**1 уровень**

**Инструментальные средства разработки программ**

1. **Для создания операции в программе Rational Rose необходимо:**

Щёлкнуть правой кнопкой мыши по классу в окне диаграмм

Щёлкнуть правой кнопкой мыши по классу в окне инструментов

Щёлкнуть правой левой мыши по классу в окне инструментов

Ввести имя нового атрибута

Щёлкнуть левой кнопкой мыши по классу в окне браузера

1. **Типы стрелок, изображающие связи в IDEF3:**

Старшая ( Precedenc)

Асинхронные (Asynchronous )

Безусловные ( Unconditional )

Объекты ссылки ( Referent )

Туннель (Arrow Tunnel)

1. **Последовательность генерации кода в программе Rational Rose:**

выбрать команду меню Tools – C++ - Code Generation

выбрать актеров, прецедентов и пакеты

выбрать команду меню File – C++-Code Generation

выбрать команду меню Add Ins – C++-Code Generation

выбрать команду меню File – C++

1. **Ядро знаний SWEBOK - это:**

Нормативный документ, разработанный IEEE

Документ, устанавливающий правовые отношения между заказчиком и разработчиком ПО

ГОСТ на разработку информационных систем

Документ, устанавливающий методику тестирования и испытания ПО

Документ, устанавливающий методику тестирования

1. **Символы ERD, соответствует отношениями:**

ограниченное отношение

композиционное отношение

правильное отношение

медианное отношение

неправильное отношение

1. **CASE – средства классифицируется по следующим признаками:**

по доступным платформам

по числу процессов

по доступным ресурсам

по числам использованной сущности

по ресурсам

1. **Концептуальная модель UML включает в себя составные части:**

правила, определяющие сочетание блоков между собой

общие механизмы модели

механизмы расширения

внешние сущности

правила определяющие классы

1. **Отладка - это …**

локализация и устранение уже найденных ошибок

корректировка и модификация программы

проверка правильности работы программы

проверка соответствия алгоритма решаемой задачи

тестирование программы

1. **Какие ошибки выявляются при компиляции программы?**

синтаксические

ошибки в данных

семантические

статические

динамические

1. **Когда имеет место отладка?**

если программа работает неправильно

если результаты работы программы не удовлетворяют пользователя

если программа морально устарела

по мере необходимости

в любых ситуациях

1. **Инструмент сред программирования, редактор - …**

создает и изменяет исходные файлы, содержащие программу

формирует исполняемый файл

копирует исполняемый файл с диска в память

инициализирует компьютер перед выполнением программы

переводит программы с языка программирования на язык машинных кодов

1. **С помощью какого инструмента сред программирования можно создавать и изменять исходные файлы, содержащие программу?**

редактор

библиотекарь

компоновщик

загрузчик

модулятор

1. **Инструмент сред программирования, транслятор - …**

переводит программы с языка программирования на язык машинных кодов

поддерживает совокупность объектных файлов с подпрограммами, типами данных

формирует исполняемый файл

копирует исполняемый файл с диска в память

инициализирует компьютер перед выполнением программы

1. **Какой инструмент сред программирования переводит программы с языка программирования на язык машинных кодов?**

транслятор

библиотекарь

компоновщик

загрузчик

модулятор

1. **Инструмент сред программирования, библиотекарь - …**

поддерживает совокупность объектных файлов с подпрограммами, типами данных

копирует исполняемый файл с диска в память

инициализирует компьютер перед выполнением программы

переводит программы с языка программирования на язык машинных кодов

создает и изменяет исходные файлы, содержащие программу

1. **Какой инструмент сред программирования поддерживает совокупность объектных файлов с подпрограммами, типами данных?**

библиотекарь

транслятор

компоновщик

загрузчик

модулятор

1. **Инструмент сред программирования, компоновщик - …**

формирует исполняемый файл

инициализирует компьютер перед выполнением программы

поддерживает совокупность объектных файлов с подпрограммами, типами данных

переводит программы с языка программирования на язык машинных кодов

создает и изменяет исходные файлы, содержащие программу

1. **Какой инструмент сред программирования формирует исполняемый файл?**

компоновщик

транслятор

библиотекарь

загрузчик

модулятор

1. **Инструмент сред программирования, загрузчик - …**

копирует исполняемый файл с диска в память и инициализирует компьютер перед выполнением программы

поддерживает совокупность объектных файлов с подпрограммами

поддерживает совокупность объектных файлов с типами данных

переводит программы с языка программирования на язык машинных кодов

создает и изменяет исходные файлы, содержащие программу

1. **Типы диаграмм, используемые в UML:**

диаграмма прецедентов, диаграмма последовательности, диаграмма кооперации

диаграмма состояний, диаграмма компонентов, диаграмма развертывания

диаграммы интерфейсов, диаграмма точки зрения, диаграммы процессов

диаграмма активных классов, диаграмма активных компонентов

диаграмма процессов, диаграмма актеров

1. **Общие механизмы языка UML:**

спецификаций

механизмы расширения

пиктограммы и обозначения

сигнатуры

обозначения

1. **Тип основных элементов, использующихся для построения концептуальной модели UML:**

базовые строительные блоки

общие механизмы языка

правила определяющие модели

общие механизмы модели

дополнительные строительные блоки

1. **Разновидности структурных сущностей:**

класс

кооперация

взаимодействие

автомат

состояние

1. **Диаграмма классов показывает:**

классы

интерфейсы

компоненты

узлы

кооперации

1. **Хорошо сформулированные модели в языке UML:**

управляется прецедентами использования

является итеративным и инкрементным

основана на классах

управляется объектами системы

является короткими

1. **Классы описывают собой описание совокупности объектов с общими:**

атрибутами

отношениями и семантикой

интерфейсами

классами

пакетами

1. **Атрибут – это:**

именованное свойство класса

абстракция данных объекта или его состояния

именованное свойство операции

свойство моделируемой операции

абстракция данных операции или его состояния

1. **Моделирование словаря системы включает этапы:**

определение разработчиков и пользователей для описания задачи и его

решения

разработать атрибуты и операции, необходимые для выполнения классами своих обязанностей

определение классов и атрибутов

определение прецедентов и операции

выявление операции и обязанностей

1. **Языки обозначений, не входящие в язык UML:**

Блокирования

Графики

Моделирования

Конструирования

Визуализации

1. **Основные элементы пакетов состоят из подпакетов:**

Механизмов расширения и типов данных

Служебных пакетов

Механизмов поиска

Вспомогательных механизмов

Элементов оболочки

1. **Отчет Model Report, сгенерированный в среде BPwin, включает в себя:**

имя модели, точку зрения, область

полный список объектов модели (работ, стрелок с указанием их типа и др.) и свойства, определяемых пользователем

информацию из словаря стрелок, информацию о работе источнике, работе назначении стрелки

1. **Программное обеспечение делится на классы:**

Системное ПО, прикладное ПО и инструментальные средства разработки программ

Системное ПО, прикладное ПО и системы программирования

Системное ПО, прикладное ПО и инструментальное ПО

Техническая документация ПО

Экспертные системы

1. **Агрегирование – это:**

ассоциация между двумя классами

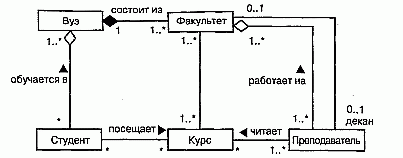
связь типа целое

связь между разными сущностями

связь типа целое/целое

связь между элементами

1. **Типы отношений, использованные на рисунке:**



композиция

5-6

1-5

обобщение

зависимость

1. **Механизмы UML позволяют:**

создавать новые свойства

создавать новые блоки

определять новую семантику

создавать новые классы

создавать новые операции

1. **Стереотипы UML позволяют:**

расширить словарь UML

вводить символы

добавлять в модель визуальные образы

расширить диаграммы UML

вводить новые элементы

1. **Ограничения UML позволяют:**

изменять существующие правила

расширить UML

создавать новые правила

изменять диаграмму

расширить диаграмму UML

1. **Диаграммы классов используются в следующих целях:**

для моделирования словаря системы

для моделирования

для моделирования логической схемы

для моделирования вариантов использования

для моделирования прецедентов

1. **Определенные в языке UML уровни видимости:**

public (открытый)

закрытый

общий

instance (экземпляр)

classifier (классификатор)

1. **Треугольник успеха для успешного проекта:**

нотация

функция

элемент

знания

тактика

1. **Роль актеров в системе:**

только снабжать информацией

смотреть информацию

снабжать информацией и получать информацию

только снабжать работой

только получать работу

1. **Описание актера в программе Rational Rose осуществляется при выполнении следующих действий:**

открыть окно описания на экране, выбрав команду меню View Documentation

из списка браузера выберите актера, щелкнув по нему мышью

установите курсор в окне описания и введите текст описания актера

закрыть окно описания на экране, выбрав команду меню View Documentation

из списка браузера выберите прецедента, щелкнув по нему мышью

1. **Для добавления описания прецеденту в программе Rational Rose:**

в списке браузер выберите прецедент, щелкнув по нему мышью

установите курсор в окне описания

откройте его с помощью команды меню View Documentation

установите курсор в окне описания и наберите краткое описание класса

установите курсор в окне описания и наберите краткое описание актера

1. **Диаграмма прецедентов – это:**

диаграмма, показывающая все прецеденты для определенного актера

диаграмма, показывающая все прецеденты

диаграмма, показывающая определенный прецедент и все его отношения

диаграмма, показывающая все актеры для определенного прецедента

диаграмма, показывающая все актеры реализованные на данной итерации

1. **Диаграммы взаимодействий содержат:**

объекты, изображаемые в виде прямоугольников

связи между объектами

сообщения в виде текста и стрелки

объекты, изображаемые в виде треугольников

объекты, изображаемые в виде линии

1. **Создать операцию в программе Rational Rose:**

щелкните правой кнопки мыши по классу в окне браузера

щелкните правой кнопки мыши по классу в окне

ввести имя новой операции

щелкните левой кнопки мыши по классу в окне браузера

щелкните левой кнопки мыши по классу в окне диаграмм

1. **Программное обеспечение делится на классы:**

системное ПО, прикладное ПО и инструментальные средства разработки программ

системное ПО, прикладное ПО и системы программирования

системное ПО, прикладное ПО и инструментальное ПО

системное ПО и прикладное ПО

операционные системы, прикладное ПО, утилиты и драйверы

1. **Аппаратные инструментальные средства разработки ПО – это:**

микропроцессор и подключаемые (внешние) устройства

устройства вычислительной системы

периферийные устройства, микропроцессор вычислительного комплекса

система для разработки новых программ на конкретном языке программирования

средства создания и редактирования текстов программ

1. **Транслятор – это:**

программа, выполняющая перевод программы с одного языка программирования на другой

техническое устройство, которое выполняет перевод программы с одного языка программирования на другой

одно из основных средств автоматизации программирования для преобразования программы, написанный на машинно независимом языке, в программу на машинном языке конкретной ЭВМ

комплекс программ мультимедийных технологий

программа переводчик с одного иностранного языка на другой

1. **Компоновщик – это:**

редактор связей

прикладная программа

программа сборки загрузочного модуля

программа для компоновки и оформления тестовых документов

комплекс программ, для создания и ведения баз данных

1. **К этапам развития технологии разработки программного обеспечения относятся:**

"стихийное" программирование

структурный метод

компонентный подход и CASE технологии

программирование на алгоритмических языках высокого уровня

программирование на языках низкого уровня

1. **"Стихийное" программирование – это:**

первый этап в истории развития технологии разработки программного обеспечения, когда программирование фактически было искусством

период в истории разработки программного обеспечения, когда программа создавалась одним программистом

характеризуется тем, что типичная программа этого периода состояла из основной программы, области глобальных данных

разработка программного обеспечения без предварительного составления плана графики работ

разработка программ с использованием различных языков программирования низкого и высокого уровня

1. **Структурный подход к программированию – это:**

совокупность рекомендуемых технологических приемов, охватывающих выполнение всех этапов разработки программного обеспечения

подход, в основе которого лежит композиция

подход, требующий представления задачи в виде иерархии

создание программного обеспечения на основе структурной схемы решаемой задачи

подход, требующий разработки структурной схемы алгоритма и программы решения задачи

1. **Объектный подход к программированию – это:**

технология создания сложного программного обеспечения, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного типа (класса)

технология создания сложного программного обеспечения

технология создания сложного программного обеспечения, в основе которой лежат новые способы организации программ

технология создания сложного программного обеспечения, основанная на представлении задачи исследования как объекта

технология создания сложного программного обеспечения, предназначенного для автоматизации технологических объектов

1. **Компонентный подход:**

предполагает построение программного обеспечения из отдельных компонентов физически отдельно существующих частей программного обеспечения

предполагает взаимодействие между компонентами

позволяет собрать объекты компоненты в динамически вызываемые библиотеки

позволяет рассматривать объект исследования, как структуру, состоящую из отдельных компонент

способ написания исходного кода программного обеспечения

1. **Управление требованиями – это:**

процесс систематического выявления, организации и документирования требований к сложной системе

задача, состоящая в том, чтобы понимать проблемы

процесс, в ходе которого вырабатывается и обеспечивается соглашение между заказчиком и выполняющей проект группой

задача выявления изначальных проблем заказчика и создание системы, удовлетворяющей этим требованиям

выявление требований заказчика и управление ими

1. **К методам выявления требований относятся:**

интервьюирование и анкетирование, мозговой штурм и отбор идей

совещания

беседы

анализ научной и технической литературы, посвященной вопросам разработки программного обеспечения

личные встречи и беседы со всеми сотрудниками предприятия

анализ технической документации и на основе нее разработка требований к системе

1. **Требования к разрабатываемой системе должны включать:**

совокупность условий, при которых предполагается эксплуатировать будущую систему

описание выполняемых системой функций

ограничения (директивные сроки завершения отдельных этапов, имеющиеся ресурсы, организационные процедуры и мероприятия, обеспечивающие защиту информации)

разработку программного обеспечения и выработка требований к изменению работы системы заказчика

построение программного обеспечения из отдельных компонентов физически отдельно существующих частей программного обеспечения

1. **Типы средств, иллюстрирующие цели моделирования системы**

функции, которые система должна выполнять

отношения между элементами

зависящее от времени поведение системы (аспекты реального времени)

способы отладки и тестирования программного обеспечения

создание программного обеспечения на основе структурной схемы исследуемого объекта или процесса

выявление требований заказчика и управление ими

1. **Метамодели ARIS в отличие от метамоделей UML обладают следующими особенностями:**

ARIS повышают структурирование, т.к. они проектируются в соответствии с моделями и жизненным циклом ARIS

ARIS расширяют возможности моделирования

ARIS ограничивают возможности моделирования и диапазон бизнес – ориентированных приложений

ARIS ориентирует процесс моделирования на решение узконаправленных задач

ARIS имеют такие же характеристики, как и метамодели UML

1. **Преимущества объектно-ориентированного подхода:**

возможность многократного использования

восприимчивость к изменениям

реалистичное моделирование

быстрота написания программного кода

статичность конфигурации системы

1. **Ядро знаний SWEBOK – это:**

Нормативный документ, разработанный IEEE

ГОСТ на разработку и комплектацию

ГОСТ на разработку ПО

Документ, устанавливающий правовые отношения между заказчиком и разработчиком ПО

Документ, устанавливающий методику тестирования и испытания ПО

1. **Требования – это:**

некоторое свойство программного обеспечения, необходимое пользователю для решения проблемы при достижении поставленной цели

возможность, которую должна обеспечивать система

некоторое свойство программного обеспечения

документ, регулирующий отношения между заказчиком информационной системы и проектировщиком

оформленное заказчиком в виде документа задание на проектирование программного обеспечения

1. Для различных представлений проектируемой системы используют типы моделей:
   * + - 1. Модель классов
         2. Модель объектов
         3. Модель размещения
         4. Динамическая модель

F) Модель декомпозиции

**62. Уровни организационного управления при планировании разработки системы:**

Тактический

Аналитический

оперативный

дополнительный

системный

1. **Допустимые объявления свойств, используемые совместно с атрибутами:**

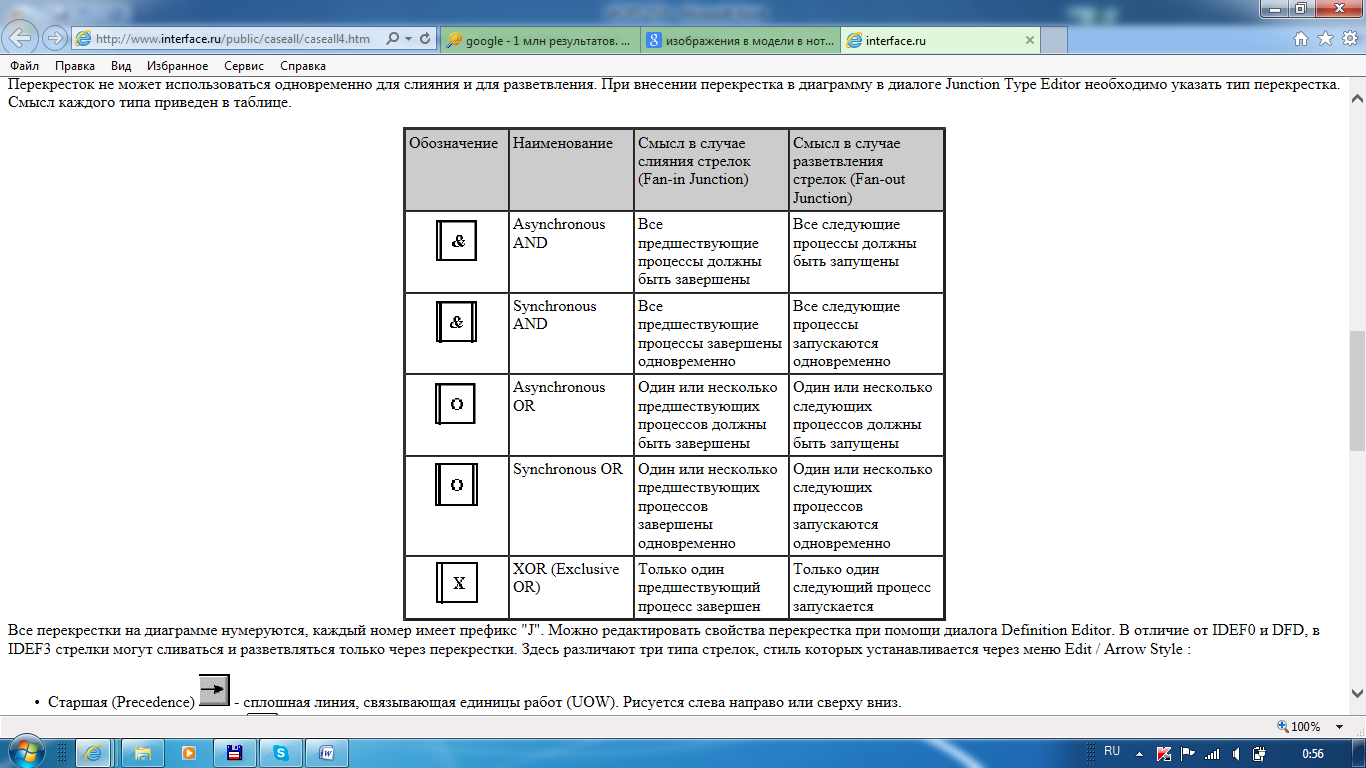
addOnly (только добавляемый)

open

active

classifier (классификатор)

instance (экземпляр)

1. **Смысл изображений  в модели в нотации IDEF3:**

Все следующие процессы запускаются одновременно

Только один предшествующий процесс завершен

Все следующие процессы завершены

Только три предшествующих процессов завершены

Только несколько предшествующих процессов завершены

1. **Основные пакеты метамодели языка UML:**

Общие механизмы

Служебные механизмы

Общие элементы

Дополнительные элементы

Элементы взаимодействия

1. **Элементы поведения состоят из подпакетов:**

Варианты использования и автоматы

Размещение

Взаимодействия

Последовательности

Слияния

1. **Основные элементы пакетов состоят из подпакетов:**

Механизмов расширения и типов данных

Служебных элементов

Механизмов поиска

Вспомогательных механизмов, Механизмов дополнений

Элементов оболочки

1. **Элементы не входящие в список стереотипов связей элементы**

external

parameter

self

association

local, global

1. **Квантор видимости Visibility может принимать одно из следующих значений:**

**“-”**

“\*”

“&”

“@”

“%”

1. **Квантор видимости Visibility может принимать одно из следующих значений:**

**“#”**

“\*”

“&”

“@”

“\*”

1. **В языке UML определены следующие уровня видимости:**

public (общий) — любой внешний класс, который "видит" данный, может пользоваться его общими свойствами.

instance (экземпляр) — у каждого экземпляра класса есть собственное значение данного свойства;

classifier (классификатор) — все экземпляры совместно используют общее значение данного свойства (выделяется на диаграммах подчеркиванием).

служебный (Abstract) - не содержащие ни одного экземпляра

содержащие ровно один экземпляр ( Singleton )

1. **В языке UML определены следующие уровня видимости:**

protected (защищенный) — только любой потомок данного класса может пользоваться его защищенными свойствами.

instance (экземпляр) — у каждого экземпляра класса есть собственное значение данного свойства;

classifier (классификатор) — все экземпляры совместно используют общее значение данного свойства (выделяется на диаграммах подчеркиванием).

служебный (Abstract) - не содержащие ни одного экземпляра

use case

1. **В языке UML определены следующие уровня видимости:**

private (закрытый) — только данный класс может пользоваться этими свойствами.

instance (экземпляр) — у каждого экземпляра класса есть собственное значение данного свойства;

classifier (классификатор) — все экземпляры совместно используют общее значение данного свойства (выделяется на диаграммах подчеркиванием).

служебный (Abstract) - не содержащие ни одного экземпляра

use case

1. **Дополнительные принципы объектного подхода:**

Сохраняемость или устойчивость

Перпендикулярность

Несохраняемость или неустойчивость

Динамичность

Реализация

1. **Создание пакетов в представлении компонентов в программе Rational Rose:**

В контекстно-зависимом меню выбрать команду New - Package

В список браузера будет добавлена Диаграмма New Diagram, дать ей название

Щёлкните в окне браузера по значку “+” слева от имени пакета

Щёлкните в окне браузера по значку “-” слева от имени пакета

Щёлкните в окне браузера по значку “\*” слева от имени пакета

1. **Унифицированный язык моделирования (UML):**

Не зависит от используемой методологии разработки

Не может поддерживать любой объектно-ориентированный язык программирования

Зависит от языков программирования

Зависит от объектно-ориентированных (ОО) языков программирования

Зависит от используемой методологии разработки проекта

1. **Типы D – требований:**

Нефункциональные требования

Производительные требования

Программные требования

Обратные требования

Прямые

1. **Управленческие принципы создания информационных систем включают:**

Системность

Наглядность

Простость

Доступность

Заменяемость

1. **Диаграмма Use Case отображает:**

Последовательность связанных транзакций, представляемых объектами или системой

Последовательность связанных транзакций, представляемых компонентами

Получение некоторой информации узлами

Образцы поведения или классов системы

Получение некоторой информации объектами

1. **Используемые пиктограммы для видимости изображаются перед именем соответствующего атрибута или операции имеют следующий смысл:**

Общий, открытый (Public) – соответствует знак “+”)

Закрытый (Private – соответствует знак “–”)

Слияние – соответствует знак “%”

Продолжение (continu- – соответствует знак “&”)

Конец (En- – соответствует знак “@”)

1. **Перечень меток имени событий:**

exit

all

begin

stop

and

1. **Типы отношения между прецедентами:**

Communicate

InterFace

Node

Note

Aggregation

1. **Модели ARIS:**

Модели процессов управления

Модели данных

Модели технологий

Модели связей

Модели аппаратных средств

1. **К организационным областям знаний SWEBOK относятся:**

Процесс инженерии программных средств, методы и средства программной инженерии

Инженерия требований, Управление конфигураций, управление проектами, Сопровождение ПО

Тестирование ПО

Требования

Цели

1. **Инструменты разработки программ выбирается на основе:**

Объема реализации и времени исполнения (реакции), надежности, устойчивости, безопасности

Мощности множества элементов разработки

Системного подхода к анализу, проектированию и реализации ПО

Упрощения связей, комментируемости функций и данных

Функциональной декомпозиций, пространственный и временной группировки информации (модульность)

1. **Виды моделей жизненного цикла ПО:**

каскадная модель, спиральная модель

имитационная модель

индивидуальная модель, лестничная модель

общая модель

разветвляющаяся модель

1. **Виды моделей жизненного цикла ПО:**

спиральная модель

имитационная модель

лестничная модель

индивидуальная модель

разветвляющаяся модель

1. **Диаграммы деятельности используется при моделировании:**

бизнес-процессов

технологических процессов

технологических потоков, параллельных действий

поведения, функций

последовательных действий

1. **Создать атрибут в программе Rational Rose для диаграмм классов:**

В контекстно-зависимом меню выбрать команду New-Attribute

Щелкнуть в окне браузере по значку слева от имени класса

В списке браузер выбрать актер, щелкнув по нему мышью

Ввести имя новой операции, Ввести имя новой диаграммы

Установить курсор в окне диаграмм и набрать атрибут

1. **Унифицированный язык моделирования (UML):**

не зависит от объектно-ориентированных (ОО) языков

не может поддерживать любой объектно-ориентированный язык программирования

зависит от объектно-ориентированных (ОО) языков

зависит от языка программирования

зависит от используемой методологии разработки проекта

1. **Язык UML - это:**

Унифицированный язык моделирования-Unified Modeling Language

Язык логического программирования

Язык управления базами данных

Язык для разработки систем искусственного интеллекта, Язык моделей

Язык управления

1. **Типы D - требований**

функциональные требования

программные требования

надежность, интерфейсные требования

цели и задачи

требования

1. **Программный продукт Model Mart позволяет решать следующие задачи**

управление доступом

создание библиотек решений

управление классами

создание старых классов

создание новых классов, совместное моделирование

1. **Сущности языка в UML подразделяются на:**

структурные и аннотационные

графические и сложные

пользовательские

функциональные, группирующие

элементарные, административные

1. **ERwin позволяет создавать модель, имеющую:**

физический уровень

абстрактный

абстрактный уровень

контекстный уровень

концептуальный уровень

1. **На логическом уровне процесса построение информационной модели в Erwin:**

задание первичных и альтернативных ключей; определение атрибутов сущностей

составление спецификации и документации на проектируемую систему, диаграмма классов

построение диаграммы взаимодействий, диаграммы размещения

построение диаграммы классов, диаграммы дерева узлов

составление линейной модели

1. **Для отображения всех атрибутов и операций в программе Rational Rose выполняются следующие действия:**

Снова вызвать контекстно-зависимое меню и выбрать команду Options-Show All Operations

Ввести имя новой операции и нового атрибута

В контекстно-зависимом меню выбрать команду Options-Show All Classes

Снова вызвать контекстно-зависимое меню и выбрать команду Options-Show All Component’s

В списке браузер выбрать класс, щелкнув по нему мышью

1. **Представление вариантов использования:**

Действующие лица, варианты использования, пакеты

Диаграмма компонентов

Компоненты, пакеты

Классы, диаграммы размещения

Классы, диаграммы классов

**Опорные точки зрения конечных пользователей системы программного обеспечения можно трактовать как:**

источник информации о системных данных

структуру представлений

структуру требований

источник событий

структуру событий

получателей требований

источник сценариев

1. **Логическое представление - это:**

Документация по вариантам использования, диаграммы вариантов использования

Диаграммы компонентов

Диаграммы взаимодействия

Компоненты, пакеты

Иерархическая диаграмма, действующие лица, варианты использования, пакеты, получателей системных сервисов

1. **Технология программирования представляет собой набор технологических инструкций, включающих:**

указание последовательности выполнения технологических операций

описания самих операций

конкретные этапы разработки

решение отдельных задач этапов

совокупность методов, используемых на разных этапах разработки, требования

1. **К основным процессам ЖЦ ПО относятся**

приобретение

управление

усовершенствование

обучение

документирование

1. **К основным процессам ЖЦ ПО относятся**

разработка

управление

усовершенствование

обучение

документирование

1. **К основным процессам ЖЦ ПО относятся**

эксплуатация

сопровождение

управление

усовершенствование

обучение, аудит

**К организационным процессам ЖЦ ПО относятся**

управление

документирование

аудит

верификация

Эксплуатация

1. **К организационным процессам ЖЦ ПО относятся**

создание инфраструктуры

документирование

аудит

верификация

эксплуатация

1. **К организационным процессам ЖЦ ПО относятся**

обучение

документирование

аудит

верификация

эксплуатация

1. **К вспомогательным процессам ЖЦ ПО относятся**

документирование

эксплуатация

сопровождение

приобретение

обучение

1. **К вспомогательным процессам ЖЦ ПО относятся**

Управление конфигурацией

Сопровождение

Приобретение

Обучение

Усовершенствование

1. **К вспомогательным процессам ЖЦ ПО относятся**

Аттестация

Эксплуатация

Сопровождение

Приобретение

Обучение

1. **К вспомогательным процессам ЖЦ ПО относятся**

Совместная работа

Эксплуатация

Сопровождение

Приобретение

Обучение

1. **Процесс проектирования сложного программного обеспечения обычно включает:**

проектирование общей структуры - определение основных компонентов и их взаимосвязей

общую логическую модель проектируемого программного обеспечения

функции, которые должно выполнять разрабатываемое программное обеспечение

требования, определяющие особенности его функционирования

совокупность взаимосвязанных действий, преобразующих некоторые входные данные в выходные

1. **Принято различать два аспекта проектирования:**

логическое и физ. проектирование

аналитическое и струк. проектирование

функциональное моделирование

структурный анализ

имитационное проектирование и управление

1. **Достоинствами каскадной схемы разработки программного обеспечения является**

получение в конце каждой стадии законченного набора проектной документации

любой уровень и внести необходимые изменения

сокращение времени до появления первых версий программного продукта

уменьшение вероятности морального устаревания системы за время разработки

совокупность взаимосвязанных действий, преобразующих некоторые входные данные в выходные

1. **Достоинствами спиральной схемы разработки программного обеспечения является:**

сокращение времени до появления первых версий программного продукта

получение в конце каждой стадии законченного набора проектной документации

простота планирования процесса разработки

при необходимости можно вернуться на любой уровень и внести необходимые изменения

получение в конце каждой стадии законченного набора проектной документации

1. **В разработке технического задания участвуют**

представители заказчика

системщики

кодировщики

техники

представители сторонних организаций

1. **Техническое задание должно содержать следующий раздел**

основание для разработки

выбор архитектуры программного обеспечения

выбор языка и среды программирования

комплексное представление проектируемого программного обеспечения

тестирование и внедрение программы

1. **Техническое задание должно содержать следующий раздел**

требования к программе или программному изделию

выбор архитектуры программного обеспечения

выбор языка и среды программирования

комплексное представление проектируемого программного обеспечения

тестирование и внедрение программы

1. **Техническое задание должно содержать следующий раздел**

технико-экономические показатели

выбор архитектуры программного обеспечения

выбор языка и среды программирования

комплексное представление проектируемого программного обеспечения

тестирование и внедрение программы

1. **Для построения диаграммы переходов состояний необходимо определить:**

основные состояния

исходные данные

результаты

управляющая информация

механизмы осуществления - человек или технические средства

1. **Словарь UML включает строительный блок:**
   1. Диаграммы
   2. Группировки
   3. Зависимости
   4. Слияния
   5. Разветвления
2. **На этапе анализа требовании ЖЦ ПО определяются:**

требования к программным и информационным компонентам ПО, необходимые аппаратные ресурсы

требования к БД, физические характеристики компонент ПО, их интерфейсы

интерфейсы и распределение функции между человеком и системой

структурная схема системы и ее декомпозиция

язык программирования

1. **Создание компонентов в программе Rational Rose:**

Щелкнуть по диаграмме и поместить на нее компонент

Открыть диаграмму пакетов

Щелкнуть по инструменту и поместить на нее компонент

Щелкнуть правой кнопкой мыши по разделу Component View в окне браузера

Щелкнуть по диаграмме

1. **Erwin поддерживает проектирование следующих типов ключей:**

Альтернативный ключ

Второстепенный ключ

Внутренний ключ

Индексный ключ

Вторичный ключ

1. **Язык моделирования, как правило, включает в себя:**

Элементы модели

Элементы пакетов

Принципы удаления

Элементы прецедентов

Стрелки

1. **Создать отчет в BPwin возможно с помощью:**

Report Template Builder

встроенных шаблонов

программных модулей, создаваемых разработчиком на языке Visual Basic

создать отчет в BPwin не возможно

отчет создается разработчиком

1. **Программные инструментальные средства разработки ПО – это:**

Программное обеспечение, используемое на всех стадиях разработки нового ПО

Программы, которые используются в ходе разработки, корректировки или развития других прикладных или системных программ

Средства создания текстовых документов

Средства создания и редактирования текстовых документов

Программное обеспечение для настройки офисных приложений на условия конкретного применения

1. **Отладчик:**

Программа, помогающая анализировать поведение отлаживаемой программы

Программа, облегчающая программисту выполнение отладки программы

Системное программное обеспечение для настройки операционной системы

Программа создания системы защиты от вирусных атак

Операционная оболочка для создания и управления файловыми структурами

1. **Стереотипы для компонентов в языке UML не входящие в список:**

Процесс

Решение

Исполнимый (executable)

Файл (filе)

Документ(document)

1. **Типы стрелок, используемые на Диаграммах**

Сплошная линия с полустрелкой, пунктирная линия с V-образной стрелкой

Сплошная линия с полукруглой стрелкой

Пунктирная линия с треугольной стрелкой

Сплошная линия с кругом

Сплошная линия без стрелки

1. **Для построения диаграммы переходов состояний необходимо определить:**

выполняемые действия

исходные данные

результаты

управляющая информация

механизмы осуществления - человек или технические средства

**61. Описание операций в программе Rational Rose:**

Выбрать нужную операцию, щелкнув по ней мышью

Установить курсор в окне диаграмм и ввести описание операции

В списке браузер выбрать прецедент, щелкнув по нему мышью

Щелкнуть в окне браузера по значку «-» слева от имени класса

В списке браузер удалить прецедент, щелкнув по нему мышью

1. **Для построения функциональной диаграммы необходимо определить:**

исходные данные

основные состояния

управляющие воздействия (или условия перехода)

выполняемые действия

возможные варианты переходов из одного состояния в другое

1. **Для построения функциональной диаграммы необходимо определить:**

управляющую информацию

основные состояния

управляющие воздействия (или условия перехода)

выполняемые действия

возможные варианты переходов из одного состояния в другое

1. **В функциональных диаграммах SADT различают следующие типы влияний блоков друг на друга:**

вход

справа налево

сверху вниз

результатная

входная

1. **В функциональных диаграммах SADT различают следующие типы влияний блоков друг на друга:**

обратная связь по входу

справа налево

сверху вниз

результатная

входная

1. **В функциональных диаграммах SADT различают следующие типы влияний блоков друг на друга:**

вход

справа налево

сверху вниз

результатная

входная

1. **В основе Диаграммы потоков данных лежит понятие**

внешней сущности, хранилища (накопителя) данных

связи, функции

контроля, управление

механизма, воздействия

абстракции, управления

1. **Модели структур данных принято делить на:**

иерархические

бинарные

унарные

каскадные

спиральные

1. **Основным параметром, который необходимо учитывать при проектировании структур данных, является:**

вид хранимой информации каждого элемента данных

выбор архитектуры программного обеспечения

выбор языка и среды программирования

комплексное представление проектируемого программного обеспечения

модели жизненного цикла, стандартов, методов и средств разработки

1. **Основными параметрами, которые необходимо учитывать при проектировании структур данных, являются:**

связи элементов данных и вложенных структур

выбор архитектуры программного обеспечения

выбор языка и среды программирования

комплексное представление проектируемого программного обеспечения

модели жизненного цикла, стандартов, методов и средств разработки

1. **Различают две базовые структуры организации данных в оперативной памяти:**

векторную

матричную

бинарную

строчную

каскадную

1. **Структурный подход к тестированию имеет недостатки**

не обнаруживают пропущенных маршрутов

необходимо большое количество тестов

необходимо большой состав тестовых наборов

не всегда приводит к выполнению каждого оператора

не обнаруживают полноту и однозначность данных

1. **Какие из перечисленных процессов относятся к группе вспомогательных в соответствии со стандартом ISO/IEC 12207**

Управление конфигурацией

Разработка

Приобретение

Поставка,

Усовершенствование

Управление

1. **Какие из перечисленных процессов относятся к группе вспомогательных в соответствии со стандартом ISO/IEC 12207**

Совместная оценка

Разработка

Приобретение

Поставка

Усовершенствование

Управление

1. **Какие из перечисленных процессов относятся к группе вспомогательных в соответствии со стандартом ISO/IEC 12207**

Обеспечение качеством

Разработка

Приобретение

Поставка

Усовершенствование

Управление

1. **Укажите свойства спиральной модели ЖЦ**

На каждом витке спирали выполняется создание очередной версии продукта, уточняются требования проекта

На каждом витке спирали планируются работы следующего витка

Требования проекта постоянно уточняются

Переход на следующий этап означает полное завершение работ на предыдущем этапе

Обучение

1. **Решение каких задач обеспечивается внедрением методологии проектирования ИС?**

Гарантировать создание системы с заданным качеством в заданные сроки и в рамках установленного бюджета проекта

Обеспечить нисходящее проектирование ИС

Обеспечить проектирование «сверху-вниз»

Одна программа должна удовлетворять потребности многих пользователей

Обеспечить проектирование «снизу-верх»

1. **Характерные особенности CASE средств:**

мощные графические средства для описания и документирования ИС, обеспечивающие удобный интерфейс с разработчиком и развивающие его творческие возможности

возможность использования CASE средств пользователями непрограммистами

отсутствие графические средства для описания и документирования ИС

использование специальным образом организованного хранилища проектных метаданных, простота и доступность в использовании

управление проектом

1. **Общие функции CАSE средств:**

документирование

технологическая среда

технические средства

проектная среда

иллюстрация альтернативной точки зрения

1. **Функциональные характеристики CАSE средств:**

среда функционирования

технологическая среда

технические средства

проектная среда

иллюстрация альтернативной точки зрения

1. **Среда функционирования CАSE средств:**

проектная среда

управление проектом

управление конфигурацией

документирование

иллюстрацию альтернативной точки зрения

1. **Процесс внедрения CАSE средств состоит из следующих этапов:**

определение потребностей в CАSE средствах

определение правил CАSE средств

определение функции и процедур CАSE средств

оценка проекта CАSE средств

иллюстрация альтернативной точки зрения

1. **Функции, ориентированные на фазы жизненного цикла CАSE средств:**

моделирование

управление конфигурацией

документирование

незавершенное и непонятное применение системы

иерархическая зависимость работ

1. **Use Case – это:**

обеспечение функциональности

повествовательное описание

описание взаимодействий между объектами

описание взаимодействий между компонентами, описание взаимодействия между пользователями

обеспечение структурности, сугубо внутренняя точка зрения

1. **Контекстная диаграмма показывает:**

общее описание системы

взаимодействия

иллюстрацию альтернативной точки зрения

динамическую модель

потоки данных

1. **Диаграмма для экспозиции (FEO) показывает:**

иллюстрацию альтернативной точки зрения

диаграмма только для функции

потоки данных

взаимодействия с внешней средой

разбиение системы на подсистемы

1. **Основные понятия методологии IDEF0:**

интерфейсные дуги или стрелки

функциональный компонент

данные или потоки данных

хранилище данных

сущность – связь

1. **Модель в нотации IDEF3 содержит следующие диаграммы:**

диаграмму описания последовательности этапов процесса

диаграмму изменений в процессе

диаграмму вариантов использования

контекстную диаграмму

диаграмму дерева узлов

1. **Смысл изображений , , в модели в нотации IDEF3:**

все предшествующие процессы должны быть завершены

все следующие процессы запускаются

один или несколько предшествующих процессов должны быть завершены

все следующие процессы завершены

только несколько предшествующий процесс завершен

1. **Смысл изображений , ,  в модели в нотации IDEF3:**

один или несколько предшествующих процессов должны быть завершены

один или несколько предшествующих процессов завершаются

только один предшествующий процесс завершен

все следующие процессы запускаются одновременно

все следующие процессы завершены

1. **Процесс построения информационной модели в ERwin включает шаги:**

определение сущностей; определение зависимостей между сущностями; задание первичных и альтернативных ключей; определение атрибутов сущностей

построение диаграммы вариантов использования, диаграммы последовательности

построение диаграммы классов, диаграммы дерева узлов

построение диаграммы взаимодействий, диаграммы размещения

задание первичных и альтернативных функций; определение атрибутов

1. **На логическом уровне выполняются этапы процесса построения информационной модели в ERwin:**

определение сущностей; определение зависимостей между сущностями

задание первичных и альтернативных функций; определение атрибутов

построение диаграммы классов, диаграммы дерева узлов

составление спецификации и документации на проектируемую систему

построение структурной схемы проектируемой системы и ее декомпозиция

1. **На физическом уровне выполняются этапы процесса построения информационной модели в ERwin:**

определение сущностей, определение зависимостей между сущностями

задание первичных и альтернативных полей; определение атрибутов

построение диаграммы использования, последовательности

построение диаграммы взаимодействий, диаграммы размещения

построение контекстной диаграммы, диаграммы декомпозиции

1. **ERwin имеет уровни отображения диаграммы:**

уровень сущностей, уровень атрибутов

уровень первичных полей

уровень спецификаций

уровень взаимодействий

уровень описаний

1. **К логическому уровню в методологии IDEF1X относятся модели**

диаграмма сущность связь

модель данных, основанная на функциях

модель структуры системы

трансформационная модель

модель СУБД

1. **Основные символы диаграммы потоков данных:**

процесс

функции

актер

точка выхода

вариант использования

1. **Символы ERD, соответствующие сущностям:**

независимая сущность

зависимая функция

агрегированная сущность

композиционная сущность

наследованная сущность

1. **Типы отношений ERD:**

1\*1

1\*%

n \*0

m\*0

\*\*9

1. **Диаграмма состояний отображает:**

автомат состояний

автомат значений

управление деятельностью

актеров и отношения между ними

классы

1. **Диаграмма в ERWin строится из блоков:**

объект

компонент

отношения

компоненты

свойства

1. **Вкладки браузеров в BPWin:**

Activities

Edit

Exports

Tools

Imports

1. **Включение допустимых в BPWin инструментов:**

View – Standard Toolbar

View Toolbox

View – Status Bar

View – Zoom

View – Diagram Manager

1. **Для внесения граничной** **стрелки входа следует:**

щелкнуть по кнопке с символом стрелки, в палитре инструментов перенести курсор к левой стороне экрана, пока не появится начальная штриховая полоска

перенести курсор к левой стороне экрана, пока не появится начальная штриховая полоска

во всплывающем меню выбрать Name и добавить имя стрелки в закладке Name диалога IDEF0 Arrow Properties

щелкнуть один раз по полоске (откуда выходит стрелка )

вернуться в палитру инструментов и выбрать опцию редактирования

выбрать стрелки

1. **Виды отчетов в BPWin:**

Model Report, Diagram Report

Diagram Report, Activity

Model DFD, Diagram DFD

Diagram Object Class, Activity Class Report

Arrow Diagram, Data Diagram Report, Model Diagram Report

1. **Разветвление и сли****яние:**

возможность слияния и расщепления моделей обеспечивает коллективную работу над проектом

объединение различных работ и данных

ход выполнения алгоритма программы

стрелки, порожденные в разных работах, представляют собой разнородные данные или объекты

соединение

1. **В IDEF3 различают типы стрелок, изображающих** **связи – это:**

старшая (Precedence)

Relational

асинхронные (Asynchronous)

процесс объектов (Object Process)

объект ссылки — (Referent)

1. **Официальная спецификация IDEF3 различает стили объектов ссылок — это:**

безусловные (unconditional)

асинхронные

потоки объектов (Object Flow)

процесс объектов (Object Process)

объект ссылки (Referent)

1. **AllFusion ERwin Data Modeler (ERwin) позволяет:**

наглядно отображать сложные структуры данных

упрощать функции

упрощать разработку классов и обьектов

многократно использовать классы

наглядно отображать графическое преставление модели

1. **ERwin Data Modeler поддерживает нотации проектирования данных**

Dimensional

IDEF5Х

DFD

IDEFX

IDEFEX

1. **Виды процессов и их количество в составе ЖЦ:**

8 вспомогательных

7 основных

3 вспомогательных

10 основных

7 организационных

1. **Основные процессы жизненного цикла программного обеспечения**

заказ, поставка

проект

аудит

решение проблем

документирование, управление конфигурацией

1. **Модели жизненного цикла ПО:**

каскадная модель

групповая

инкрементная модель

лестничная модель

периодическая модель

1. **Жизненный цикл ПО по методологии RAD состоит из фаз:**

анализа и планирования требований

функционирования

тестирования и внедрения

программного обеспечения

планирования

1. **Основные принципы создания информационных систем:**

управленческие

информационные

юридические

вспомогательные

дополнительные

1. **Управленческие принципы создания информационных систем включают принципы:**

системности

функциональности

наглядности

простоты

заменяемости

1. **Модели, дающие полное описание системы:**

модель классов

модель поведения

модель последовательности

модель коопераций

модель компонентов

1. **Виды диаграмм, с помощью которых описывается модель взаимодействия:**

диаграмма прецедентов

диаграмма элементов

диаграмма развертывания

диаграмма состояния

диаграмма пакетов

1. **Строительные блоки UML:**

сущность

функции

компоненты

узлы

пакеты

связи

1. **Типы сущностей языка UML:**

структурные

управлющие

аппаратные

концептуальные

логические и физические

1. **При оформлении графического изображения технологического процесса на схеме отображаются:**

Носители информации

Отношения

Технологические операции

Графические операции

Страница всех графических символов

1. **В среде Rational Rose поддерживаются варианты устойчивости:**

Временный

Постоянный

Относительно устойчивый

Динамичный

Временно устойчивый

1. **Язык UML предназначен для:**

Визуализации

Разделения

Тестирования

Сопровождения

Анализа требований

1. **Язык UML – это:**

Унифицированный язык моделирования

Язык для разработки систем искусственного интеллекта

Язык создания запросов в базах данных

Язык управления базами данных

Язык логического программирования

1. **IEEE – это:**

Institute of Electrical and Electronic Engineers, Inc

Просто принятое обозначение, расшифровки не имеет

Такая аббревиатура нигде не используется

Американская организация ученых экономистов

Коммерческая организация ученых и исследователей

1. **Структура ЖЦ ПО по стандарту ISO/IEC 12207 базируется на группах процессов:**

Основные процессы ЖЦ ПО

Стандартные процессы

Управляющие процессы

Ожидающих выполнение

Целые процессы

1. **Организационные процессы жизненного цикла программного обеспечения:**

Обучение

Документирование

Аудит, решение проблем

Ожидающих выполнение

Целые процессы

1. **Уровни логической модели:**

Модель данных, основанная на ключах

Полная операционная модель

Модель данных, основанная на классах

Диаграмма пакетов

Целые процессы

1. **Объектно ориентированная методология (ООМ) включает в себя составные части:**

Объектно ориентированный анализ

Объектно ориентированная декомпозиция

Объектно ориентированная парадигма

Объектно ориентированное моделирование

Объектно ориентированный подкласс

1. **Типы объектов ссылок не входящие в IDEF3:**

CALL

ELAB (Elaboration)

NOTE

UOB (Unit of behaviour)

GOTO

1. **Символы ERD, соответствуют отношениям:**

Ограниченное отношение

Композиционное отношение

Правильное отношение

Диапазонное отношение

Медианное отношение

1. **Диаграмма моделирования данных «сущность связь» (ERD) состоит из:**

Атрибуты

Операции

Компоненты

Классы

Вариантов использования

1. **Создание компонентов в программе Rational Rose:**

Щелкнуть по кнопке Component на панели инструментов

Щелкнуть по инструменту и поместить на нее компонент

Щелкнуть правой кнопкой мыши по разделу Component View в окне браузера

Щелкнуть по диаграмме

Щелкнуть по инструменту

1. **Концептуальная модель UML включает в себя составные части:**

Базовые строительные блоки

Дополнительные строительные блоки

Правила, определяющие классы

Технические средства разработки ПО

Аналитические средства разработки ПО

1. **Технология АВС включает следующие основные понятия:**

Двигатель затрат

Класс затрат

Управление затратами

Технические средства разработки ПО

Аналитические средства разработки ПО

1. **Инструментальные средства разработки программ – это:**

Аппаратные и программные инструменты разработки нового ПО

Средства тестирования ПО

Средства отладки ПО

Технические средства разработки ПО

Аналитические средства разработки ПО

1. **Объектно ориентированное проектирование использует инструментальные средства:**

Rational Rose

JAM

Bpwin

UML

Use case

1. **Вспомогательные процессы жизненного цикла программного обеспечения**

Документирование, управление конфигурацией

Эксплуатация, сопровождение

Управление, создание инфраструктуры, усовершенствование

Анализ и синтез систем управления

Документировать принимаемые решения, используя полученные модели

1. **Моделирование в UML позволяет решать задачи:**

Визуализировать систему в ее текущем или востребованном для нас состоянии

Раccчитать экономическую эффективность от внедрения программного обеспечения

Смоделировать разрабатываемую информационную систему

Анализ и синтез систем управления

Документировать принимаемые решения, используя полученные модели

1. **Методы, указывающие уровни качества:**

Инспектирование

Внедрение и проверка

Вид и окружение

Неформальные методы

Документирование

1. **Получение отчета в BPWin:**

Tools Reports

Model – Model Properties…

Model – New Model Properties…

File Export

Model – New Model

1. **Виды моделей жизненного цикла ПО:**

Спиральная модель

Периодическая модель

Имитационная модель

Поэтапная модель

UML model

1. **Для отражения модели прецедентов на диаграмме используются:**

Прецедент

Кооперации

Поведения

Клиенты (“Клиент”)

Use case

1. **Количество и виды процессов в составе жизненного цикла ПО:**

4 организационных

3 организационных

7 организационных

3 основных

10 основных

1. **Активизация диаграммы классов в окне диаграммы:**

Раскрыть логическое представление в браузере (Logical View) и дважды щелкнуть на пиктограмме Main (Главная)

Щёлкнуть по кнопке с изображением диаграммы размещения на стандартной панели инструментов

Щёлкнуть по кнопке с изображением диаграммы прецедентов на стандартной панели инструментов

Через пункт меню Main – Class Diagramm (Главная – Диаграмма классов)

Щёлкнуть на кнопке с изображением диаграммы взаимодействия на стандартной панели инструментов

1. **Компоновщик – это:**

Программа, которая из объектных модулей стандартных подпрограмм формирует загрузочный модуль

Комплекс программ, для создания и ведения баз данных

Программа для компоновки и оформления тестовых документов

Программа для поиска синтаксических и семантических ошибок в программе

Программа для компоновки

1. **Построение SADT – модели включает в себя выполнение следующих действий:**

Определение цели и точки зрения модели, построение, обобщение и декомпозиция диаграмм

Разработка, отладка и тестирование программного обеспечения

Представление исследуемого объекта средствами системного моделирования

Написание программного обеспечения для разрабатываемой системы по требованиям заказчика

Использование графических пакетов для представления системы в виде цели

1. **Диаграмма DFD описывает:**

Функции обработки стрелок (arrow)

Документы ( стрелки, arrow ), объекты сотрудников или отделы, которые участвуют в обработке внешних стрелок

Функции обработки внешних ссылок

Объекты сотрудников или отделы

Использование графических пакетов для представления системы в виде цели

1. **CASE – средства классифицируются по следующим признакам:**

По степени интегрированности c СУБД

По используемым языкам программирования

По используемому программному обеспечению

По этапам жизненного цикла программного обеспечения

По степени сложности

1. **Что не относится к основным компонентам диаграмм потоков данных:**

Работы

Внешние сущности

Системы/подсистемы

Потоки данных

Накопители данных

1. **Кратность ассоциации, записанная неправильно:**

Ноль или число (0, 8)

Равна единице ( 1 )

Определенное число ( 3 )

Бесконечность

Неопределенное число

1. **Преимущества спиральной модели:**

Анализ риска и издержек в процессе проектирования

Простота отладки и компоновки ПО

Быстрота разработки программного обеспечения

Отсутствие ошибок при проектировании

Быстрота и простота

1. **Допустимые объявления свойств, используемые совместно с атрибутами:**

addOnly (только добавляемый)

private ( закрытый )

protected (защищенный)

classifier (классификатор)

instance (экземпляр)

1. **Особо важные отношения для объектно ориентированного моделирования:**

Зависимость

Независимые

Многие ко многим

Упрощенная

Один к многим

1. **Создание диаграммы состояния в программе Rational Rose:**

В контекстно- зависимом меню выбрать команду New – Statechart Diagram

Снова вызвать контекстно-зависимое меню и выбрать команду New – Diagram

New – Diagram

New-FlowChart

New – Diagram- FlowChart

1. **К инструментальным средствам представления функциональных моделей относятся:**

Model Mart

ARIS

JAM

Mathcad

Matlab

1. **К организационным областям знаний SWEBOK относятся:**

Управление конфигурацией, управление проектами

Инженерия требований

Тестирование ПО

Проектирование ПО

Конструирование ПО

1. **Объектно – ориентированный подход в проектирование не использует:**

Диаграммы потоков данных DFD

Диаграмму объектов

Диаграмму последовательности

Диаграмму классов

Диаграмму взимосвязи

1. **Модель ЖЦ ПО включает в себя:**

Стадии

Слияние работ

Упровления

Тестирование

Верификацию

1. **Внутренние стрелки, не входящие в состав диаграммы IDEF0:**

mechanism output

output input

output control

output

mechanism

1. **Проблемная область в разработке определяет:**

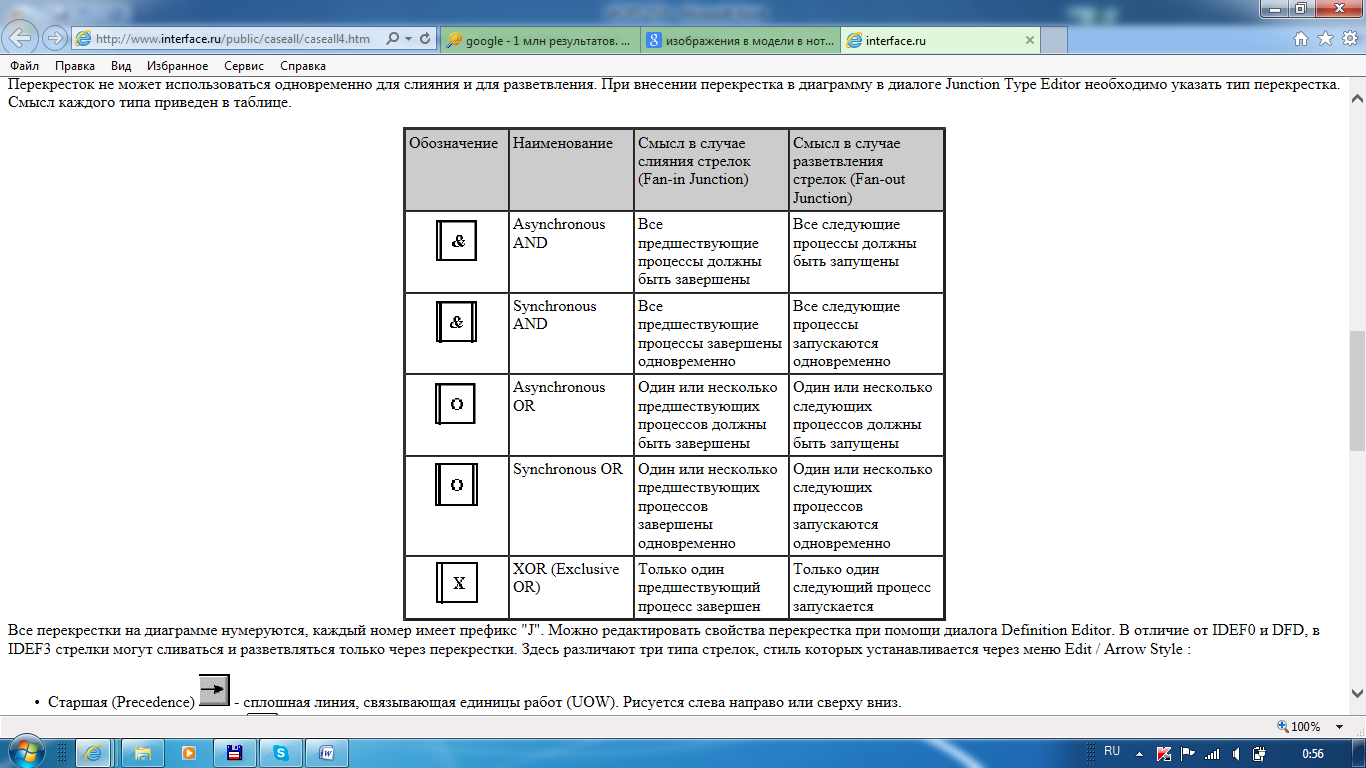
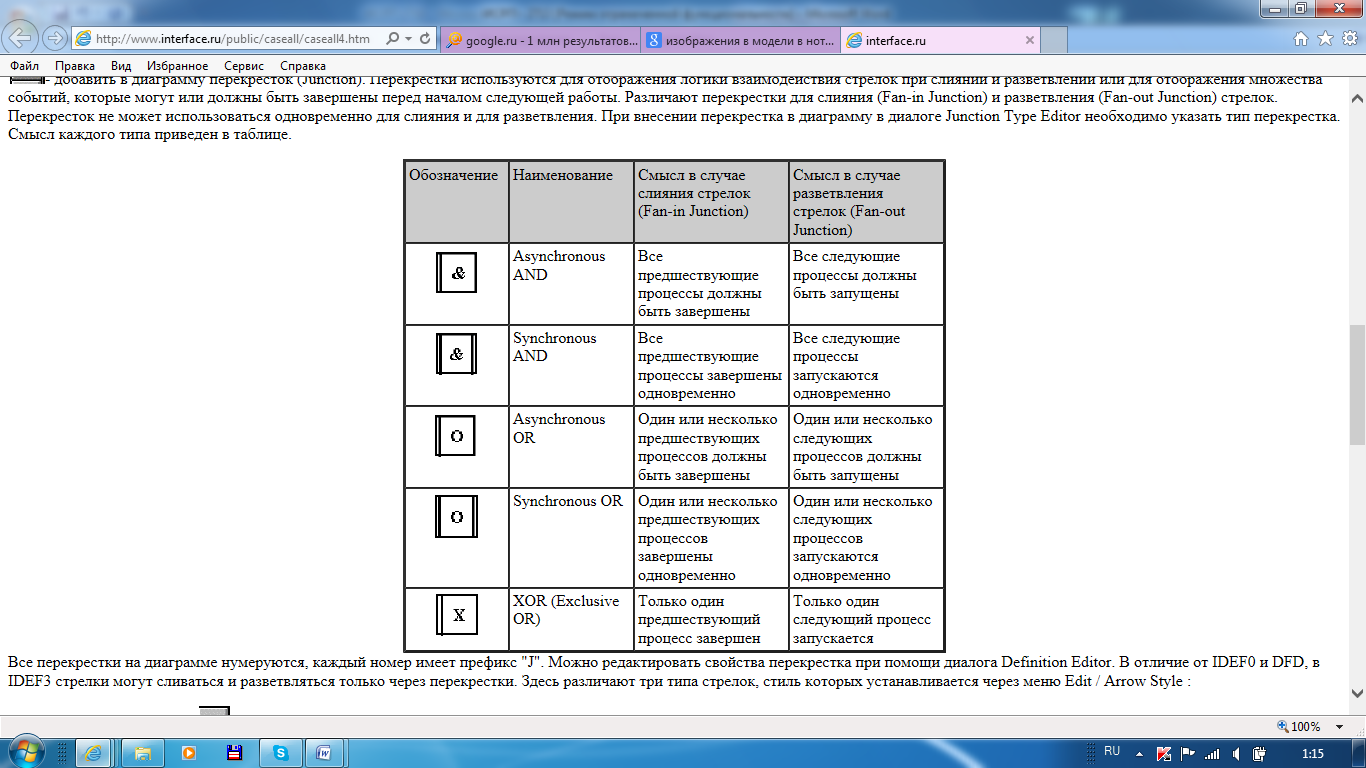
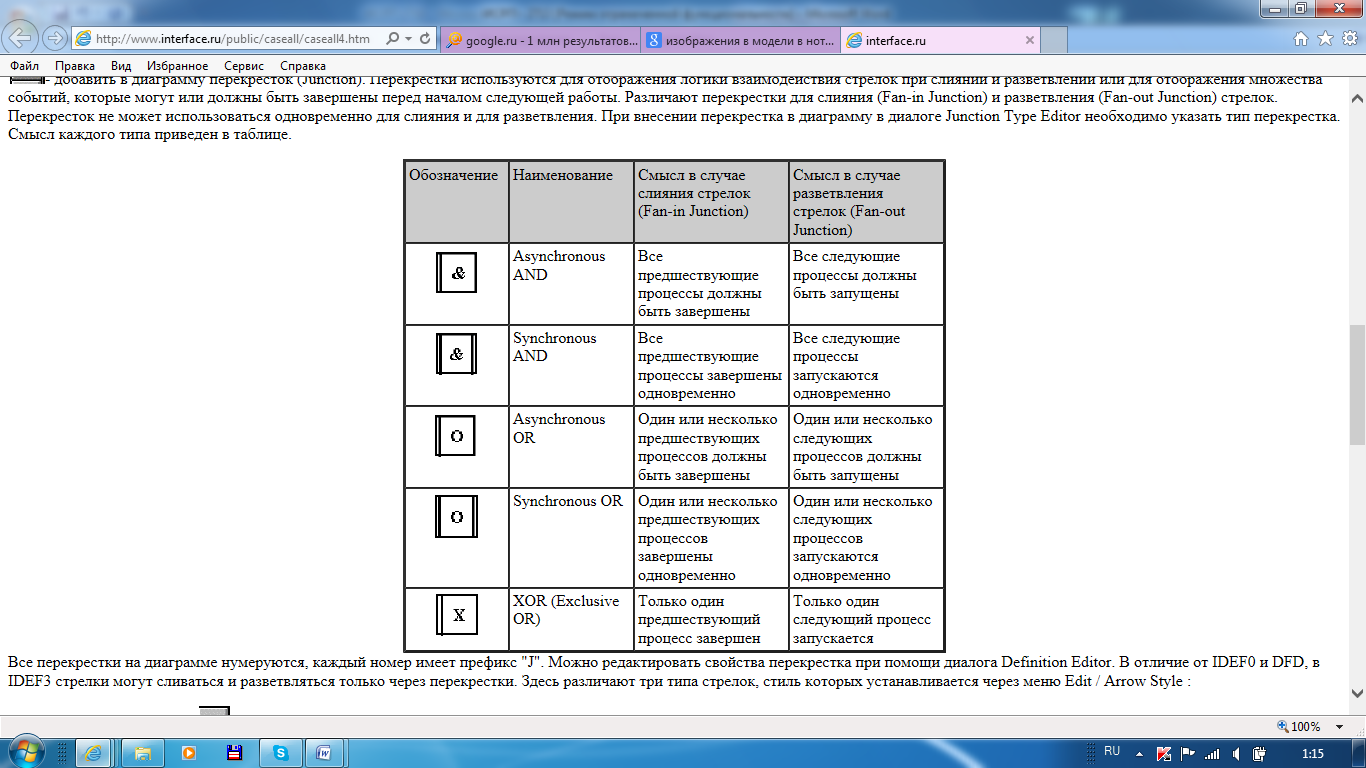
структуру данных, функциональность

порядок действий, данные

последовательность вызова

реализации функции

данные

1. **Смысл изображений , ,  в модели в нотации IDEF3:**

Все следующие процессы запускаются одновременно

Все предшествующие процессы должны быть запущены

Только несколько предшествующих процессов завершены

Только три предшествующих процессов завершены

Один или несколько предшествующих процессов должны быть завершены

1. **Жизненный цикл ПО по методологии RAD состоит из фаз:**

Проектирования

Разделения

Проверки

Программного обеспечения

Тестирования

1. **Для задания стоимости работы (для каждой работы на диаграмме декомпозиции) следует:**

Щелкнуть правой кнопкой мыши по работе и на всплывающее меню выбрать Cost

Произвести бухгалтерский расчет стоимости работ и занести в диаграмму

Создать дополнительный программный модуль расчета стоимости работ

В рамках диаграммы декомпозиции такое действие выполнить не возможно

Стоимость работ рассчитывается и оформляется отдельным документом

1. **Язык моделирования, как правило, включает в себя:**

Принципы использования

Элементы пакетов

Принципы удаления

Элементы прецедентов

Стрелки

1. **Основные стереотипы класса:**

Управляющий элемент

Интерфейсный элемент

Отношения

Подчинённый элемент

Зависимый элемент

1. **Представление размещения – это:**

Диаграммы размещения

Диаграммы взаимодействия

Диаграммы последовательности и кооперации

Компоненты, пакеты

Диаграммы компонентов

1. **На этапе анализа требований ЖЦ ПО определяются:**

Архитектура системы, ее функции, внешние условия, распределения функций между аппаратурой и ПО

Коллектив разработчиков

Коллектив сопровождения программного обеспечения

Язык программирования

Состав технических средств реализации системы

1. **Типы стрелок, используемые на Диаграммах кооперации для обозначения сообщений:**

Пунктирная линия с треугольной стрелкой

Сплошная линия с полукруглой стрелкой

Сплошная линия с кругом

Сплошная линия без стрелки

Сплошная линия

1. **Основные понятие объектно – ориентированного подхода:**

Полиморфизм

Реализация

Обобщение

Ассоциация

Композиция

1. **К инструментальным средствам объектно- ориентированного анализа и проектирования относятся:**

Rational Rose

BPwin

JAM

Model Mart

MAJ

1. **Для создания моделей Erwin используется международно признанные системы обозначений (нотации):**

IDEFX

IDEF0

IDEF3

DFD

IDEF4

1. **Основные понятия объектно-ориентированного подхода:**
   * + 1. Инкапсуляция
       2. Агрегирование
       3. Композиция
       4. Ассоциация
       5. Реализация
2. **К основным компонентам диаграммы ERwin относятся:**
   * 1. Связи
     2. Классы
     3. Слияния
     4. Использования
     5. Разветвления
3. **К основным областям знаний SWEBOK относятся:**

Инженерия требований, проектирование ПО

Управление проектами

Управление конфигурации

Инженерия качества программных средств

Сопровождение ПО

1. **Имитационная модель включает следующие основные элементы:**

Процессы (Process)

Потоки (Thredes)

Удаление (Delets)

Вставка

Очистка

1. **Классы позволяют дать описание совокупности объектов с общими:**

атрибутами

пакетами

компонентами и узлами

классами

связями

1. **Основными преимуществами CASE средств являются:**

Уменьшение времени на разработку

Поддержка возможности моделирования совместно со средними CASE средствами

Усложнение доступа к данным

Увеличение времени на разработку

Модификация

1. **Основные пакеты метамодели языка UML:**

общие механизмы

общие элементы

дополнительные элементы

стрелки

связи

1. **Создать отчет возможно с помощью:**

встроенных шаблонов

программных модулей, создаваемых разработчиком на языке Visual Basic

создать отчет BPwin не возможно

отдельно поставляемых программ

программных модулей

1. **Каждая область ядра знаний SWEBOK представляется:**

структурной схемой

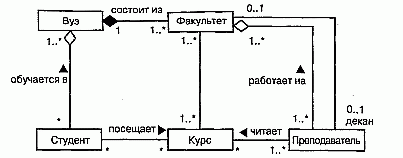
диаграммой UML

описанием и комментариями

определением языка программирования

определением инструментов поддержки инженерной деятельности

1. **Типы отношений, использованные на рисунке:**



агрегация

обобщение

зависимость

реализация

независимость

1. **Для добавления описания прецедента в программе Rational Rose необходимо:**

В списке браузер выбрать прецедент, щелкнув по нему мышью

Установить курсор в окне описания и набрать краткое описание класса

Закрыть его с помощью команды меню View-Documentation

Установить курсор в окне описания и набрать краткое описание актера

В списке браузер выбрать актера, щелкнув по нему мышью

1. **Что не относится к основным компонентам диаграмм потоков данных?**

внутренние сущности

накопители данных

процессы

системы/подсистемы

функции

1. **Стереотипы сообщений диаграммы последовательности в нотации UML:**

«View» (представить)

Удалить

«Call» (вызвать), «Return» (возвратить)

“Destroy”(уничтожить), “Send” (послать)

«Copy» (копировать)

1. **Получение отчета в PBWin:**

Model Report….

File Export

Tools – Reports Builder….

Model Model Properties…

Report

1. **Характеристика программных инструментов разработки:**

Пространственная и временная группировка информации (модульность)

Функциональность

Декомпозиция, атомарность (функции и данных), поэтапность для разработки

Визуальность и тестируемость разработки

Мера автоматизма в работе реализации и инструментах разработки

1. **Типы отношений между прецедентами:**

Communicate

* Note
* Node

Interface

Function

1. **Функции, выполняемые нотацией в визуальном моделировании:**

обеспечивает достаточными классами

обеспечивает нотацию

обеспечивает достаточными компонентами

предлагает конкретные компоненты

является языком для описания взаимодействий

1. **Описание атрибута в диаграмме классов в программе Rational Rose:**

Щелкнуть в окне браузера по значку слева от имени класса

Установить курсор в окне описания

Установить курсор в окне описания и набрать краткое описание прецедента

Ввести имя нового атрибута

Щелкнуть правой кнопкой мыши по классу в окне браузера

1. **Количество и виды процессов в составе жизненного цикла ПО:**

5 основных

7 вспомогательных

3 основных

3 организационных

4 основных

1. **В среде Rational Rose поддерживаются варианты устойчивости:**

статичный

временный неустойчивый

временно устойчивый

постоянный

временно статичный

1. **Элементы поведения состоят из подпакетов:**

варианты использования и автоматы

общее поведение

последовательности

размещение

ограничение доступа

1. **перечень меток имен событий:**

entry

do end

end

begin

and

1. **Главные принципы объектного подхода:**

абстрагирование

agretations

агрегирование

relation

обощение и специализация

1. **Диграмма моделирования данных «сущность связь» (ERD) состоит:**

связи

элементы

классы

отношения

объекты

1. **Представление компонентов –это:**

Пакеты

Классы

Классы, диаграммы классов

Действующие лица и варианты использования

Диаграммы последовательности, диаграммы кооперации

1. **Программная инженерия:**

software engineering

коллектив инженеров программистов, разрабатывающих программное обеспечение для компьютеров

комплекс программ, предназначенный для решения инженерных задач

ООП

**Моделирование основывается на принципах:**

выбор модели оказывает определяющее влияние на подход к решению проблемы и на то, как будет выглядеть это решение

каждая модель может быть представлена с различной степенью точности; лучшие модели – те, что ближе к реальности

нельзя ограничиваться созданием только одной модели.

декомпозиции системы на отдельные подзадачи

инкапсуляции и полиморфизма

1. **В бизнес процессах выделяют классы процессов:**

основные бизнес процессы

управляющие

решающие бизнес процессы

регламентирующие бизнес процессы

бизнес процессы поведения системы

1. **CASE средства классифицируются по следующим признакам:**

по применяемым методологиям и моделям систем и БД

по типам ПО

по используемому программному обеспечению

по этапам жизненного цикла программного обеспечения

по уровням детализации и декомпозиции проектируемой системы

1. **К малым интегрированным средствам моделирования относятся:**

ERwin

Model

ARIS Toolset

Design/IDEF

Designer/2000

1. **К средним интегрированным средствам моделирования относятся:**

Design/IDEF

Rational Rose

BPwin

Model Mart

Designer

1. **К основным понятиям объектно ориентированного подхода относятся:**

полиморфизм

обобщение

реализация

агрегирование

отношения

1. **Дополнительные принципы объектного подхода:**

типизация

реализация

внедрение

перпендикулярность

отношения

1. **Методологии, поддерживаемые в BPwin**

IDEF0

IDEF

IDEF1Х

IDEF1

IDEFХ

1. **Диаграмма IDEF0 может содержать следующие типы диаграмм:**

контекстную диаграмму, диаграмму декомпозиции

диаграмму классов

диаграмму компонентов

диаграмму взаимодействий

диаграмму элементов

1. **Внутренние стрелки, не входящие в состав диаграммы IDEF0:**

mechanism output

output input

output control

output input feedback

output

1. **Типы стрелок, не входящие в состав диаграммы IDEF0:**

Editor

Input

Control

Output

File

1. Элементы, входящие в состав интерфейса Rational Rose

Browser

Reports

Query

Private

File

1. **Стрелка управления (Control) в модели в нотации IDEF0 служит:**

для описания правил

для управления

для указания на другую модель работы

для описания материала, которые производятся работой

для описания информации, которые производятся работой

1. **ERwin позволяет создавать модель имеющую:**

А) как логический уровень, так и физический уровень

1. абстрактный уровень
2. контекстный уровень

G) функциональный уровень

d) ООП

1. **Quick Reports – создание простейших отчетов – позволяет создавать отчеты:**

Columnar. Простой табличный отчет

Group/Totals. Табличный отчет с автоматической группировкой и сортировкой данных

Report Header. Печатается единожды в начале отчета

Header

Footer

1. **BPwin допускает следующие переходы с одной нотации на другую:**

IDEF0 → DFD

IDEF → IDEF4

IDEF3 → DFD

DFD → IDEF0

DFD → IDEF4

1. **DFD описывает:**

функции обработки информации (работы)

функции обработки стрелок (arrow)

внешние ссылки (external references), объекты, сотрудников или отделы, которые участвуют в обработке информации

внутренние ссылки

функции

1. **. BPwin позволяет создавать на диаграмме DFD типы граничных стрелок:**

обычная граничная стрелка

специальная стрелка

внутренняя ссылка

внешняя ссылка

управляющая стрелка

1. **В BPwin 4.0 отчеты могут быть экспортированы в распространенные форматы:**

текстовый

символьный

графический

видео

звук

1. **Поддерживаемые в RPTwin типы операторов:**

текстовый оператор конкатенации (&)

символ

текст

графика

данные

1. **Инструментальное средство ERwin позволяет:**

проектировать на физическом и логическом уровне модели данных

редактировать и отлаживать программы

управлять процессом конструирования ПО

редактировать

окрывать

1. **К основным компонентам диаграммы ERwin относятся:**

сущности

переходы

классы

стрелки

схемы

1. **Точки зрения организации в ARIS:**

организационная структура

структура внедрения и структура потоков

управленческая структура

структура потоков

внедрения

1. **Уровни точки зрения в ARIS:**

описание требований

описание структуры

описание поведения

описание объектов

элементы

1. **Методы описания, используемые в ARIS:**

EPC метод описания процессов

ЕРТ – метод описания потоков

ERM модель сущность связь для описания структуры объектов

Описание объектов

Описание классов

1. **К основным компонентам инструментов ARIS Toolset относятся:**

Explorer (проводник)

Design

Таблица элементов

Internet (интернет)

WordPad (ввод текстовых данных)

1. **ARIS Business Optimizer позволяет:**

определять целевые затраты и рассчитывать стоимость продукта: во что компании обходится предоставление отдельных продуктов

принимать решения

планировать требования

принимать решения о времени начала и окончания работы над проектом

определять последовательность работ , выполняемых в ходе работы над проектом

1. **"Взгляды" ARIS:**

процессы

элементы

объекты

потоки

процедуры

1. **Уровни анализа ARIS для каждого "взгляда":**

требования

функции

редактирование

поведение

просмотр

1. **MS Visio позволяет создавать схемы, чертежи, диаграммы с помощью:**

встроенных шаблонов

функций

панели инструментов

графических редакторов

палитры

1. **В UML существуют следующие виды сущностей:**

структурные

разветвляющиеся

декомпозиционные

простые

составные

1. **В UML существуют следующие виды связей:**

зависимость

слияние

разветвление

1-5

1-10

1. **В процессе разработки и внедрения программных систем UML позволяет специфицировать все важные решения касающиеся:**

анализа

выбора инструментальных средств

выбора технических средств

теста

условий

1. **UML, как язык документирования, помимо исполняемого кода производит и другие продукты, включающие:**

требования, архитектуру, проектные решения

спецификацию технических средств

требования к уровню квалификации разработчиков

документы

элементы

1. **UML включает синтаксические и семантические правила для:**

имен, областей действия

агрегации

тестирования

отношений

связей

1. **Применение языка UML существенно упрощает последовательное использование механизмов:**

спецификации, дополнения

выработки требований

создания плана работ

документов

условий

1. **Механизмы расширения UML включают:**

стереотипы

значения

ограничения

исключения

дополнения

1. **В объектное ориентированном моделировании между классами существуют типы связей:**

зависимость

2-2

3-3

слияние

линейность