ЯЗЫКИ ПРОГРАММИРОВАНИЯ. ОСНОВНЫЕ ПОНЯТИЯ.

Основы алгоритмизации и программирования.

ЯЗЫК ПРОГРАММИРОВАНИЯ

- *Язык программирования* – это формальная знаковая система, предназначенная для записи компьютерных программ.
- *Язык программирования* определяет набор лексических, синтаксических и семантических правил, задающих внешний вид программы и действий, которые выполнит ЭВМ под управлением.

НАЗНАЧЕНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ

- Язык программирования предназначен для написания компьютерных программ, которые применяются для передачи компьютеру инструкций по выполнению того или иного вычислительного процесса.

СТАНДАРТИЗАЦИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

- Язык программирования может быть представлен в виде набора спецификаций, определяющих его синтаксис и семантику.
- Для многих широко распространенных языков программирования созданы *международные стандарты.*
- Специальные организации проводят регулярное обновление и публикацию спецификаций и формальных определений соответствующего языка.

ОРГАНИЗАЦИИ, ЗАНИМАЮЩИЕСЯ ВОПРОСАМИ СТАНДАРТИЗАЦИИ

- Американский национальный институт стандартов ANSI (American National Standards Institute);
- Институт инженеров по электротехнике и электронике IEEE (Institute of Electrical and Electronic Engineers);
- Организация международных стандартов ISO (International Organization for Standardization).

ТИПЫ ДАННЫХ

Особая система, по которой данные организуются в программе – это *система типов языка программирования.*

Разработка и изучение систем типов данных известна под названием теория типов.

Языки могут быть классифицированы как системы со статической типизацией и динамической типизацией.

ТИПЫ ДАННЫХ

- **Статическая типизация** означает, что типы определяются на этапе компиляции. То есть ошибки в типах будут видны *еще до того, как программа запустится.*
- В языках с **динамической типизацией** типы определяются *во время выполнения программы.*

Например, в Python (динамическая типизация) можно сделать вот так:

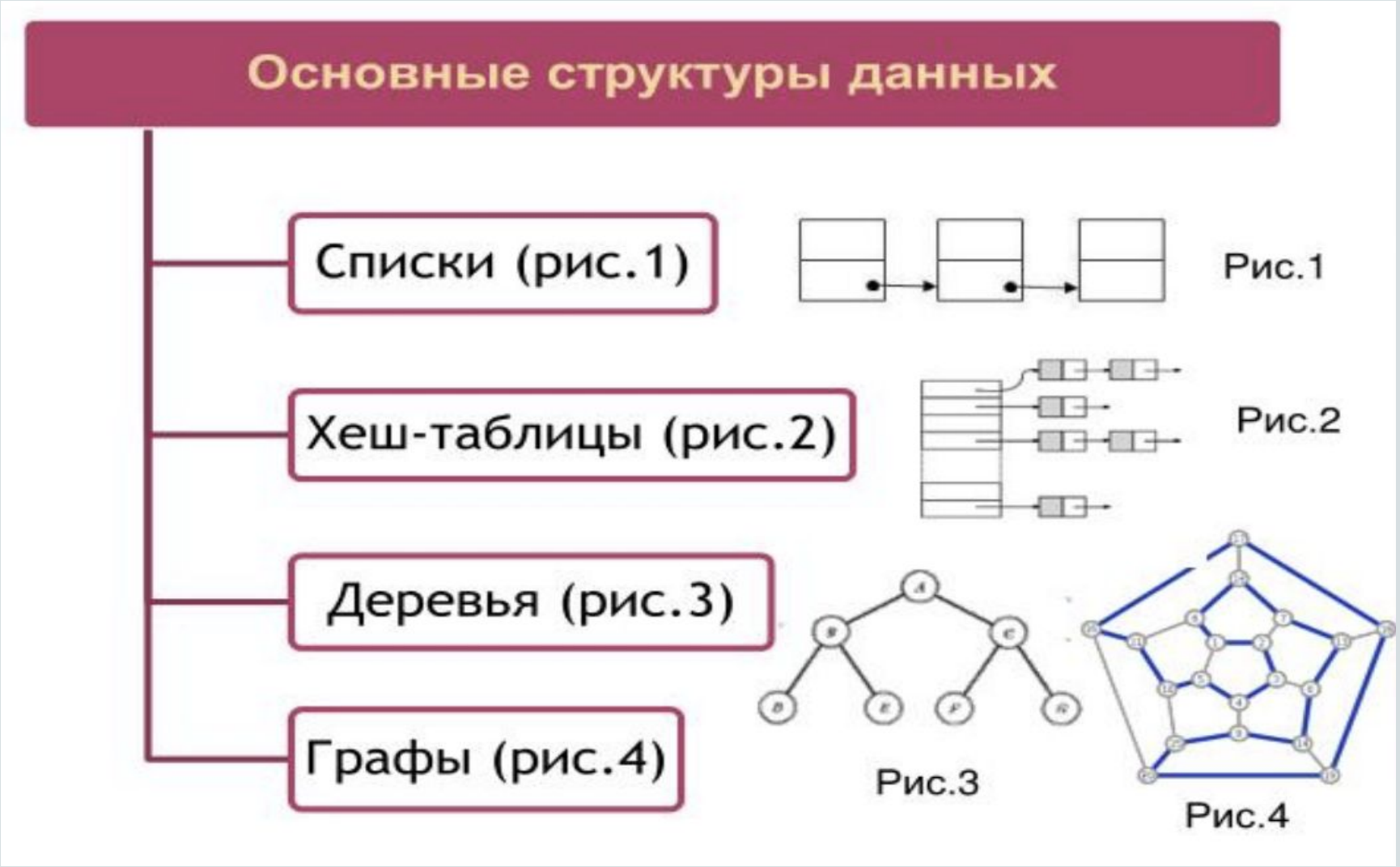
```
x = 5
```

а потом так:

```
x = “text”
```

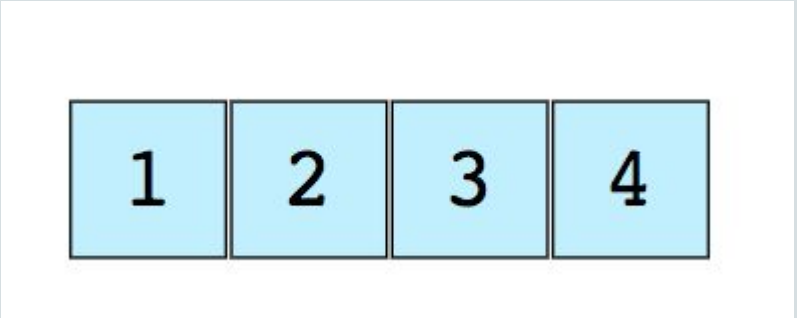
И язык не будет возражать. В Java (статическая типизация) так сделать нельзя.

СТРУКТУРЫ ДАННЫХ



МАССИВЫ

- *Массив* — это простейшая и наиболее распространенная структура данных. Другие структуры данных, например, стеки и очереди, производны от массивов.
- Каждому элементу данных присваивается положительное числовое значение, именуемое **индексом** и соответствующее положению этого элемента в массиве. В большинстве языков программирования элементы в массиве нумеруются с 0 (это ноль).

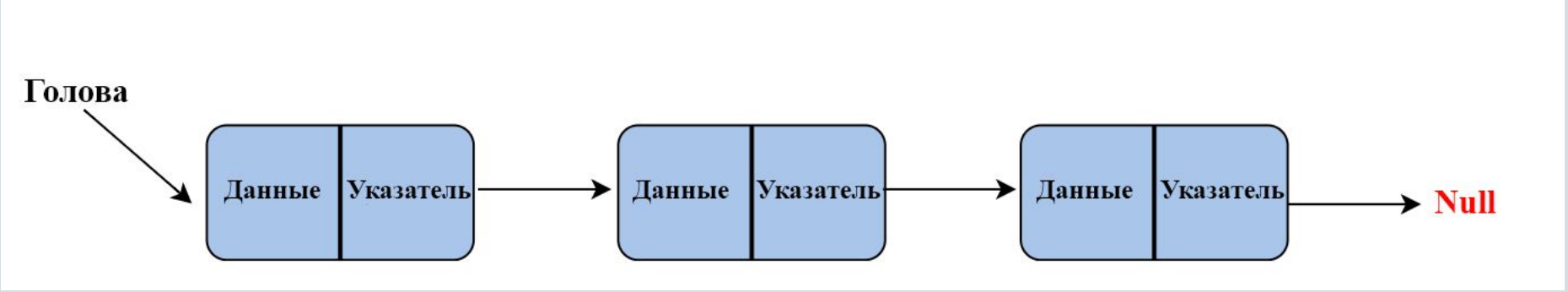


Существуют массивы двух типов:

- *Одномерные* (такие, как показанный выше)
- *Многомерные* (массивы, в которые вложены другие массивы)

СПИСКИ

- ***Связный список*** —важная линейная структура данных, на первый взгляд напоминающая *массив*. Однако, связный список отличается от массива по выделению памяти, внутренней структуре и по тому, как в нем выполняются базовые операции вставки и удаления.
- ***Связный список*** напоминает цепочку узлов, в каждом из которых содержится информация: например, данные и указатель на следующий узел в цепочке. Есть головной указатель, соответствующий первому элементу в связном списке, и, если список пуст, то он направлен просто на null (ничто).



ГРАФЫ

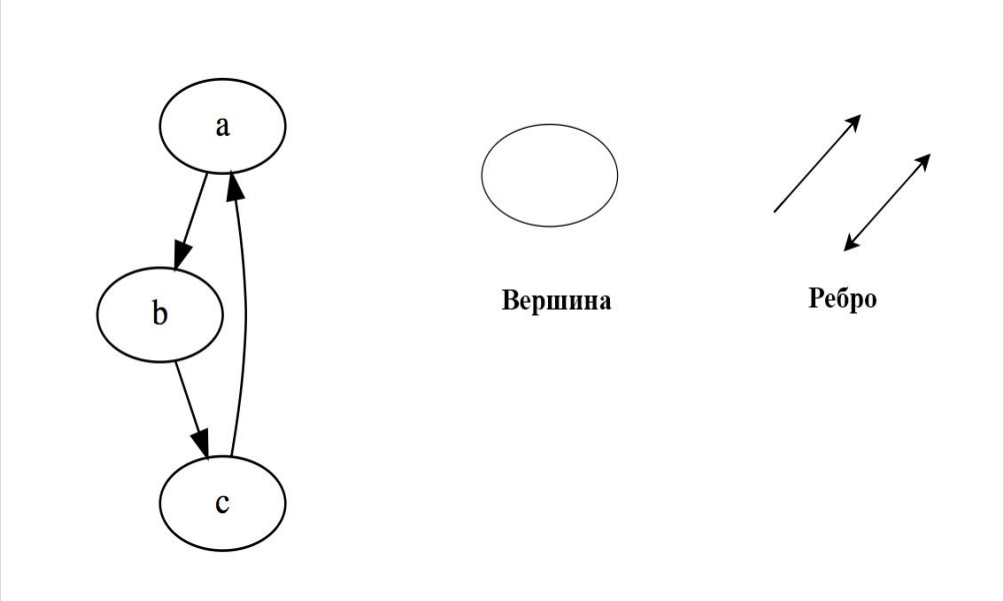
- ***Граф*** — это множество узлов, соединенных друг с другом в виде сети. Узлы также называются вершинами. Пара (x,y) называется ребром, это означает, что вершина x соединена с вершиной y. Ребро может иметь вес/стоимость — показатель, характеризующий, насколько затратен переход от вершины x к вершине y.

Типы графов:

- Неориентированный граф
- Ориентированный граф

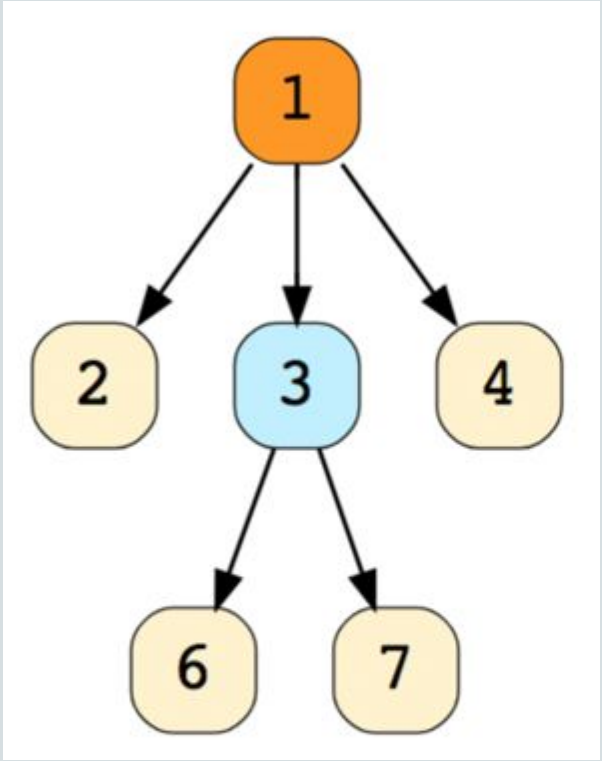
Распространенные алгоритмы обхода графа:

- Поиск в ширину
- Поиск в глубину



ДЕРЕВЬЯ

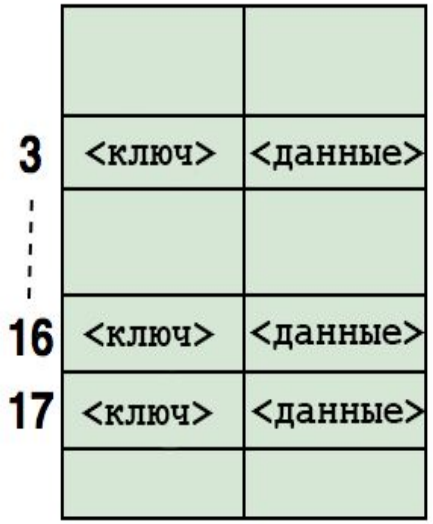
- ***Дерево*** — это иерархическая структура данных, состоящая из вершин (узлов) и ребер, которые их соединяют. Деревья подобны графам, однако, ключевое отличие дерева от графа таково: в дереве не бывает циклов.
- Деревья широко используются в области искусственного интеллекта и в сложных алгоритмах, выступая в качестве эффективного хранилища информации при решении задач.



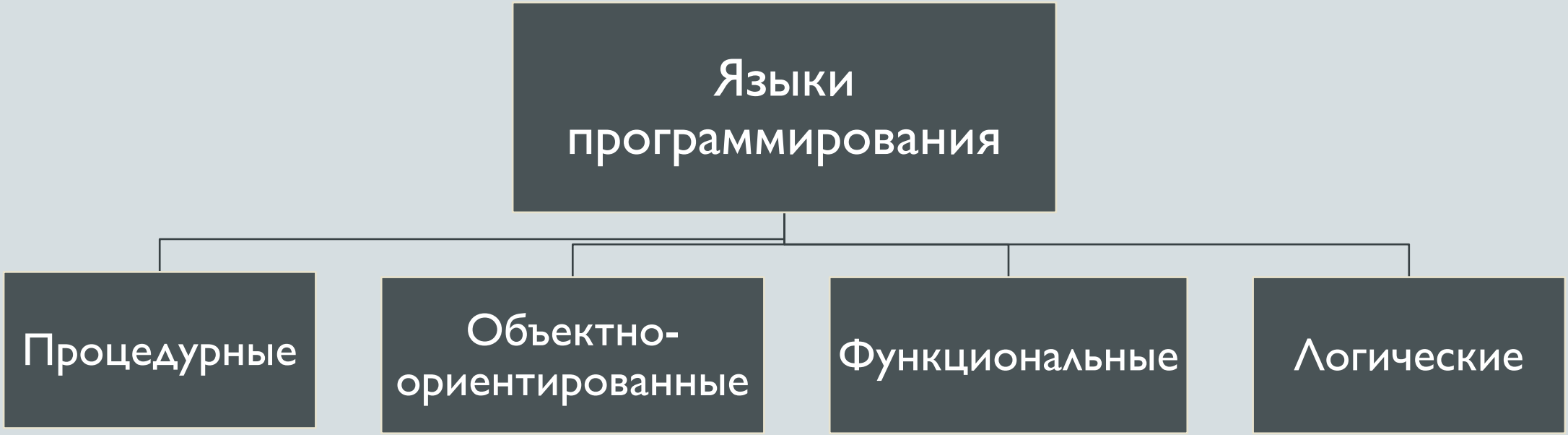
ХЕШ-ТАБЛИЦА

- ***Хеширование*** — это процесс, применяемый для уникальной идентификации объектов и сохранения каждого объекта по заранее вычисленному индексу, именуемому его «ключом». Таким образом, объект хранится в виде «ключ-значение», а коллекция таких объектов называется «словарь». Каждый объект можно искать по его ключу. Существуют разные структуры данных, построенные по принципу хеширования, но чаще всего из таких структур применяется **хеш-таблица**.

Как правило, хеш-таблицы реализуются при помощи массивов.



ОСНОВНЫЕ КЛАССЫ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ



ОСНОВНЫЕ КЛАССЫ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

- ***Процедурный язык программирования***. Особенность таких языков программирования состоит в том, что задачи разбиваются на шаги и решаются шаг за шагом. Используя процедурный язык, программист определяет языковые конструкции для выполнения последовательности алгоритмических шагов.
- ***Функциональное программирование*** объединяет разные подходы к определению процессов вычисления на основе достаточно строгих абстрактных понятий и методов символьной обработки данных.

ОСНОВНЫЕ КЛАССЫ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

- ***Логическое программирование*** – парадигма программирования, основанная на автоматическом доказательстве теорем, а также раздел дискретной математики, изучающий принципы логического вывода информации на основе заданных фактов и правил вывода.
- ***Объектно-ориентированный язык программирования***– язык, построенный на принципах объектно-ориентированного программирования. В основе концепции объектно-ориентированного программирования лежит понятие объекта – некой субстанции, которая объединяет в себе поля и методы.

ТРАНСЛЯТОРЫ

- **Транслятор** – обрабатывающая программа, предназначенная для преобразования исходной программы в объектный модуль.

Транслятор обычно выполняет также диагностику ошибок, формирует словари идентификаторов, выдаёт для печати тексты программы и т. д.

- **Трансляция** — преобразование программы, представленной на одном из языков программирования, в программу на другом языке и, в определённом смысле, равносильную первой.

Виды трансляторов: Адресный, диалоговый, однопроходной, многопроходной, обратный, оптимизирующий, текстовый, синтаксически-ориентированный (синтаксически-управляемый).

ИНТЕРПРЕТАТОРЫ

Интерпретатор — программа (иногда аппаратное средство), анализирующая команды или операторы программы и тут же выполняющая их.

Типы интерпретаторов:

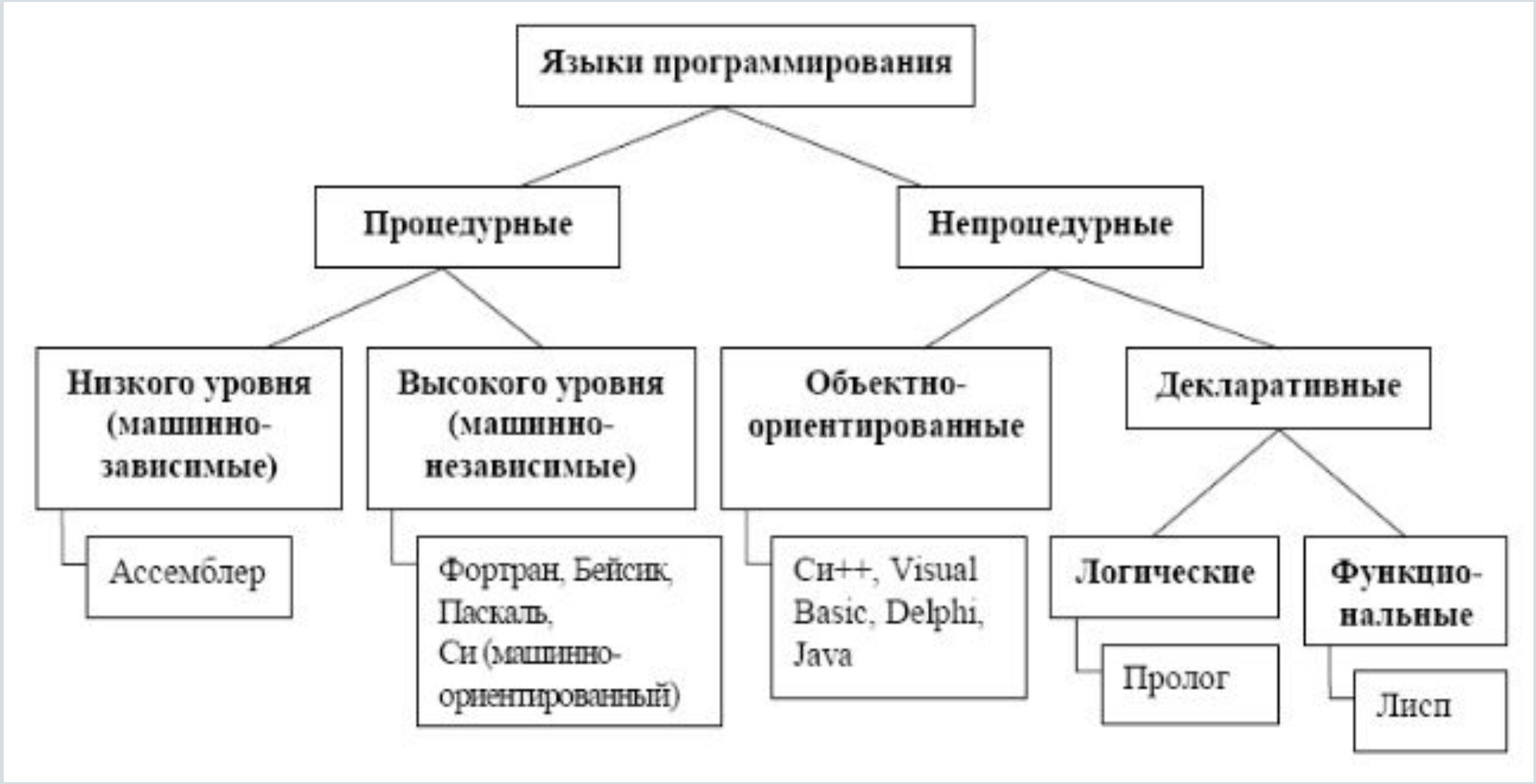
- простой интерпретатор;
- интерпретатор компилирующего типа.

Алгоритм работы простого интерпретатора:

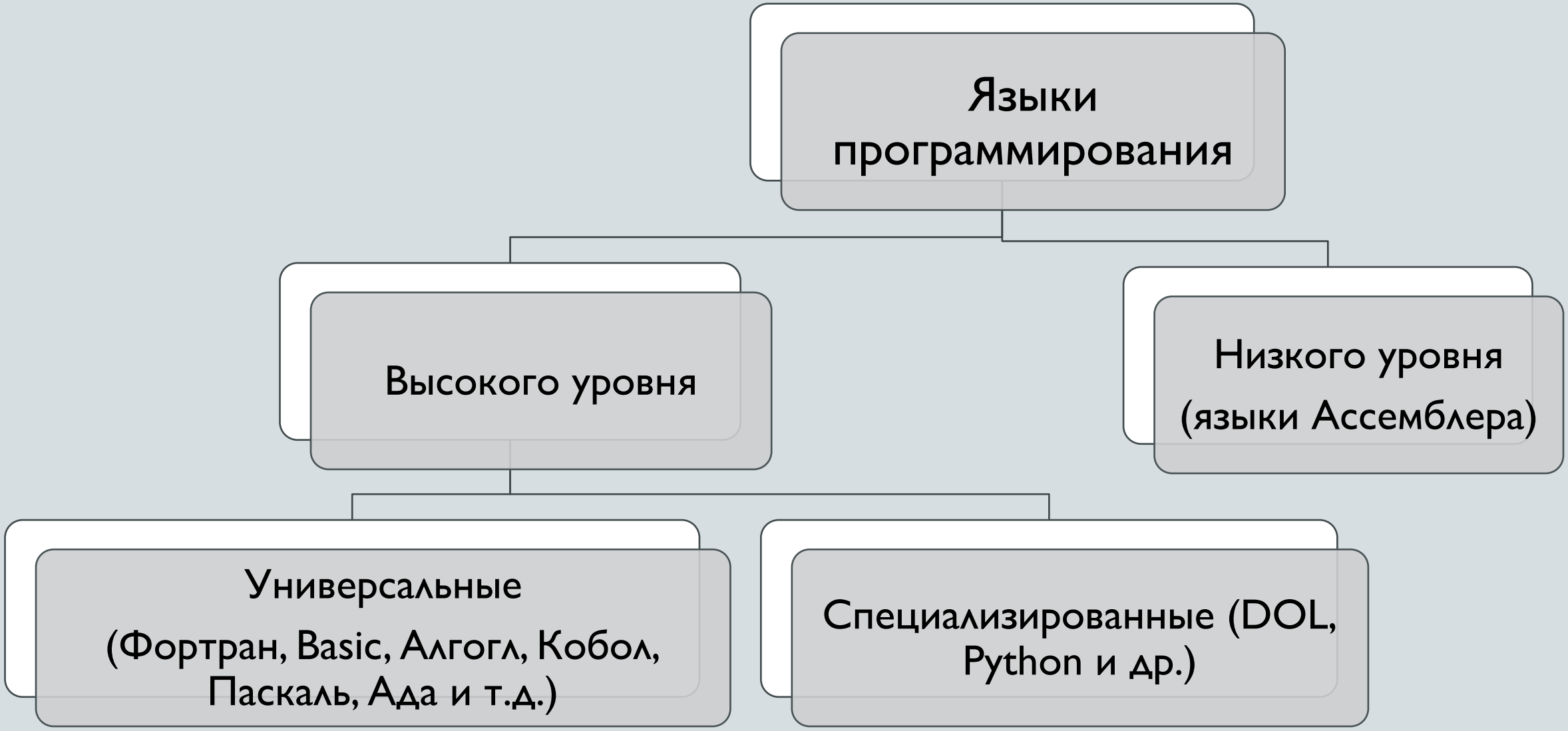
- прочитать инструкцию;
- проанализировать инструкцию и определить соответствующие действия;
- выполнить соответствующие действия;
- если не достигнуто условие завершения программы, перейти к пункту 2.

КОМПИЛЯТОРЫ

- **Компилятор** – программа или техническое средство, выполняющее компиляцию.
- **Компиляция** — трансляция программы на машинный язык или близкий к машинному.
- ***Виды компиляторов:*** Векторизующий, гибкий, диалоговый, инкрементальный, интерпретирующий (пошаговый), компилятор компиляторов, отладочный, резидентный, универсальный, самокомпилируемый.


Языки программирования
Процедурные
Непроцедурные
Низкого уровня (машинно-зависимые)
Высокого уровня (машинно-независимые)
Объектно-ориентированные
Декларативные
Ассемблер
Фортран, Бейсик, Паскаль, Си (машинно-ориентированный)
Си++, Visual Basic, Delphi, Java
Логические
Функцио-нальные
Пролог
Лисп

УРОВНИ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ



ЯЗЫКИ НИЗКОГО УРОВНЯ

- Языком самого низкого уровня (НУ) является **«Машинный код»** – язык конкретной вычислительной машины, программа которого интерпретируется микропроцессором данной вычислительной машины.
- Каждая модель процессора имеет свой собственный машинный язык, хотя во многих моделях эти наборы команд сильно перекрываются.

Достоинства языков низкого уровня

- • позволяют писать самый быстрый и компактный код;
- • максимальное использование возможностей конкретной платформы;
- • возможность непосредственного доступа к аппаратуре;
- • эффективно используются программно-аппаратные ресурсы.

Недостатки языков низкого уровня

- • большая трудоемкость создания программ;
- • требуется высокая квалификация программиста;
- • высока вероятность внесения ошибок в программный код;
- • отсутствует переносимость программ на компьютеры с другой архитектурой и системой команд.

ЯЗЫКИ ASSEMBLER

- ASM (assembler – сборщик) является языком низкого уровня. В отличие от языка машинных кодов, позволяет использовать более удобные для человека мнемонические (символьные) обозначения команд.
- Команды языка ассемблера соответствуют командам процессора, фактически, они представляют собой более удобную символьную форму записи (мнемокод) команд и их аргументов. При этом одной команде языка ассемблера может соответствовать несколько команд процессора.

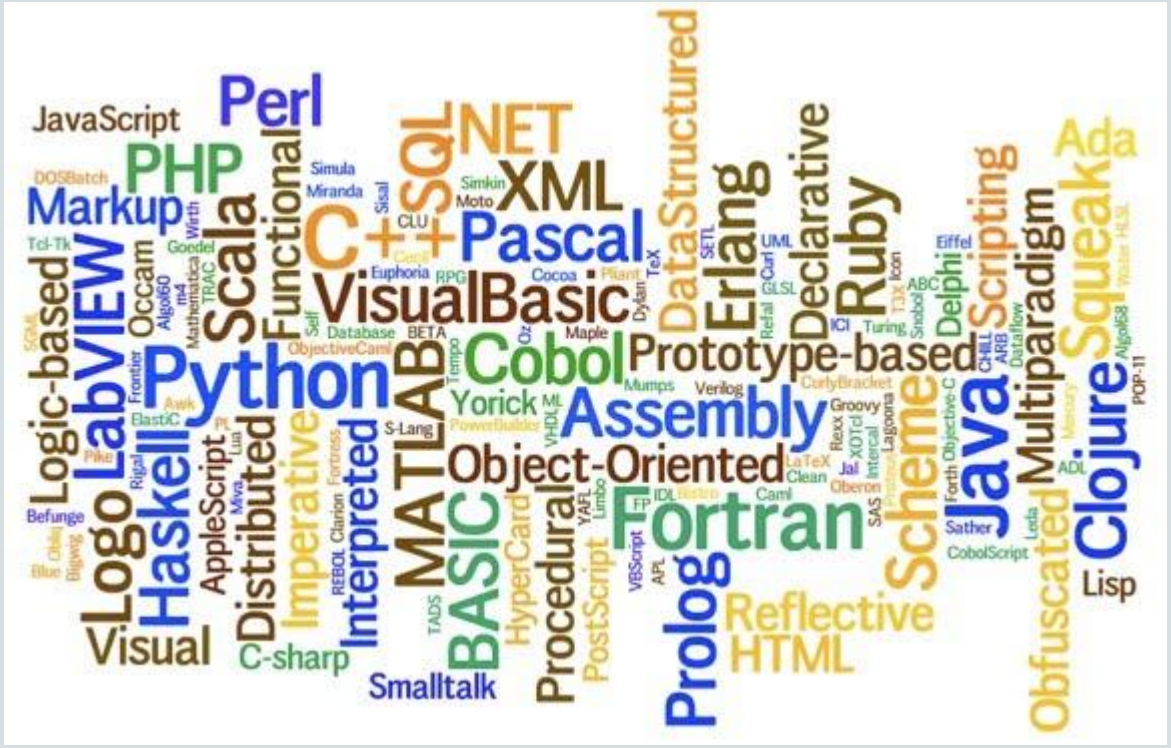
Область применения языков ASM

- драйверы устройств;
- оптимизация программного кода (рендеринг, кодеки);
- программы для бытовых устройств;
- программирование микроконтроллеров;
- взлом и защита программ;
- аппаратно-зависимые части ядер операционных систем;
- программирование средств связи;
- написание вирусов;
- для виртуализации аппаратного обеспечение (виртуальные машины).

ВЫСОКОУРОВНЕВЫЙ ЯЗЫК ПРОГРАММИРОВАНИЯ

- **Высокоуровневый язык программирования** – средство записи компьютерных программ, обеспечивающее высокую скорость и удобство работы.
- Его отличительной чертой является ***абстракция***.
- Другими словами, высокоуровневый язык программирования обеспечивает возможность введения смысловых конструкций, способных коротко описать форматы данных и операции с ними в тех случаях, когда описания на низкоуровневом языке (например, на машинном коде) будут сложными для восприятия и очень длинными.

- Первые высокоуровневые языки программирования создавались с целью предотвращения зависимости сути алгоритмов от платформы.
- В этом случае платформенная независимость обеспечивается перекладыванием связей на инструментальные программы, которые осуществляют перевод текстов с высокоуровневых языков на машинный код. Инструментальные программы выступают своего рода ***трансляторами***.



- Основным ***достоинством*** машинно-независимых языков программирования являются их простота и универсальность. Как следствие, значительно сокращается продолжительность написания кода и отладки. Одна и та же программа может быть выполнена на компьютерах разной архитектуры.

- Также к ***преимуществам*** языков программирования высокого уровня следует отнести такие факты:

1. алфавит существенно шире машинного. Он содержит 256 символов и позволяет описать любые конструкции;
2. для операторов и ключевых слов используются осмысленные слова естественного языка;
3. развитые операторы управления и огромный арсенал средств описания структур данных;
4. существует понятие типа данных и поддерживается их широкий набор.

ЗАДАНИЕ К СЛЕДУЮЩЕЙ ПАРЕ. ПОДГОТОВИТЬ ПРЕЗЕНТАЦИЮ (МАКСИМУМ 2 ЧЕЛОВЕКА) ОБ ОДНОМ ИЗ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ.

- C++
- C#
- PHP
- BASIC
- SQL

- PASCAL
- JAVA
- JavaSc
- PYTHON
- HTML
- RUBY

Что нужно сделать:

- Кратко описать история создания (кем, когда, почему)
- Основные особенности языка
- Основные достоинства и недостатки
- Отличия от других языков
- Объем до 10-12 слайдов, до 10 минут.
- Выступить с презентации на лекции
- Получить 5 :)