УДК 004.43

Классификация языков программирования

Шабакин Николай Владимирович
Национальный исследовательский Мордовский государственный университет им. Н. П. Огарёва
Студент

Прокин Александр Александрович
Национальный исследовательский Мордовский государственный университет им. Н. П. Огарёва
Преподаватель

Аннотация
В течение нескольких последних десятилетий сфера информационных и компьютерных технологий шагнула далеко вперёд. Программирование, само собой, – идёт бок о бок с техническим прогрессом и развивается столь же стремительно, как и остальные технологии. Развиваются старые языки программирования, разрабатываются новые. Необходимо языки программирования классифицировать. На данный момент существует несколько различных классификаций по отдельным признакам, и каждый автор толкует их по-своему, по-своему распределяя языки по классам.
Ключевые слова: программирование; языки программирования; классификация; уровни языков программирования.

What programming language to learn first properly

Shabakin Nikolay Vladimirovich
National Research Ogarev Mordovia State University
Student

Prokin Alexander Alexandrovich
National Research Ogarev Mordovia State University
Lector

Abstract
Within several last decades the sphere of information and computer technologies evolved dramatically forward. Programming, by itself, – goes side by side with technical progress and develops so promptly, as well as other technologies. Old programming languages develop, new are developed. It is necessary to classify programming languages. At the moment there are several various classifications by separate signs, and each author interprets them in own way, in own way distributing languages on classes.

Keywords: programming; programming languages; classification; levels of programming languages

Сразу можно оговориться, что классификация языков программирования – это очень странное понятие, т.к. языков программирования настолько много и столько же много возможных понятий классификации, что сказать вот эта классификация правильная, а вот это не правильная, это весьма глупо.

Можно выделить три основных вида классификации, которые лучше представляют человеку, в чем суть языка программирования.

Самый обширный способ классификация – это разбить языки программирования на алгоритмические и не алгоритмические языки программирования[1].

Алгоритмический язык – любой язык, в котором программист выражает алгоритм. Так же можно сказать, что это языки, в которых программист ставит задачу, и он сам описывает, как эту задачу решать.

Не алгоритмические языки – это те языки, которые не относятся к первым. Они так же подразделяются на другие, среди них можно выделить: язык моделирования. Примером такого языка является XML, в котором программист просто моделирует какие-то данные. Или SQL, где моделируется проблема, которую нужно решить, и база данных возвращает нам результат.

Второй, более интересный подход, как можно классифицировать языки программирования, это по уровню языка программирования, по уровню абстракции[2].

а) Машинный код. Это тот уровень, на котором «говорит» компьютер, это язык нулей и единиц. Он не включает в себя вмешательства со стороны программиста, т.е. он пишет единицы и нули. Это тот код который будет загружаться в процессор и будет исполняться самим процессором. В связи с этим можно сказать, что это язык программирования, это язык на котором разговаривает компьютер. Т.е. в данном случае полностью зависит от компьютера. Нет, никакой абстракции от компьютера, как компьютер воспринимает информацию, так же программист должен ее выразить

б) Машинно-ориентированные языки. Ярким примером этого языка является Ассемблер. В Ассемблере отсутствует большое количество абстракции, единственное, что программист делает, он берет какую-то последовательность бит, и выражает эту последовательность бит каким-то словом. Это сделано для того, чтобы программисту было чуть удобней работать на машинном уровне. Хоть это и на один уровень выше чем машинный, но все равно этот уровень не позволяет нам полностью абстрагироваться от конкретной машины, программист должен знать как работает процессор, как работает все железо. Ассемблер пишется под конкретную архитектуру.

в) Структурные языки программирования. В них мы уже можем создавать некие структуры, в которых мы можем создать функции, вызывать эти функции. Эти функции и структуры не будут написать под конкретную архитектуру. Эти языки программирования находятся на еще один уровень выше, поэтому в этих языках программирования, появляется такое понятие как компилятор. Компилятор берет программу написанную человеком и превращает ее во что-то, что может понимать и обрабатывать процессор. В то время, в ассемблере компилятора нет, он просто собирает код. Здесь мы компилируем программу, собирая ее сначала в объектный код, потом уже в машинный.

г) Проблемно-ориентированные языки. Примером такого языка является SQL. Это языки, которые ориентируются на то, что программист должен хорошо описать проблему. Здесь идет полное абстрагирование от какого либо компьютера, программист не задумывается на какой архитектуре будет программа. Определенное описание проблемы, в результате мы получаем какое-либо решение данной проблемы.

д) Натуральный язык. Таких языков еще не существует в природе. Хотя многие люди думают, что они такое написали, на самом деле нет. Если представить что-то из научной фантастики, где человек сидит и общается с компьютером, и просто говорит, что ему нужно что-то. Компьютер сам решает, что представляет собой эта проблема, создает алгоритм действий, и решает этот алгоритм. Т.е. это язык программирования, в котором компьютер полностью идет навстречу программисту. Если на первом уровне программист полностью идет навстречу компьютеру, то на пятом уровне компьютер полностью переходит на сторону человека. Т.е. компьютер начинает разговаривать на языке человека.

Это наверное один из правильных и понятных способов понять на каком языке программист программирует. Обычно программисты остаются на третьем и четвертом уровне. К третьему уровню относятся такие языки как C, C++, C#, Java и т.д. К четвертому уровню относятся: SQL, некоторые возможности C#. Т.е. из этого можно сделать вывод, что какой-либо один язык не привязан к конкретному уровню. Тот же самый C позволяет легко переходить и начинать программировать на Ассемблере или начинать программировать по конкретную архитектуру. Так же C с некоторыми библиотеками позволяет дойти до четвертого уровня.

Третий подход является очень размытым. Это классификация по структуре самого кода, на котором пишет программист. Скорее даже не получится разделить на 1, 2, 3, потому что-то они очень сильно отличаются друг от друга[3].

Неструктурный язык программирования. Его сразу можно отличить по наличию оператора go to или оператора jump, который позволяет независимо от того как он вызывается перейти в любую точку программы. Этот оператор очень опасен для структурных языков программирования, но он обязательно используется в неструктурных языках программирования, т.е. Ассемблер, машинный язык, в котором не обойдешься без этого оператора перехода.

Структурный язык программирования. Это тот язык, который позволяет структурировать программный код. Т.е. создается структура, могут создаваться функции, могут создаваться функции обратного вызова и т.д.

Функциональный язык программирования. В нем так же создаются функции, но в отличие от структурного языка, каждая функция выполняется как черный ящик, т.е. во-первых каждый вызов данной функции с одинаковыми данными должен получить один и тот же результат. Например такие функции как sin, cos, tan, ctg. Если мы вызовем функцию с одним и тем же параметром несколько раз, мы обязаны получить один и тот же результат. В структурном программирование, не факт что так получится. Второй отличительной особенностью является отсутствие глобальных переменных, отсутствие статуса программы. При выполнении программы мы можем предсказать что произойдет, смотря на линию кода, программист понимает что происходит. В структурном программирование мы должны знать, что происходить в данный момент, чтобы узнать, что произойдет в следующий раз.

Объектно-ориентированный язык программирования. В этих языках программист создает объекты. Программа выражается тем, что у нас есть какая-то коммуникация между разными объектами. Эти объекты могут вызывать функции друг у друга или делать что-либо другое. Практически во всех структурно-ориентированных языках уже появляются объекты, тем не менее, не все языки в которых есть объекты являются объектно-ориентированными. У объектно-ориентированных не только есть объекты, а его исполнение зависит от этих объектов, без них невозможно программировать. Также структура языка подходит под работу с объектами. Например, в языке С есть объекты, но он не является объектно-ориентированным, т.к. это всего лишь дополнение. С другой стороны C++, C#, являются объектно-ориентированными, потому что работа в этих языках тесно связана с созданием объектов.

Визуальные языки программирования. Фактически это создание визуальных объектов. Например, программист создал диаграмму, и эта диаграмма является программным кодом. Программисту не нужно брать диаграмму и переводить ее в программный код, сама диаграмма уже является программным кодом. Очень часто в языках визуальных программирования используется цвет объекта и методы подсоединения этого объекта, программист всего лишь задает записи, как один объект связан с другим, типы связей и соединений. Т.е. программист программирует графические элементы, вместо написания слов, он рисует картинки.

Это далеко не единственные варианты классификации языков программирования, но даже их достаточно, чтобы понять, насколько это объемная и сложная тема.

Библиографический список

1. Классификация языков программирования.

URL:http://bourabai.ru/alg/classification.htm
2. Пять поколений языков программирования. URL: http://life-prog.ru/view_zam2.php?id=194&cat=5&page=11
3. Эволюция языков программирования. URL:http://www.urtt.ru/bib/dataindex/oaip/lection/_html/lect_05.htm