ПРИЛОЖЕНИЕ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ. В.Ф. УТКИНА»

МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ «Инструментальные средства информационных систем»

Направление подготовки

09.03.02 «Информационные системы и технологии»

Направленность (профиль) подготовки

Информационные системы и технологии

Квалификация выпускника – бакалавр

Форма обучения – очная, заочная

Рязань 2021

1. ПЛАНЫ ЛАБОРАТОРНЫХ РАБОТ

Лабораторная работа № 1
Инструментальные средства обработки регулярных выражений

Цель работы: изучение инструментальных средств, применяемых при обработке регулярных выражений.

Задание

Задача 1. Проверить вхождение шаблона в заданные выражения.

Задача 2. Найти первую подстроку, соответствующую шаблону в заданном выражении.

Задача 3. Найти все подстроки, соответствующие шаблону в заданном выражении.

Задача 4. Выполнить замену подстроки, соответствующей шаблону в заданном выражении.

Задача 5. Вывести массив строк, полученный в результате разделения входящей строки в местах соответствия шаблону регулярного выражения.

Задача 6. Написать программу для проверки корректности адреса электронной почты.

Задача 7. Написать программу с использованием параметра поиска (игнорирование регистра).

Задача 8. Написать программу для проверки правильности ввода логина с заданными требованиями.

Задача 9. Создать регулярное выражение, которому соответствовали бы любые ошибочные написания заданного слова, чтобы иметь возможность отыскивать это слово в документе, не полагаясь на грамотность автора. Предполагается, что на месте любой гласной буквы может использоваться один из заданных символов.

Задача 10. Создать регулярное выражение, которому соответствовала бы единственная шестнадцатеричная цифра.

Задача 11. Создать регулярное выражение, которому соответствовал бы единственный символ, не являющийся шестнадцатеричной цифрой.

Задача 12. Создать регулярное выражение, которому должно соответствовать заданное слово, только если оно находится в самом начале испытуемого текста.

Задача 13. Создать регулярное выражение, которому должно соответствовать заданное слово, только если оно находится в самом конце испытуемого текста.

Задача 14. Создать регулярное выражение, которому должно соответствовать заданное слово, только если оно находится в самом начале строки.

Задача 15. Создать регулярное выражение, которому должно соответствовать заданное слово, только если оно находится в самом конце строки.

Задача 16. Создать регулярное выражение, которому соответствовало бы заданное слово (например, кот) в тексте, но которое не находило бы соответствие внутри слов (например, котел, котомка).

Задача 17. Создать регулярное выражение, которому соответствовало бы символ заданного слова (например, кот) в тексте, но которое не находило бы соответствие слова (например, кот).

Задача 18. Создать регулярное выражение, которому соответствовало бы вхождение одной из последовательностей (например, суп, жаркое, сок**)** в тексте.

Задача 19. Создать регулярное выражение, которому соответствовало бы вхождению заданных слов (например, суп, жаркое, сок) в тексте как целых слов.

Задача 20. Создать регулярное выражение, которому соответствовали бы любые даты в формате yyyy-mm-dd, и сохраняющее по отдельности год, месяц и день. Цель состоит в том, чтобы облегчить работу с отдельными значениями в программном коде, обрабатывающем совпадение. Введем ограничение, что все даты, присутствующие в испытуемом тексте,

корректны. Регулярное выражение не должно исключать такие даты, как 9999-99-99, так как они вообще не будут появляться в испытуемом тексте.

Задача 21. Создать регулярное выражение, совпадающее с «магическими» датами в формате yyyy-mm-dd. Магической считается дата, когда последние две цифры года, месяц и день являются одним и тем же числом. Например, магической считается дата 2008-08-08. Введем ограничение, что все даты, присутствующие в испытуемом тексте, корректны. Регулярное выражение не должно исключать такие даты, как 9999-99-99, так как они вообще не будут появляться в испытуемом тексте. Требуется отыскать только магические даты.

Задача 22. Создать регулярное выражение, которому соответствовали бы любые даты в формате yyyy-mm-dd и сохраняющее по отдельности год, месяц и день. Цель состоит в том, чтобы облегчить работу с отдельными значениями в программном коде, обрабатывающем совпадение. Введем ограничение, что все даты, присутствующие в испытуемом тексте, корректны. Содействуя достижению этой цели, необходимо присвоить сохраняемым фрагментам текста описательные имена «year», «month» и «day».

Задача 23. Создать другое регулярное выражение, совпадающее с «магическими» датами в формате yyyy-mm-dd. Магической считается дата, когда последние две цифры года, месяц и день являются одним и тем же числом. Например, магической считается дата 2012-12-12. Необходимо сохранить магическое число (12 в примере) и пометить его именем «magic».

Варианты заданий приведены в источнике [1].

Рекомендуемая литература:

1. № 6012 Регулярные выражения: методические указания / Рязан. гос. радиотехн. ун-т; сост. С.В. Челебаев. Рязань, 2021. 48 с. https://elib.rsreu.ru/ebs/download/2948 (требуется авторизация).

Лабораторная работа № 2
Инструментальные средства распараллеливания вычисленйи в информационных системах

Цель работы: многопоточное обучение искусственной нейронной сети на основе дельта-правила с использованием класса *Thread* языка *C#*.

Краткие теоретические сведения

В языке C# для создания пользовательских потоков и работы с ними используется класс *Thread*. Для создания объекта класса *Thread* нужно вызвать конструктор *Thread()*, указав в качестве аргумента имя метода, который будет вызван при запуске потока, например:

```
Thread trd1 = new Thread(fun1);
```

В примере создан объект класса *Thread* с именем *trd1*, при запуске которого будет вызван метод *fun1*.

Для запуска потока необходимо вызвать метод *Start()*, например:

```
trd1.Start();
```

Задать приоритет потока можно с помощью свойства *Priority*. Возможные значения: Highest, AboveNormal, Normal, BelowNormal, Lowest.

```
trd1.Priority = ThreadPriority.AboveNormal;
```

Метод *Join()* блокирует вызывающий поток до завершения потока, продолжая отправлять стандартные сообщения COM и SendMessage. Может иметь аргумент Int32.

Задание

1. Оформить в вире подпрограммы алгоритм обучения искусственной нейронной сети (дельта-правило) для заданного количества нейронов. В качестве обязательных параметров

подпрограмма должна иметь: параметр *b* активационной функции, целевая погрешность, количество эпох обучения, скорость обучения).

2. Вызвать написанную подпрограмму в четырех одновременно работающих потоках для целевой погрешности 0,001 и наиболее целесообразных значений параметров активационных функций (взятых из прошлой работы). Количество эпох обучения – не менее 100 000.

Таблица 1 – Варианты заданий для работы № 2

Вариант	Количество нейронов
1	30
2	31
3	32
4	33
5	34
6	35
7	36
8	37
9	38
10	39
11	30
12	31
13	32
14	33
15	34
16	35
17	36
18	37
19	38
20	39
21	30
22	31
23	32
24	33
25	34
26	35
27	36
28	37
29	38
30	39
31	40

Контрольные вопросы

1. Класс *Thread*.
2. Приоритеты потоков.
3. Делегирование.
4. Создание объекта потока.
5. Запуск потока на выполнение.
6. Метод *Join()*.

Лабораторная работа № 3

Многопоточная реализация алгоритма обучения нейронной сети с разбиением нейронов 1-го слоя на задачи на основе класса Task

Цель работы: многопоточное обучение искусственной нейронной сети на основе дельта-правила с использованием класса Task языка C#.

Задание.

1. Оформить в вире подпрограммы алгоритм обучения искусственной нейронной сети (из лабораторной работы № 2) для заданного количества нейронов. В качестве обязательных параметров подпрограмма должна иметь: параметр b активационной функции, целевая погрешность, количество эпох обучения, скорость обучения).

2. Вызвать написанную подпрограмму в четырех одновременно работающих потоках для целевой погрешности в два раза ниже, чем в лаб. работе № 3 и наиболее целесообразных значений параметров активационных функций (взятых из прошлой работы). Функция – из работы № 2. Количество эпох обучения – не менее 100 000.

Варианты заданий взять из таблицы № 1.

Контрольные вопросы

1. Назначение класса *Task*.
2. Методы класса *Task*.

Лабораторная работа № 4

Инструментальные средства проектирования интеллектуальных информационных систем

Цель работы: **Цель работы:** изучение принципов работы динамических баз данных в языке Пролог, разработка динамической базы данных на заданную тему.

Задание

Написать программу, реализующую компьютерный вариант телефонного справочника. Основное назначение программы — находить по фамилии человека его телефонный номер или, наоборот, по телефонному номеру — фамилию владельца телефона. У пользователя программы должна быть возможность добавлять информацию в базу данных, а также удалять и изменять устаревшую информацию.

Таблица 4 – Варианты заданий

№ варианта	Поле
1	Имя
2	Год рождения
3	СНИЛС
4	ИНН
5	Номер паспорта
6	Город
7	Место работы
8	Специальность
9	Год окончания университета
10	Год окончания школы
11	Хобби
12	Любимое блюдо
13	Любимый напиток
14	Стаж работы

15	Номер группы
16	Факультет
17	Кафедра
18	Место отдыха
19	Любимый фильм
20	Год поступления в университет

2. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ СТУДЕНТАМ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Перед началом изучения дисциплины студенту необходимо ознакомиться с содержанием рабочей программы дисциплины, с целями и задачами дисциплины, ее связями с другими дисциплинами образовательной программы, методическими разработками по данной дисциплине, имеющимися на образовательном портале РГРТУ и сайте кафедры.

Методические рекомендации студентам по работе над конспектом лекции

Основу теоретического обучения студентов составляют лекции. Они дают систематизированные знания студентам о наиболее сложных и актуальных проблемах изучаемой дисциплины. На лекциях особое внимание уделяется не только усвоению студентами изучаемых проблем, но и стимулированию их активной познавательной деятельности, творческого мышления, развитию научного мировоззрения, профессионально-значимых свойств и качеств.

Перед каждой лекцией студенту необходимо просматривать рабочую программу дисциплины, что позволит сэкономить время на записывание темы лекции, ее основных вопросов, рекомендуемой литературы.

Перед очередной лекцией необходимо просмотреть по конспекту материал предыдущей лекции. При затруднениях в восприятии материала следует обратиться к основным литературным источникам. Если разобраться в материале опять не удалось, то обратитесь к лектору (по графику его консультаций) или к преподавателю на практических занятиях. Не оставляйте «белых пятен» в освоении материала.

Во время лекции студенты должны не только внимательно воспринимать действия преподавателя, но и самостоятельно мыслить, добиваться понимания изучаемого предмета. Студенты должны аккуратно вести конспект. В случае недопонимания какой-либо части предмета следует задать вопрос в установленном порядке преподавателю. В процессе работы на лекции необходимо так же выполнять в конспектах модели изучаемого предмета (рисунки, схемы, чертежи и т.д.), которые использует преподаватель.

Слушая лекцию, нужно из всего получаемого материала выбирать и записывать самое главное. Следует знать, что главные положения лекции преподаватель обычно выделяет интонацией или повторяет несколько раз. Именно поэтому предварительная подготовка к лекции позволит студенту уловить тот момент, когда следует перейти к конспектированию, а когда можно просто внимательно слушать лекцию. В связи с этим нелишне перед началом сессии еще раз бегло просмотреть учебники или прежние конспекты по изучаемым предметам. Это станет первичным знакомством с тем материалом, который прозвучит на лекции, а также создаст необходимый психологический настрой.

Чтобы правильно и быстро конспектировать лекцию важно учитывать, что способы подачи лекционного материала могут быть разными. Преподаватель может диктовать материал, или рассказывать его, не давая ничего под запись, или проводить занятие в форме диалога со студентами. Чаще всего можно наблюдать соединение двух или трех вышеназванных способов.

Эффективность конспектирования зависит от умения владеть правильной методикой записи лекции. Конечно, способы конспектирования у каждого человека индивидуальны. Однако существуют некоторые наиболее употребляемые и целесообразные приемы записи лекционного материала.

Запись лекции можно вести в виде тезисов – коротких, простых предложений, фиксирующих только основное содержание материала. Количество и краткость тезисов может определяться как преподавателем, так и студентом. Естественно, что такая запись лекции требует впоследствии

обращения к дополнительной литературе. На отдельные лекции можно приносить соответствующий иллюстративный материал на бумажных или электронных носителях, представленный лектором на портале или присланный на «электронный почтовый ящик группы» (таблицы, графики, схемы). Данный материал будет охарактеризован, прокомментирован, дополнен непосредственно на лекции.

Кроме тезисов важно записывать примеры, доказательства, даты и цифры. Значительно облегчают понимание лекции те схемы и графики, которыми преподаватель иллюстрирует теоретический материал. По мере возможности студенты должны переносить их в тетрадь рядом с тем текстом, к которому эти схемы и графики относятся.

Хорошо если конспект лекции дополняется собственными мыслями, суждениями, вопросами, возникающими в ходе прослушивания содержания лекции. Те вопросы, которые возникают у студента при конспектировании лекции, не всегда целесообразно задавать сразу при их возникновении, чтобы не нарушить ход рассуждений преподавателя. Студент может попытаться ответить на них сам в процессе подготовки к практическим занятиям либо обсудить их с преподавателем на консультации.

Важно и то, как будет расположен материал в лекции. Если запись тезисов ведется по всей строке, то целесообразно отделять их время от времени красной строкой или пропуском строки. Примеры же и дополнительные сведения можно смещать вправо или влево под тезисом, а также на поля. В тетради нужно выделять темы лекций, записывать рекомендуемую для самостоятельной подготовки литературу, внести фамилию, имя и отчество преподавателя. Наличие полей в тетради позволяет не только получить «ровный» текст, но и дает возможность при необходимости вставить важные дополнения и изменения в конспект лекции.

При составлении конспектов необходимо использовать избыточность русского языка, сокращая слова. Так в процессе совершенствования навыков конспектирования лекций важно выработать индивидуальную систему записи материала, научиться рационально сокращать слова и отдельные словосочетания.

Практика показывает, что не всегда студенту удается успевать записывать слова лектора даже при использовании приемов сокращения слов. В этом случае допустимо обратиться к лектору с просьбой повторить сказанное. При обращении важно четко сформулировать просьбу, указать какой отрывок необходимо воспроизвести еще раз. Однако не всегда удобно прерывать ход лекции. В этом случае можно оставить пропуск, и после лекции устранить его при помощи конспекта соседа. Важно сделать это в короткий срок, пока свежа память о воспринятой на лекции информации.

Работу над конспектом следует начинать с его доработки, желательно в тот же день, пока материал еще легко воспроизводим в памяти (через 10 часов после лекции в памяти остается не более 30-40 % материала). С целью доработки необходимо прочитать записи, восстановить текст в памяти, а также исправить описки, расшифровать не принятые ранее сокращения, заполнить пропущенные места, понять текст, вникнуть в его смысл. Далее следует прочитать материал по рекомендуемой литературе, разрешая в ходе чтения возникшие ранее затруднения, вопросы, а также дополняя и исправляя свои записи. Записи должны быть наглядными, для чего следует применять различные способы выделений. В ходе доработки конспекта углубляются, расширяются и закрепляются знания, а также дополняется, исправляется и совершенствуется конспект.

Подготовленный конспект и рекомендуемая литература используются при подготовке к лабораторным работам и практическим занятиям. Подготовка сводится к внимательному прочтению учебного материала, к выводу с карандашом в руках всех утверждений и формул, к решению примеров, задач, к ответам на вопросы. Примеры, задачи, вопросы по теме являются средством самоконтроля.

Непременным условием глубокого усвоения учебного материала является знание основ, на которых строится изложение материала. Обычно преподаватель напоминает, какой ранее изученный материал и в какой степени требуется подготовить к очередному занятию. Обращение к ранее изученному материалу не только помогает восстановить в памяти известные положения, выводы, но и приводит разрозненные знания в систему, углубляет и расширяет их. Каждый возврат к старому материалу позволяет найти в нем что-то новое, переосмыслить его с иных позиций, определить для него наиболее подходящее место в уже имеющейся системе знаний. Неоднократное обращение к пройденному материалу является наиболее рациональной формой приобретения и закрепления знаний.

Методические рекомендации студентам по работе с литературой

В рабочей программе дисциплины для каждого раздела и темы дисциплины указывается основная и дополнительная литература, позволяющая более глубоко изучить данный вопрос. Обычно список всей рекомендуемой литературы преподаватель озвучивает на первой лекции или дает ссылки на ее местонахождение (на образовательном портале РГРТУ, на сайте кафедры и т.д.).

При работе с рекомендуемой литературой целесообразно придерживаться такой последовательности. Сначала лучше прочитать заданный текст в быстром темпе. Цель такого чтения заключается в том, чтобы создать общее представление об изучаемом материале, понять общий смысл прочитанного. Затем прочитать вторично, более медленно, чтобы в ходе чтения понять и запомнить смысл каждой фразы, каждого положения и вопроса в целом.

Чтение приносит пользу и становится продуктивным, когда сопровождается записями. Это может быть составление плана прочитанного текста, тезисы или выписки, конспектирование и др. Выбор вида записи зависит от характера изучаемого материала и целей работы с ним. Если содержание материала несложное, легко усваиваемое, можно ограничиться составлением плана. Если материал содержит новую и трудно усваиваемую информацию, целесообразно его законспектировать.

План – это схема прочитанного материала, перечень вопросов, отражающих структуру и последовательность материала.

Конспект – это систематизированное, логичное изложение материала источника. Различаются четыре типа конспектов:

- план-конспект – это развернутый детализированный план, в котором по наиболее сложным вопросам даются подробные пояснения,
- текстуальный конспект – это воспроизведение наиболее важных положений и фактов источника,
- свободный конспект – это четко и кратко изложенные основные положения в результате глубокого изучения материала, могут присутствовать выписки, цитаты, тезисы; часть материала может быть представлена планом,
- тематический конспект – составляется на основе изучения ряда источников и дает ответ по изучаемому вопросу.

В процессе изучения материала источника и составления конспекта нужно обязательно применять различные выделения, подзаголовки, создавая блочную структуру конспекта. Это делает конспект легко воспринимаемым и удобным для работы.

Методические рекомендации студентам по подготовке к лабораторным работам

Лабораторная работа — это форма организации учебного процесса, когда обучающиеся по заданию и под руководством преподавателя самостоятельно проводят вычислительные расчеты и экспериментальные исследования на основе специально разработанных заданий.

Для проведения лабораторных работ используется вычислительная техника, которая размещается в специально оборудованных учебных лабораториях. Перед началом цикла лабораторных работ преподаватель или другое ответственное лицо проводит с обучающимися инструктаж о правилах техники безопасности в данной лаборатории, после чего студенты расписываются в специальном журнале техники безопасности.

По каждой лабораторной работе разрабатываются методические указания по их проведению. Они используются обучающимися при выполнении лабораторной работы.

Применяются разные формы организации обучающихся на лабораторных работах: фронтальная, групповая и индивидуальная. При фронтальной форме организации занятий все обучающиеся выполняют одновременно одну и ту же работу. При групповой форме организации занятий одна и та же работа выполняется группами по 2-5 человек. При индивидуальной форме организации занятий каждый обучающийся выполняет индивидуальное задание. Выбор метода зависит от учебно-методической базы и задач дисциплины.

До начала лабораторной работы студент должен ознакомиться с теоретическими вопросами, которые будут изучаться или исследоваться в этой работе. Также необходимо познакомиться с принципами работы лабораторного оборудования, используемого в лабораторной работе. Перед началом лабораторной работы преподаватель может провести проверку знаний обучающихся - их теоретической готовности к выполнению задания. По итогам этой проверки студент допускается или не допускается к

данной работе. О такой исходной проверке преподаватель информирует студентов заранее. Также возможна ситуация, когда допуском к очередной лабораторной работе является своевременная сдача предыдущей лабораторной работы (или подготовка отчета по ней).

Во время лабораторной работы обучающиеся выполняют запланированное лабораторное задание. Все полученные результаты необходимо зафиксировать в черновике отчета или сохранить в электронном виде на сменном носителе.

Завершается лабораторная работа оформлением индивидуального отчета и его защитой перед преподавателем.

Методические рекомендации студентам по подготовке к экзамену

При подготовке к экзамену студент должен повторно изучить конспекты лекций и рекомендованную литературу, просмотреть решения основных задач, решенных самостоятельно и на практических занятиях.

Необходимо помнить, что промежутки между очередными экзаменами обычно составляют всего несколько дней. Поэтому подготовку к ним нужно начинать заблаговременно в течение семестра. До наступления сессии уточните у преподавателя порядок проведения промежуточной аттестации по его предмету и формулировки критериев для количественной оценивания уровня подготовки студентов. Для итоговой положительной оценки по предмету необходимо вовремя и с нужным качеством выполнить или защитить контрольные работы, лабораторные работы, так как всё это может являться обязательной частью учебного процесса по данной дисциплине.

Рекомендуется разработать план подготовки к каждому экзамену, в котором указать, какие вопросы или билеты нужно выучить, какие задачи решить за указанный в плане временной отрезок.

Также бывает полезно вначале изучить более сложные вопросы, а затем переходить к изучению более простых вопросов. При этом желательно в начале каждого следующего дня подготовки бегло освежить в памяти выученный ранее материал.

В период экзаменационной сессии организм студента работает в крайне напряженном режиме и для успешной сдачи сессии нужно не забывать о простых, но обязательных правилах:

- по возможности обеспечить достаточную изоляцию: не отвлекаться на разговоры с друзьями, просмотры телепередач, общение в социальных сетях;
- уделять достаточное время сну;
- отказаться от успокоительных. Здоровое волнение – это нормально. Лучше снимать волнение небольшими прогулками, самовнушением;
- внушать себе, что сессия – это не проблема. Это нормальный рабочий процесс. Не накручивайте себя, не создавайте трагедий в своей голове;
- помогите своему организму – обеспечьте ему полноценное питание, давайте ему периоды отдыха с переменой вида деятельности;
- следуйте плану подготовки.

Методические рекомендации студентам по проведению самостоятельной работы

Самостоятельная работа студента над учебным материалом является неотъемлемой частью учебного процесса в вузе.

В учебном процессе образовательного учреждения выделяются два вида самостоятельной работы:

1) аудиторная – выполняется на учебных занятиях, под непосредственным руководством преподавателя и по его заданию), студентам могут быть предложены следующие виды заданий:

– выполнение самостоятельных работ;
– выполнение лабораторных работ;
– составление схем, диаграмм, заполнение таблиц;
– решение задач;
– работу со справочной, нормативной документацией и научной литературой;
– защиту выполненных работ;
– тестирование и т.д.

2) внеаудиторная – выполняется по заданию преподавателя, но без его непосредственного участия, включает следующие виды деятельности.

– подготовку к аудиторным занятиям (теоретическим и лабораторным работам);

– изучение учебного материала, вынесенного на самостоятельную проработку: работа над определенными темами, разделами, вынесенными на самостоятельное изучение в соответствии с рабочими программами учебной дисциплины или профессионального модуля;

– выполнение домашних заданий разнообразного характера;

– выполнение индивидуальных заданий, направленных на развитие у студентов самостоятельности и инициативы;

– подготовку к практической работе, экзамену;

– другие виды внеаудиторной самостоятельной работы.

Внеаудиторные самостоятельные работы представляют собой логическое продолжение аудиторных занятий, проводятся по заданию преподавателя, который инструктирует студентов и устанавливает сроки выполнения задания.

При планировании заданий для внеаудиторной самостоятельной работы используются следующие типы самостоятельной работы:

– воспроизводящая (репродуктивная), предполагающая алгоритмическую деятельность по образцу в аналогичной ситуации. Включает следующую основную деятельность: самостоятельное прочтение, просмотр, конспектирование учебной литературы, прослушивание записанных лекций, заучивание, пересказ, запоминание, Internet–ресурсы, повторение учебного материала и др.

– реконструктивная, связанная с использованием накопленных знаний и известного способа действия в частично измененной ситуации, предполагает подготовку отчетов по лабораторным работам, подбор литературы по дисциплинарным проблемам, подготовка к защите лабораторных работ и др.

– эвристическая (частично-поисковая) и творческая, направленная на развитие способностей студентов к исследовательской деятельности.

Одной из важных форм самостоятельной работы студента является работа с литературой ко всем видам занятий. Самостоятельная работа студента с литературой позволяет ему более углубленно вникнуть в изучаемую тему.

Один из методов работы с литературой – повторение: прочитанный текст можно заучить наизусть. Простое повторение воздействует на память механически и поверхностно. Полученные таким путем сведения легко забываются.

Более эффективный метод – метод кодирования: прочитанный текст нужно подвергнуть большей, чем простое заучивание, обработке. Чтобы основательно обработать информацию и закодировать ее для хранения, важно провести целый ряд мыслительных операций: прокомментировать новые данные; оценить их значение; поставить вопросы; сопоставить полученные сведения с ранее известными. Для улучшения обработки информации очень важно устанавливать осмысленные связи, структурировать новые сведения.

Изучение научной, учебной и иной литературы требует ведения рабочих записей. Форма записей может быть весьма разнообразной: простой или развернутый план, тезисы, цитаты, конспект.

План – структура письменной работы, определяющая последовательность изложения материала. Он является наиболее краткой и потому самой доступной и распространенной формой записей содержания исходного источника информации. По существу, это перечень основных вопросов, рассматриваемых в источнике. План может быть простым и развернутым. Их отличие состоит в степени детализации содержания и, соответственно, в объеме.

Преимущество плана состоит в том, что план позволяет наилучшим образом уяснить логику мысли автора, упрощает понимание главных моментов произведения. Кроме того, он позволяет быстро и глубоко проникнуть в сущность построения произведения и, следовательно, гораздо легче ориентироваться в его содержании и быстрее обычного вспомнить прочитанное. С помощью плана гораздо удобнее отыскивать в источнике нужные места, факты, цитаты и т. д.

Выписки представляют собой небольшие фрагменты текста (неполные и полные предложения, отдельные абзацы, а также дословные и близкие к дословным записи об излагаемых в нем фактах), содержащие в себе основной смысл содержания прочитанного. Выписки представляют собой более сложную форму записи содержания исходного источника информации. По сути, выписки – не что иное,

как цитаты, заимствованные из текста. Выписки позволяют в концентрированные форме и с максимальной точностью воспроизвести наиболее важные мысли автора. В отдельных случаях – когда это оправдано с точки зрения продолжения работы над текстом – вполне допустимо заменять цитирование изложением, близким дословному.

Тезисы – сжатое изложение содержания изученного материала в утвердительной (реже опровергающей) форме. Отличие тезисов от обычных выписок состоит в том, что тезисам присуща значительно более высокая степень концентрации материала. В тезисах отмечается преобладание выводов над общими рассуждениями. Записываются они близко к оригинальному тексту, т. е. без использования прямого цитирования.

Аннотация – краткое изложение основного содержания исходного источника информации, дающее о нем обобщенное представление. К написанию аннотаций прибегают в тех случаях, когда подлинная ценность и пригодность исходного источника информации исполнителю письменной работы окончательно неясна, но в то же время о нем необходимо оставить краткую запись с обобщающей характеристикой.

Резюме – краткая оценка изученного содержания исходного источника информации, полученная, прежде всего, на основе содержащихся в нем выводов. Резюме весьма сходно по своей сути с аннотацией. Однако, в отличие от последней, текст резюме концентрирует в себе данные не из основного содержания исходного источника информации, а из его заключительной части, прежде всего, выводов. Но, как и в случае с аннотацией, резюме излагается своими словами – выдержки из оригинального текста в нем практически не встречаются.

Конспект представляет собой сложную запись содержания исходного текста, включающая в себя заимствования (цитаты) наиболее примечательных мест в сочетании с планом источника, а также сжатый анализ записанного материала и выводы по нему.

При выполнении конспекта требуется внимательно прочитать текст, уточнить в справочной литературе непонятные слова и вынести справочные данные на поля конспекта. Нужно выделить главное, составить план. Затем следует кратко сформулировать основные положения текста, отметить аргументацию автора. Записи материала следует проводить, четко следуя пунктам плана и выражая мысль своими словами. Цитаты должны быть записаны грамотно, учитывать лаконичность, значимость мысли.

В тексте конспекта желательно приводить не только тезисные положения, но и их доказательства. При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной последовательности, отвечающей логической структуре произведения. Для уточнения и дополнения необходимо оставлять поля. Необходимо указывать библиографическое описание конспектируемого источника.

3. ТЕСТОВЫЕ ВОПРОСЫ ДЛЯ САМОПОДГОТОВКИ

ОПК-2.1.

1. Инструментальные программные средства – это …

1) программы, которые используются в ходе разработки, корректировки или развития других прикладных или системных программ;

2) аппаратное обеспечение;

3) аппаратно-программное обеспечение;

4) алгоритмы, которые используются в ходе разработки, корректировки или развития других прикладных или системных программ;

5) совокупность аппаратного и программного обеспечения информационных систем.

2. Интегрированная среда разработки – это …

1) система программных средств, используемая программистами для разработки программного обеспечения;

2) система программных средств, используемая программистами для разработки аппаратного обеспечения;

3) текстовые редакторы;

4) компиляторы;
5) интерпретаторы.

3. Компилятор – это …
1) программа или техническое средство, выполняющее трансляцию программы, составленной на исходном языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду;
2) программа или техническое средство, выполняющее трансляцию программы, составленной на исходном языке низкого уровня, в эквивалентную программу на высокоуровневом языке;
3) устройство разработки алгоритма решения задачи;
4) устройство отладки программы;
5) программное средство отладки программы.

4. Интерпретатор – это …
1) программа или аппаратное устройство, выполняющее пооператорный анализ, обработку и выполнение исходной программы или запроса;
2) программа или техническое средство, выполняющее трансляцию программы, составленной на исходном языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду;
3) устройство разработки алгоритма решения задачи;
4) устройство отладки программы;
5) программное средство отладки программы.

5. Компоновщик – это …
1) программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль;
2) программа или техническое средство, выполняющее трансляцию программы, составленной на исходном языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду;
3) программа или аппаратное устройство, выполняющее пооператорный анализ, обработку и выполнение исходной программы или запроса;
4) устройство отладки программы;
5) программное средство отладки программы.

6. Синтаксический анализ – это …
1) процесс сопоставления линейной последовательности лексем языка с его формальной грамматикой;
2) программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль;
3) программа или аппаратное устройство, выполняющее пооператорный анализ, обработку и выполнение исходной программы или запроса;
4) программа или техническое средство, выполняющее трансляцию программы, составленной на исходном языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду;
5) система программных средств, используемая программистами для разработки программного обеспечения.

7. Отладчик – это …
1) компьютерная программа, предназначенная для поиска ошибок в других программах, ядрах операционных систем, SQL-запросах и других видах кода;
2) процесс сопоставления линейной последовательности лексем языка с его формальной грамматикой;
3) программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль;
4) программа или аппаратное устройство, выполняющее пооператорный анализ, обработку и выполнение исходной программы или запроса;

5) система программных средств, используемая программистами для разработки программного обеспечения.

8. Профилирование – это …

1) сбор характеристик работы программы, таких как время выполнения отдельных фрагментов, число верно предсказанных условных переходов, число кэш промахов;

2) компьютерная программа, предназначенная для поиска ошибок в других программах, ядрах операционных систем, SQL-запросах и других видах кода;

3) процесс сопоставления линейной последовательности лексем языка с его формальной грамматикой;

4) программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль;

5) программа или аппаратное устройство, выполняющее пооператорный анализ, обработку и выполнение исходной программы или запроса.

9. Генератор документации – это …

1) программа или пакет программ, позволяющая получать документацию, предназначенную для программистов и/или для конечных пользователей системы, по особым образом комментированному исходному коду и по исполняемым модулям;

2) сбор характеристик работы программы, таких как время выполнения отдельных фрагментов, число верно предсказанных условных переходов, число кэш промахов ;

3) компьютерная программа, предназначенная для поиска ошибок в других программах, ядрах операционных систем, SQL-запросах и других видах кода;

4) программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль;

5) программа, которая используется в ходе разработки, корректировки или развития других прикладных или системных программ.

10. Система управления версиями – это …

1) программное обеспечение для облегчения работы с изменяющейся информацией;

2) сбор характеристик работы программы, таких как время выполнения отдельных фрагментов, число верно предсказанных условных переходов, число кэш промахов;

3) компьютерная программа, предназначенная для поиска ошибок в других программах, ядрах операционных систем, SQL-запросах и других видах кода;

4) программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль;

5) программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль.

11. Машинные языки – это …

1) языки программирования, воспринимаемые аппаратной частью компьютера;

2) языки программирования, которые отражают структуру конкретного типа компьютера;

3) языки программирования, не зависящие от архитектуры компьютера, служащие для отражения структуры алгоритма;

4) языки программирования, где имеется возможность описания программы как совокупности процедур, подпрограмм;

5) языки программирования, предназначенные для решения задач определенного класса.

12. Машинно-ориентированные языки – это …

1) языки программирования, которые отражают структуру конкретного типа компьютера;

2) языки программирования, воспринимаемые аппаратной частью компьютера;

3) языки программирования, не зависящие от архитектуры компьютера, служащие для отражения структуры алгоритма;

4) языки программирования, где имеется возможность описания программы как совокупности процедур, подпрограмм;

5) языки программирования, предназначенные для решения задач определенного класса.

13. Алгоритмические языки – это …

1) языки программирования, не зависящие от архитектуры компьютера, служащие для отражения структуры алгоритма;

2) языки программирования, которые отражают структуру конкретного типа компьютера;

3) языки программирования, воспринимаемые аппаратной частью компьютера;

4) языки программирования, предназначенные для решения задач определенного класса;

5) ассемблеры.

14. Процедурно-ориентированные языки – это …

1) языки программирования, где имеется возможность описания программы как совокупности процедур, подпрограмм;

2) языки программирования, не зависящие от архитектуры компьютера, служащие для отражения структуры алгоритма;

3) языки программирования, воспринимаемые аппаратной частью компьютера;

4) языки программирования, предназначенные для решения задач определенного класса;

5) ассемблеры.

15. Проблемно-ориентированные языки – это …

1) языки программирования, предназначенные для решения задач определенного класса;

2) языки программирования, не зависящие от архитектуры компьютера, служащие для отражения структуры алгоритма;

3) языки программирования, воспринимаемые аппаратной частью компьютера;

4) языки программирования, где имеется возможность описания программы как совокупности процедур, подпрограмм;

5)) ассемблеры.

ОПК-2.2.

16. Базовое программные средства включают …

1) операционные системы;

2) языки программирования;

3) программные среды;

4) системы управления базами данных;

5) компьютерные игры.

17. Инструментальные средства CASE – это …

1) специальные программы, которые поддерживают одну или несколько методологий анализа и проектирования информационных систем;

2) текстовые процессоры;

3) электронные таблицы;

4) браузеры;

5) почтовые клиенты.

18. Лексический анализатор …

1) просматривает литеры исходной программы слева направо и строит символы программы – целые числа, идентификаторы, служебные слова;

2) выполняет компиляцию программы;

3) выполняет трансляцию программы;

4) выполняет отладку программы;

5) ищет вирусы.

19. Инструментальная система поддержки проекта – это …

1) открытая система, способная поддерживать разработку программного средства на разных языках программирования после соответствующего ее расширения программными инструментами, ориентированными на выбранный язык;

2) программа для просмотра литер исходной программы слева направо и построения символов программы;

3) специальная программа, которая поддерживают одну или несколько методологий анализа и проектирования информационных систем;

4) язык программирования, предназначенный для решения задач определенного класса;

5) язык программирования, в котором имеется возможность описания программы как совокупности процедур, подпрограмм.

20. Программная среда – это …

1) интегрированная совокупность технических и программных средств, при помощи которых осуществляется разработка программ;

2) открытая система, способная поддерживать разработку программного средства на разных языках программирования после соответствующего ее расширения программными инструментами, ориентированными на выбранный язык;

3) специальные программы, которые поддерживают одну или несколько методологий анализа и проектирования информационных систем;

4) язык программирования, в котором возможность описания программы как совокупности процедур, подпрограмм;

5) язык программирования, не зависящий от архитектуры компьютера, служащий для отражения структуры алгоритма.

ОПК-5.1.

21. Символьный класс \w в регулярных выражениях задает соответствие:

1) цифре, букве латинского алфавита или знаку нижнего подчеркивания;
2) цифре, букве русского алфавита или знаку нижнего подчеркивания;
3) цифре, букве знаку нижнего подчеркивания;
4) цифре, букве русского алфавита или пробельному символу;
5) любому символу, кроме цифр, букв и знака нижнего подчеркивания.

22. Символьный класс \s в регулярных выражениях задает соответствие:

1) пробельному символу;
2) цифре;
3) букве;
4) букве русского алфавита;
5) букве латинского алфавита.

23. Символ повторения * в регулярных выражениях означает соответствие …

1) предыдущему элементу 0 или более раз;
2) предыдущему элементу один или более раз;
3) предыдущему элементу ноль или один раз;
4) предыдущему элементу один раз;
5) предыдущему элементу два раза.

24. Символ повторения + в регулярных выражениях означает соответствие …

1) предыдущему элементу один или более раз;
2) предыдущему элементу 0 или более раз;
3) предыдущему элементу ноль или один раз;
4) предыдущему элементу один раз;
5) предыдущему элементу два раза.

25. Символ повторения ? в регулярных выражениях означает соответствие …

1) предыдущему элементу ноль или один раз;
2) предыдущему элементу один или более раз;
3) предыдущему элементу 0 или более раз;
4) предыдущему элементу один раз;
5) предыдущему элементу два раза.

ОПК-5.2.

26. Класс Thread определяет …

1) ряд методов и свойств, которые позволяют управлять потоком и получать информацию о нем;
2) ряд методов и свойств, которые позволяют управлять подпрограммой и получать информацию о ней;
3) ряд методов и свойств, которые позволяют управлять обработчиком прерываний и получать информацию о нем;
4) ряд методов и свойств, которые позволяют управлять событием и получать информацию о нем;
5) ряд методов и свойств, которые позволяют управлять потоком и получать информацию о нем.

27. Статическое свойство CurrentContext класса Thread …
1) позволяет получить контекст, в котором выполняется поток;
2) возвращает ссылку на выполняемый поток;
3) указывает, является ли поток фоновым;
4) содержит имя потока;
5) хранит приоритет потока.

28. Статическое свойство CurrentThread класса Thread …
1) возвращает ссылку на выполняемый поток;
2) позволяет получить контекст, в котором выполняется поток;
3) указывает, является ли поток фоновым;
4) содержит имя потока;
5) хранит приоритет потока.

29. Свойство IsAlive класса Thread …
1) указывает, работает ли поток в текущий момент;
2) указывает, является ли поток фоновым;
3) возвращает ссылку на выполняемый поток;
4) содержит имя потока;
5) хранит приоритет потока.

30. Свойство IsBackground класса Thread …
1) указывает, является ли поток фоновым;
2) указывает, работает ли поток в текущий момент;
3) возвращает ссылку на выполняемый поток;
4) содержит имя потока;
5) хранит приоритет потока.

ОПК-7.1.

31. Конструкция Parallel.For является параллельным аналогом …
1) цикла for;
2) цикла while;
3) цикла do while;
4) цикла repeat until;
5) цикла foreach.

32. Конструкция Parallel.ForEach является параллельным аналогом …
1) цикла foreach;
2) цикла for;
3) цикла while;
4) цикла do while;
5) цикла repeat until.

33. Статический метод Parallel.Invoke позволяет …
1) распараллелить исполнение блоков операторов;
2) распараллелить обработку прерываний;
3) выполнить цикл for;
4) выполнить цикл while;
5) выполнить цикл do while.

34. Подход Work Sharing – это …

1) централизованное планирование, при котором планировщик контролирует единый пул задач в системе и назначает задачи процессорам;

2) децентрализованное планирование, при котором единый планировщик отсутствует, а процессоры сами выбирают какие задачи им исполнят;

3) параллельное выполнение цикла for;

4) параллельное выполнение цикла while;

5) параллельное выполнение цикла do while.

35. Подход Work Stealing – это …

1) децентрализованное планирование, при котором единый планировщик отсутствует, а процессоры сами выбирают какие задачи им исполнят;

2) децентрализованное планирование, при котором единый планировщик отсутствует, а процессоры сами выбирают какие задачи им исполнят;

3) параллельное выполнение цикла for;

4) параллельное выполнение цикла while;

5) параллельное выполнение цикла do while.

ОПК-7.2.

36. Мультиагентная система состоит из …

1) множества организационных единиц, в котором выделяются: подмножество агентов, манипулирующих подмножеством объектов;

2) множества задач, решаемых агентами в рамках выполнения общей задачи;

3) среды, т.е. некоторого пространства, в котором существуют агенты и объекты;

4) множества отношений между агентами;

5) множества формул.

37. Адаптивный планировщик мультиагентной системы …

1) обрабатывает поток входящих событий;

2) обрабатывает поток исходящих событий;

3) вызывает подпрограммы;

4) вызывает методы;

5) обрабатывает прерывания.

38. Конструктор сцены мультиагентной системы …

1) позволяет редактировать начальную конфигурацию сети и определить все параметры ресурсов компании;

2) позволяет редактировать конечную конфигурацию сети и определить все параметры ресурсов компании;

3) обрабатывает поток входящих событий;

4) обрабатывает поток исходящих событий;

5) вызывает подпрограммы.

39. Редактор онтологии мультиагентной системы позволяет …

1) ввести и изменить общую онтологию компании, описывающую модель знаний предметной области;

2) редактировать начальную конфигурацию сети и определить все параметры ресурсов компании;

3) обрабатывать поток входящих событий;

4) обрабатывать поток исходящих событий;

5) вызывать подпрограммы.

40. Эволюционный дизайн мультиагентной системы – это …

1) модуль, вырабатывающий предложения по улучшению конфигурации сети в части увеличения или уменьшения определенного числа ресурсов, изменению географии ресурсов;

2) модуль ввода и изменения общей онтологии компании, описывающей модель знаний предметной области;

3) модуль редактирования начальной конфигурации сети и определения всех параметров ресурсов компании;

4) модуль обработки потока входящих событий;

5) модуль вызова подпрограмм.

4. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

а) основная литература

1. Вичугова А.А. Инструментальные средства информационных систем [Электронный ресурс] : учебное пособие / А.А. Вичугова. — Электрон. текстовые данные. — Томск: Том-ский политехнический университет, 2015. — 136 c. — 978-5-4387-0574-1. — Режим доступа: http://www.iprbookshop.ru/55190.html

2. Инструментальные средства математического моделирования [Электронный ресурс] : учебное пособие / А.А. Золотарев [и др.]. — Электрон. текстовые данные. — Ростов-на-Дону: Южный федеральный университет, 2011. — 90 c. — 978-5-9275-0887-7. — Режим до-ступа: http://www.iprbookshop.ru/46963.html

3. Михеев А.Г. Процессное управление на свободном программном обеспечении [Электронный ресурс] / А.Г. Михеев. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 230 c. — 2227-8397. — Ре-жим доступа: http://www.iprbookshop.ru/39562.html

4. Кондратьев В.К. Введение в операционные системы [Электронный ресурс] : учебное пособие / В.К. Кондратьев. — Электрон. текстовые данные. — М. : Евразийский открытый институт, Московский государственный университет экономики, статистики и информа-тики, 2007. — 232 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/10637.html

5. Федотова Д.Э. ОС Windows & ОС Linux [Электронный ресурс] : лабораторные рабо-ты по курсу «Операционные системы» / Д.Э. Федотова. — Электрон. текстовые данные. — М. : Российский новый университет, 2009. — 224 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/21256.html

6. Командная строка UNIX [Электронный ресурс] : лабораторный практикум по дисци-плине «Операционные системы» / . — Электрон. текстовые данные. — М. : Московский гос-ударственный строительный университет, ЭБС АСВ, 2013. — 44 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/23729.html

7. Мельников С.В. Perl для профессиональных программистов. Регулярные выражения [Электронный ресурс] : учебное пособие / С.В. Мельников. — Электрон. текстовые данные. — Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Ву-зовское образование, 2017. — 200 c. — 978-5-4487-0034-7. — Режим доступа: http://www.iprbookshop.ru/67400.html

б) дополнительная литература

1. Практикум по администрированию программного обеспечения [Электронный ре-сурс] : лабораторный практикум / . — Электрон. текстовые данные. — Ставрополь: Северо-Кавказский федеральный университет, 2017. — 85 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/75589.html

2. Назаров С.В. Современные операционные системы [Электронный ресурс] / С.В. Назаров, А.И. Широков. — Электрон. текстовые данные. — М. : Интернет-Университет Ин-формационных Технологий (ИНТУИТ), 2016. — 351 c. — 978-5-9963-0416-5. — Режим до-ступа: http://www.iprbookshop.ru/52176.html

5. КОНТРОЛЬНЫЕ ВОПРОСЫ ДЛЯ ПОДГОТОВКИ К ЭКЗАМЕНУ

ОПК-2.1.

1. Понятие, содержание, назначение инструментальных средств.

2. Виды классификаций инструментальных средств.

3. История и перспективы развития инструментальных средств.

ОПК-2.2.

4. Современные CASE-средства как инструмент многочисленных технологий проектирования информационных систем.

5. Классификация CASE -средств.

6. Характеристики CASE-средств.

7. Организация документооборота и эффективное представление структур и данных документов.

ОПК-5.1.

8. Диалекты регулярных выражений: Perl, PRCE (Perl-Compatible Regular Expressions), .NET, Java, JavaScript, Python, Ruby.

9. Литералы регулярных выражений в исходных текстах.

10. Импортирование библиотеки регулярных выражений.

11. Создание объектов регулярных выражений.

12. Установка параметров регулярных выражений.

13. Проверка возможности совпадения в пределах испытуемой строки.

14. Проверка совпадения со всей испытуемой строкой.

15. Извлечение текста совпадения.

16. Определение позиции и длины совпадения.

17. Извлечение части совпавшего текста.

18. Извлечение списка всех совпадений.

19. Обход всех совпадений в цикле.

20. Проверка полученных совпадений в программном коде.

21. Поиск совпадения внутри другого совпадения.

22. Замена всех совпадений.

23. Замена совпадений с повторным использованием частей совпадений.

24. Разбиение строки.

25. Построчный поиск.

ОПК-5.2.

26. Понятие параллельных вычислений.

27. Требования, приводящие к достижению параллелизма.

28. Режимы выполнения независимых частей программы.

29. Примеры параллельных вычислительных систем (суперкомпьютеры, кластеры, персональные мини-кластеры).

30. Классификация вычислительных систем (по способам взаимодействия потоков выполняемых команд и обрабатываемых данных).

31. Характеристика системных платформ для построения кластеров: Microsoft Compute Cluster Server.

32. Показатели эффективности параллельного алгоритма: ускорение, эффективность, стоимость вычислений.

33. Оценка максимально достижимого параллелизма.

34. Анализ масштабируемости параллельных вычислений.

35. Моделирование параллельных программ.

36. Методика разработки параллельных алгоритмов.

ОПК-7.1.

37. Библиотека Parallel Extensions to the .NET Framework.

38. Конструкция Parallel.For.

39. Планирование исполнения процессов: work sharing – централизованное планирование, work stealing – децентрализованное планирование.

40. Конструкция Parallel.Invoke.

41. Программирование с использованием Task Parallel Library (TPL).

42. Класс System.Threading.Tasks.Future.

43. Координирующие структуры данных.

44. PLINQ (Parallel Language-Integrated Query) – параллельный интегрированный язык запросов.

45. Обработка исключений с использованием библиотеки Parallel FX.

46. Реализация конструкций ContinueWhenAll и ContinueWhenAny.

47. Асинхронное выполнение последовательности задач.

48. Ожидание завершения множества задач.

49. Реализация конструкции ParallelWhileNotEmpty.

50. Оценка производительности памяти с помощью теста Random Access.

51. Высокоуровневый язык параллельного программирования MC#.

52. Async- и movable-методы.

53. Каналы и обработчики канальных сообщений.

54. Синхронизация в языке MC#.

55. Примеры программирования на языке MC#.

ОПК-7.2.

56. Стандарты разработки интеллектуальных информационных систем.

57. Инструментальные средства разработки интеллектуальных и агентно-ориентированных приложений.