4134 гр.

Басов.А

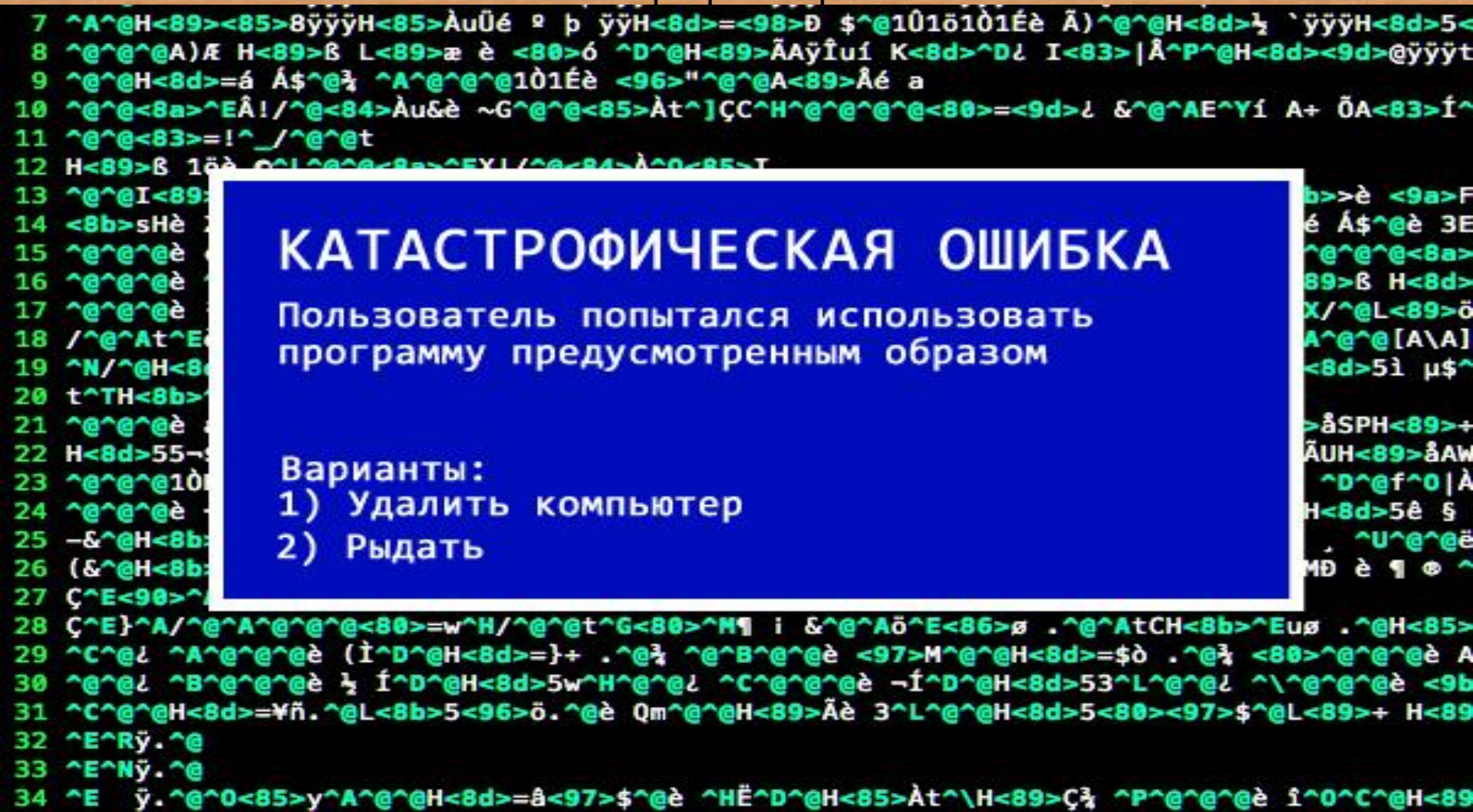
Языки программирования

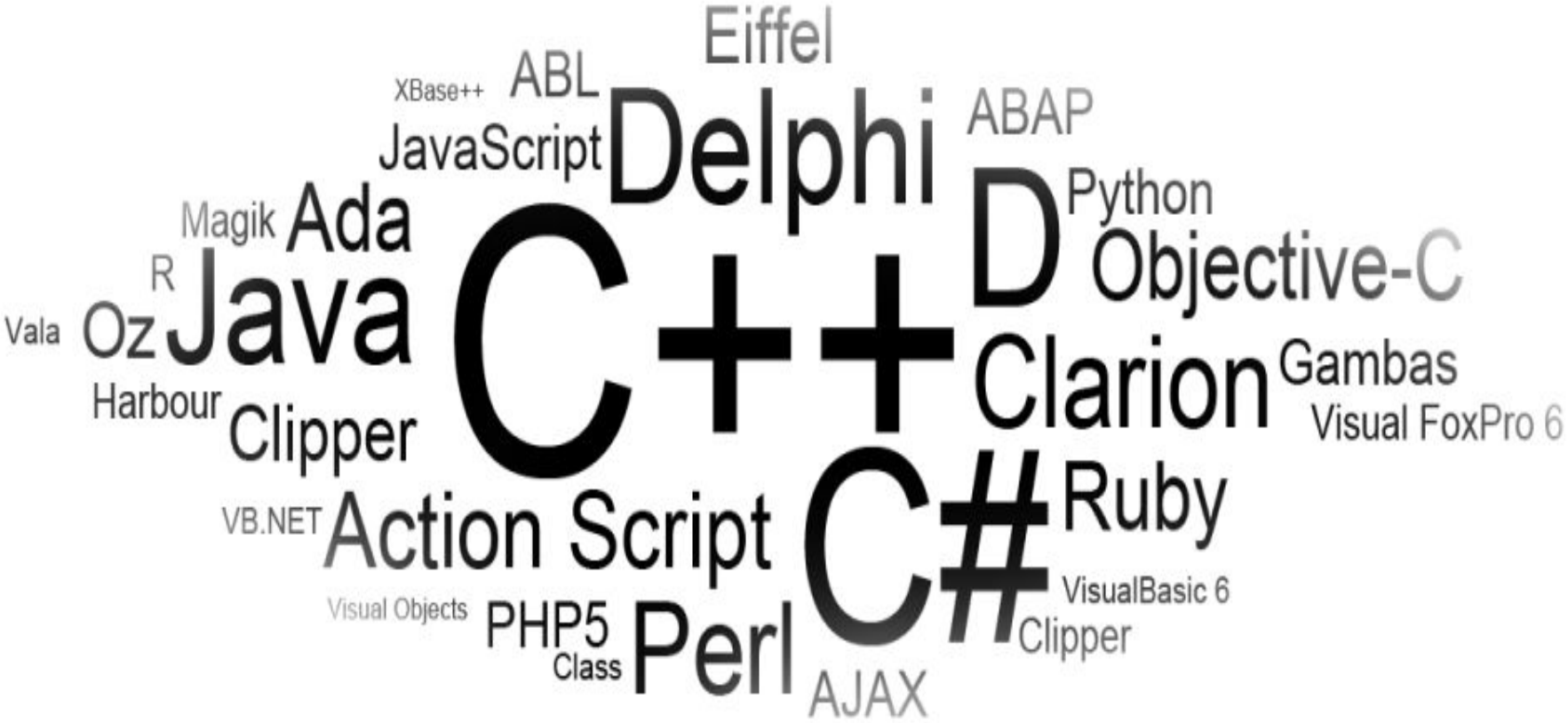
Язы́к программи́рования — формальная знаковая система, предназначенная для записи компьютерных программ. Язык программирования определяет набор лексических, синтаксических и семантических правил, определяющих внешний вид программы и действия, которые выполнит исполнитель (обычно — ЭВМ) под её управлением.

Программирование - это искусство создавать программные продукты, которые написаны на языке программирования. **Язык программирования** – это формальная знаковая система, которая предназначена для написания программ, понятной для исполнителя (в нашем рассмотрении – это компьютер).

Язык программирования (англ. *Programming language*) - система обозначений для описания алгоритмов и структур данных, определенная искусственная формальная система, средствами которой можно выражать алгоритмы. Язык программирования определяет набор лексических, синтаксических и семантических правил, задающих внешний вид программы и действия, которые выполняет исполнитель (компьютер) под ее управлением.

Со времени создания первых программируемых машин было создано более двух с половиной тысяч языков программирования. Ежегодно их число пополняется новыми. Некоторыми языками умеет пользоваться только небольшое число их собственных разработчиков, другие становятся известны миллионам людей. Профессиональные программисты обычно применяют в своей работе несколько языков программирования.

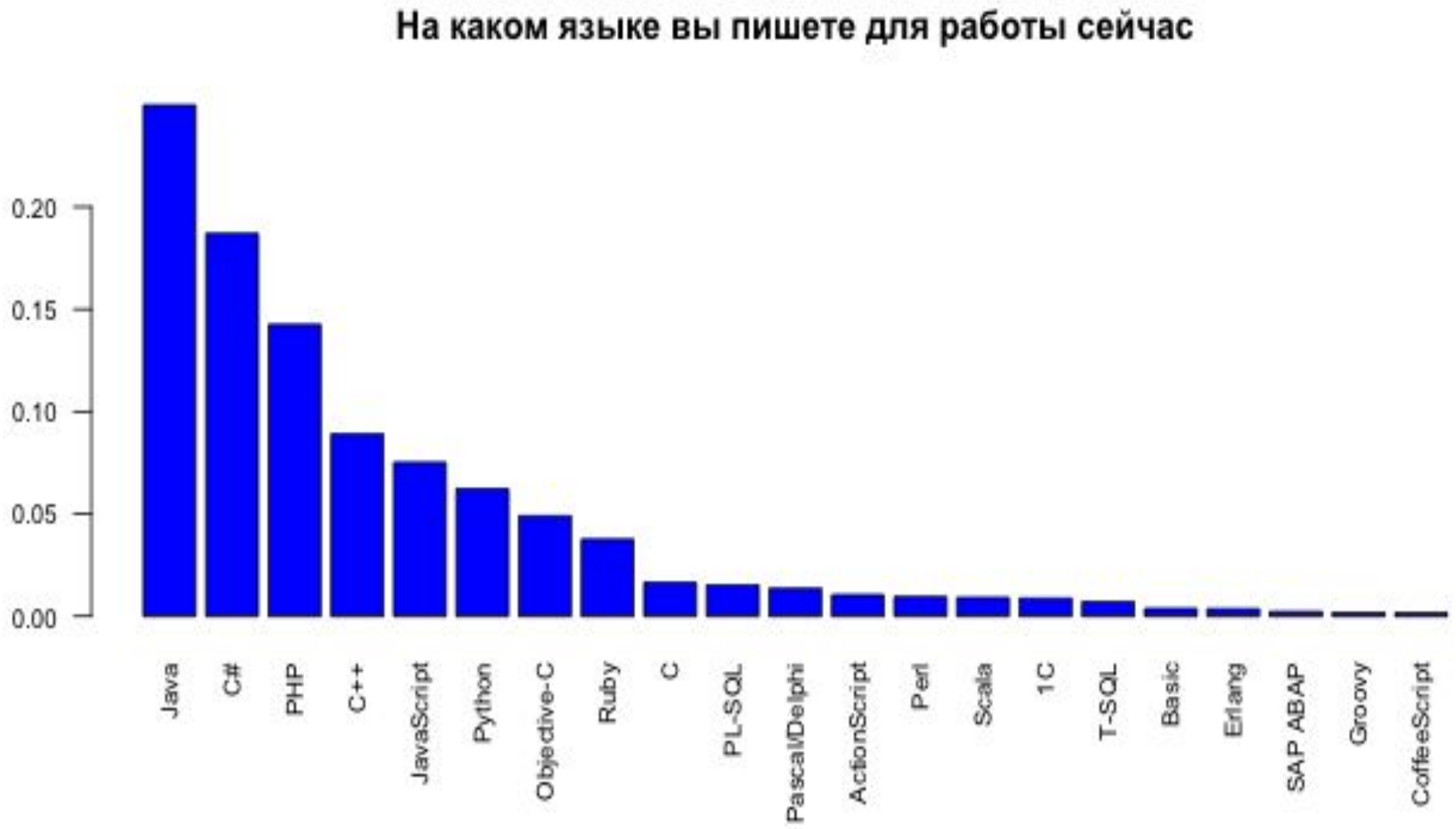



Eiffel
XBase++
ABL
JavaScript
Delphi
ABAP
Python
Magik
Ada
D
Objective-C
R
Java
C++
Vala
Oz
Clarion
Gambas
Harbour
Clipper
Visual FoxPro 6
C#
Ruby
VB.NET
Action Script
VisualBasic 6
Visual Objects
PHP5
Perl
Clipper
Class
AJAX

ИНДЕКС ПОПУЛЯРНОСТИ	ЯЗЫК	СТАТИЧЕСКИЙ/ДИНАМИЧЕСКИЙ
1	Java	Статический
2	Си	Статический
3	C++	Статический
4	Visual Basic	Статический
5	PHP	Динамический
6	Perl	Динамический
7	C#	Статический
8	Python	Динамический
9	JavaScript	Динамический
10	Ruby	Динамический
11	SAS	Динамический
12	Delphi	Статический
13	PL/SQL	Динамический
14	D	Статический
15	ABAP	Динамический
16	Lisp/Scheme	Динамический
17	Ada	Статический
18	Cobol	Динамический
19	Pascal	Статический
20	Transact/SQL	Статический

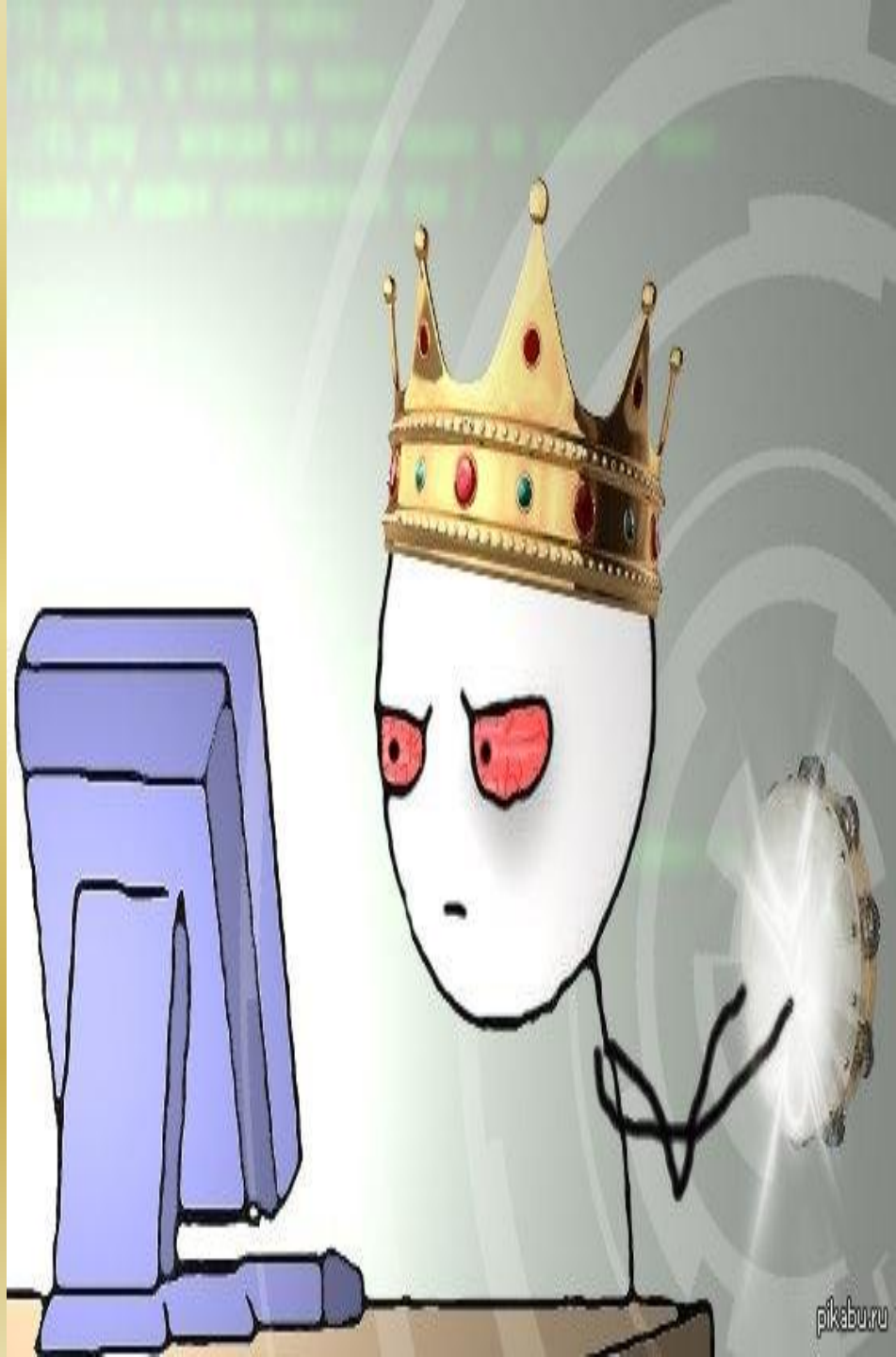
Источник: Tiobe Software, 2006.

какие же бывают языки программирования?



- **Классификация**
- Программные языки условно можно разделить на 4 вида, это:
  1. Полнофункциональные;
  2. По отдельным признакам;
  3. Не полнофункциональные;
  4. Эзотерические.

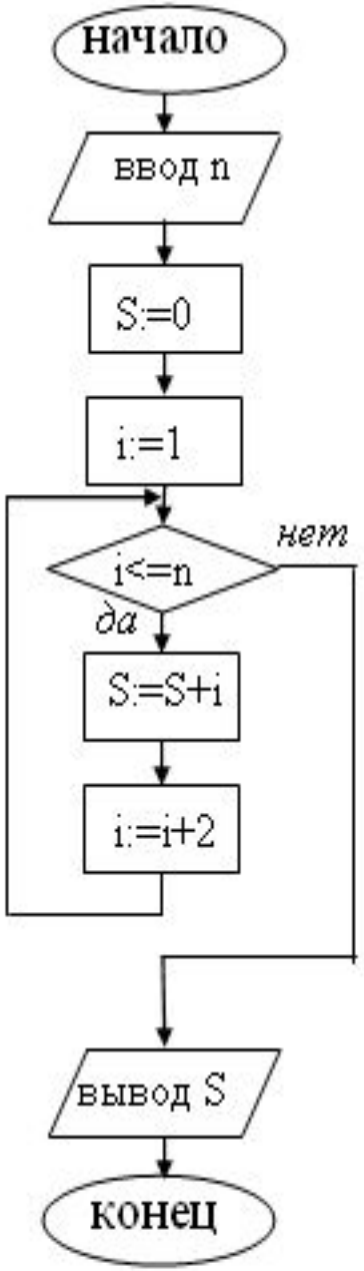
- Естественно, в каждом типе есть подразделы, но в них особо мы вникать не будем.
- Главным и основным языком разработки в группе полнофункциональных является — C#, Паскаль, C++, Delphi, ява и яваскрипт, а также PHP и Бэйсик



Паскаль

- Это один из самых известных языков по разработке компьютерных приложений. Паскаль является базой под другие языки. Когда я стал интересоваться программированием, то первым вопросом было — какой же язык программирования изучать? Паскаль был в моем списке изучаемых языков, и я не ошибся. Для меня изучение паскаля было нетрудным, к тому же существует большое количество материалов по изучению языка

Блок – схема	Псевдокоды	Pascal

	**алг** сумма нечетных чисел
	нач
	ввод n
	S:=0
	i:=1
	нц пока i<=n
	S:=S+i
	i:=i+2
	кц
	вывод S
	кон
		program summa_nech;
		var i, n, S: **integer**;
		begin
		readln (n);
		S:=0;
		i:=1;
		while i<=n do
		begin
		S:=S+i;
		i:=i+2
		end;
		writeln (S)
		end.

C#

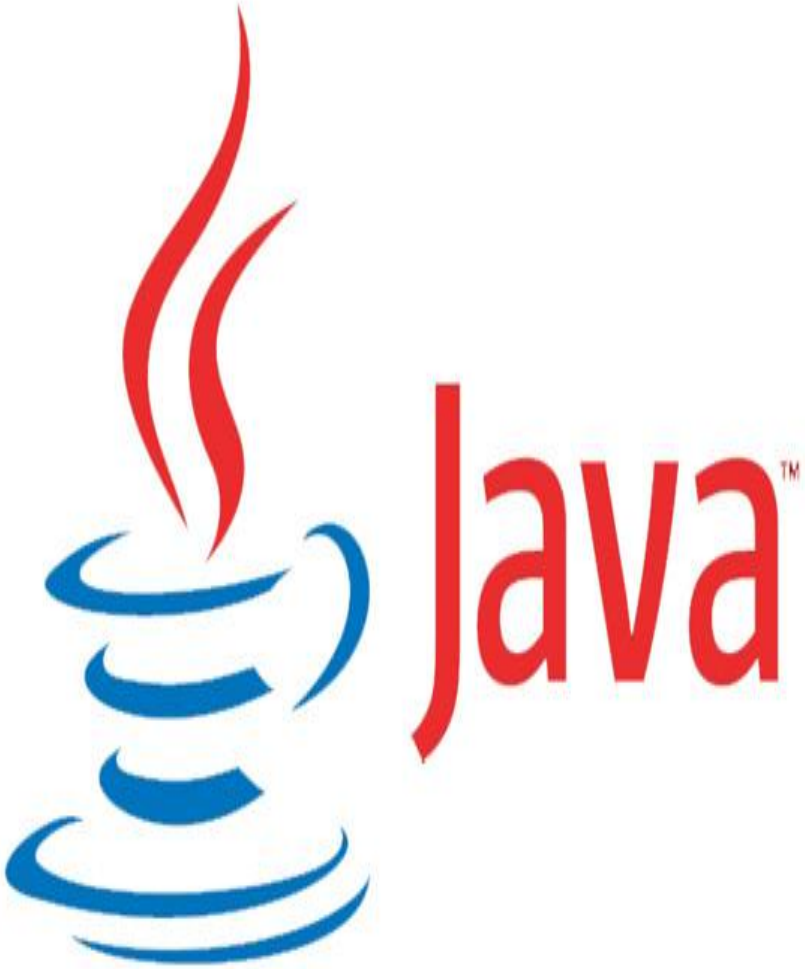
- Данный язык применяется для создания программного обеспечения. На нем строится большое количество осей (ОС — операционная система). На C# разрабатываются драйвера, и другие прикладные программы.

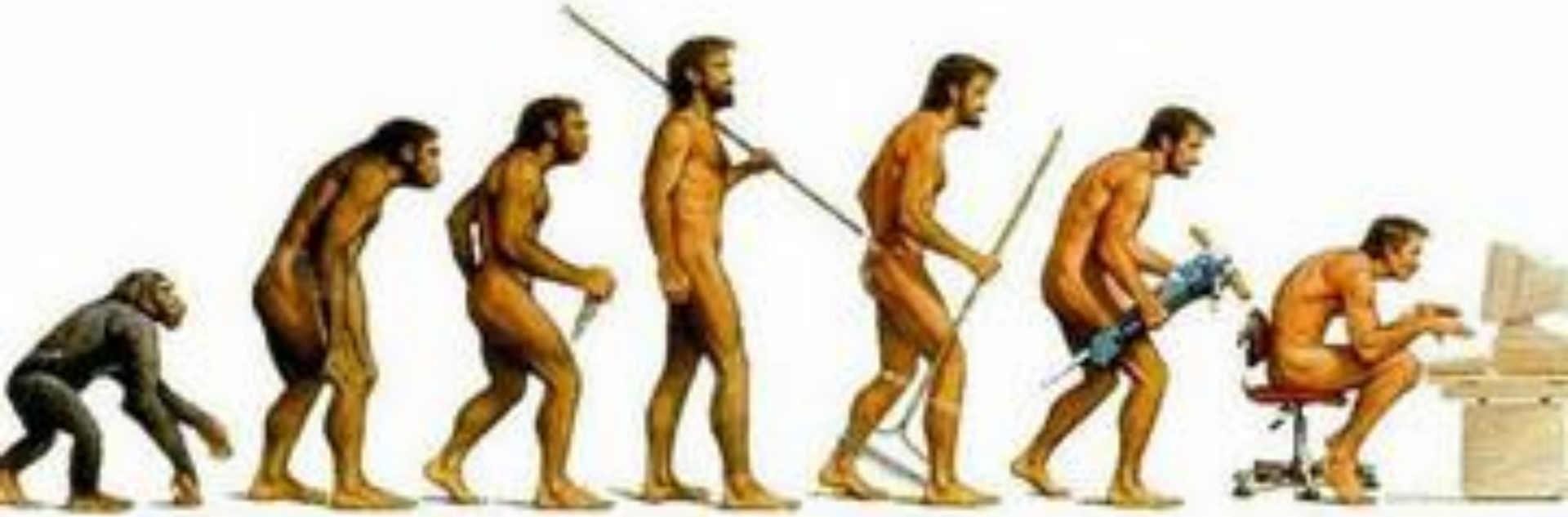
PHP

- Пи-Эйч-Пи очень удобен при создании сайтов. Если взглянуть на код страницы, то можно заметить на каком языке прописан сайт. Благодаря широкому функционалу php дает широкие возможности для веб-программирования и строения сайта.

Java

- **Java**
- Главным преимуществом данного языка является независимость от ОС и оборудования. На Джава языке строятся как обычные программы, игры, а также СУБД, базы данных.
- Я рассказал вам о самых распространенных языках, теперь вы имеете представление о каждом из них. Главный выбор остается за вами — какой изучать. На просторах интернета вы найдете большое количество полезного и разнообразного материала по изучению языков программирования, данной тематике посвятили себя многие сайты. Удачи в изучении и выборе.





11111011101

Как выучить язык программирования ?

- Если вы хотите научиться создавать компьютерные программы, мобильные приложения, сайты, игры или любое другое программное обеспечение, вам, собственно, нужно научиться программировать. Программы пишутся на языках программирования, которые и дают программе возможность работать – быть выполненной компьютером, смартфоном или иным вычислительным устройством.

Часть 1

Выбираем язык

- **Определитесь с тем, что вам интересно.** Конечно, вы можете начать учить любой язык программирования (далее ЯП). Правда, некоторые ЯП существенно легче в освоении других… Как бы то ни было, вам нужно определиться с тем, ради чего вы будете учить язык программирования. Это позволит вам определиться с тем, что именно вам надо учить, да и просто станет хорошей отправной точкой . Веб-разработка греет вам душу? Список ЯП, которые вам пригодятся, существенно отличается от списка ЯП, необходимых для написания компьютерных программ. Мобильная разработка – мечта вашего детства? Это уже третий список. То, что вы будете учить, зависит от того, чем вы хотите заниматься.

- **Начните с более простого языка.** Что бы вы там для себя ни решили, а начинать стоит с относительно простых языков высокого уровня. Такие языки особенно хороши для начинающих, так как они позволяют овладеть базовыми принципами и понять общую логику программирования.[1]Чаще всего в этом контексте вспоминают про Python и Ruby. Это два объектно-ориентированных ЯП с вполне понятным синтаксисом, используемых преимущественно для создания веб-приложений.
- "Объектно-ориентированный ЯП" опирается представление всего в виде «объектов», соединяющих внутри себя данные и методы их обработки, и последующую манипуляцию объектами. Такими ЯП являются, в частности, C++, Java, Objective-C и PHP.

Почитайте обучающие пособия базового уровня к нескольким ЯП. Если вы до сих пор не определились с тем, что учить, почитайте обучающие пособия к нескольким ЯП. Если что-то вас зацепит – попробуйте разобраться в том ЯП чуть лучше. Задача эта простая, так как различного рода обучающих материалов вводного уровня по ЯП в сети можно найти более чем достаточно :Python – замечательный язык для начинающих, который, впрочем, обладает достаточно мощным потенциалом, если научиться им пользоваться. Сфера использования – веб-приложения и игры.

Java – используется… ох, проще сказать, где этот ЯП не используется! Почти что все, от игр и до софта для банкоматов - Java.

HTML – не ЯП, а язык разметки, но для любого веб-разработчика просто необходим.

C – один из самых старых ЯП, не утративший актуальность и по сей день. C – не только мощный инструмент сам по себе, но еще и основа для более современных ЯП: C++, C# и Objective-C.

Часть 2
Начинаем с малого

- **Выучите базовые принципы ЯП.** Тут, конечно, все зависит от выбранного вами языка, однако есть у ЯП и общие моменты, исключительно для написания полезных программ важные. Чем раньше вы овладеете всеми этими понятиями и научитесь применять их на практике, тем лучше для вас и ваших навыков программиста. Итак, вот лишь некоторые из вышеупомянутых «моментов»:Переменные – в переменной можно хранить и вызывать меняющиеся данные. Переменными можно управлять, у переменных есть типы (очень упрощенно говоря – числа, символы и так далее), которыми и определяется тип хранящихся в переменной данных. Наименования переменных принято задавать такими, чтобы человек, читающий исходный код, мог получить представление о том, что в переменной хранится – так будет проще понять логику работы программы.
- Условные конструкции (они же – условные выражения) – это действие, которые выполняется в том случае, если выражение или конструкция истинно или ложно. Наиболее распространенной формой таких выражений можно назвать конструкцию "If-Then" (если-то). Если выражение истинно (например, если x = 5), то произойдет действие №1, а если ложно (x != 5), то действие №2.
- Функции – в разных языках программирования их называют по-разному: где-то это процедуры, где-то – методы, где-то – вызываемые единицы. По сути же, функции представляют собой мини-программы, входящие в состав большой программы. Функцию можно вызывать несколько раз, что позволяет программисту создавать сложные программы.
- Ввод данных – достаточно широко трактуемое понятие, присутствующее в почти что каждом ЯП. Суть его – обработка данных, введенных пользователем, и их хранение. То, как будут собраны данные, зависит от программы и доступных пользователю способов ввода данных (с клавиатуры, из файла и так далее). Понятие ввода данных тесно связано с понятием вывода данных – то есть того, каким образом данные будут возвращены пользователю (выведены на экран, записаны в файл и так далее).

Установите все необходимое ПО.

- Многим ЯП нужны компиляторы – программы, переводящие программный код в понятные для компьютера инструкции. Впрочем, есть и ЯП другого типа (вроде Python), в которых программы выполняются сразу, а компиляция их не требуется . У некоторых ЯП есть так называемые IDE (Integrated Development Environment, интегрированная среда разработки), в которые входят редактор кода, компилятор/интерпретатор и отладчик (дебаггер). Это дает программисту возможность работать над программой, образно выражаясь, по принципу одного окна. Также в IDE могут входить визуальные представления иерархий объектов и директорий.
- Есть и онлайновые редакторы кода. Эти программы подсвечивают синтаксис программного кода несколько иначе, а также предлагают разработчику доступ к ряду

Часть 3
Пишем свою первую программу

- **1**
- **Осваивайте базовые концепты по одному за раз.** Первой программой, которую пишут на любом ЯП, является классическая "Hello World". Она очень проста, вся ее суть – вывести на экран текст "Hello, World" (или его вариацию). Из этой программы люди, изучающие ЯП, должны усвоить синтаксис простейшей рабочей программы, а также способ вывода данных на экран. Меняя текст, можно заметить, как обрабатываются программой простые данные.

- **2**
- **Учитесь на основе разбора онлайн-примеров.** В сети для любого ЯП можно найти сотни, тысячи примеров программ, программок и просто кусков кода. Изучайте различные аспекты выбранного вами ЯП на основе этих примеров. Создавая свои собственные программы, опирайтесь на такие вот фрагменты знаний.

- 3
- **Выучите синтаксис ЯП.** Что такое синтаксис в контексте ЯП? Способ написания программ особым образом, понятным для компиляторов. У каждого ЯП собственные правила синтаксиса, хотя, конечно, встречаются и общие элементы. Изучение синтаксиса языка – один из краеугольных камней изучения ЯП. Довольно часто люди даже думают, что именно изучение синтаксиса сделает их программистами. В реальности, конечно, все не так – синтаксис суть основа, фундамент.

- 4
- **Экспериментируйте!** Каким именно образом? Вносите изменения в примеры программ и проверяйте получаемые результаты. Такой подход куда быстрее позволит вам понять, что дает результаты, а что – нет, чем если бы вы занимались по книге. Не бойтесь испортить или «сломать» программу, ибо исправление ошибок является одним из ключевых этапов в процессе разработки ПО. Да и потом, с первого раза написать работающую программу… ну, это почти что фантастика! [2]

- **5**
- **Начните работать с дебаггером.** Программные ошибки (баги) – это то, с чем вы обязательно столкнетесь, занявшись программированием. Ошибки будут везде, готовьтесь. Они могут быть безвредными, относительно безвредными или, увы, критическими, не дающими программе скомпилироваться. Процесс отладки программы является одним из ключевых этапов разработки ПО, повторим. Приучитесь к исправлению ошибок как можно раньше.Экспериментируя с программами, вы непременно что-то сломаете, и это хорошо. Умение исправить программу – один из самых ценных навыков для программиста.

- **6**
- **Не забывайте комментировать код.** Почти что все ЯП позволяют вносить в программный код комментарии – текст, не обрабатываемый компилятором. С помощью комментариев вы можете внести в программу простые и понятные описания того, что делает та или иная функция (и не только функция). Комментарии пригодятся не только вам самим (порой можно и в собственном коде запутаться), но и другим людям, совместно с которыми вы будете работать над программой.

Часть 4
Программируем регулярно

- 1
- **Программируйте ежедневно.** На что, чтобы овладеть языком программирования, уйдет много времени. Очень много. Даже Python, относительно простой ЯП, чей синтаксис можно освоить за день-другой, требует от любого, кто вознамерится овладеть им в совершенстве, сотен и тысяч часов работы. Программирование – навык, а потому тот, кто хочется овладеть таким навыком в совершенстве, должен регулярно практиковаться. Старайтесь программировать каждый день, пусть даже по часу перед сном, если нет другой возможности.

- 2
- **Ставьте вашим программам цели.** Устанавливая сложные, но все же достижимые цели, вы научитесь решать проблемы, находить решения, бороться со сложностями. Например, представьте себе простую программу – скажем, калькулятор, а потом подумайте, как вы ее напишете. Примените на практике все то, чему вы уже научились.

- **3**
- **Обменивайтесь опытом и читайте чужие программы.** Вокруг каждого ЯП собралось огромное сообщество. Если вы присоединитесь к соответствующему сообществу, то очень сильно себе поможете, так как получите доступ к более чем качественному обучающему материалу. Чтение чужого кода может вдохновить вас, может придать вам сил и помочь лучше понять те особенности программирования, на которых вы до этого буксовали.[3]Форумы и онлайн-сообщества, посвященные программированию на выбранном вами ЯП – вот что следует искать в первую очередь. Не надо постоянно лишь задавать вопросы, участвуйте в жизни сообщества полноценно – это, как никак, места, где люди сотрудничают друг с другом, а не проводят бесплатные обучающие курсы. Иными словами, не стесняйтесь просить о помощи, но и не сидите, сложа руки!
- Наработав более-менее приличный опыт, принимайте участие в хакатонах или иных аналогичных мероприятиях – соревнования, где надо успеть написать особую программу в отведенное время. Такие мероприятия увлекательны и полезны.

- **4**
- **Веселитесь.** Делайте то, что еще не умеете делать. Изучайте способы решения проблем, а затем используйте их по-своему. Старайтесь не радоваться тому, что «программа работает и ладно» - делайте все возможное, чтобы программа работала безукоризненно!

Часть 5
Расширяем кругозор

- **1**
- **Запишитесь на курсы.** Университеты, колледжи и образовательные центры (и не только) проводят курсы и семинары по программированию, что может стать отличным вариантом для начинающих. Сами посудите, где еще новички смогут вживую пообщаться с матерыми специалистами?

- **2**
- **Читайте тематические книги.** Как вы получите к книгам доступ – дело ваше, суть в том, что по любому ЯП можно найти сотни книг разной степени полезности. Конечно, ваши знания не должны быть сугубо книжными, это факт. Тем не менее, и в книгах есть своя

- **3**
- **Учите логику и математику.** Программирование во многом завязано на базовой арифметике, но и более сложные моменты могут пригодиться, особенно в тех случаях, когда вы занимаетесь алгоритмами или пишете сложную программу. Впрочем, скорее всего, если только вы не будете зарываться в сложные области, сложная математика вам не понадобится, зато понадобится логика, в частности – компьютерная, так как с ее помощью вы сможете лучше понять, как надо решать задачи, возникающие по ходу работы

- 4
- **Никогда не прекращайте программировать.** Есть популярная теория «10 тысяч часов», гласящая, что мастерство приходит спустя 10000 часов, проведенных за тем или иным занятием. Точное количество часов как точка достижения мастерства – вопрос, конечно, спорный, но в целом теория верна – мастерство суть результат приложенного труда и затраченного времени. Не опускайте руки, и однажды вы станете экспертом.[4]

- 5
- **Выучите другой ЯП.** Конечно, овладение даже одним ЯП будет вам только в плюс, однако многие программисты не останавливаются на достигнутом и учат несколько языков. Будет неплохо, если второй или третий ЯП, который вы выберете, будет дополнять первый – тогда можно будет создать даже еще более сложные и интересные программы. Само собой, учить новое надо лишь тогда, когда старым вы уже овладели на приличном уровне.Есть все шансы, что второй язык вы выучите быстрее первого, но это вполне понятно, так как многие концепты программирования распространены более чем широко, особенно у «родственных» языков.

Часть 6
Применяем полученные навыки

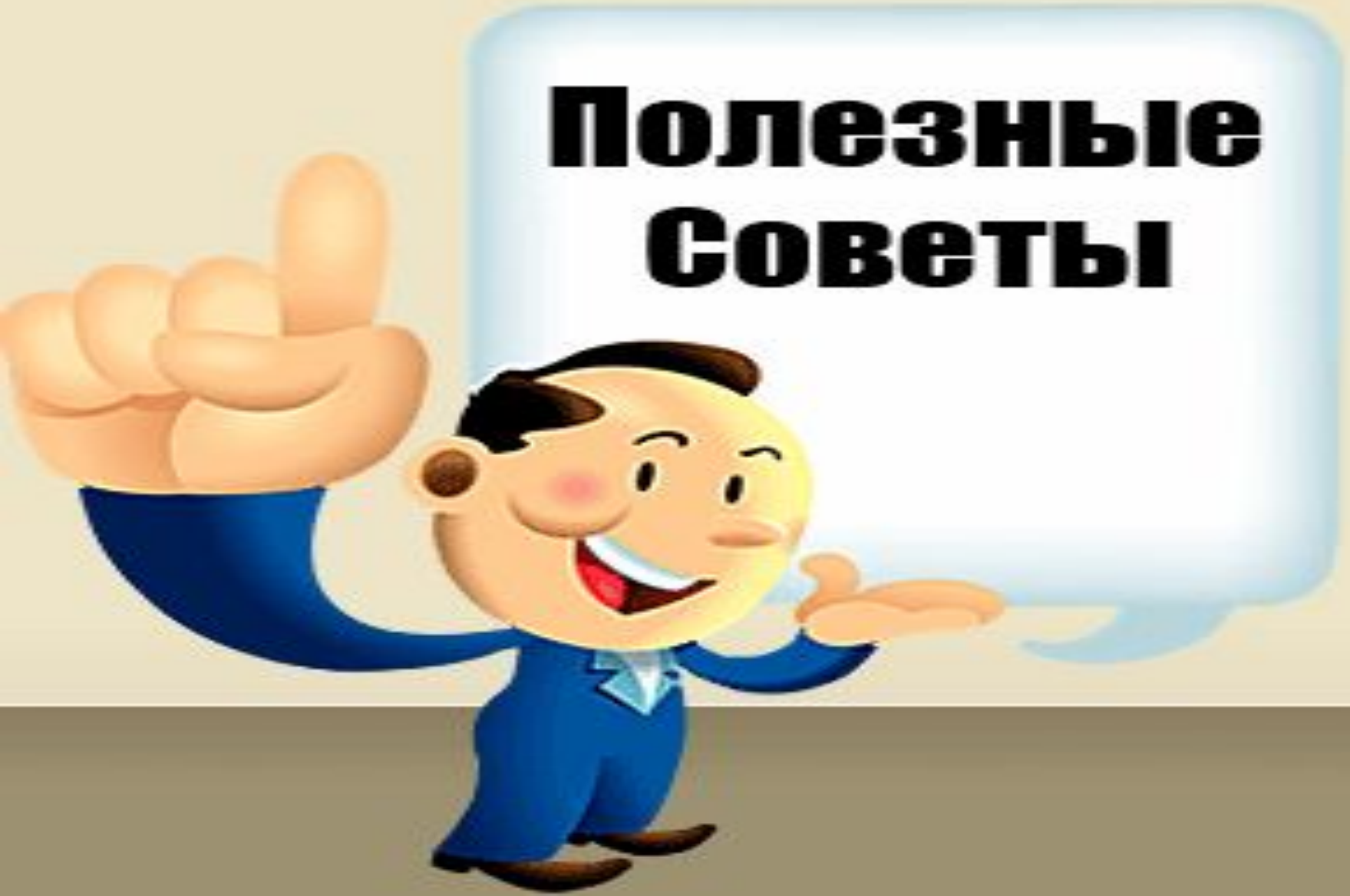
- 1
- **Получите университетское образование.** Этот пункт обязательным не является, однако годы учебы могут открывать что-то новое (а могут и не открыть) и познакомить вас с нужными людьми (тоже не факт). Опять же – этот шаг не обязателен, есть много успешных программистов, которые диплома о высшем образовании не имеют.

- 2
- **Соберите портфолио.** Создавая программы и развиваясь как специалист, обязательно откладывайте лучшие образцы ваших работ отдельно – в портфолио. Именно портфолио вы будете показывать рекрутерам и интервьюерам в качестве примеров того, на что вы способны. Те проекты, которые вы выполняли самостоятельно и по собственной инициативе, можно добавлять в портфолио без размышлений, а вот те, над которыми вы работали, будучи сотрудником той или иной компании, лишь с разрешения соответствующих лиц.

- **3**
- **Станьте фрилансером.** Программисты (особенно те, которые специализируются на мобильных приложениях) сейчас нарасхват. Выполните пару-тройку проектов как фрилансер – это и для портфолио полезно, и для кошелька, и для опыта.
-

- **4**
- **Разработайте собственный программный продукт.** Платным он будет или нет – решать вам. В конце концов, вовсе не обязательно работать на кого-то, чтобы заработать программированием деньги! Если вы умеете писать программы и продавать их, то дело почти что в шляпе! Главное – не забывать предоставлять пользователям поддержку после релиза программы.Модель Freeware (бесплатное распространение) популярна в случае небольших программ и утилит. В таком случае разработчик ничего не зарабатывает в финансовом плане, зато получает репутацию и узнаваемое имя среди коллег по цеху.

Советы



Хотите создавать игры? Учите Python, C# и Java. Из этой троицы C# дает наилучшую производительность, Python самый простой, а Java запустится на всех ОС без особых проблем

- Изучайте исходный код программ. Зачем, сами подумайте, изобретать велосипед, когда можно взять готовый велосипед и просто его улучшить? Главное – понимать, что именно вы программируете.

- Изучая что-то новое, полезно будет самостоятельно это реализовать, затем внести изменения, попытаться угадать результаты и, как следствие, приблизиться к понимаю сути.
- Используйте современные интерфейсы и актуальные версии ЯП.

- Дополнительные материалы – ваши друзья. Нет ничего плохо в том, что вы что-то забыли или не запомнили. Всему свое время, не переживайте. Главное – знать. Где подсмотреть!
- Хорошей практикой будет обучение других – это позволит вам не только лучше понять материал, но и взглянуть на него со стороны.

Где работать ?

- Наиболее популярная область работы — разработка и создание программного обеспечения, используемого в текстовых редакторах, бухгалтерских программах, играх, базах данных и даже системах видеонаблюдения. Востребованы сегодня и специалисты, адаптирующие уже готовые программы (в частности 1С: Бухгалтерия) под особенности конкретного предприятия. Не останутся без работы и web-разработчики. Первой ступенькой карьеры может стать должность программиста-стажера. Необходимо знать языки высокого уровня, желательно иметь

Зарплаты

- Стажер, помощник программиста

  30-40 000 р.

- Специалист

  80-90 000 р.

Ведущий программист
110 000 р.

Где учиться ?

- Какой бы вуз вы ни выбрали, учиться на программиста будет сложно. Предстоит получить серьезную математическую подготовку, изучить алгоритмические языки и программирование, методы и средства защиты компьютерной информации.

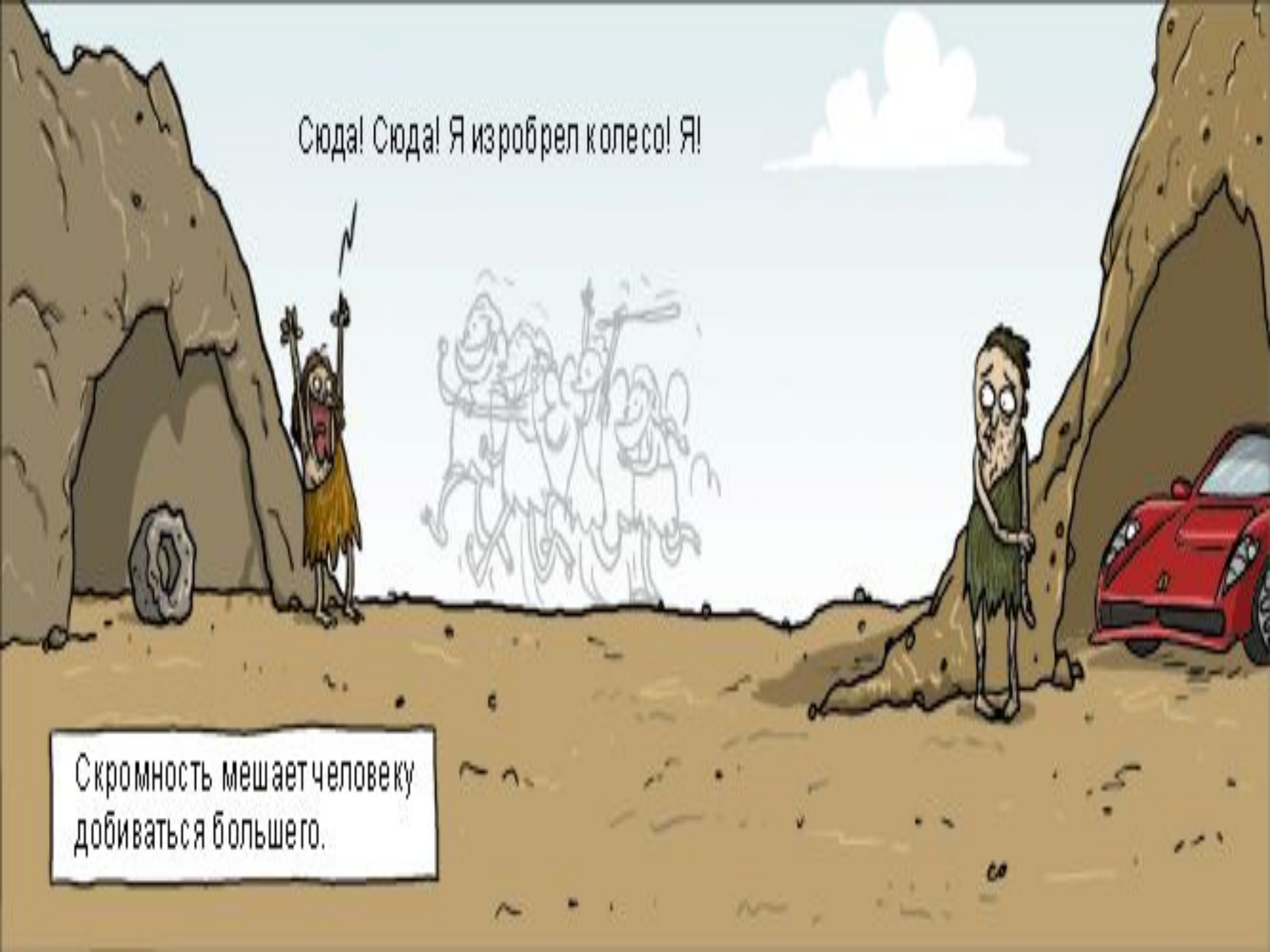
  Придется «подружиться» с рядом специальных дисциплин, в числе которых: структуры и алгоритмы обработки данных, функциональное, логическое и объектно-ориентированное программирование. Много времени придется проводить непосредственно за компьютером.


КАК ПОЛЬЗОВАТЕЛИ ВИДЯТ ПРОГРАММИСТОВ
КАК ПРОГРАММИСТЫ ВИДЯТ ПОЛЬЗОВАТЕЛЕЙ
UGH!
OOGH!
UG UG!!
TOC TOC

Что ждет программиста?

Быть программистом круто!
Это лучшая профессия в мире!

- Лет через 8-10 профессиональной деятельности, если ты все эти годы учился, ты становишься желанным гостем в любой ИТ-компании. Наконец-то, появляется **большая зарплата**, о которой так давно мечтал и **комфортные условия работы**.
- Программистами, да и в целом, ИТ-шниками любого направления, становятся только самые умные люди. Еще более умных людей можно встретить только в научной среде.


Сюда! Сюда! Я изробрел колесо! Я!
Скромность мешает человеку добиваться большего.


01011 10100
110 1011001

Насколько программисту нужно математика

?

5200
150647 + 4564
65600 + 850000
8000 + 720000
72 + 950
456700
96.000
Z
A
Y
G
40
56700
456000
56000
75000
8000
456.000
62000 + 62100

- Зависит от того, что называть математикой. Умение складывать числа тоже математика, и такое знание крайне желательно. А, например, без понимания того, что именно доказал Перельман, вполне можно программировать. Любые попытки провести грань, до которой необходимо знать математику, чтобы стать программистом, заведомо обречены на провал. Одно можно сказать наверняка — умение оперировать абстрактыми понятиями (одно из основных для математика) несомненно помогает и программистам в их работе.
-

Как и в любом деле, все зависит от решаемой задачи.

- слова программиста из яндекса
- Когда я участвовал в разработке рекомендательной и репутационных систем, математика была очень нужна. Приходилось придумывать и разрабатывать алгоритмы, использующие интегральное и дифференциальное счисление, находить экстремумы, строить регрессии, вводить метрики для определения близости в многомерном пространстве. И все ради каких-то лишних сотых долей в RMSE рекомендательного движка.

- Последние годы я занимаюсь разработкой высокопроизводительных бэкендов, работающих в режиме 24/7, обслуживающих миллионы онлайн-пользователей, держащих сотни тысяч постоянных соединений. Теперь мне достаточно знаний основ теории алгоритмов, алгоритмической сложности, теперь главное — надежный, поддерживаемый, расширяемый, быстрый код.

И я бы не сказал, что из-за меньшей «математичности» работы я получаю от нее меньшее удовольствие и признание.

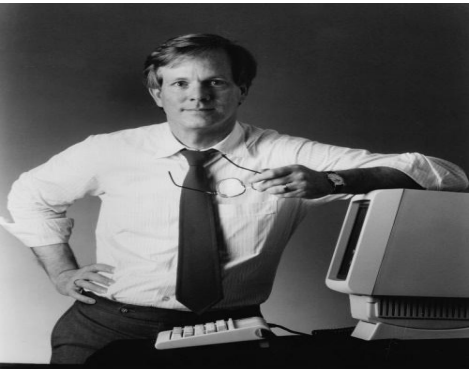
В любом случае, программист, впрочем, как и обычный человек, должен знать и любить математику!

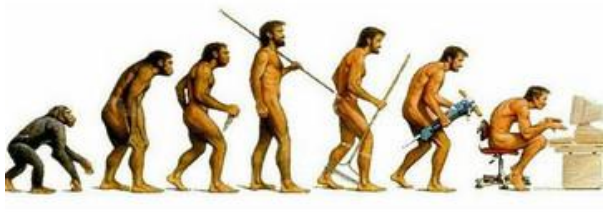

Я ГОВОРИЛ ЧТО ПРОГРАММИСТАМ НЕ НУЖНА МАТЕМАТИКА
Я ещё никогда в жизни так не ошибался

История языков программирования

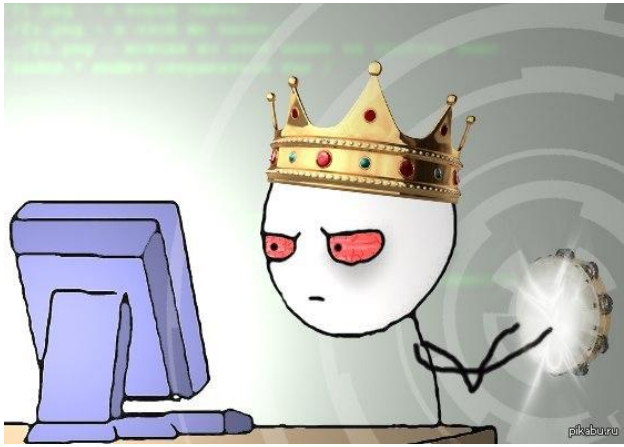
- Одной из самых революционных идей приведших к созданию автоматических цифровых вычислительных машин, была высказанная в 20-х годах 19 века Чарльзом Бебиджем мысль о предварительной записи порядка действия машины для последующей автоматической реализации вычислений – программе. И, хотя использованная Бебиджем запись программы на перфокартах, придуманная для управления такими станками французским изобретателем Жозефом Мари Жаккаром, технически не имеет ничего общего с современными приемами хранения программ в ПК, принцип здесь по существу один.

С этого момента начинается история программирования.



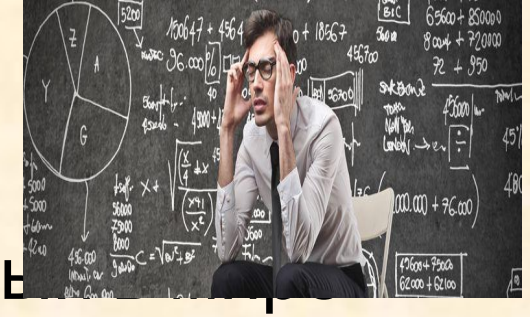


11111011101



Аду Левлейс, современницу Бебиджа, называют первым в мире программистом. Она теоретически разработала некоторые приемы управления последовательностью вычислений, которые используются в программировании и сейчас. Ею же была описана и одна из важнейших конструкций практически любого современного языка программирования – цикл.

- Революционным моментом в истории языков программирования стало появление системы кодирования машинных команд с помощью специальных символов, предложенной Джоном Моучли.
  Система кодирования, предложенная им, вдохновила одну из его сотрудниц Грейс Мюррей Хоппер. При работе на компьютере «Марк-1» ей и ее группе пришлось столкнуться со многими проблемами и все, что ими придумано, было впервые. В частности они придумали подпрограммы. И еще одно фундаментальное понятие техники программирования впервые ввели Хоппер и ее группа – «отладка».
  В конце 40-х годов Дж. Моучли создал систему под названием «Short Code», которая являлась примитивным языком программирования высокого уровня. В ней программист записывал решаемую задачу в виде математических формул, а затем, используя специальную таблицу, переводил символ за символом, преобразовывал эти формулы в двухлитерные коды. В дальнейшем специальная программа компьютера превращала эти коды в двоичный машинный код. Система, разработанная Дж. Моучли, считается одним из первых примитивных интерпретаторов.



Уже в 1951 г. Хоппер создала первый [illegible] компилятор и ею же был введен сам этот термин. Компилятор Хоппер осуществлял функцию объединения команд и в ходе трансляции производил организацию подпрограмм, выделение памяти компьютера, преобразование команд высокого уровня (в то время псевдокодов) в машинные команды. «Подпрограммы находятся в библиотеке (компьютера), а когда вы подбираете материал из библиотеки – это называется компиляцией» – так она объясняла происхождение введенного ею термина.

- В 1954 году группа под руководством Г. Хоппер разработала систему, включающую язык программирования и компилятор, которая в дальнейшем получила название Math-Matic. После удачного завершения работ по созданию Math-Matic Хоппер и ее группа принялись за разработку нового языка и компилятора, который позволил бы пользователями программировать на языке, близком к обычному английскому. В 1958 г. появился компилятор Flow-Matic. Компилятор Flow-Matic был первым языком для задач обработки коммерческих данных. Разработки в этом направлении привели к созданию языка Кобол (COBOL – Common Business Oriented Language). Он был создан в 1960 году. В этом языке по сравнению с Фортраном и Алголом, слабее развиты математические средства, но зато хорошо развиты средства обработки текстов, организация вывода данных в форме требуемого документа. Он задумывался как основной язык для массовой обработки данных в сферах управления и бизнеса.

- Середина 50-х годов характеризуется стремительным прогрессом в области программирования. Роль программирования в машинных командах стала уменьшаться. Стали появляться языки программирования нового типа, выступающие в роли посредника между машинами и программистами. Первым и одним из наиболее распространенных был Фортран (FORTRAN, от FORmula TRANslator – переводчик формул), разработанный группой программистов фирмы IBM в 1954 году (первая версия). Этот язык был ориентирован на научно-технические расчеты математического характера и является классическим языком программирования при решении на ПК математических и инженерных задач.
  Для первых языков программирования высокого уровня предметная ориентация языков была характерной чертой. Особое место среди языков программирования занимает Алгол, первая версия которого появилась в 1958 году. Одним из разработчиков Алгола был «отец» Фортрана Джон Бэкус. Название языка ALGOrithmic Language подчеркивает то обстоятельство, что он предназначен для записи алгоритмов. Благодаря четкой логической структуре Алгол стал стандартным средством записи алгоритмов в научной и технической литературе.

- В середине 60-х годов Томас Курц и Джон Камени (сотрудники математического факультета Дартмунтского колледжа) создали специализированный язык программирования, который состоял из простых слов английского языка. Новый язык назвали «универсальным символическим кодом для начинающих» (Beginner All-Purpose Symbolic Instruction Code, или, сокращенно, BASIC). Годом рождения нового языка можно считать 1964. Сегодня универсальный язык Бейсик (имеющий множество версий) приобрел большую популярность и получил широкое распространение среди пользователей ПК различных категорий во всем мире. В значительно мере этому способствовало то, что Бейсик начали использовать как встроенный язык персональных компьютеров, широкое распространение которых началось в конце 70-х годов. Однако Бейсик неструктурный язык, и поэтому он плохо подходит для обучения качественному программированию. Справедливости ради следует заметить, что последние версии Бейсика для ПК (например, QBasic) стали более структурными и по своим изобразительным возможностям приближаются к таким

- Разработчики ориентировали языки на разные классы задач, в той или иной мере привязывали их к конкретной архитектуре ПК, реализовывали личные вкусы и идеи. В 60-е годы были предприняты попытки преодолеть эту «разноголосицу» путем создания универсального языка программирования. Первым детищем этого направления стал PL/1 (Programm Language One), разработанный фирмой IBM в 1967 году. Этот язык претендовал на возможность решать любые задачи: вычислительные, обработки текстов, накопления и поиска информации. Однако он оказался слишком сложным, транслятор с него – недостаточно оптимальным и содержал ряд невыявленных ошибок. Однако линия на универсализацию языков была поддержана. Старые языки были модернизированы в универсальные варианты: Алгол-68 (1968 г.), Фортран-77. Предполагалось, что подобные языки будут развиваться и усовершенствоваться, станут вытеснять все остальные. Однако ни одна из этих попыток не

- Язык ЛИСП появился в 1965 году. Основным в нем служит понятие рекурсивно определенных функций. Поскольку доказано, что любой алгоритм может быть описан с помощью некоторого набора рекурсивных функций, то ЛИСП по сути является универсальным языком. С его помощью ПК может моделировать достаточно сложные процессы, в частности – интеллектуальную деятельность людей.
  Пролог разработан во Франции в 1972 году для решения проблем «искусственного интеллекта». Пролог позволяет в формальном виде описывать различные утверждения, логику рассуждений и заставляет ПК давать ответы на заданные вопросы.
  Значительным событием в истории языков программирования стало создание в 1971 году языка Паскаль. Его автор – швейцарский ученый Никлаус Вирт. Вирт назвал его в честь великого французского математика и религиозного философа XVII века Блеза Паскаля, который изобрел первое суммирующее устройство, именно поэтому новому языку было присвоено его имя. Этот язык первоначально разрабатывался как учебный язык структурного программирования, и, действительно, сейчас он является одним из основных языков обучения программированию в школах и вузах.

В 1975 году два события стали вехами в истории программирования – Билл Гейтс и Пол Аллен заявили о себе, разработав свою версию Бейсика, а Вирт и Йенсен выпустили классическое описание языка «Pascal User Manual and Report».

- Не менее впечатляющей, в том числе и финансовой, удачи добился Филип Кан, француз, разработавший в 1983 году систему Турбо-Паскаль. Суть его идеи состояла в объединении последовательных этапов обработки программы – компиляции, редактирования связей, отладки и диагностики ошибок – в едином интерфейсе. Турбо-Паскаль – это не только язык и транслятор с него, но еще и операционная оболочка, позволяющая пользователю удобно работать на Паскале. Этот язык вышел за рамки учебного предназначения и стал языком профессионального программирования с универсальными возможностями. В силу названных достоинств Паскаль стал источником многих современных языков программирования. С тех пор появилось несколько версий Турбо-Паскаля, последняя – седьмая.
  Фирма Borland/Inprise завершила линию продуктов Турбо-Паскаль и перешла к выпуску системы визуальной разработки для Windows – Delphi.

- Большой отпечаток на современное программирование наложил язык Си (первая версия – 1972 год), являющийся очень популярным в середе разработчиков систем программного обеспечения (включая операционные системы). Этот язык создавался как инструментальный язык для разработки операционных систем, трансляторов, баз данных и других системных и прикладных программ. Си сочетает в себе как черты языка высокого уровня, так и машинно-ориентированного языка, допуская программиста ко всем машинным ресурсам, чего не обеспечивают такие языки как Бейсик и Паскаль.
  Период с конца 60-х до начала 80-х годов характеризуется бурным ростом числа различных языков программирования, сопровождавшим кризис программного обеспечения. В январе 1975 года Пентагон решил навести порядок в хаосе трансляторов и учредил комитет, которому было предписано разработать один универсальный язык. В мае 1979 года был объявлен победитель – группа ученых во главе с Жаном Ихбиа. Победивший язык окрестили Ада, в честь Огасты Ады Левлейс. Этот язык предназначен для создания и длительного (многолетнего) сопровождения больших программных систем, допускает возможность параллельной обработки, управления процессами в реальном времени.

В течение многих лет программное обеспечение строилось на основе операционных и процедурных языков, таких как Фортран, Бейсик, Паскаль, Ада, Си. По мере эволюции языков программирования получили широкое распространение и другие, принципиально иные, подходы к созданию программ