Содержание:

Введение

Язык программирования — формальный язык, предназначенный для записи компьютерных программ. Язык программирования заранее определяет набор лексических, синтаксических и семантических правил, в той или иной степени строгости определяющих внешний вид программы или действия, которые реализует исполнитель (чаще всего – ЭВМ). Со времен зарождения программирования создано более 8000 различных языков и с каждым годом по мере роста числа и разнообразия задач, появляющихся перед программистами, их становится все больше [1].

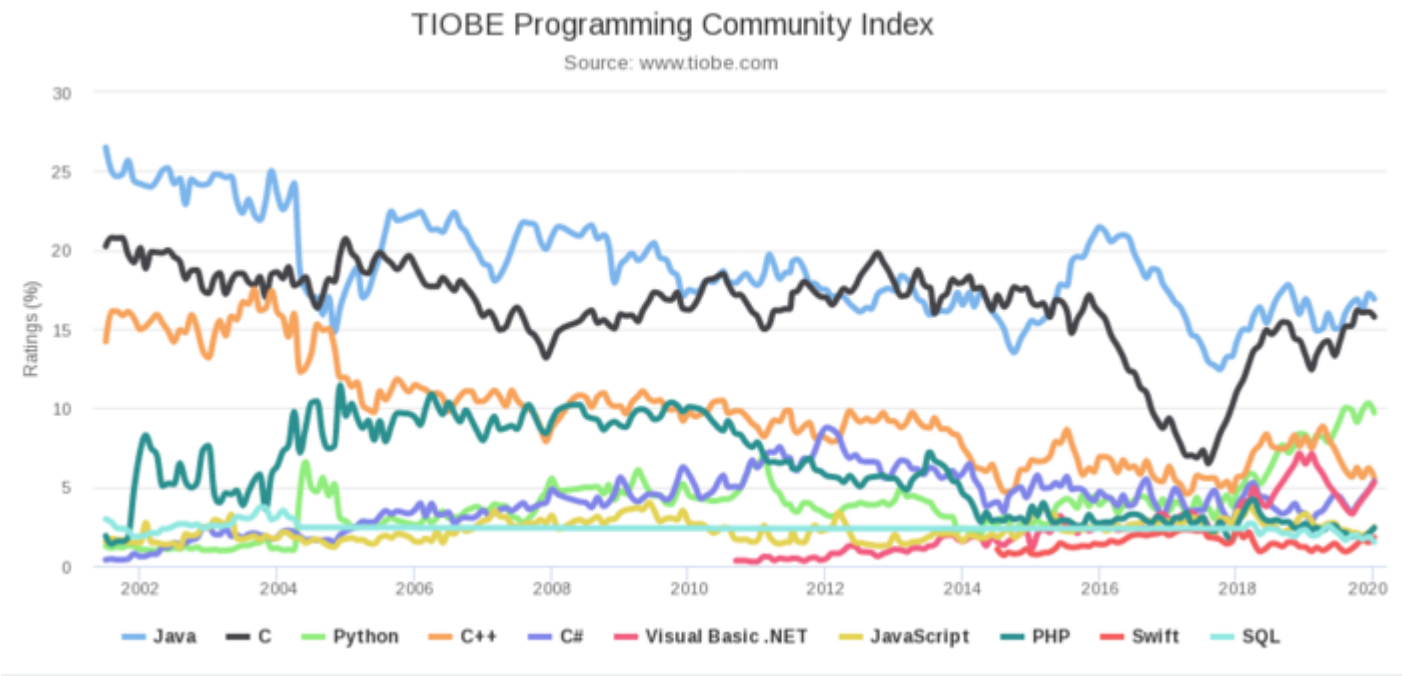
Идея языка программирования появилась так же давно, как и универсальные вычислительные машины – на рубеже 40-50 годов. Уже на первых шагах их эксплуатации выяснились недостатки использования машинного кода, определились методы устранения или уменьшения этих недостатков: использование библиотек стандартных подпрограмм, имен вместо адресов, предварительного распределения памяти и т.д. Программы для первых ЭВМ программисты писали на языках машинных команд. Это очень трудоемкий и длительный процесс. Проходило значительное время между началом составления программы и началом ее использования. Решить эту проблему можно было лишь путем создания средств автоматизации программирования. Изначально программист записывал решаемую задачу в виде математических формул, преобразовывал формулы в двухбуквенные коды. В дальнейшем специальная программа переводила эти коды в двоичный машинный код [1].

Область применения ЭВМ расширялась, расширялся и круг пользователей, обосновывая необходимость приближения машинного языка к разговорному. Новый язык позволил программировать на языке, близком к обычному английскому.

В последнее время одним из основных направлений в развитии программного обеспечения компьютера стал объектно-ориентированный подход. В 1985 году лаборатория Bell Labs (США) сообщила о создании языка программирования C++ (СИ++). Этот язык является сегодня наиболее популярным среди языков объектно-ориентированного программирования. С его помощью возможно создание

программных приложений, ориентированных на любые машины – от персональных до суперкомпьютеров. Создателем языка является Бьорн Страуструп.

Представителем языков объектно-ориентированного программирования является и язык JAVA, созданный в 1995 году под руководством Джеймса Гослинга группой инженеров компании Sun Microsystems. При его разработке была поставлена цель - создать простой язык, не требующий специального изучения. Язык JAVA был разработан так, чтобы быть максимально похожим на C++. JAVA является идеальным инструментом при создании приложений для Интернета.



За последние два десятилетия множество языков программирования вышло в свет. Однако не все они одинаково хорошо прижились в мире разработки ПО. Компания TIOBE опубликовала рейтинг популярности языков программирования за 2019 год. Индекс популярности TIOBE не пытается найти самый лучший язык программирования по самому большому количеству написанных строк кода, а строит свои доводы по изменению интереса к языкам, на основе анализа статистики поисковых запросов в таких системах, как Google, Google Blogs, Yahoo!, Wikipedia, MSN, YouTube, Bing, Amazon и Baidu [3].

Согласно данному рейтингу, лидерами среди языков программирования остаются Java, C, Python и C++.

Данная работа посвящена истории создания и развития Java и C++, а также изменениям, которые претерпевали языки программирования в ходе движения

технологического прогресса.

1. Язык программирования JAVA

1.1 История возникновения и развития

Истоками появления история языка программирования Java восходит к 1991 году, когда группа инженеров из компании Sun Microsystems под руководством Патрика Нотона (Patrick Naughton) и члена Совета директоров (и разностороннего компьютерного «гения») Джеймса Гослинга (James Gosling) занялась разработкой компактного языка, который подходил бы для программирования бытовых устройств, например, контроллеров для переключения каналов кабельного телевидения (cable TV switchboxes). Ввиду того, что такие устройства не потребляют много энергии и имеют небольшие микросхемы памяти, главное требование к языку было продиктовано условием компактности для того, чтобы он мог бы генерировать очень малообъемные программы. Кроме того, поскольку разные производители могут выбирать разные центральные процессоры (Central Processor Unit – CPU), было важно не завязнуть в какой-то одной архитектуре компьютеров. Проект получил кодовое название "Green Project".

До появления языка Java программистам приходилось выбирать между языками, обеспечивающие только узкий спектр возможностей. Одни из первых языков программирования BASIC, COBOL и FORTRAN были созданы без учета принципов структурирования. Структурированный язык Pascal не был предназначен для написания разнообразных программ. Язык C++ частично решал эти проблемы. Но критическое неудобство применения C и C++ состоит в том, что написанные на них программы должны быть скомпилированы для конкретной платформы.

В поисках решения этих проблем, Джеймс Гослинг и другие разработчики начали работу над переносимым языком, не зависящим от конкретной платформы. Исходный код, написанный на таком языке, должен был выполняться на разнотипных процессорах и в различных операционных системах. И в итоге их усилия привели к созданию первого в мире кроссплатформенного языка под названием Java.

Платформа и язык Java начинались как внутренний проект в Sun Microsystems в декабре 1990 года, потенциально обеспечивая альтернативу языкам

программирования C и C++. В разработке и развитии нового языка приняли участие многие другие специалисты: Билл Джой (Bill Joy), Артур ван Хофф (Arthur van Hoff), Джонатан Пэйн (Jonathan Payne), Франк Йеллин (Frank Yellin) и Тим Линдхольм (Tlm Lindholm), которые внесли основной вклад в развитие исходного прототипа Java.

Изначально команда рассматривала вариант с использованием модифицированного языка C++ как исходника, но отклонила его по следующим причинам: требовательность к объемам памяти и высокая сложность. Отсутствие в C++ сборки мусора означало, что программистам приходилось вручную управлять системной памятью, сложной и подверженной ошибкам задачей. Тем более, что они разрабатывали встроенную систему с ограниченными ресурсами. Наконец, они хотели получить платформу, перенос которой на любые типы устройств совершенно не является трудностью. И все-таки синтаксис Java во многом заимствован из C и C ++, а объектно-ориентированные функции смоделированы на основе языков C++, Smalltalk and Objective-C [4].

Стремясь изобрести небольшой, компактный и кроссплатформенный код, разработчики возродили модель, что использовалась при реализации первых версий языка Pascal на заре эры персональных компьютеров. Никлаус Вирт, создатель языка Pascal, в свое время разработал машинонезависимый язык, генерирующий промежуточный код для некоей гипотетической машины. Этот язык стал коммерческим продуктом под названием UCSD Pascal. (Такие гипотетические машины часто называются виртуальными – например, виртуальная машина языка Java, или JVM). Такой промежуточный код можно выполнять на любой машине, имеющей соответствующий интерпретатор. Инженеры, работавшие над проектом "Green", также использовали виртуальную машину, что позволило решить их основную проблему.

Однако большинство сотрудников компании Sun Microsystems имели опыт работы с операционной системой UNIX, поэтому в основу разрабатываемого ими языка был положен язык C++, а не Pascal. В частности, они сделали язык объектно-, а не процедурно-ориентированным.

Как сказал Гослинг в своем интервью: «Язык – это всегда средство, а не цель». Разработкой нового языка вплотную занимался Джеймс Гослинг и изначально назвал его Oak (в переводе с английского Дуб). Рядом с офисом разработчика действительно рос дуб. Наверное, Джеймс, работая в офисе, поглядывал в окно и, видя крепкий дуб, излучающий здоровье и долголетие, решил назвать новый язык

в честь зеленого дерева. В 1994 году язык Oak был переименован из-за того, что название «Oak» уже использовалось как торговая марка компании Oak Technology (американская корпорация, поставщик электронных компонентов). Oak был переименован в Java, в честь одного из популярных в то время брендов кофе, которое получило название одноименного острова Ява. Потому на официальной эмблеме изображена чашка с кофе с поднимающейся из нее струйкой пара. Существует и другая версия происхождения названия языка: она связана с шутливым намеком на кофе-машину, как пример бытового устройства, для программирования которых изначально язык и был задуман. К лету 1992 года команда смогла продемонстрировать части новой платформы, включая операционную систему Green OS, язык Oak, библиотеки и аппаратное обеспечение. В полноценном варианте Java 1.0 была окончательно выпущена в 1996 году.

Их первая попытка, продемонстрированная 3 сентября 1992 года, была сосредоточена на создании устройства PDA (Personal Digital Assistant, карманный компьютер) под названием «Star7», которое имело графический интерфейс и смарт-агента под названием «Duke» для помощи пользователю. Карманный компьютер Star7 был принципиально новым устройством, которое намного опередило своё время. Это было инновационное устройство для чрезвычайно интеллектуального дистанционного управления. Оно имело мощность рабочей станции SPARK, помещаясь в коробочке размером 6x4x4 дюйма. К сожалению, из-за высокой стоимости оно не смогло произвести переворот в мире технологии и постепенно было забыто. Возможно, в настоящее время именно оно вернулось к нам в виде умных андроид устройств.

Затем разработчики стали заниматься созданием устройства для кабельного телевидения, которое способно было бы осуществлять новые виды услуг, например, включать видеосистему по запросу. Вновь они не заключили ни одного контракта. (Забавно, что одной из компаний, отказавшихся подписать с ними контракт, руководил Джим Кларк (Jim Clark) – основатель компании Netscape, впоследствии сделавшей очень много полезного для успеха языка Java.)

Весь 1993 год и половину следующего года продолжались бесплодные поиски покупателей продукции, разработанной в рамках проекта "Green" (под новым названием "First Person, Inc."). (Патрик Нотон, один из основателей группы, впоследствии в основном занимавшийся маркетингом, налетал в общей сложности более 300 тысяч миль, пытаясь продать разработанную технологию.) Проект "First Person, Inc." был закрыт в 1994 году.

1.2 Эволюция и развитие Java

Тем временем в рамках Интернет разрасталась сеть World Wide Web. Ключом к этой сети является браузер, превращающий гипертекст в изображение на экране.

В 1994 году большинство людей пользовалось броузером Mosaic, некоммерческим Web-браузером, разработанным в суперкомпьютерном центре Университета штата Иллинойс (University of Illinois) в 1993 году. (Частично этот браузер был написан Марком Андреессеном (Mark Andreessen) за 6,85 доллара в час. В то время Марк заканчивал университет и браузер являлся его дипломной работой. Уже после он стал одним из основателей и главным программистом компании Netscape, и к нему пришли слава и богатство.)

В своем интервью журналу Sun World Гослинг сказал, что в середине 1994 года разработчики языка поняли: «Нам нужно создать действительно крутой браузер. Такой браузер должен представлять собой одно из немногих приложений модной клиент-серверной технологии, в которой жизненно важным было бы именно то, что мы сделали: архитектурная независимость, выполнение в реальном времени, надежность, безопасность – вопросы, не являвшиеся чрезвычайно важными для рабочих станций. И мы создали такой браузер».

На самом деле браузер был разработан Патриком Нотоном и Джонатаном Пэйном (Johnatan Payne). Позже он превратился в современный браузер HotJava. Этот браузер был создан с помощью языка Java, чтобы продемонстрировать всю его мощь. Однако разработчики не забывали о мощных средствах, которые теперь называются апплетами, наделив свой браузер способностью выполнять код внутри Web-страниц. «Демонстрация технологии» была представлена на выставке Sun World '95 23 мая 1995 года и вызвала всеобщее помешательство на почве успешных достижений поставленных целей языка Java, продолжающееся и до сих пор, в какой-то мере.

Компания Sun Microsystems выпустила первую версию языка Java в начале 1996 года. Через несколько месяцев после нее появилась версия Java 1.02. Люди быстро поняли, что версия Java 1.02 не подходит для разработки серьезных приложений. Конечно, эту версию можно применять для разработки Web-страниц с пляшущими человечками, однако в версии Java 1.02 ничего нельзя даже напечатать [5].

Если говорить откровенно – версия Java 1.02 была еще довольно сырой. Ее преемница, версия Java 1.1, заполнила большинство зияющих пробелов, в разы улучшив возможность отражения и добавив новую модель событий для программирования графического пользовательского интерфейса. Несмотря на всё это, она всё еще была достаточно ограниченной.

Выпуск версии Java 1.2 стал основной новостью конференции JavaOne в 1998 году. В новой версии слабые средства для создания графического пользовательского интерфейса и графических приложений были заменены сложным и масштабным инструментарием. Это был шаг вперед, к реализации лозунга «Write Once, Run Anywhere»™ («Один раз напиши – и везде выполняй»), выдвинутого при разработке предыдущих версий.

В 1997 году большинство версий Java оставались бесплатными, за исключением Java Enterprise System. В этом же году вышла существенно обновленная версия – Java 2, а также выделились отдельные платформы: J2SE, J2EE, J2ME.

В декабре 1998 года через три дня (!) после выхода в свет название новой версии было изменено на громоздкое словосочетание Java 2 Standart Edition Software Development Kit Version 1.2 (Стандартное издание пакета инструментальных средств для разработки программного обеспечения на языке Java 2, версия 1.2).

Для более наглядного понимания этапности развития Java приведем ключевые события приводятся в хронологическом порядке:

Июнь 1991 г. – Гослинг начинает работу над интерпретатором Oak, который через несколько лет (при поисках торговой марки) переименован в Java.

19 августа 1991 г. – Коллектив разработчиков Green демонстрирует идеи базового пользовательского интерфейса и графическую систему сооснователям компании Sun Скотту Макнили и Биллу Джою.

17 октября 1991 г. – Шеридан и Нотон присваивают конструкторской философии своего коллектива девиз «1st Person», который со временем становится названием компании.

17 ноября 1991 г. – Офис проекта Green снова подключается к главной сети компании Sun линией на 56 Кбит/с

1 марта 1992 г. – К проекту Green присоединяется Джонатан Пейн, который позднее участвует в написании HotJava.

Лето 1992 г. – Интенсивная деятельность по доработке Oak, Green OS, пользовательского интерфейса, аппаратуры Star7 и соответствующих компонентов.

4 сентября 1992 г. – Завершена разработка устройства Star7; оно продемонстрировано Джою и Макнили.

1 октября 1992 г. – Из компании SunLabs переходит Уэйн Розинг, принимающий на себя руководство коллективом.

1 ноября 1992 г. – Организована корпорация FirstPerson.

15 января 1993 г. – Коллектив переезжает в Пало Альто в здание, где раньше находилась лаборатория Western Research Lab компании DEC и была основана исходная группа Hamilton Group (она же OSF).

15 марта 1993 г. – После ознакомления с результатами испытаний кабельного интерактивного телевидения, проведенных компанией Time Warner, корпорация FirstPerson сосредотачивается на этой тематике.

Апрель 1993 г. – Выпуск первого графического браузера для Internet – Mosaic 1.0, разработанного в центре NCSA.

14 июня 1993 г. – Компания Time Warner продолжает проводить свои испытания интерактивного кабельного ТВ с компанией SGI, несмотря на признанное превосходство технологии компании Sun и уверения, что Sun выиграла эту сделку.

Лето 1993 г. – Нотон пролетает 300 тысяч миль, продавая Oak всем, занимающимся бытовой электроникой и интерактивным телевидением; тем временем темп, с которой люди получают доступ к Internet, головокружительно нарастает.

Август 1993 г. – Через несколько месяцев многообещающих переговоров с компанией 3DO относительно разработки ОС для приставок, президент 3DO Трип Хокинс предлагает купить технологию. Макнили отказывается, и сделка срывается.

Сентябрь 1993 г. – К коллективу присоединяется Артур Ван Хофф, поначалу – чтобы создать среду разработки приложений, предназначенных для интерактивного телевидения, а потом разрабатывающий, главным образом, сам язык.

7 декабря 1993 г. – Экспертиза операций на высоком уровне в FirstPerson обнаруживает, что эта группа не имеет реальных партнеров или маркетинговой стратегии и неясно представляет себе дату выпуска.

8 февраля 1994 г. – Отменено публичное заявление компании FirstPerson о выпуске, которое должно было состояться на конференции Technology, Entertainment and Design (TED).

17 февраля 1994 г. – Исполнительным лицам компании Sun для разносторонней экспертизы представлен альтернативный бизнес-план корпорации FirstPerson по разработке мультимедийной платформы для CD-ROM и онлайновой работы.

25 апреля 1994 г. – Создана компания Sun Interactive; в нее переходит половина сотрудников FirstPerson.

Июнь 1994 г. – Начат проект Liveoak, нацеленный Биллом Джоем на использование Oak в крупном проекте небольшой операционной системы.

Июль 1994 г. – Нотон ограничивает область применения проекта Liveoak, просто переориентировав Oak на Internet.

16 сентября 1994 г. – Пейн и Нотон начинают писать WebRunner – браузер-аналог Mosaic, позднее переименованный в HotJava.

29 сентября 1994 г. – Прототип HotJava впервые продемонстрирован исполнительным лицам компании Sun.

11 октября 1994 г. – Нотон уходит в компанию Starwave.

Осень 1994 г. – Ван Хофф реализует компилятор Java на языке Java. Ранее Гослинг реализовывал его на языке C.

23 мая 1995 г. – Компания Sun официально представляет Java и HotJava на выставке SunWorld '95.

23 мая 1995 г. – Netscape объявляет о намерении использовать Java при разработке браузера Netscape.

21 сентября, 1995 г. – В Нью-Йорке проходит конференция по Java-разработке

25 Сентября, 1995 г. – Sun объявляет о расширенном сотрудничестве с Toshiba.

26 Сентября, 1995 г. – Sun анонсирует пакет инструментов для разработки с использованием Java-технологий.

30 Октября, 1995 г. – На Internet World Conference в Бостоне компании Lotus Development Corp., Intuit Inc., Borland International Inc., Macromedia Inc. и Spyglass Inc. объявляют о намерении лицензировать Java.

4 декабря, 1995 г. – Sun, Netscape и Silicon Graphics создают альянс для разработки инструментария для интернета.

4 декабря, 1995 г. – Borland, Mitsubishi Electronics, Sybase и Symatec объявляют о планах лицензировать Java.

6 декабря, 1995 г. – IBM и Adobe объявляют о планах лицензировать Java.

7 декабря, 1995 г. – Microsoft объявляет о планах лицензировать Java.

23 января, 1996 г. – релиз JDK 1.0

В 1997 году большинство версий Java оставались бесплатными, за исключением Java Enterprise System. В этом же году вышла существенно обновленная версия – Java 2, а также выделились отдельные платформы: J2SE, J2EE, J2ME.

- J2SE 1.3 (Май 8, 2000)
- J2SE 1.4 (Февраль 6, 2002)
- J2SE 5.0 (Сентябрь 30, 2004)
- Java SE 6 (Декабрь 11, 2006)
- Java SE 7 (Июль 28, 2011)
- Java SE 8 (Март 18, 2014)

В ноябре 2006 года Sun объявила часть кода виртуальной машины Java (JVM) свободном распространяемым и начала выпускать его под лицензией GNU General Public License (GPL). К маю 2007 года компания распространяла бесплатно почти весь код JVM, за исключением малой его части, на которую Sun не имеет прав.

В 2009-10 годах корпорация Oracle поглотила компанию Sun Microsystems. В апреле 2010 Джеймс Гослинг, перешедший в Oracle после поглощения, покинул компанию.

Кроме стандартного издания пакета ("Standart Edition") были предложены еще два варианта: "микроиздание" ("Micro Edition") для портативных устройств, например, для мобильных телефонов, и. "промышленное издание" ("Enterprise Edition") для создания серверных приложений. В нашей книге в центре внимания находится стандартное издание.

Версии 1.3 и 1.4 стандартного издания пакета инструментальных средств намного совершеннее первоначального выпуска языка Java 2. Они обладают расширенными возможностями, включая новые, и, разумеется, содержат намного меньше ошибок. В таблице 1. ниже показан стремительный рост объема библиотеки API по мере появления новых версий стандартного издания пакета SDK.

Таблица 1. Рост объема библиотеки API из пакета Java Standart Edition

Версия	**Год выпуска**	**Новые языковые средства**	**Количество классов и интерфейсов**
1.0	1996	Выпуск самого языка	211
1.1	1997	Внутренние классы	477
1.2	1998	Отсутствуют	1524
1.3	2000	Отсутствуют	1840
1.4	2002	Утверждения	2723
5.0	2004	Обобщенные классы, цикл в стиле for each, автоупаковка, аргументы переменной длины, метаданные, перечисления, статический импорт	3279
6	2006	Отсутствуют	3793
7	2011	Оператор switch со строковыми метками ветвей, ромбовидный оператор, двоичные литералы, усовершенствованная обработка исключений	4024

8	2014	Лямбда-выражения, библиотеки потоков и даты/времени, интерфейсы с методами по умолчанию	4240
9	2017	Литералы в коллекциях, оператор Элвиса, Class Optional, Streams, IO, Regrexp, обработка процессов ProcessHandle	более 4500

С 2018 года коренным образом изменилась парадигма разработки и выпуска релизов Java. Java всегда была известна медленным темпом выхода новых версий. В среднем новые релизы Java выходили приблизительно раз в 3 года: Java 7 вышла в 2011 году, Java 8 – в 2014, Java 9 – в 2017. При этом релизы были очень крупными, например в Java 9 вошло аж 99 JEP'ов [6]. Постепенно Oracle понял, что в современном мире такой медленный темп выхода может навредить развитию и продвижению Java, поэтому принял решение перейти к 6-месячному релизному циклу. Таким образом, выход новых улучшения, которые могут быть выпущены прямо сейчас, не будет задерживаться из-за других нововведений, более трудоемких к реализации.

1.3. Сферы применения языка Java

На сегодняшний день существует большое разнообразие областей применения языка программирования Java, от сайтов электронной коммерции до Android приложений, от научных до финансовых приложений, таких как трейдинговые системы, и игр, типа Minecraft, до настольных программных средств, таких как Eclipse, Netbeans и IntelliJ, от open source фреймворков до J2ME приложений и т.д. Узнаем поподробнее о каждом из них.

Java крайне обширно применяется в финансовой сфере. Очень многие мировые инвестиционные банки, типа Goldman Sachs, Citigroup, Barclays, Standard Charted и другие используют Java для написания front-end и back-end офисных электронных систем, систем регулирования и конфирмации, систем обработки данных и некоторых других. Преимущественно Java используется при написании серверных приложений, в общей массе своей без какого-либо пользовательского интерфейса, которые получают данные с одного сервера, обрабатывают их и отправляют

дальше. Java Swing был также популярен для создания «толстоклиентных» интерфейсов, но сейчас C# более успешен, а значит и востребован на рынке в этой области, а Swing уже выдыхается.

Еще один яркий пример успешного использования Java – операционная система Android. Взглянем на смартфон или другое устройство на Android: абсолютно все приложения написаны на Java, с использованием Google и Android API, которые очень схожи с JDK. Несколько лет назад Android предоставил необходимые возможности, благодаря чему сегодня многие Java программисты – Android разработчики. Кстати, Android использует другую JVM и другой способ компоновки, но код всё ещё написан на Java [5,7,8].

Также Java активно используется в электронной коммерции и в области веб-приложений. Огромное количество RESTful сервисов было разработано с использованием Spring MVC, Struts 2.0 и похожих фреймворков. Даже простейшие приложения, основанные на Servlet, JSP и Struts, в определенной мере популярны в различных государственных проектах. Многие веб-приложения государственных, оздоровительных, страховых, образовательных, оборонительных и некоторых других отделений написаны на Java, что в очередной раз доказывает востребованность и универсальность этого языка среди других.

Много полезных и достойных программных средств и средств разработки написаны и разработаны на Java, например Eclipse, IntelliJ Idea и Netbeans IDE. К примеру, Eclipse представляет собой свободную интегрированную среду разработки модульных кроссплатформенных приложений. Она развивается и поддерживается компанией «Eclipse Foundation». Самые известные из приложений на основе Eclipse Platform – разнообразные «Eclipse IDE» для разработки программного обеспечения на различных языках программирования. По сути, это «инструментарий для разработки инструментариев». Поскольку она не является набором API, инфраструктура Eclipse будет состоять из реального кода, созданного для исполнения реальных задач [7].

Отдельно внимание стоит обратить на Java To Micro Edition. Несмотря на то, что появление iOS и Android практически уничтожило J2ME рынок, в мире ещё огромное количество дешёвых телефонов от Nokia и Samsung, использующих J2ME. Было время, когда практически все игры и приложения, доступные на Android, были написаны с использованием MIDP и CLDC, которые являются частью платформы J2ME. Все, кто использовал кнопочные мобильные телефоны с 2000-ых по 2010-е годы, вкушали плоды активного использования программистами этого языка. J2ME

всё ещё популярен в таких средствах, как Blu-ray, карточки и телевизионные приставки (конечно, в гораздо менее широком смысле, чем те же 10 лет назад). Кстати, одна из причин такой популярности месенджера «WhatsApp» – он также доступен на Java To Micro Edition.

В наше время очень популярен трейдинг – неважно, криптовалюты или ценные бумаги, локальные местные банки или крупнейшие финансовые концерны – всякая идущая в ногу с современными тенденциями организация имеет собственное мобильное приложение для удобства клиентов – трейдинговая платформа. И, да, чаще всего она написана на Java. Также те самые популярные банковские приложения, типа Murex, тоже созданы с помощью Java.

Hadoop и другие технологии обработки больших данных так или иначе используют Java, например Hbase и Accumulo от Apache, или ElasticSearch. Хоть Java и не доминирует в этой области, поскольку существуют такие технологии, как MongoDB, которые написаны на C++.

Обширна Java и в области встраиваемых систем. Для того, чтобы увидеть на что способна платформа, вам нужно всего 130 KB для использования Java (на смарт-картах и сенсорах). Изначально Java разрабатывалась для встраиваемых систем. В действительности эта область была частью начальной кампании Java «пиши один раз, запускай, где угодно» и похоже, что она приносит свои плоды. Один из немногих языков с конкретной целью в разработке, что не уходил в сторону с намеченного пути и, более того, пришел к этой цели [9].

Java в целом улучшила свои эксплуатационные показатели и с современными JIT-ами она может выдавать производительность на уровне C++. Это одна из причин, почему Java так популярна и при написании высокопроизводительных систем. Хоть производительность и проигрывает в сравнении с родным языком, но вы жертвуете безопасностью, мобильностью и надёжностью ради большей скорости исполнения. Это, безусловно, было бы оправданно, если бы не одно «но»: требуется всего один неопытный C++ программист, чтобы сделать приложение медленным и ненадёжным, что в конечном итоге сведет все жертвы и компромиссы на нет.

Таким образом очевидно, что сфер применения у языка Java превеликое множество и в каждом из них он по-своему преуспевает. Исходя из сравнения с другими языками, в частности C++, статистики и степени разработки и поддержки можно с уверенностью прочить Java дальнейшего роста и развития, что не может не радовать нынешних программистов на Java.

2. Язык программирования C++

2.1 Краткая характеристика

C++ – язык общего назначения и, по словам создателя, задуман для того, чтобы настоящие программисты получили удовольствие от самого процесса программирования. За исключением второстепенных деталей он содержит язык C как подмножество. Язык C расширяется введением гибких и эффективных средств, предназначенных для построения новых типов. Программист структурирует свою задачу, определив новые типы, которые точно соответствуют понятиям предметной области задачи. Такой метод построения программы обычно называют абстракцией данных. Информация о типах содержится в некоторых объектах типов, определенных пользователем. С такими объектами можно работать надежно и просто даже в тех случаях, когда их тип нельзя установить на стадии трансляции. Программирование с использованием таких объектов обычно называют объектно-ориентированным. Если этот метод применяется правильно, то программы становятся короче и понятнее, а сопровождение их упрощается.

Ключевым понятием C++ является класс. Класс – это определяемый пользователем тип. Классы обеспечивают сокрытие данных, их инициализацию, неявное преобразование пользовательских типов, динамическое задание типов, контролируемое пользователем управление памятью и средства для перегрузки операций. Методы класса – это функции, которые смогут применяться к экземплярам класса, иными словами – функция, объявленная внутри класса и предназначенная для работы с его объектами [10]. Методы объявляются в теле класса. Описываться могут там же, но могут и за пределами класса (внутри класса в таком случае достаточно представить прототип метода, а за пределами класса определять метод поставив перед его именем – имя класса и оператор ::). Методы и поля входящие в состав класса называются членами класса. При этом методы часто называют функциями-членами класса.

В языке C++ концепции контроля типов и модульного построения программ реализованы более полно, чем в C. Кроме того, C++ содержит усовершенствования, прямо с классами не связанные: символические константы, функции-подстановки, стандартные значения параметров функций, перегрузка имен функций, операции управления свободной памятью и ссылочный тип [10]. В

C++ сохранены все возможности C эффективной работы с основными объектами, отражающими аппаратную «реальность» (разряды, байты, слова, адреса и т.д.). Это позволяет достаточно эффективно реализовывать пользовательские типы.

2.2 История создания

Язык программирования C++ был создан в начале 1980-х годов, его создатель – Бьёрн Страуструп.

Бьерн Страуструп является разработчиком языка C++ и создателем первого транслятора. Он – сотрудник научно-исследовательского вычислительного центра AT&T Bell Laboratories в Мюррей Хилл (Нью-Джерси, США).Он получил звание магистра математики и вычислительной техники в университете г. Аарус (Дания), а докторское звание по вычислительной технике в Кэмбриджском университете (Англия).Он специализируется в области распределенных систем, операционных систем, моделирования и программирования. Вместе с М.А. Эллисон является автором полного руководства по языку C++ – «Руководство по C++ с примечаниями». Стоит отметить, что у языка C++ отсутствует правообладатель – язык является свободным. Однако сам документ стандарта языка (за исключением черновиков) не доступен бесплатно.

Изначально создания языка программирования C++ не планировалось, а Бъерн Страуструп лишь придумал ряд усовершенствований к существующему языку программирования C для собственных нужд. Когда в конце 1970-х годов Страуструп начал работать в Bell Labs над задачами теории очередей (в приложении к моделированию телефонных вызовов), он обнаружил, что попытки применения существующих в то время языков моделирования оказываются неэффективными, а применение высокоэффективных машинных языков слишком сложно из-за их ограниченной выразительности. Так, язык Симула имеет такие возможности, которые были бы очень полезны для разработки большого программного обеспечения, но работает слишком медленно, а язык BCPL достаточно быстр, но слишком близок к языкам низкого уровня и не подходит для разработки большого программного обеспечения [11]. Ранние версии языка C++, известные под именем «C with classes» («Си с классами»), начали появляться с 1980 года. Вспомнив опыт своей диссертации, Страуструп решил дополнить язык C (преемник BCPL) возможностями, имеющимися в языке Симула. Язык C, будучи базовым языком системы UNIX, на которой работали компьютеры фирмы Bell,

является быстрым, многофункциональным и переносимым. Страуструп решил дополнить язык C (преемник BCPL) возможностями, имеющимися в языке Симула, и добавил к нему возможность работы с классами и объектами, тем самым зародил предпосылки нового, основанного на синтаксисе C, языка программирования. В первую очередь в C были добавлены классы (с инкапсуляцией), наследование классов, строгая проверка типов, inline-функции и аргументы по умолчанию. В результате практические задачи моделирования оказались доступными для решения как с точки зрения времени разработки (благодаря использованию Симула-подобных классов), так и с точки зрения времени вычислений (благодаря быстродействию C). Синтаксис C++ был основан на синтаксисе C, так как Бьёрн Страуструп стремился сохранить совместимость с языком C.

Разрабатывая C с классами, Страуструп написал программу cfront[en] – транслятор, перерабатывающий исходный код C с классами в исходный код простого C. Это позволило работать над новым языком и использовать его на практике, применяя уже имеющуюся в UNIX инфраструктуру для разработки на C. Новый язык, неожиданно для автора, приобрёл большую популярность среди коллег и вскоре Страуструп уже не мог лично поддерживать его, отвечая на тысячи вопросов. К 1983 году в язык были добавлены новые возможности, такие как виртуальные функции, перегрузка функций и операторов, ссылки, константы, пользовательский контроль над управлением свободной памятью, улучшенная проверка типов и новый стиль комментариев (//). Получившийся язык уже перестал быть просто дополненной версией классического C и был переименован из C с классами в «C++». Его первый коммерческий выпуск состоялся в октябре 1985 года. Имя языка, получившееся в итоге, происходит от оператора унарного постфиксного инкремента C ++ (увеличение значения переменной на единицу) и общего именования через «+», чтобы указать на расширенные возможности программы компьютера. Проще говоря, знак плюса означает усовершенствование программы и придание ей нового функционала. Название языка принадлежит Рику Маскитти (Rick Mascitti) и впервые было использовано в декабре 1983 года. По Страуструпу: «Это имя означает эволюционный характер изменения из Си». Хотя большинство C кода действительно для C++, но C не образует подмножество C++.

Некоторые программисты C отметили, что если объявить x=3; и y=x++; то при выполнении x= 4, а y = 3; так как x увеличивается после того, как его значение присваивается y. Однако, если написать y=++x; то y=4 и x=4. После таких рассуждений, более подходящее название для C++ может быть фактически ++C. Однако, C++ и ++C это увеличение C, поэтому форма C++ является более

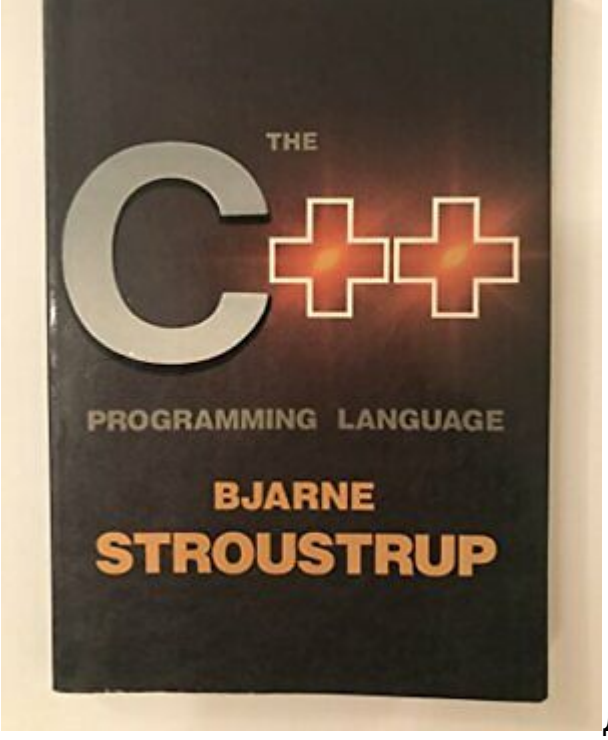
распространенной, чем ++C. Педанты могут отметить, что после введения C++, язык C сам себя не изменил и наиболее точное название может быть «C +1».

При создании C++ Бьёрном Страуструпом преследовались следующие цели:

- создание универсального языка со статическими типами данных, эффективностью и переносимостью языка C;
- всесторонняя поддержка множества стилей программирования, в том числе обобщённого программирования, абстракции данных, процедурного и объектно-ориентированного программирования;
- возможность предоставить программисту свободу выбора, даже при условии, что этот выбор будет неверным;
- сохранение максимальной совместимости с C, таким образом обеспечивая легкость перехода от исходной системы программирования;
- недопущение разночтений между C и C++ – если конструкция допустима, то в каждом из языков она имеет идентичный смысл и обеспечивает равный результат в поведении программы;
- максимальная универсальность, избегание особенностей, зависящих от платформы;
- недопущение падения производительности программы при использовании языка;

Несмотря на ряд известных недостатков языка C, Страуструп пошёл на его использование в качестве основы, так как «в C есть свои проблемы, но их имел бы и разработанный с нуля язык, а проблемы C нам известны». Кроме того, использование готового языка дало возможность быстро получить прототип компилятора (cfront), который лишь реализовывал трансляцию нововведенных синтаксических элементов в оригинальный язык C.

По мере разработки C++ в него были включены другие средства, которые перекрывали возможности конструкций C, в связи с чем неоднократно поднимался вопрос об отказе от совместимости языков путём удаления устаревших конструкций. Тем не менее, совместимость была сохранена по следующим причинам: был сохранен действующий код, написанный на C, а также ради исключения путаницы между языками, то есть различия, имеющиеся в них, требовались быть сведенными к минимуму. Кроме того, при данном раскладе отсутствовала необходимость переучивания программистов, ранее изучавших C [11,12].



До начала официальной стандартизации язык развивался в основном силами Страуструпа в ответ на запросы программистского сообщества. Функцию стандартных описаний языка выполняли написанные Страуструпом печатные работы по C++ (описание языка, справочное руководство и так далее). В 1985 году вышло первое издание «Язык программирования C++», обеспечивающее первое описание этого языка, что было чрезвычайно важно из-за отсутствия официального стандарта.

Бьерн Страуструп: «Я придумал C++, записал его первоначальное определение и выполнил первую реализацию. Я выбрал и сформулировал критерии проектирования C++, разработал его основные возможности и отвечал за судьбу предложений по расширению языка в комитете по стандартизации C++", - пишет автор самого популярного языка программирования. - "Язык C++ многим обязан языку C, и язык C остается подмножеством языка C++ (но в C++ устранены несколько серьезных брешей системы типов C). Я также сохранил средства C, которые являются достаточно низкоуровневыми, чтобы справляться с самыми критическими системными задачами. Язык C, в свою очередь многим обязан своему предшественнику, BCPL; кстати, стиль комментариев // был взят в C++ из BCPL. Другим основным источником вдохновения был язык Simula-67. Концепция классов (с производными классами и виртуальными функциями) была позаимствована из него. Средства перегрузки операторов и возможность помещения объявлений в любом месте, где может быть записана инструкция, напоминает Algol 68.»

В 1989 году вышла версия 2.0 языка C++. Его новые возможности включали множественное наследование, абстрактные классы, статические функции-члены, функции-константы и защищённые члены. В 1990 году вышло «Комментированное справочное руководство по C++», положенное впоследствии в основу стандарта. Последние обновления включали шаблоны, исключения, пространства имён, новые способы приведения типов и логический тип [11].

После первого издания книги запросы пользователей определили развитие C++. Авторов направлял опыт широкого круга пользователей, работающих в разных областях программирования. За шесть лет, прошедших от момента выхода в свет первого издания описания C++, число пользователей возросло в сотни раз. За эти годы были усвоены многие уроки, были предложены и подтвердили практикой свое право на существование различные приемы программирования. Внедренные за это время расширения языка прежде всего были направлены на повышение выразительности C++ как языка абстракции данных и объектно-ориентированного программирования вообще и как средства для создания высококачественных библиотек с пользовательскими типами данных в частности. Библиотекой высокого качества Бьёрн Страуструп называл библиотеку, позволяющую пользователю определять с помощью классов понятия, работа с которыми сочетает удобство, эффективность и надежность. Под надежностью понимается то, что класс предоставляет защищенный по типам интерфейс между пользователями библиотеки и ее разработчиками. Эффективность предполагает, что использование классов не влечет за собой больших накладных расходов по памяти или времени по сравнению с «ручными» программами на C.

В связи с лавинообразным процессом увеличения числа пользователей C++ стало очевидно, что работа по стандартизации C++ неизбежна и что следует незамедлительно приступить к созданию основы для нее.

2.3 История стандартизации

В 1998 году был опубликован стандарт языка ISO/IEC 14882:1998 (известный как C++98), разработанный комитетом по стандартизации C++ (ISO/IEC JTC1/SC22/WG21 working group). Стандарт состоит из двух частей – основы языка (core language) и стандартной библиотеки языка, которая включает Standard Template Library (STL) и модифицированный вариант стандартной библиотеки языка C.

В 2003 году был опубликован стандарт языка ISO/IEC 14882:2003, где были исправлены выявленные ошибки и недочёты предыдущей версии стандарта.

В 2005 году был выпущен отчёт Library Technical Report 1. Отчёт описывает расширения стандартной библиотеки, которые, должны быть включены в следующую версию языка C++.

С 2009 года велась работа по обновлению предыдущего стандарта, предварительной версией нового стандарта сперва был C++09, а спустя год C++0x, сегодня – C++11, куда были включены дополнения в ядро языка и расширение стандартной библиотеки.

В 2011 году в стандарт языка C++ были внесены изменения, упрощающие написание программ, добавлены параллелизм и лямбда-выражения, которые стали необходимостью в современных языках программирования.

В 2011 году был принят также новый стандарт языка C, в котором также добавлен параллелизм. Стоит отметить, что некоторые кардинальные изменения в языке C, которые появились еще в стандарте 1999 года, не нашли отражения в новом стандарте C++ (например, массивы переменного размера, указание имени поля или номера элемента массива при инициализации, комплексный тип). Напротив, в стандарт C99 добавлены возможности из C++ или их аналоги, позволяющие минимизировать изменения при копировании кода (например, объявление переменных в любом месте программы, макросы, генерирующие вызов нужной функции в зависимости от типа аргумента).

В 2014 году был разработан очередной стандарт C++, который немного увеличил удобство языка (см. константы), добавил некоторые обобщения (см. лямбда-выражения здесь и тут) к революционным изменениям, которые были сделаны в стандарте 2011 года.

В стандарт 2017 было решено добавить параллельные версии алгоритмов, работу с файловой системой, дополнительные атрибуты для управления предупреждениями при компиляции. К сожалению, в новый стандарт не попало предложение Б.Страуструпа и Г.Саттера об унификации вызовов методов и функций: можно писать как x.size(), так и size(x).

В текущем году анонсировано появление C++20 – неофициальное название стандарта ISO/IEC языка программирования C++, который ожидается после после C++17 [13].

Хотя язык программирования C++ справедливо называют продолжением C и любая работоспособная программа на языке C будет поддерживаться компилятором C++, при переходе от C к C++ был сделан весьма существенный скачок. Язык C++ выигрывал от своего родства с языком C в течение многих лет, поскольку многие программисты обнаружили, что для того, чтобы в полной мере воспользоваться преимуществами языка C++, им нужно отказаться от некоторых своих прежних знаний и приобрести новые, а именно: изучить новый способ концептуальности и решения проблем программирования. Перед тем как начинать осваивать C++, Страуструп и большинство других программистов, использующих C++ считают изучение языка C необязательным [14].

Заключение

Язык программирования решает две взаимосвязанные задачи: позволяет программисту записать подлежащие выполнению действия и формирует понятия, которыми программист оперирует, размышляя о своей задаче. Первой цели идеально отвечает язык, который очень "близок машине". Тогда со всеми ее основными "сущностями" можно просто и эффективно работать на этом языке, причем делая это очевидным для программиста способом. Второй цели идеально отвечает язык, который настолько "близок к поставленной задаче", что на нем непосредственно и точно выражаются понятия, используемые в решении задачи.

Язык Java является объектно-ориентированным и поставляется с достаточно объемной библиотекой классов. Библиотеки классов Java значительно упрощают разработку приложений, предоставляя в распоряжение программиста мощные средства решения распространенных задач. Поэтому программист может больше внимания уделить решению прикладных задач, а не таких, как, например, организация динамических массивов, взаимодействие с операционной системой или реализация элементов пользовательского интерфейса.

C++ является языком программирования общего назначения. Естественная для него область применения – системное программирование, понимаемое в широком смысле этого слова. Кроме того, C++ успешно используется во многих областях приложения, далеко выходящих за указанные рамки. Реализации C++ теперь есть на всех машинах, начиная с самых скромных микрокомпьютеров – до самых больших супер-ЭВМ, и практически для всех операционных систем.

Связь между языком, на котором мы думаем и программируем, а также между задачами и их решениями, которые можно представить в своем воображении, довольно близка. По этой причине ограничивать возможности языка только поиском ошибок программиста - в лучшем случае опасно. Как и в случае естественных языков, очень полезно обладать, по крайней мере, двуязычием. Язык предоставляет программисту некоторые понятия в виде языковых инструментов; если они не подходят для задачи, их просто игнорируют. Например, если существенно ограничить понятие указателя, то программист будет вынужден для создания структур, указателей и т.п. использовать вектора и операции с целыми. Хороший проект программы и отсутствие в ней ошибок нельзя гарантировать только наличием или отсутствием определенных возможностей в языке.

В качестве итога можно сказать, что в наше время в современном мире правила развития и поддержки наиболее популярных и успешных языков программирования определяются потребностями программистов и разработчиков. Это значительно упрощает понимание стоящих задач и пути их решения. И несмотря на столь стремительное и быстрое (в сравнении с остальными эволюционными прорывами в различных сферах) развитие достигнутые результаты позволяют эффективно и без длительных поисков решений заниматься прикладным программированием, что крайне важно в условиях бурного развития информационных технологий.

Список используемой литературы

1. Шабаев М.Б., Магомедов И.А. Популярные языки программирования // Тенденции развития науки и образования. - 2019. - №56-3. - С. 39-41.
2. Computer Languages History URL: https://www.levenez.com/lang/ (дата обращения: 14.08.2020).
3. TIOBE Index for August 2020 URL: https://www.tiobe.com/tiobe-index/ (дата обращения: 14.08.2020).
4. Осокин М.С. Язык JAVA как один из лучших языков программирования // Конкурентоспособность территорий. - Екатеринбург: Уральский государственный экономический университет, 2017. - С. 54-59.
5. Шилдт Г. JAVA. Полное руководство. - 10 изд. - М.: Диалектика, 2018. - 1488 с.
6. Шаран К. JAVA 9. Полный обзор нововведений. - 10 изд. - М.: «ДМК Пресс, 2018. - 544 с.
7. V. Denisov. Overview of Java application configuration frameworks. International Journal of Open Information Technologies ISSN: 2307-8162, 1 (6), 2013. URL:

http://injoit.org/index.php/j1/article/view/33 (дата обращения: 14.08.2020)
8. V. Denisov. Functional requirements for a modern application configuration framework. International Journal of Open Information Technologies ISSN: 2307-8162 3(10), 2015. URL: http://injoit.org/index.php/j1/article/view/236
9. Денисов В.С. Управление настройками JAVA-приложений с использованием механизма аннотаций // Современные информационные технологии и ИТ-образование. - 2015. - №11(2). - С. 37-40.
10. Цыдыпова Е.Г., Юданова В.В. Сравнительный анализ возможностей языков программирования семейства С // Наука и инновации В XXI Веке: Актуальные вопросы, открытия и достижения. - Пенза: "Наука и Просвещение", 2017. - С. 43-45.
11. Страуструп Б. Дизайн и эволюция C++. Второе издание. - М.: ДМК Пресс, 2016. - 448 с.
12. Страуструп Б. Язык программирования C++. Специальное издание. - М.: Бином, 2015. - 1136 с.
13. Working Draft, Standard for Programming Language C ++ // open-std.org URL: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2017/n4713.pdf (дата обращения: 14.08.2020).
14. Prinz P., Kirch-Prinz U. A complete guide to programming in C++ . - Sudbury, MA: Jones & Bartlett Learning, 2002. - 825 с.