

ЛЕКЦИЯ 1.

ВВЕДЕНИЕ В ПРЕДМЕТ. КЛАССИФИКАЦИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

Цель и задачи дисциплины

Целью преподавания дисциплины «Основы программирования» является ознакомление студентов с практическими задачами, созданию проектов, программных модулей и базы данных с помощью языка высокого уровня Visual Basic, развитие способности программирования для управления современными компьютерами.

Задачи:

- сформировать у студентов понимание основ теории визуального программирования и сущности используемых ныне языков программирования высокого уровня.
- Обучение алгоритмического языка Visual Basic
- Изучить основные операторы языка при решении задач, создание программ и модулей, использование в программах анимации и графических возможностей языка;
- Работа в интегрированной среде IDE, дополнительные возможности языка, настройка и изменение параметров, работа с библиотекой и базой данных.
- научиться работать с типами переменных, использовать формы, разрабатывать проекты, создавать EXE приложения.
- освоить процесс создания программных комплексов

Под *языком программирования* мы будем понимать совокупность средств и правил представления алгоритма в виде, приемлемом для компьютера. Отсюда неискушенный вы можете решить, что компьютер, оказывается, при всей его кажущейся могущественности, не поймет задачи, поставленной ему на простом человеческом языке, будь то русский, английский или даже китайский.

Существует разделение всех языков программирования на две большие группы.

➤ Языком самого высокого уровня считается "человеческий язык, и когда компьютер станет его легко понимать, то он вплотную приблизится к человеку.

- Языком самого низкого уровня считается язык так называемых машинных кодов.
- Все остальные алгоритмические языки лежат где-то посередине.

История появления языка программирования Visual Basic

Язык **Basic** был разработан преподавателями Дартмутского колледжа (США) Джоном Кемени и Томасом Куртцом в 1964 году как средство обучения и работы начинающих программистов.

Обычно Basic ассоциируется с каким-то очень простым в освоении и использовании средством программирования. Это действительно так. На заре компьютерных технологий язык Basic был создан для создания простых программ и использовался в качестве учебного языка для первых шагов при изучении основ программирования с последующим переходом на более сложные и универсальные языки. Это было заложено в название языка *BASIC — Beginners All-purpose Symbolic Instructional Code*, то есть многоцелевой код символьных инструкций для начинающих. С прогрессом компьютерных технологий развивался и Basic. В настоящее время версия Visual Basic 6 дает возможность решать любые современные задачи разработки приложений. При этом Visual Basic 6 остался достаточно простым в освоении, став в то же время одним из мощных современных языков программирования.

По сути дела последняя версия языка Visual Basic 6 представляет собой интегрированную среду разработки — *IDE (Integrated Development Environment)*. Это означает, что в Visual Basic 6 интегрирован набор инструментов, облегчающих и значительно ускоряющих процесс разработки готового приложения. Такая реализация языка Visual Basic 6 с полным правом позволяет отнести его к средствам быстрой разработки приложений — *RAD (Rapid Application Development)* и ставит практически в один ряд с такими средствами разработки, как Visual C++, Delphi и другими.

С помощью Visual Basic 6 можно создавать приложения практически для любой области современных компьютерных технологий: бизнес-приложения, игры, мультимедиа, базы данных. При этом приложения могут быть как простыми, так и очень сложными, в зависимости от поставленной задачи.

Простота и мощность языка Visual Basic 6 позволили сделать его встроенным языком для приложений Microsoft Office. Многие независимые разработчики, как например известная своими программами в области бухгалтерского учета фирма "1С", приобретают лицензии на использование языка

Visual Basic в качестве внутреннего языка своих приложений. Поэтому в настоящее время Basic уже не считается учебным языком — знание Visual Basic и его диалектов (VBA, VBScript) становится необходимостью для современного программиста любого уровня.

Языки программирования принято делить на пять поколений.

Первое поколение

К первому поколению (англ. first-generation programming language, 1GL) относят машинные языки — языки программирования на уровне команд процессора конкретной машины. Для программирования не использовался транслятор, команды программы вводились непосредственно в машинном коде переключателями на передней панели машины. Такие языки были хороши для детального понимания функционирования конкретной машины, но сложны для изучения и решения прикладных задач.

Термины «первое поколение» и «второе поколение» были введены одновременно с термином «третье поколение». Фактически, эти термины в то время не использовались. С появлением языков высокого уровня, языки ассемблера стали относить к первому поколению языков.

Второе поколение

Языки второго поколения (2GL) создавались для того, чтобы облегчить тяжёлую работу по программированию, перейдя в выражениях языка от низкоуровневых машинных понятий ближе к тому как обычно мыслит программист. Эти языки появились в 1950-е годы, в частности, такие языки как Фортран и Алгол. Наиболее важной проблемой, с которыми столкнулись разработчики языков второго поколения, стала задача убедить клиентов в том, что созданный компилятором код выполняется достаточно хорошо, чтобы оправдать отказ от программирования на ассемблере. Скептицизм по поводу возможности создания эффективных программ с помощью автоматических компиляторов был довольно распространён, поэтому разработчикам таких систем должны были продемонстрировать, что они действительно могут генерировать почти такой же эффективный код, как и при ручном кодировании, причём практически для любой исходной задачи.

Третье поколение

Под третьим поколением (3GL) первоначально понимались все языки более высокого уровня, чем ассемблер. Главной отличительной чертой языков третьего поколения стала независимость от аппаратного обеспечения,

то есть выражение алгоритма в форме, не зависящей от конкретных характеристик машины, на которой он будет исполняться. Код, написанный на языке третьего поколения, перед исполнением транслируется либо непосредственно в машинные команды, либо в код на ассемблере и затем уже ассемблируется. При компиляции, в отличие от предыдущих поколений, уже нет соответствия один-к-одному между инструкциями программы и генерируемым кодом.

Стала широко использоваться интерпретация программ — при этом инструкции программы не преобразуются в машинный код, а исполняются непосредственно одна за другой. Независимость от «железа» достигается за счёт использования интерпретатора, скомпилированного под конкретную аппаратную платформу. Одним из ранних интерпретируемых языков стал Лисп.

Четвёртое поколение

Термин языки программирования четвёртого поколения (4GL) лучше представлять как среды разработки четвёртого поколения. Они относятся к временному периоду с 1970-х по начало 1990-х.

Языки этого поколения предназначены для реализации крупных проектов, повышают их надёжность и скорость создания, ориентированы на специализированные области применения, и используют не универсальные, а проблемно-ориентированные языки, оперирующие конкретными понятиями узкой предметной области. В эти языки встраиваются мощные операторы, позволяющие одной строкой описать такую функциональность, для реализации которой на языках младших поколений потребовались бы тысячи строк исходного кода.

Пятое поколение

Рождение языков пятого поколения произошло в середине 90-х годов. К ним относятся также системы автоматического создания прикладных программ с помощью визуальных средств разработки, без знания программирования. Основная идея была заключена в возможности автоматического формирования результирующего текста на универсальных языках программирования (который необходимо откомпилировать). Инструкции же вводятся в компьютер в максимально наглядном виде с помощью методов, наиболее удобных для человека, не знакомого с программированием.