 3));
        Console.ReadLine();
    }
}
class A
{                public virtual int Sum(int a, int b)
                        {                    return a + b;
     }
}
class B : A
{
}
class C : A
{       public override int Sum(int a, int b)
     {
         return a + b + 1;
     }
}
```

Варианты ответа:

1. 55
2. 56
3. 66
4. Возникнет ошибка на этапе компиляции

16. Что будет выведено в результате выполнения программы?

```
class Program
{     static void Main(string[] args)
   {          var a = new A {B = 3};
      Console.Write(a is System.Object);
        var o = a as System.Object;
      Console.Write(((A)o).B);
        Console.ReadLine();
   }
}
 public class A
{     public int B;
}
```

Варианты ответа:

1. True
2. False3
3. True3
4. Возникнет ошибка на этапе компиляции

Тестирование по теме

«Язык Java»

1. Имеется следующий код:

```
public class Overload{
```

```
  public void method(Object o) {
    System.out.println("Object");
  }
  public void method(java.io.FileNotFoundException f) {
    System.out.println("FileNotFoundException");
  }
  public void method(java.io.IOException i) {
    System.out.println("IOException");
  }
  public static void main(String args[]) {
    Overload test = new Overload();
    test.method(null);
  }
}
```

Результатом его компиляции и выполнения будет:

1. Ошибка компиляции
2. Ошибка времени выполнения
3. «Object»
4. «FileNotFoundException»
5. «IOException»

2. Имеется следующий код:

```
Float f1 = new Float(Float.NaN);
Float f2 = new Float(Float.NaN);
System.out.println( ""+ (f1 == f2)+" "+f1.equals(f2)+ " "+(Float.NaN ==
Float.NaN) );
```

Что будет выведено в результате выполнения данного куска кода:

1. false false false
2. false true false
3. true true false
4. false true true
5. true true true

3. Имеется следующий код:

```
class Mountain {
  static String name = "Himalaya";
  static Mountain getMountain() {
    System.out.println("Getting Name ");
    return null;
  }
  public static void main(String[ ] args) {
    System.out.println( getMountain().name );
  }
}
```

Что произойдет при попытке выполнения данного кода:

1. Будет выведено «Himalaya» но НЕ будет выведено «Getting Name „
2. Будет выведено “Getting Name » и «Himalaya»
3. Ничего не будет выведено
4. Будет выброшен NullPointerException
5. Будет выведено «Getting Name », а потом выброшено NullPointerException

4. Имеется следующий код:

```
Integer a = 120;
Integer b = 120;
Integer c = 130;
```

```
Integer d = 130;
System.out.println(a==b);
System.out.println(c==d);
```

В результате выполнения данного кода будет выведено:

1. true true
2. false false
3. false true
4. true false
5. произойдет ошибка времени выполнения

5. Имеется следующий код:

5a. Имеется следующий код:

```
//In File Other.java
package other;
public class Other { public static String hello = "Hello"; }

//In File Test.java
package testPackage;
import other.*;
class Test{
  public static void main(String[] args) {
    String hello = "Hello", lo = "lo";
    System.out.print((testPackage.Other.hello == hello) + " ");
    System.out.print((other.Other.hello == hello) + " ");
    System.out.print((hello == ("Hel"+"lo")) + " ");
    System.out.print((hello == ("Hel"+lo)) + " ");
    System.out.println(hello == ("Hel"+lo).intern());
  }
```

```
}
class Other { static String hello = "Hello"; }
```

В результате мы получим:

1. false true true false true
2. false false true false true
3. true true true true true
4. true true true false true
5. Все ответы неверны

6. Имеется следующий код:

Дана сигнатура метода:

```
public static <E extends CharSequence> List<? super E> doIt(List<E> nums)
```

Который вызывается как-то так:

```
result = doIt(in);
```

Какого типа должны быть result и in?

1. ArrayList<String> in; List<CharSequence> result;
2. List<String> in; List<Object> result;
3. ArrayList<String> in; List result;
4. List<CharSequence> in; List<CharSequence> result;
5. ArrayList<Object> in; List<CharSequence> result;

7. Имеется следующий код:

```
public static void doIt(String String) { //1
  int i = 10;
  i : for (int k = 0 ; k< 10; k++) { //2
    System.out.println( String + i); //3
```

```
    if( k*k > 10) continue i; //4
  }
}
```

Данный код:

1. Не скомпилируется из-за строки 1
2. Не скомпилируется из-за строки 2
3. Не скомпилируется из-за строки 3
4. Не скомпилируется из-за строки 4
5. Скомпилируется и запустится без проблем

8. Имеется следующий код:

```
public class Main {
  static void method(int... a) {
    System.out.println("inside int...");
  }
  static void method(long a, long b) {
    System.out.println("inside long");
  }
  static void method(Integer a, Integer b) {
    System.out.println("inside INTEGER");
  }
  public static void main(String[] args) {
    int a = 2;
    int b = 3;
    method(a,b);
  }
}
```

В результате мы получим:

1. Ошибку компиляции
2. Ошибку времени выполнения
3. «inside int...»
4. «inside long»
5. «inside INTEGER»

9. Имеется следующий код:

```
class Super { static String ID = "QBANK"; }
class Sub extends Super{
  static { System.out.print("In Sub"); }
}
class Test{
  public static void main(String[] args) {
    System.out.println(Sub.ID);
  }
}
```

В результате выполнения данного кода:

1. Он даже не скомпилируется
2. Результат зависит от реализации JVM
3. Будет выведено «QBANK»
4. Будет выведено «In Sub» и «QBANK»
5. Все ответы неверны

10. Имеется следующий код:

Имеется два класса:

```
//in file A.java
package p1;
public class A{
  protected int i = 10;
  public int getI() { return i; }
}

//in file B.java
package p2;
import p1.*;
public class B extends A{
  public void process(A a) {
    a.i = a.i*2;
  }
  public static void main(String[] args) {
    A a = new B();
    B b = new B();
    b.process(a);
    System.out.println( a.getI() );
  }
}
```

В результате выполнения класса B мы получим:

1. Будет выведено «20»
2. Будет выведено «10»
3. Код не скомпилирутся
4. Возникнет ошибка времени выполнения
5. Все ответы неверны