Муниципальное бюджетное общеобразовательное учреждение
«Школа № 13» города Сарова

Востребованные языки программирования в настоящее время

Тип проекта: информационный

Автор проекта: Терешкина Вероника Сергеевна, учащаяся 10А класса МБОУ Школы №13

Руководитель проекта: Гурина Наталия Александровна, учитель информатики МБОУ Школы № 13

Саров

2022

Содержание

Введение……3

Глава 1. Понятие о языках программированиях……5

1.1. Языки программирования: основные требования и компоненты ……..5

1.2. Компиляторы и интерпретаторы……..7

1.3. Классификация и обзор языков программирования……. 10

Глава 2. Востребованные языки программирования в настоящее время……14

2.1. Python……15

2.2. Java ……20

2.3. C……22

2.4. C++……25

2.5 C#……30

2.6 JavaSkript……33

Заключение……37

Список литературы……38

Введение

Программирование настолько глубоко вошло в быт, что люди престали замечать, как изменилась их жизнь. Сотни тысяч привычных вещей не существовали бы без программирования или были бы гораздо менее удобными в использовании. Привычные бытовые приборы: микроволновая печь, стиральная машина – работают благодаря заложенным в них программам. Всего 50 лет назад невозможно было представить, как легко будет найти любую необходимую информацию, насколько экономнее станет использоваться время, затрачиваемое на решение некоторых задач. Сейчас же программ написано такое множество, что в их разнообразии трудно ориентироваться.

Компьютерные программы не могут существовать без определённого программирования – своеобразного языка между человеком и компьютером. Ныне языки программирования развиваются очень быстро, и их количество постоянно увеличивается. А ведь не так давно процесс работы компьютера заключался в выполнении команд, состоящих из комбинаций нулей и единиц. И чтобы задать машине конкретное действие, нужно было написать тысячи команд, соответственно, потратив на это много сил и времени. Когда же программисты пришли к тому, что намного проще создать алгоритм действий на более понятном им языке, а работу по его переводу в двоичные коды поручить компьютеру, возникли языки программирования.

Язык программирования - формальная знаковая система, предназначенная для записи компьютерных программ. Язык программирования определяет набор лексических, синтаксических и семантических правил, задающих внешний вид программы и действия, которые выполнит исполнитель (компьютер) под ее управлением.

Программирование помогает реализовать много сложных операций, позволяя компьютеру решать крайне сложные задачи, в том числе и выполнять защиту ПК от всевозможных вирусов с помощью bitdefender, antivirus. Ныне количество языков программирования стремительно возрастает. Множество из них соответствуют специальным международным стандартам; одни языки применяются только узким кругом людей, например, самими разработчиками; другие становятся популярными среди миллионов пользователей.

Возникает вопрос, какие же языки программирования востребованы в 2022 году.

Целью моей работы является выявление востребованных языков программирования в настоящее время

Для достижения поставленной цели были поставлены следующие **задачи**:

1. Изучение источников информации по теме: «Востребованные языки программирования в настоящее время» .
2. Анализ полученных результатов.
3. Подведение итогов по данной теме.

Объектом исследования являются языки программирования

Предметом исследования являются востребованные языки программирования.

Результат проекта (продукт): проект.

Глава 1. Понятие о языках программирования.

1.1. Языки программирования: основные требования и компоненты.

Язык программирования — это формальный язык, предназначенный для записи компьютерных программ. Языки программирования являются искусственными языками. От естественных языков они отличаются ограниченным числом “слов” и очень строгими правилами записи команд (операторов). Поэтому при применении их по назначению они не допускают свободного толкования выражений, характерного для естественного языка. Программирование основывается на использовании языков программирования, на которых записываются исходные тексты программ.

Компьютерный код — тот же иностранный язык, только он позволяет разговаривать с компьютером, ставить ему задачи и контролировать их выполнение.

Компьютерная программа – это набор инструкций, следуя которым компьютер выполняет поставленные задачи.

Программировать, значит писать для компьютера пошаговые инструкции, объясняющие, что и как ему нужно делать. Компьютеры могут казаться очень умными, но это всего лишь напичканные электроникой ящики, которые умеют очень быстро и точно выполнять инструкции. Мы разумные существа, можем давать компьютерам задачи, описывая их в виде программ, то есть пошаговых инструкций.

По выражению одного из основателя языков программирования Никлауса Вирта : **«Программы = алгоритмы + структуры данных».**

Можно сформулировать ряд требований к языкам программирования и классифицировать языки по их особенностям.

Основные требования, предъявляемые к языкам программирования:

1) **наглядность** - использование в языке по возможности уже существующих символов, хорошо известных и понятных как программистам, так и пользователям ЭВМ;

2) **единство** - использование одних и тех же символов для обозначения одних и тех же или родственных понятий в разных частях алгоритма. Количество этих символов должно быть как можно меньше;

3) **гибкость** - возможность описания распространенных приемов математических вычислений с помощью имеющегося в языке ограниченного набора изобразительных средств;

4) **однозначность** - недвусмысленность записи любого алгоритма. Отсутствие ее могло бы привести к неправильным ответам при решении задач.

5) **модульность** - возможность описания сложных алгоритмов в виде совокупности простых модулей, которые могут быть составлены отдельно и использованы в различных сложных алгоритмах;

Любой алгоритм - последовательность предписаний, выполнив которые, можно за некоторое число шагов перейти от исходных данных к результату.

В зависимости от степени детализации предписаний обычно определяется уровень языка программирования — чем меньше детализация, тем выше уровень языка.

По этому критерию можно выделить следующие уровни языков программирования:

1) машинные;

2) машинно-ориентированные (ассемблеры);

3) машинно-независимые (языки высокого уровня).

Языки программирования – это формальные искусственные языки. Как и естественные языки, они имеют алфавит, словарный запас, грамматику и синтаксис, а также семантику.

Основные компоненты алгоритмического языка:

1) алфавит,
2) синтаксис,
3) семантика.

Алфавит — это фиксированный для данного языка набор основных символов, т.е. букв алфавита, из которых должен состоять любой текст на этом языке — никакие другие символы в тексте не допускаются.

Синтаксис — это правила построения фраз, позволяющие определить, правильно или неправильно написана та или иная фраза.

Семантика – система правил однозначного толкования каждой языковой конструкции, позволяющих производить процесс обработки данных.

Взаимодействие синтаксических и семантических правил определяет основные понятия языка, такие как операторы, идентификаторы, константы, переменные, функции, процедуры и т.д. В отличие от естественных, язык программирования имеет ограниченный запас слов (операторов) и строгие правила их написания, а правила грамматики и семантики, как и для любого формального языка, явно однозначно и четко сформулированы.

1.2. Компиляторы и интерпретаторы

С помощью языка программирования создается текст программы, описывающий разработанный алгоритм. Чтобы программа была выполнена, надо либо весь ее текст перевести в машинный код (это действие и выполняет программа – ***компилятор)*** и затем передать на исполнение процессору, либо

сразу выполнять команды языка, переводя на машинный язык и исполняя каждую команду поочередно (этим занимаются программы – ***интерпретаторы***).

Интерпретатор функционирует следующим образом: берет очередной оператор языка из текста программы, анализирует его структуру и затем сразу исполняет. После успешного выполнения текущей команды интерпретатор переходит к анализу и исполнению следующей. Если один и тот же оператор в программе выполняется несколько раз, интерпретатор всякий раз воспринимает его так, будто встретил впервые. Поэтому программы, в которых требуется произвести большой объем повторяющихся вычислений, будут работать медленно. Для выполнения программы на другом компьютере также необходимо установить интерпретатор, так как без него программа представляет собой набор слов, и работать не может.

Компиляторы полностью обрабатывают весь текст программы (его называют *исходным* ***кодом*** или **source code**). Они осуществляют поиск синтаксических ошибок, выполняют семантический анализ и только затем, если текст программы в точности соответствует правилам языка, его автоматически переводят (транслируют) на машинный язык (говорят: генерируют ***объектный код*** или **object code**).

Нередко при этом выполняется оптимизация с помощью набора методов, позволяющих повысить быстродействие программы. Сгенерированный объектный код обрабатывается специальной программой ***сборщиком*** или ***редактором связей***, который производит связывание объектного и машинного кодов. Текст программы преобразуется в готовый к исполнению EXE-файл ***(исполнимый код)***, его можно сохранить в памяти компьютера или на диске. Этот файл имеет самостоятельное значение, и может работать под управлением операционной системы. Его можно перенести на другие компьютеры с процессором, поддерживающим соответствующий машинный код.

Основной недостаток компиляторов – трудоемкость трансляции языков программирования, ориентированных на обработку данных сложной структуры, заранее неизвестной или динамически меняющейся во время работы программы. Для таких программ в машинный код вводятся дополнительные проверки и анализ наличия ресурсов операционной системы, средства динамического захвата и освобождения памяти компьютера, что на уровне статически заданных машинных инструкций осуществить достаточно сложно, а для которых задач практически невозможно.

С помощью интерпретатора, наоборот, для исследования содержимого памяти допустимо в любой момент прервать работу программы, организовать диалог с пользователем, выполнить любые сложные преобразования данных и при этом постоянно контролировать программно-аппаратную среду, что и обеспечивает высокую надежность работы программы.

Интерпретатор при выполнении каждой команды подвергает проверке и анализу необходимые ресурсы операционной системы, при возникающих проблемах выдает сообщения об ошибках.

В реальных системах программирования смешаны технологии компиляции и интерпретации. В процессе отладки программу можно выполнять по шагам (трассировать), а результирующий код не обязательно будет машинным, он может быть, например, аппаратно-независимым промежуточным кодом абстрактного процессора, который в дальнейшем будет транслироваться в различных компьютерных архитектурах с помощью интерпретатора или компилятора в соответствующий машинный код.

Процесс создания программы включает (1):

- Составление исходного кода программы на языке программирования.
- Этап трансляции, необходимый для создания объектного кода программы
- Построение загрузочного модуля, готового к исполнению.

(1) Процесс создание программы, готовой к исполнению

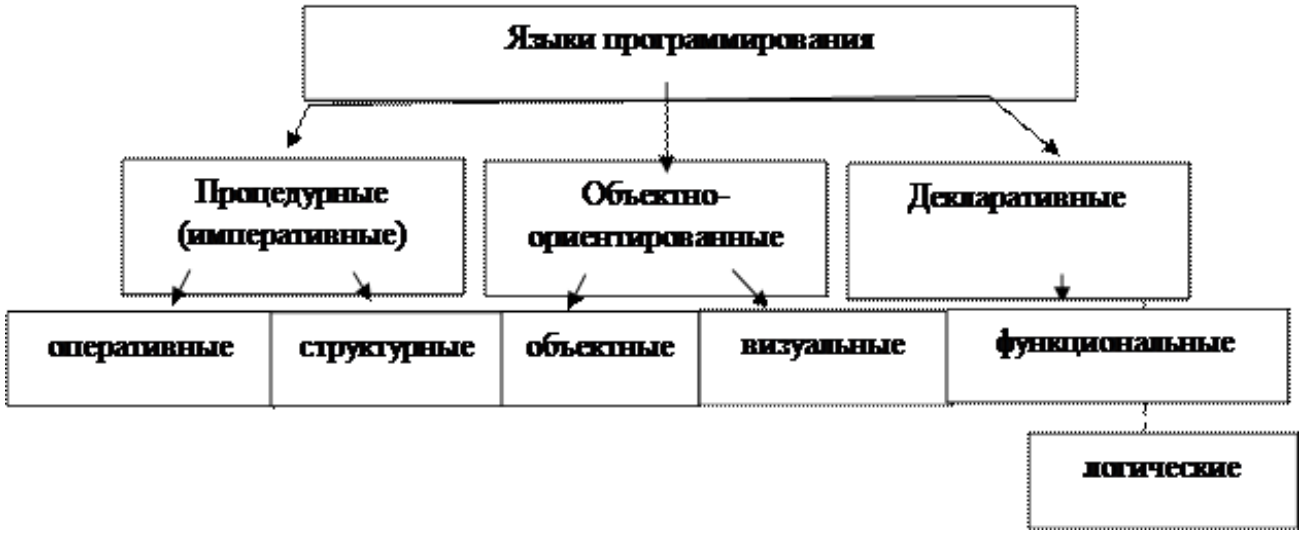
Все перечисленные выше действия требуют наличия специальных программных средств.

Совокупность этих программных средств входит в состав ***системы программирования***:

- Текстовый редактор (необходимый для создания и редактирования исходного кода программы на языке программирования).
- Компилятор.
- Редактор связей.
- Отладчик.
- Библиотеки функций.
- Справочная система.

1.3. Классификация и обзор языков программирования

Современное состояние языков программирования можно представить в виде следующей классификации (2).



(2) Классификация языков программирования

Сначала нужно узнать общие виды языков программирования и их назначение. Все они подразделяются на две категории:

- процедурные;
- непроцедурные.

Процедурная (алгоритмическая) программа — это система формальных предписаний, направленных на решение конкретных задач, которые выполняет ЭВМ.

Непроцедурное программирование представляет собой прямо противоположную методологию (парадигму) разработки, когда компьютеру ставится определённая задача в более или менее общем виде, без написания формализованного алгоритма, который отдаётся на усмотрение машины.

Процедурные языки отличаются тем, на кого в первую очередь направлены: на машину или человека. Они подразделяются на две категории:

- низкого уровня (или машинно-ориентированные);
- высокого уровня.

Языки программирования, ориентированные на команды процессора и учитывающие его особенности, называют языками **низкого уровня**. «Низкий уровень» не означает неразвитый, имеется в виду, что операторы этого языка

близки к машинному коду и ориентированы на конкретные команды процессора. Они учитывают их особенности и следуют конкретным указаниям, исходящим от процессора. Работать с ними тяжело, но созданные с их помощью программы (обычно это системные программы и драйверы) занимают меньше места в памяти и работают быстрее.

Языком самого низкого уровня является *ассемблер*. Программа, написанная на нем, представляет последовательность команд машинных кодов, но записанных с помощью *символьных мнемоник*. С помощью языков низкого уровня создаются компактные оптимальные программы, так как программист получает доступ ко всем возможностям процессора.

С другой стороны, при этом требуется хорошо понимать устройство компьютера, а использование такой программы на компьютере с процессором другого типа невозможно. Такие языки программирования используются для написания небольших системных приложений, драйверов устройств, модулей стыковки с нестандартным оборудованием, когда важнее компактность, быстродействие, прямой доступ к аппаратным ресурсам.

Языки программирования, имитирующие естественные, обладающие укрупненными командами, ориентированные «на человека», называют языками **высокого уровня**. Чем выше уровень языка, тем ближе структуры данных и конструкции, использующиеся в программе, к понятиям исходной задачи. Особенности конкретных компьютерных архитектур в них не учитываются, поэтому исходные тексты программ легко переносимы на другие платформы, имеющие трансляторы этого языка. Разрабатывать программы на языках высокого уровня с помощью понятных и мощных команд значительно проще, число ошибок, допускаемых в процессе программирования, намного меньше. В настоящее время насчитывается несколько сотен таких языков (без учета их диалектов).

Таким образом, языки программирования высокого уровня, ориентированные на решение больших содержательных прикладных задач, являются аппаратно-независимыми и требуют использования

соответствующих программ-переводчиков для преобразования текста программы в машинный код, который в итоге и обрабатывается процессором.

Непроцедурные языки включают две основные языковые группы:

- объектно-ориентированные;
- декларативные.

Объектно-ориентированные состоят из ряда независимых объектов, которые функционируют как отдельные компьютеры. С помощью этих блоков можно решать задачи, не вникая во «внутреннюю кухню» их работы.

Работа с **декларативным языком** подразумевает установление взаимосвязей между исходными информационными структурами и свойствами конечного результата. При этом в нём не существует понятия «команда», а программист не создаёт алгоритмы.

Декларативные языки подразделяются на два семейства:

- логические;
- функциональные.

Логическое программирование описывает проблемы в виде фактов и формул, а система решает их посредством механизмов логического вывода.

Функциональное, в свою очередь, формулирует задачу как совокупность определённых функций.

Глава 2. Востребованные языки программирования в настоящее время.

Алгоритмический язык (язык программирования) представляет собой один из способов записи алгоритма. Язык программирования является строго формализованным, то есть все команды записываются по определенным правилам и отступления от этих правил не допускаются. Например, в русском языке можно при разделении элементов перечисления поставить запятую (,) или точку с запятой (;). А в языке программирования при записи команд нельзя изменить ни одного знака - возникает ошибка.

Программа, написанная на языке программирования, состоит из команд (операторов), задающих последовательность действий. Эти действия выполняются над некоторыми объектами. Объектами могут быть числа, текстовые строки, переменные. Языки отличаются друг от друга множеством допустимых объектов и набором операций, которые можно выполнять над этими объектами.

Первым значительным шагом представляется переход к языку ассемблера. Не очень заметный, казалось бы, шаг — переход к символическому кодированию машинных команд — имел на самом деле огромное значение.

Программисту не надо было больше вникать в хитроумные способы кодирования команд на аппаратном уровне. Более того, зачастую одинаковые по сути команды кодировались совершенно различным образом в зависимости от своих параметров.

Появилась также возможность использования макросов и меток, что также упрощало создание, модификацию и отладку программ. Появилось даже некое подобие переносимости — существовала возможность разработки целого семейства машин со сходной системой команд и некоего общего ассемблера для них, при этом не было нужды обеспечивать двоичную совместимость.

Среди существующих языков программирования в настоящее время наиболее распространены следующие (3)

(3) диаграмма востребованных языков, сделанная с помощь результатов сайтов: PYPL, Индекс TIOBE, DOU, Wappalyzer, IEEE Spectrum, Stack Overflow.

2.1 Python

Python — активно развивающийся язык программирования. На официальном сайте Python этот язык описывается так: «Python — это язык программирования, который позволяет вам работать быстрее и более эффективно интегрировать ваши системы». Он очень прост в изучении как для начинающих веб-мастеров, так и для опытных, работающих с другими языками, программистов.

Python относительно легко читаем, часто его преподают на курсах начинающим программистам. Считается языком общего назначения, поэтому подходит для решения множества задач. Собственно, благодаря этому и популярен. Вообще, чтобы им воспользоваться и применить для собственных нужд в ИТ, не обязательно даже быть программистом.

Разработка языка Python была начата в конце 1980-х годов сотрудником голландского института CWI Гвидо ван Россумом. В феврале 1991 года Гвидо опубликовал исходный текст в группе новостей alt.sources. С самого начала Python проектировался как объектно-ориентированный язык.

Название языка произошло вовсе не от названия семейства пресмыкающихся. Автор назвал язык в честь популярного британского комедийного телешоу 1970-х «Летающий цирк Монти Пайтона». Впрочем, всё равно название языка чаще связывают именно со змеёй, нежели с передачей — пиктограммы файлов в KDE или в Microsoft Windows и даже эмблема на сайте python.org изображают змеиные головы.

Синтаксис Питона всегда выделял его на фоне других языков программирования. Он не страдает избыточностью, схожесть синтаксиса с обычным английским позволяет понять код даже обычному пользователю, кроме того, программист пишет меньше строк кода, потому что нет

необходимости использовать символы: «;», «{», «}». Вложенность обозначается отступами, что повышает читаемость кода и приучает новичков к правильному оформлению.

Python используют в сферах от веб-разработки до работы с искусственным интеллектом. Опрос ***JetBrains*** выделил пять главных направлений, в которых питон применяется чаще всего.

Веб-разработка

В веб-разработке питон используется для работы с серверной частью веб-приложений.

Анализ данных

Python — наиболее эффективный язык для работы с большими данными. Для анализа, визуализации и прогнозирования существуют несколько популярных и производительных библиотек. Так Pandas — фундаментальная библиотека питона, которую используют для анализа данных.

Машинное обучение и AI

Машинное обучение и искусственный интеллект часто программируют на питоне. Это удобно делать на простом языке с эффективной производительностью при обработке данных.

Парсинг, скрапинг и краулинг сайтов

Питон позволяет эффективно извлекать необходимую информацию с веб-сайтов для последующего анализа. Любую интернет-страницу можно исследовать на информацию, а всё что находится на ней — извлечь парсерами.

Скриптинг

Чаще всего скриптинг используют для написания несложных программ и автоматизации простых задач. Это позволяет делегировать рутинный ручной труд машине. Следует только написать пару десятков или сотен строк кода, а после забыть о нём. Остальную часть задачи скрипт возьмёт на себя.

Сейчас язык находится на пике своей популярности, и будет на нём ещё не менее пяти-десяти лет. Вот пять причин, почему питон всё ещё актуален.

1) **Питон перечеркнул миф о сложности разработки**.

У языка понятный синтаксис, который базируется на английском языке. На питоне легко писать и его легко читать.

2) **Большое количество справочной литературы.**

Вы не будете испытывать недостатка актуальной информации, потому что её много в открытом доступе — книги, сайты, форумы, видеоролики, платные и бесплатные курсы.

3) **Множество инструментов.**

Для питона создано множество инструментов, фреймворков и сред разработки, которые позволяют упростить решение многих задач. Можно воспользоваться готовым решением и не тратить время.

4) Минимализм.

Не нужно писать полотна лишнего кода. Динамическая типизация и другие функции языка дают возможность меньше заморачиваться над шаблонностью кода и упрощать его.

5) **Востребованность специалистов.**

Если вы сейчас начнёте изучать питон, то у вас не будет проблемы, что через пять лет вы останетесь без работы. И став питон-разработчиком легко освоить любой другой язык.

Основные недостатки, которые выделяют программисты:

1) Скорость исполнения кода.

Его быстродействия достаточно, чтобы покрыть запросы большинства задач, но в сравнении с конкурентами язык отстаёт. Писать что-то высокопроизводительное на одном только питоне не получится — придётся подключать другие языки или пользоваться типизированными расширениями.

2) **Динамическая типизация.**

Эта функция даёт возможность писать код проще и быстрее, но из-за этого могут возникнуть ошибки времени выполнения — это ошибки, которые генерируются, когда программа находится в запущенном состоянии. Из-за этого часто требуется дополнительное тестирование кода.

3) **Динамические ограничения видимости.**

Это головная боль многих разработчиков, ведь чтобы оценить выражение, компилятор ищет текущий блок и все вызывающие функции. Это означает, что каждое выражение тестируется во всех возможных контекстах, и на это уходит много времени.

4) **Сложность мобильной разработки.**

Теоретически, создавать приложения под мобильные устройства на питоне можно. Но для этого есть много более подходящих инструментов, вроде фреймворков Flutter и Iconic.

Но несмотря на ряд проблем языка, программисты закрывают на это глаза, ведь на кону — удобство и скорость работы.

Список компаний мирового масштаба, которые активно используют этот язык программирования.

1) Компания Google положительно отнеслась к языку с самого начала, решив использовать python и C++. C++ используется в компании для задач, где необходима высокая скорость и полный контроль над памятью. В остальных задачах инструментов достаточно python. Сегодня пайтон — один из официальных языков разработки Google.

2) Известный сервис стриминговой музыки использует python для анализа данных. Чтобы предоставить пользователям рекомендации, Spotify использует большое количество аналитики. В общей сложности сервис использует более 6000 сервисов питона, которые работают одновременно.

3) Самая популярная компания-поставщик фильмов и сериалов

мирового масштаба Netflix использует пайтон для тех же целей, что и Spotify. Питон стал основой для многих сервисов и ПО, так как язык имеет большое сообщество, множество удобных библиотек, а писать на нём код довольно удобно. Язык также используется для анализа данных на стороне сервиса Netflix.

Будущее Python — предмет горячих споров и дискуссий в сообществе программистов. Одни уверены, что лучше этого языка нет и не будет в ближайшее десятилетие точно. Другие говорят, что эпоха языка вот-вот пройдёт, как прошла эпоха языка Perl.

Программирование на питоне можно назвать скилом будущего. Будущее индустрии за автоматизацией и упрощением решения задач, а питон предоставляет для этого большие возможности. На этом языке работают самые разные сферы: от веб-программирования до создания игр. И он удобен и прост для первых шагов в программировании.

2.2 Java

Java — строго типизированный объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle).

Изначально язык назывался Oak («Дуб»), разрабатывался Джеймсом Гослингом для программирования бытовых электронных устройств. Из-за того, что язык с таким названием уже существовал, Oak был переименован в Java. Назван в честь марки кофе Java, которая, в свою очередь, получила наименование одноимённого острова (Ява), поэтому на официальной эмблеме языка изображена чашка с горячим кофе. Существует и другая версия происхождения названия языка, связанная с аллюзией на кофе-машину как пример бытового устройства, для программирования которого изначально язык создавался.

Java — это серьёзный объектно-ориентированный язык, на котором пишут программы для компьютеров и мобильные приложения. Он интересен тем, что один и тот же код можно скомпилировать под множество разных платформ. Java — один из основных языков для разработки под Android.

Язык программирования Java работает с веб-приложениями, которые транслируются в байт-код. Он может работать на любой компьютерной архитектуре, так как код преобразуется с помощью Java-машины.

Перечислить все бренды, которые используют Java, невозможно. В качестве примера можно взять YouTube, Netflix, Facebook, EBay, PayPal.

То, что в свое время Google выбрал Java для разработки Android, подогрело интерес к этому языку среди разработчиков. Сегодня это самая популярная операционная система, и практически все мобильные приложения для нее написаны на Java.

Java — универсальный язык и нашёл применение практически во всех областях экономики и IT-специализациях. На нём создают десктопные и мобильные приложения, софт для умной техники, программное обеспечение и игры. Java применяют банки, торговые и строительные фирмы, образовательные организации, государственные структуры и IT-корпорации.

Популярность.

Около 7 млрд устройств по всему миру используют приложения, созданные на джава. Миллионы программистов владеют языком программирования java и спрос на таких специалистов остаётся высоким.

Универсальность и гибкость.

Джава применяют в научных разработках, мобильных приложениях, геймдеве, создании десктопного ПО, веб-сайтов и софта для встраиваемых систем. На джава создают программы любой сложности: банковские приложения, интернет-магазины, софт для умной техники и умных домов.

Сообщество.

Благодаря большому числу программистов и реализованных решений, а также библиотек и дополнений практически любую проблему можно решить, обратившись за помощью к коллегам на форумах.

Кроссплатформенность.

Для джава создана специальная виртуальная машина JVM, исполняющая код. У неё две функции — запускать джава-приложения на любых ОС и устройствах и управлять памятью приложений. Единожды написанный код будет работать с любой операционной системой и на любой платформе.

Надёжность.

Джава — строго типизированный язык. Это значит, что всякая переменная или выражение имеют определённый тип и программа-компилятор проверяет код и не даёт совершать ошибки разработчику.

Простота изучения.

Для понимания java необходимы базовые знания. Нужно понимать специфику объектно-ориентированного программирования, разбираться в компьютерном «железе», изучать достаточно сложный синтаксис и учиться работать с дополнительными программными инструментами.

На джава созданы приложение Google Docs, игра Minecraft, серверная часть большинства приложений Netflix, социальные сети и линкедин, сервис такси Uber и веб-сервисы компании Amazon. А ещё на джава проектируют системы виртуальной и дополненной реальности, средства разработки программного обеспечения, файловые системы и контроллеры беспилотных автомобилей.

Синтаксис Java сильно отличается от синтаксис Python и Javascript, поскольку он относится к C-подобным языкам.

С точки зрения популярности Java стабильно входит в тройку самых популярных языков программирования.

2.3. C

Сегодня многие считают язык **Си** устаревшим. В каком-то смысле это так, ведь он появился в далеком 1972 году. Разрабатывался он с учетом того времени, то есть в соответствии с характеристиками компьютеров, которые существовали полвека назад. А какими были эти компьютеры? Если сказать упрощенно, то по своему функционалу они напоминали современный калькулятор.

Несмотря на все вышесказанное, спустя полвека Си совсем не умер, о нем не забыли. Секрет долголетия прост — **язык постоянно развивается и поддерживается**, несмотря на все «но». А еще он обеспечивает быстрое выполнение и хороший отклик, то есть быстродействие и производительность.

Многие компании успешно применяют Си десятки лет, ведь он до сих пор часто работает быстрее, чем конкуренты. Почему он так быстр? Потому что выполняется, по сути, на уровне процессора.

Си [C] - Многоцелевой язык программирования высокого уровня, разработанный Деннисом Ритчи в начале 1970-х гг. Является базовым языком операционной системы Unix, однако применяется и вне этой системы, для написания быстродействующих и эффективных программных продуктов, включая и операционные системы.

Язык Си разрабатывался как язык системного программирования, для которого можно создать однопроходный компилятор. Стандартная библиотека также невелика. Как следствие данных факторов — компиляторы разрабатываются сравнительно легко. Поэтому данный язык доступен на самых различных платформах. К тому же, несмотря на свою низкоуровневую природу, язык ориентирован на переносимость. Программы, соответствующие стандарту языка, могут компилироваться под различные архитектуры компьютеров.

Целью языка было облегчение написания больших программ с минимизацией ошибок по сравнению с ассемблером, следуя принципам процедурного программирования, но избегая всего, что может

привести к дополнительным накладным расходам, специфичным для языков высокого уровня.

Языки семейства C используют в разных сферах — в играх, операционных системах, кроссплатформенных приложениях и в интернете вещей.

C помогает понять логику и архитектуру компьютеров, операционных систем и сетевых технологий.

Вот основные особенности C:

Универсальность.

На C пишут код практически для всего — для робототехники, нейронных сетей, микроконтроллеров, операционных систем и интернета вещей.

Востребованность.

Программистам, которые знают C, проще найти работу. Это связано с тем, что реализованных решений на C большое количество. Все их нужно поддерживать в актуальном состоянии, «лечить», добавлять новые функции и оптимизировать.

Совместимость.

В основе кроссплатформенных приложений лежит язык C. Поэтому, если вы планируете заниматься разработкой для Linux и Windows, знание C — весомое преимущество.

Каким бы языком программирования вы ни владели — PHP, Python и прочие, для работодателя в приоритете будет кандидат, у которого в арсенале есть ещё и один из языков C.

Примеры использования данного языка:

Операционные системы.

Когда-то давно Unix была написана на ассемблере. Потом появился Си, и Unix переписали. В 1985 году C пригодился при написании Windows. Сегодня компьютеры Apple работают с помощью ОС macOS, ядро которой создано тоже с помощью C. Девять из десяти наиболее мощных

суперкомпьютеров — это тоже герой сегодняшнего разговора. Еще добавим ядра для iOS , Android и Windows Phone. Результат очевиден: Си нередко находится в фундаменте работы популярного ПО из разных сфер: от мобильных устройств до суперкомпьютеров;

Open Source-программы.

Проекты, имеющие открытый исходный код, тоже нередко создаются на C.

Драйверы устройств.

Они необходимы для подключения к вашему компьютеру различных устройств: клавиатуры, мыши, принтера, сканера и пр. Именно драйвер взаимодействует с операционной системой.

Языки программирования.

До сих пор при создании нового языка нередко применяют именно универсальный C;

Базы данных.

О, да… Самые популярные БД, такие как Oracle Database, MS SQL Server MySQL, SQLite и PostgreSQL, написаны на C. Почему? Потому что базы должны обладать максимальной надежностью и производительностью.

Графические библиотеки.

Опять же, работая с графикой, мы снова ожидаем максимального быстродействия, которое может обеспечить C. В крайнем случае, если речь идет о наиболее ответственных местах, некоторые участки кода частично пишутся на языке ассемблера. Примеры библиотек: Cairo, OpenGL, SDL;

Встроенные устройства.

Торговые автоматы, кассовые аппараты, парковочные роботы, программно-техническая начинка вашего автомобиля — все это Си

Космические и авиационные системы.

Раз опять нужна максимальная надежность, то по традиции вопрос выбора становится риторическим.

2.4 C++

Си++ [C++] - Язык программирования высокого уровня, созданный Бьярном Страуструпом на базе языка Си. Является его расширенной версией, реализующей принципы объектно-ориентированного программирования. Используется для создания сложных программ.

C++ — очень гибкий язык. Он позволяет обращаться к памяти напрямую и в то же время писать высокоуровневый код на примитивах стандартной библиотеки. Или даже на чём-нибудь посложнее.

Главное преимущество C++ — на нём можно создавать программы, которые быстро исполняются. Но только если правильно написать.

Вот где сегодня используют C++:

Веб-сервера.

Их пишут на C++ чаще всего — например, в том же «Яндексе». Для этого нужно понимать, как работают протоколы и сетевые соединения.

Графика и сложные вычисления.

Почему бы не C++? Хотя есть Python, программы на котором будут работать так же быстро, если их «правильно готовить», — ведь в конечном итоге это тоже просто вызовы функций.

Десктоп.

Раньше большую часть десктопных приложений писали на C++, но теперь всё чаще используют Java и C#.

Мультимедиа.

Приложения для аудио и видео пишут на C или C++. У «плюсов» есть замечательная библиотека FFmpeg для работы с видео — всем приходится её использовать, никуда от неё не денешься.

Перспективы C++

Иногда в Twitter и блогах встречаю мнение, что не сегодня завтра C++ умрёт — а его заменит, например, Rust. Я в это не верю. Как минимум потому, что есть куча кода, который нельзя переписать на другие языки, а значит, кому-то придётся поддерживать «плюсы». Бизнес уже вложил слишком много денег в язык — выбрасывать его на помойку он не станет.

Да, со временем C++ наверняка заменит какой-нибудь молодой и перспективный язык — и пусть это будет Rust. Однако такая замена произойдёт не через пять и даже не через десять лет. Тот же Fortran был довольно популярным десять лет назад. Угасать он начал относительно недавно — всё-таки старичку уже 60 лет, а нашим «плюсам» всего 40. Поэтому работы для программистов, пишущих на C++, на наш век точно хватит.

Для чего используется C ++?

C ++ имеет множество реальных приложений, в том числе:

- Разработка видеоигр
- Приложения на основе графического интерфейса
- Базы данных
- Операционные системы
- Веб-браузеры
- Вычисления и графика
- Банковское дело
- Облако
- Распределенные системы
- Компиляторы

- Встроенные системы
- Корпоративное программное обеспечение
- Библиотеки
- Крупномасштабные серверные приложения
- Компиляторы кода

C ++ также используется для создания многих популярных сервисов, таких как MySQL, Microsoft Windows и Office, macOS и других. Это популярный язык для крупных встраиваемых систем. Он часто используется для системного программирования и создания приложений с ограниченными ресурсами. C ++ — отличный язык для использования всякий раз, когда у вас большой буфер, а также в случаях, когда у вас высокий уровень параллелизма и требуется минимальная задержка. Это касается серверных приложений и игр.

Есть причина, по которой C ++ остается одним из самых популярных языков программирования. У языка есть много важных функций и преимуществ, в том числе:

Обработка исключений

Обработка исключений встроена в C ++. Это инструмент, который разделяет код, который обнаруживает и обрабатывает исключительные обстоятельства, возникающие при выполнении программ.

Перегрузка функций

Перегрузка функций — это процесс наличия двух или более функций с одним и тем же именем, но с разными параметрами. Эта функция C ++ позволяет вам определять более одного определения для имени функции или оператора в одной и той же области.

Управление памятью

C ++ поддерживает динамическое распределение памяти (DMA), которое помогает освобождать и выделять память. Его возможности

манипулирования памятью позволяют вам настраивать вещи и напрямую обращаться к аппаратным данным, а также писать высокопроизводительный код.

C ++ Стандартной библиотека

C ++ стандартной библиотека шаблонов (STL) заполнена шаблонами готовых к использованию библиотек для различных структур данных, арифметических операций и алгоритмов.

Объектно-ориентированный

Концепции объектно-ориентированного программирования (ООП) позволяют обрабатывать данные как объекты и классы.

Мультипарадигма

C ++ — это мультипарадигмальный язык. Это позволяет вам выбрать единый подход или смешивать аспекты разных парадигм программирования (таких как общие, императивные и объектно-ориентированные).

Высокая переносимость

C ++ отличается высокой переносимостью и используется для создания сценариев приложений систем, которые составляют значительную часть операционных систем Windows, Linux и Unix.

Универсальность

C ++ универсален и имеет большой рынок труда. Он используется во многих различных отраслях, таких как финансы, разработка игр, машинное обучение и т.д.

Масштабируемость

C ++ отлично подходит для ресурсоемких приложений из-за его масштабируемости и производительности.

Это чрезвычайно быстрый и эффективный язык. Многие инструменты и фреймворки полагаются на скорость и эффективность C ++. Сейчас он пользуется большим спросом, и он будет оставаться востребованным в 2022 году из-за своей надежности, производительности и эффективности.

C ++ — отличный язык для изучения, если вы программист, который хочет глубоко понять, как работают компьютеры. C ++ позволяет вам познакомиться с низкоуровневыми концепциями программирования и помогает понять, как компьютеры думают и работают. Другие языки и концепции программирования могут иметь для вас больше смысла после того, как вы изучите C ++.

Язык программирования C ++ остается одним из самых популярных языков в программной инженерии и информатике. Несмотря на то, что для него характерна крутая кривая обучения, это широко используемый низкоуровневый язык программирования, используемый для создания многих соответствующих приложений. Есть много преимуществ в изучении языка, близкого к «голому железу», поскольку он помогает лучше понять, как работают компьютеры.Э

2.5 C#

C# (C Sharp) – “Си Шарп”: объектно-ориентированный яык программирования, о разработке которого в 2000 г. объявила фирма Microsoft. По своему характеру он напоминает языки C++ и Java и предназначен для разработчиков программ, использующих языки C и C++ для того, чтобы они могли более эффективно создавать интернет-приложения. Указывается, что C# будет тесно интегрирован с языком XML.

Это объектно-ориентированный язык, который использует платформу .NET для создания программного обеспечения, приложений и веб-разработки.

Язык популярен среди разработчиков игр. Unity 3D использует этот язык практически на всех этапах производства, изредка разбавляя его Java. В целом, очень влиятельный язык, предназначенный для непростых задач.

C# позволяет создавать функциональные и производительные приложения. Если хотите написать ОС, свой язык программирования,

программировать серьезную аппаратуру и даже роботов… Тогда вам нужен C#.

Его можно использовать для продвинутых бизнес-приложений, видеоигр, функциональных веб-приложений, приложений для Windows, macOS, мобильных программ для iOS и Android.

C#— популярная технология, однако в сравнении с другими этот язык считается более сложным для новичков. Но не смотря на низкий порог входа, язык продолжают активно изучают. Потому что спрос явно есть.

Является братом языков C и C++ со схожим синтаксисом. Однако имеет сходство и с Java.

Помимо этого, язык C# отличается медленной скоростью работы, но несмотря на это имеет безопасный код, что немаловажно.

Где применяется C#

Разработка игр

Unity – один из наиболее популярных игровых движков. Он включает все необходимое для разработки: физику, редактор графики и множество различных инструментов. Для создания игр на Unity используется C#, что делает этот язык востребованным в сфере геймдева.

Связка Unity + C# популярна в небольших компаниях и среди инди-разработчики из-за отличной функциональности, дружелюбного сообщества и возможности использования для написания любых 2D- и 3D-игр.

Веб-разработка (Backend и Frontend)

Во времена появления C# и платформы .NET библиотеки еще не были развиты, но даже тогда платформа ASP.NET имела большую аудиторию. Сейчас Microsoft поменяла почти все и добавила кроссплатформенность, а скорость работы приложения выросла многократно. Современный свободно распространяемый фреймворк ASP.NET Core позволяет запускать сервисы на Windows, Linux и macOS. На нем можно разрабатывать бэкенд-приложения, REST API и приложения MVC.

Если этого мало, то после выхода фреймворка Blazor на C# можно писать и фронтенд, отбросив JavaScript. Зная HTML, CSS и C#, вы создадите полноценные фуллстек-проекты и даже настольные приложения.

Машинное обучение (ML)

Фреймворк для машинного обучения ML.NET поддерживается корпорацией Microsoft.

Строгая типизированность C# и его нацеленность на объектно-ориентированное программирования очень помогает при разработке. Еще стоит упомянуть комфортную интеграцию с разнообразными встраиваемыми системами, датчиками и другими приспособлениями.

Будучи объектно-ориентированным языком, он много перенял у Java и C++. Как и Java, C# изначально предназначался для веб-разработки, и примерно 75% его синтаксических возможностей такие же, как у Java. C# также называют «очищенной версией Java». Ещё 10% наш герой позаимствовал из C++ и 5% – из Visual Basic. Оставшиеся 10% C# — это реализация собственных идей разработчиков. Объектно-ориентированный подход позволяет строить с помощью C# крупные, но в то же время гибкие, масштабируемые и расширяемые приложения.

Он всё ещё активно развивается, и с каждой новой версией появляется всё больше интересного .

По сравнению с другими языками C# довольно молод, но в то же время он уже прошёл большой путь.

У «шарпа» выделяют много преимуществ:

- Поддержка подавляющего большинства продуктов Microsoft
- Бесплатность ряда инструментов для небольших компаний и некоторых индивидуальных разработчиков — Visual Studio, облако Azure, Windows Server, Parallels Desktop для Mac Pro и др.
- Типы данных имеют фиксированный размер (32-битный int и 64

битный long), что повышает «мобильность» языка и упрощает программирование, так как вы всегда знаете точно, с чем вы имеете дело.

- Автоматическая «сборка мусора» Это значит, что нам в большинстве случаев не придётся заботиться об освобождении памяти. Вышеупомянутая общеязыковая среда CLR сама вызовет сборщик мусора и очистит память.
- Большое количество «синтаксического «сахара» — специальных конструкций, разработанных для понимания и написания кода. Они не имеют значения при компиляции.
- Низкий порог вхождения. Синтаксис C# имеет много схожего с другими языками программирования, благодаря чему облегчается переход для программистов. Язык C# часто признают наиболее понятным и подходящим для новичков.
- С помощью Xamarin на C# можно писать программы и приложения для таких операционных систем, как iOS, Android, MacOS и Linux;

Но есть у C# и некоторые недостатки:

- Приоритетная ориентированность на платформу Windows;
- Язык бесплатен только для небольших фирм, индивидуальных программистов, стартапов и учащихся . Крупной компании покупка лицензионной версии этого языка обойдётся в круглую сумму.
- Инструментарий C# позволяет решать широкий круг задач, язык действительно очень мощный и универсальный. На нём часто разрабатывают: веб-приложения, игры, мобильные приложения для Android или iOS, программы под Windows.

Перечень возможностей разработки практически не имеет ограничений благодаря широчайшему набору инструментов и средств. Конечно, всё это можно реализовать при помощи других языков. Но некоторые из них

узкоспециализированные, а в некоторых придётся использовать дополнительные инструменты сторонних разработчиков. В C# решить широкий круг задач возможно быстрее, проще и с меньшими затратами времени и ресурсов.

2.6 JavaSkript

JavaScript (JS) — высокоуровневый язык программирования, который поддерживает императивный, функциональный, событийно-ориентированный и другие подходы. Относится к языкам с динамической типизацией, входит в группу интерпретируемых языков.

В число основных особенностей JS входят:

Динамическая типизация — тип данных определяется в момент присваивания значения константе или переменной.

Интерпретируемый язык — код приложения интерпретируется при обращении, не требуется предварительная компиляция.

Функции как объекты первого класса, то есть функции в JavaScript можно возвращать из функций, передавать в качестве параметров в другие функции, присваивать переменным.

Поддержка прототипного и объектно-ориентированного подхода.

Универсальность — все популярные браузеры поддерживают JavaScript.

Важная особенность JavaScript — развитая инфраструктура. Вокруг этого языка программирования сформировано многочисленное сообщество.

Разработчикам доступен широкий выбор инструменты, например:

- Библиотеки и фреймворки для создания приложений (React, Vue).
- Сборщики (Webpack, Gulp).
- Вспомогательные библиотеки (Lodash, Underscore).
- Генераторы статических сайтов (Gatsby.js, Next.js).

Сферы применения JavaScript

В первую очередь JavaScript широко используется во фронтенд-разработке. Этот язык вместе с HTML и CSS входит в базовый набор инструментов фронтендера. На JavaScript создаются приложения, которые исполняются в браузере на стороне клиента. Они обеспечивают интерактивность сайтов. Например, когда пользователь заполняет форму и нажимает кнопку «Подписаться», мгновенная реакция на это действие обычно обеспечивается кодом, написанным на JavaScript.

Сферы применения JavaScript не ограничиваются браузерами и веб-приложениями. На этом языке. например, можно:

1. Разрабатывать нативные приложения. Например, с помощью фреймворка React Native создаются приложения для Android и iOS.
2. Серверные приложения. Node.js применяется для бэкенд-разработки.
3. Десктопные приложения. JS применяется в офисных пакетах Microsoft и OpenOffice, в приложениях компании Adobe.
4. Программировать оборудование и бытовую технику, например, платёжные терминалы и телевизионные приставки.

Для начинающего JavaScript-программиста есть три основные пути развития, если связывать свою деятельность с веб-разработкой:

Frontend-разработка.

В ходе нее создается интерфейс сайта или веб-приложения, прорабатывается базовый функционал, анимации и переходы. На этом этапе JavaScript используется для создания анимации, всплывающих окон, базовой проверки правильности заполнения форм.

Backend-разработка.

Отвечает за проработку внутреннего функционала сайта или приложения: обработка форм, реакция на действия пользователей и так далее. Здесь уже желательны более углубленные знания JavaScript. На начальных позициях можно обойтись только знаниями JS, но для дальнейшего роста

потребуется изучение других языков программирования: PHP, Python, Ruby и так далее.

Fullstack-разработка.

Включает полную разработку сайта или приложения, то есть Frontend и Backend. Требует знания хотя бы на среднем уровне: HTML, CSS, JavaScript. Еще желательно будет изучить и другие языки программирования, используемые в Backend.

JavaScript используется почти во всех веб-проектах, плюс, по нему много учебных материалов и документации. Сам язык программирования легко выучить на базовом уровне и уже можно будет применять в разработке.

JavaScript использует понятный синтаксис и логику построения команд, что позволяет быстро разобраться в его работе. Также в интернете очень много технической документации и других обучающих материалов, в том числе и на русском языке. Есть даже интерактивные игры, которые подойдут детям и взрослым для изучения основ JavaScript.

JavaScript встречается почти на всех сайтах и веб-приложениях, за исключением самых примитивных. Он может как просто отвечать за анимацию некоторых элементов и всплывающих окон, так и за более сложный функционал, например обработку форм. Пока JS нет адекватной замены в сфере веб-разработки, следовательно, новые проекты тоже активно используют JavaScript для работы.

Помимо веба JS еще используется в разработке игр, десктопных программ, для научных вычислений. В ближайшее время вряд ли JavaScript будет терять свою популярность.

Возможности языка можно расширить с помощью сторонних библиотек, например, npm, React и так далее. Они значительно облегчают процесс разработки, а благодаря использованию почти того же синтаксиса и правил взаимодействия, что и в классическом JS, легки в освоении.

Последнее крупное обновление версии JavaScript произошло в 2009 году. JavaScript постоянно меняется, адаптируясь к новым задачам. Регулярно

обновляются версии популярных библиотек, типа, React, также создаются новые. В ближайшем будущем основные нововведения коснутся библиотек и плагинов, написанных на JS. Возможно, что будут внесены какие-то обновления и в сам язык.

Многие крупные проекты, например, Netflix, PayPal и другие в качестве своей основы используют JavaScript. Все больше крупных проектов стараются перенести часть своего функционала на JS или его библиотеку Node.js. Данное решение обусловлено тем, что конечный продукт становится более отзывчивым, а затраты на его обслуживание снижаются.

Заключение

Изобретение языков программирования высшего уровня, а также их постоянное совершенствование и развитие, позволило человеку не только общаться с машиной и понимать ее, но использовать ЭВМ для сложнейших расчетов в области самолетостроения, ракетостроения, медицины и даже экономики.

На сегодняшний день любое среднее и крупное предприятие имеет в своем штате группу программистов, обладающих знаниями

программирования различными языками, которые редактируют, изменяют и модифицируют программы, используемые сотрудниками предприятия. Это говорит о том, что на рынке труда пользуются спросом обладающие знаниями и опытом работы с различными языками программирования люди.

Несмотря на то, что современный уровень развития языков программирования является довольно высоким, тенденция их развития, а также развития информационных технологий в целом, складывается таким образом, что можно предположить, что в ближайшем будущем, человеческие познания в этой сфере помогут произвести на свет языки, умеющие принимать, обрабатывать и передавать информацию в виде мысли, слова, звука или жеста.

Сейчас нам необходимы языки программирования для повседневной жизни, к которой мы привыкли. Так как без языков программирования не было ни ПК, ни программ, ничего, что связано с компьютерами. Мы бы не общались в соц.сетях, не покупали вещи в интернет-магазинах. Не зарабатывали с помощью интернета и компьютера. Я считаю, что языки программирования имеют очень большую роль в жизни каждого человека.

Тему выбора языка программирования можно раздуть до очень больших размеров, перечисляя все возможные и невозможные тонкости в этом деле.

Список литературы

1) Трофимов, В. В. Основы алгоритмизации и программирования : учебник для СПО / В. В. Трофимов, Т. А. Павловская ; под ред. В. В. Трофимова. — М. : Издательство Юрайт, 2019. — 137 с.

2) Федоров, Д. Ю. Программирование на языке высокого уровня python : учеб. пособие для прикладного бакалавриата / Д. Ю. Федоров. — 2-е изд., перераб. и доп. — М. : Издательство Юрайт, 2019. — 161 с.
3) Александреску, А. Язык программирования D / А. Александреску. — СПб.: Символ-плюс, 2017. — 544 с.
4) Ашарина, И.В. Основы программирования на языках С и С++: Курс лекций для высших учебных заведений / И.В. Ашарина. — М.: Гор. линия-Телеком, 2018. — 208 с.
5) Баженова, И.Ю. Языки программирования: Учебник для студентов учреждений высш. проф. образования / И.Ю. Баженова; Под ред. В.А. Сухомлин. — М.: ИЦ Академия, 2018. — 368 с.
6) Бьянкуцци, Ф. Пионеры программирования: Диалоги с создателями наиболее популярных языков программирования / Ф. Бьянкуцци, Ш. Уорден; Пер. с англ. С. Маккавеев. — СПб.: Символ-Плюс, 2017. — 608 с.
7) Гавриков, М.М. Теоретические основы разработки и реализации языков программирования: Учебное пособие / М.М. Гавриков, А.Н. Иванченко, Д.В. Гринченков. — М.: КноРус, 2016. — 184 с.
8) Керниган, Б. Язык программирования С. 2-е изд. / Б. Керниган, Д.М. Ритчи. — М.: Вильямс, 2016. — 288 с.
9) Златопольский Д.М. Основы программирования на языке Python. – М.: ДМК Пресс, 2017. – 284 с.
10) Блох, Д. Java Эффективное программирование / Д. Блох. - М.: Лори, 2016. - 440 с.
11) Изучаем программирование на JavaScript / Фримен Э., Робсон Э. Ф88. - СПб.: Питер, 2015. —-640 с
12) Боровский, А.Н. Qt4.7+. Практическое программирование на С++. / А.Н. Боровский. - СПб.: BHV, 2012. - 496 с.
13) Васильев, А. C#. Объектно-ориентированное программирование: Учебный курс / А. Васильев. - СПб.: Питер, 2012. - 320 с.