Убедитесь, что вы поняли и выполнили предыдущий предмет.

Большинство людей на этой планете используют десятичные числа. Это может показаться естественным для человека, имеющего 10 пальцев и часто использующего их для счета.

Но это не так естественно для компьютеров, построенных на кремниевых интегральных схемах. Поскольку у нас есть 10 пальцев, кусочки кремния могут иметь два состояния: они могут проводить электричество или нет. Такое поведение ДА или НЕТ называется двоичным.

Нам нравится ассоциировать вещи с числами. В этом случае для описания состояния кремния нужны только два числа. Когда он может проводить электричество, мы говорим ДА или 1 (один). Другое состояние связано с 0 (нулем). С информационной точки зрения эти два состояния дают нам наименьшую часть информации, которую мы называем BIT. Значение BIT может быть 0 (ноль) или 1 (единица).

Компьютеры построены как двоичные машины, которые понимают только ноль и единицу на самом базовом, фундаментальном уровне. На самом деле это всего лишь кусочки информации. Комбинации этих битов могут давать то, что мы называем байтами, которые могут вырасти до довольно больших чисел. Те же комбинации байтов можно интерпретировать как символы, предоставляющие нам читаемые (и не очень читаемые) данные.

Один байт имеет 8 бит и может быть числом от 0 до 255 (2 в степени 8). Один байт может также выражать один американский символ (ASCII), например "A".

Два байта могут представлять символ Unicode, который может быть выражен на многих иностранных языках; эти символы Unicode могут быть числами от 1 до 65500.

Нужно ли нам все это помнить? Не совсем. В самом начале компьютерной эры программисты разговаривали с компьютерами с помощью последовательностей ONE-Zero. Но это было в прошлом.

Программисты изобрели множество языков, которые пытаются выражать человеческие мысли в точных формах, приемлемых для компьютеров. Каждый язык программирования требует компилятора, который может переводить все, что мы пишем в виде исходного кода, в двоичный код.

В прошлом разные аппаратные платформы имели разные языки программирования. На заре программирования такие языки, как COBOL, FORTRAN и C были адаптированы для основных платформ.

До появления Java у каждого языка на любой конкретной платформе были свои особенности. Обычным процессом было написание программного обеспечения специально для конкретной платформы, а затем компиляция программного обеспечения с помощью определенного компилятора и смешивание с конкретными би,лиотеками для запуска этой программы на выбранной платформе.

https://ituniversity.us/content/1-6.png

Обычным процессом было написание программного обеспечения специально для конкретной платформы, а затем компиляция программного обеспечения с помощью определенного компилятора и смешивание с конкретными библиотеками.

https://ituniversity.us/content/2-6.png

Java не только предлагала отличный объектно-ориентированный язык, но также представила совершенно новую технологию. Официальный слоган Java: напиши один раз, беги везде! -Это связано с тем, что исходный код Java компилируется в стандартный двоичный формат, который гарантированно работает на любой основной платформе и на нескольких аппаратных устройствах.

Что делает эту технологию Java такой гибкой? Стандартная спецификация Java! Введенная Sun Microsystems, эта конкретная спецификация требует, чтобы любой компилятор создавал стандартный двоичный код, только после этого спецификация предоставляет единый стандарт для любой платформы о том, как выполнять этот Java-код через виртуальную машину Java (JVM).

JVM выполняет трюк своевременного перевода двоичного кода в машинный код, специфичный для каждой платформы. Sun Microsystems предоставила первую JVM для основных платформ, включая мэйнфреймы IBM, Apple, Windows и UNIX.

Было ли это ясно до сих пор? Выделите нужный текст и нажмите «Я» или «Пообщайтесь с Ассистентом».

JVM работает вместе со стандартными (для конкретной платформы) библиотеками Java, образуя так называемую среду выполнения Java (JRE).

Виртуальная машина Java - это абстрактная (т.е. виртуальная) система, определенная этой спецификацией и обычно работающая с операционной системой. Эта спецификация включает четкие инструкции, защищающие код Java от потоков безопасности (загрузчик классов и т. Д.), Но опускает детали реализации, такие как алгоритмы сборки мусора или оптимизации.

https://ituniversity.us/content/3-6.png

Microsoft, пытаясь догнать Java, предложила аналогичную, но другую технологию под названием .NET.

Платформа Dot NET также является виртуальной исполнительной системой со своим собственным стандартом: Common Language Infrastructure (CLI). Common Language Runtime (CLR) - это реализация стандарта CLI с функциональностью, аналогичной JVM: загрузка классов, своевременная компиляция для конкретной платформы Windows и выполнение кода.

Подобная идеологии Java сосуществуют со следующими отличиями. Dot NET позволяет разработчикам писать исходный код на многих языках: C, C ++, C #, Perl и других. Любой исходный код может быть скомпилирован согласно стандарту CLI в один и тот же двоичный код, который понимается CLR.

Мир Java предлагает множество инструментов интерактивной среды разработки (IDE), Eclipse, IDEA, NetBeans, JDeveloper и другие.

Microsoft не предоставляет разработчикам .NET выбора. Но их единственная IDE, Microsoft Visual Studio, действительно хороша в разработке, компиляции, развертывании и выполнении приложений в среде .NET.

Девиз .NET: «Пишите на (почти) любом языке и работайте на любых платформах WINDOWS: Windows XP, Windows 8, Windows Tablet и Windows phone».

https://ituniversity.us/content/4-6.png

https://ituniversity.us/content/5-6.png

Жизнь за пределами Java не так увлекательна, но все же возможна. Microsoft доказала эту возможность, изобрав свой собственный язык под названием C #, который произносится как see-sharp.

Java и C # очень похожи по своей языковой структуре, функциональности и идеологии.

В книге «Архитектура и дизайн, готовая к интеграции» есть глава («Сага о братьях и сестрах»), в которой они дословно сравниваются.

В книге описывается разработка программного обеспечения с использованием Java, C #, распознавания голоса, беспроводной связи и технологий знаний, а также приводятся теоретические и практические примеры исходных текстов в этих областях.

Несколько глав книги, включая «Сагу о братьях и сестрах», доступны на веб-сайте МСЭ.

C #, этот счастливый второй ребенок, унаследовал хорошие манеры, элегантность, стиль и даже немного одежды от своего отчима, наслаждаясь заботой матери и ее богатой и обширной платформой .NET.

Задания:

1. Прочтите главу "Сага о братьях и сестрах" из книги.

2. Google "Сравнение C # и Java"

3. Напишите в MS Word абзац с тремя сильнейшими функциями Java (не в C #) и тремя сильнейшими функциями C # (не в Java).

Отправьте документ с именем 1.1.2.JavaTechnology.Your.Name по адресу dean@ituniversity.us.