

Программное обеспечение (ПО)

—это совокупность программ, хранящихся во внешней памяти компьютера в виде файлов и предназначенных для выполнения электронными системами.

Программа

— упорядоченная последовательность команд.

Команда

— это описание операции, которую должен выполнить компьютер. Как правило, у команды есть свой код (условное обозначение), исходные данные (операнды) и результат.

- **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** - определяет набор лексических, синтаксических и семантических правил, задающих внешний вид программы и действия, которые выполняет исполнитель (компьютер) под ее управлением.

1 этап

машинно-зависимые, представляя собой по сути набор машинных команд, который определяется архитектурой конкретной ЭВМ.

2 этап

Второе поколение характеризуются созданием ЯП *ассемблерного типа*, позволяющих вместо бинарных и иных форматов машинных команд использовать символьные обозначения. Являясь существенным шагом вперед, ассемблерные языки все еще оставались машинно-зависимыми и пользователь должен был быть хорошо знаком с аппаратной средой конкретного типа ЭВМ. Ассемблерные программы затруднительны для чтения, более трудоемки при отладке, хотя и по сей день они используются для создания минимального по объему и максимального по производительности ПО.

Языки низкого уровня

Первым компьютерам приходилось программировать двоичными машинными кодами. Однако программировать таким образом - достаточно трудоемкая и сложная задача. Для упрощения этой задачи стали появляться языки программирования низкого уровня, которые позволяли задавать машинные команды в более понятном для человека виде. Для преобразования их в двоичный код были созданы специальные программы - трансляторы.

ПРЕИМУЩЕСТВА

- *С помощью языков низкого уровня создаются эффективные и компактные программы, поскольку разработчик получает доступ ко всем возможностям процессора.*

НЕДОСТАТКИ

- *Программист, работающий с языками низкого уровня, должен быть высокой квалификации, хорошо понимать устройство микропроцессорной системы, для которой создается программа.*
- *результатирующая программа не может быть перенесена на компьютер или устройство с другим типом процессора.*
- *значительное время разработки больших и сложных программ.*

3 этап

В 1954 году начинается **третье поколение языков программирования**: в недрах корпорации IBM группой разработчиков во главе с Джоном Бэкусом (John Backus) был создан язык программирования **Fortran**. Значение этого события трудно переоценить. Это **первый язык программирования высокого уровня**. Ключевой идеей, отличающей новый язык от ассемблера, была концепция подпрограмм. В 1960 году был создан язык программирования **Cobol**. Он задумывался как язык для создания коммерческих приложений, и он стал таковым. На Коболе написаны тысячи прикладных коммерческих систем. Отличительной особенностью языка является возможность эффективной работы с большими массивами данных, что характерно именно коммерческих приложений. Популярность Кобола столь высока, что даже сейчас, при всех его недостатках (по структуре и замыслу Кобол во многом напоминает Фортран) появляются новые его диалекты и реализации.

- В 1960 году командой во главе с Петером Науром (Peter Naur) был создан язык программирования **Algol**. Этот язык дал начало целому семейству Алгол-подобных языков (важнейший представитель — Pascal). В 1968 году появилась новая версия языка.
- В 1963 году в Дартмурском колледже (Dartmouth College) был создан язык программирования **BASIC** (Beginners' All-Purpose Symbolic Instruction Code — многоцелевой язык символических инструкций для начинающих). Язык задумывался в первую очередь как средство обучения и как первый изучаемый язык программирования. Он предполагался легко интерпретируемым и компилируемым. Надо сказать, что BASIC действительно стал языком, на котором учатся программировать. Было создано несколько мощных реализаций BASIC, поддерживающих самые современные концепции программирования (ярчайший пример — Microsoft Visual Basic).
- В 1964 году все та же корпорация IBM создала язык **PL/1**, который был призван заменить Cobol и Fortran в большинстве приложений. Язык обладал исключительным богатством синтаксических конструкций. В нем впервые появилась обработка исключительных ситуаций и поддержка параллелизма.
- В 1970 году Никлаусом Виртом был создан язык программирования **Pascal**. Язык замечателен тем, что это первый широко распространенный язык для **структурного программирования**. Впервые оператор безусловного перехода перестал играть основополагающую роль при управлении порядком выполнения операторов.
- В 1972 году Керниганом и Ритчи был создан язык программирования **C**. Он создавался как язык для разработки операционной системы UNIX. С часто называют «переносимым ассемблером», имея в виду то, что он позволяет работать с данными практически так же эффективно, как на ассемблере, предоставляя при этом структурированные управляющие конструкции и абстракции высокого уровня (структуры и массивы).

4 ЭТАП

- Языки **четвертого поколения** носили **непроцедурный** характер, т.е. программы на таких языках описывают только что надо делать, но не как это сделать. Типичными примерами являются языки **искусственного интеллекта** или **логического программирования**. Родоначальником таких языков стал язык **Prolog**, появившийся в 1971 году.
- Вторым направлением эволюции ЯВУ явилось появление **объектно-ориентированных** языков программирования. Первым объектно-ориентированным языком был язык **Simula** (1967). Этот язык был предназначен для моделирования различных объектов и процессов, и объектно-ориентированные черты появились в нем именно для описания свойств модельных объектов.

В **объектно-ориентированном** программировании имеют дело с тремя базовыми элементами:

- **объектами**, которые представляют собой многократно используемые программные модули, содержащие связанные данные и процедуры. Структурно объекты состоят из 2 элементов: **свойств** и **методов**. Свойства - это переменные элементы, изменяющие состояние объекта, а методы - процедуры и функции, определяющие функционирование объекта.
- **событиями**, которые происходят с данными объектами
- **классами объектов**, экземпляры которых используются при программировании.

5 ЭТАП

- Широкое распространением так называемые **скриптовые языки**.
- К пятому поколению языков программирования также можно отнести **языки искусственного интеллекта, экспертных систем и баз знаний**, а также **естественные языки**.
- Язык **JavaScript** был создан в компании Netscape Communications в качестве языка для описания сложного поведения веб-страниц. Интерпретируется браузером во время отображения веб-страницы. По синтаксису схож с Java и (отдаленно) с C/C++. Имеет возможность использовать встроенную в браузер объектную функциональность, однако подлинно объектно-ориентированным языком не является.
- **VBScript** был создан в корпорации Microsoft во многом в качестве альтернативы JavaScript. Имеет схожую область применения. Синтаксически схож с языком Visual Basic (и является усеченной версией последнего). Так же, как и JavaScript,
- исполняется браузером при отображении веб-страниц и имеет ту же степень объектно-ориентированности.
- **Perl** создавался в помощь системному администратору операционной системы Unix для обработки различного рода текстов и выделения нужной информации. Развился до мощного средства работы с текстами. Является интерпретируемым языком и реализован практически на всех существующих платформах. Применяется при обработке текстов, а также для динамической генерации веб-страниц на веб-серверах.
- **Python** - интерпретируемый объектно-ориентированный язык программирования. По структуре и области применения близок к Perl, однако менее распространен и более строг и логичен. Имеются реализации для большинства существующих платформ.

БАЗОВЫЙ УРОВЕНЬ

Базовый уровень является низшим уровнем программного обеспечения. Отвечает за взаимодействие с базовыми аппаратными средствами. Базовое программное обеспечение содержится в составе базового аппаратного обеспечения и сохраняется в специальных микросхемах постоянного запоминающего устройства (ПЗУ), образуя базовую систему ввода-вывода BIOS. Программы и данные записываются в ПЗУ на этапе производства и не могут быть изменены во время эксплуатации.

СИСТЕМНЫЙ УРОВЕНЬ

Системный уровень - является переходным. Программы этого уровня обеспечивают взаимодействие других программ компьютера с программами базового уровня и непосредственно с аппаратным обеспечением. От программ этого уровня зависят эксплуатационные показатели всей вычислительной системы. При подсоединении к компьютеру нового оборудования, на системном уровне должна быть установлена программа, обеспечивающая для остальных программ взаимосвязь с устройством.

Конкретные программы, предназначенные для взаимодействия с конкретными устройствами, называют драйверами.

Другой класс программ системного уровня отвечает за взаимодействие с пользователем. Благодаря ему, можно вводить данные в вычислительную систему, руководить ее работой и получать результат в удобной форме. Это средства обеспечения пользовательского интерфейса, от них зависит удобство и производительность работы с компьютером.

Совокупность программного обеспечения системного уровня образует **ядро операционной системы компьютера**. Наличие ядра операционной системы - это первое условие для возможности практической работы пользователя с вычислительной системой. Ядро операционной системы выполняет такие функции: управление памятью, процессами ввода-вывода, файловой системой, организация взаимодействия и диспетчеризация процессов, учет использования ресурсов, обработка команд и т.д.

СЛУЖЕБНЫЙ УРОВЕНЬ

Программы этого уровня взаимодействуют как с программами базового уровня, так и с программами системного уровня. Назначение служебных программ (утилит) состоит в автоматизации работ по проверке и настройке компьютерной системы, а также для улучшения функций системных программ. Некоторые служебные программы (программы обслуживания) сразу входят в состав операционной системы, дополняя ее ядро, но большинство являются внешними программами и расширяют функции операционной системы. То есть, в разработке служебных программ отслеживаются два направления: интеграция с операционной системой и автономное функционирование.

Классификация служебных программных средств

1. Диспетчеры файлов (файловые менеджеры). С их помощью выполняется большинство операций по обслуживанию файловой структуры: копирование, перемещение, переименование файлов, создание каталогов (папок), уничтожение объектов, поиск файлов и навигация в файловой структуре. Базовые программные средства содержатся в составе программ системного уровня и устанавливаются вместе с операционной системой
2. Средства сжатия данных (архиваторы). Предназначены для создания архивов. Архивные файлы имеют повышенную плотность записи информации и соответственно, эффективнее используют носители информации.
3. Средства диагностики. Предназначены для автоматизации процессов диагностики программного и аппаратного обеспечения. Их используют для исправления ошибок и для оптимизации работы компьютерной системы.
4. Программы инсталляции (установки). Предназначены для контроля за добавлением в текущую программную конфигурацию нового программного обеспечения. Они следят за состоянием и изменением окружающей программной среды, отслеживают и протоколируют образование новых связей, утраченных во время уничтожения определенных программ. Простые средства управления установлением и уничтожением программ содержатся в составе операционной системы, но могут использоваться и дополнительные служебные программы.
5. Средства коммуникации. Разрешают устанавливать соединение с удаленными компьютерами, передают сообщения электронной почты, пересылают факсимильные сообщения и т.п..
6. Средства просмотра и воспроизведения. Преимущественно, для работы с файлами, их необходимо загрузить в "родную" прикладную программу и внести необходимые

исправления. Но, если редактирование не нужно, существуют универсальные средства для просмотра (в случае текста) или воспроизведения (в случае звука или видео) данных.

7. Средства компьютерной безопасности. К ним относятся средства пассивной и активной защиты данных от повреждения, несанкционированного доступа, просмотра и изменения данных. Средства пассивной защиты - это служебные программы, предназначенные для резервного копирования. Средства активной защиты применяют антивирусное программное обеспечение. Для защиты данных от несанкционированного доступа, их просмотра и изменения используют специальные системы, базирующиеся на криптографии.

ПРИКЛАДНОЙ УРОВЕНЬ

Программное обеспечение этого уровня представляет собой комплекс прикладных программ, с помощью которых выполняются конкретные задачи (производственных, творческих, развлекательных и учебных). Между прикладным и системным программным обеспечением существует тесная взаимосвязь. Универсальность вычислительной системы, доступность прикладных программ и широта функциональных возможностей компьютера непосредственно зависят от типа имеющейся операционной системы, системных средств, помещенных в ее ядро и взаимодействии комплекса человек-программа-оборудование

ХАРАКТЕРИСТИКА ОПЕРАЦИОННЫХ СИСТЕМ

Важнейшим достоинством большинства ОС является модульность. Это свойство позволяет объединить в каждом модуле определенные логически связанные группы функций. Если возникает необходимость в замене или расширении такой группы функций, это можно сделать путем замены или модификации, лишь одного модуля, а не всей системы.

Большинство ОС состоит из следующих основных модулей:

- базовая система ввода-вывода (BIOS - Basic Input Output System);
- загрузчик операционной системы (Boot Record);
- ядро ОС;
- драйверы устройств;
- командный процессор;
- внешние команды (файлы).

Базовая система ввода-вывода (BIOS) - это набор микропрограмм, реализующих основные низкоуровневые (элементарные) операции ввода-вывода. Они хранятся в ПЗУ компьютера и записываются туда при изготовлении материнской платы. Данная система, по сути, «встроена» в компьютер и является одновременно его аппаратной частью и частью операционной системы.

Первая функция BIOS - автоматическое тестирование основных компонентов компьютера при его включении. При обнаружении ошибки на экран выводится соответствующее сообщение и / или выдается звуковой сигнал.

Далее BIOS осуществляет вызов блока начальной загрузки операционной системы, находящейся на диске (эта операция выполняется сразу по окончании тестирования).

Загрузив в ОЗУ этот блок, BIOS передает ему управление, а он в свою очередь загружает другие модули ОС.

Еще одна важная функция BIOS - обслуживание прерываний. При возникновении определенных событий (нажатие клавиши на клавиатуре, щелчок мыши, ошибка в программе и т.д.) вызывается одна из стандартных подпрограмм BIOS по обработке возникшей ситуации.

Загрузчик операционной системы - это короткая программа, находящаяся в первом секторе любого загрузочного диска (дискеты или диска с операционной системой).

Функция этой программы заключается в считывании в память основных дисковых файлов ОС и передаче им дальнейшего управления ЭВМ.

Ядро ОС реализует основные высокоуровневые услуги, загружается в ОЗУ и остается в ней постоянно. В ядре ОС выделяют несколько подсистем, каждая из которых отвечает за выполнение той или иной задачи:

- файловая система (отвечает за размещение информации на устройствах хранения);
- система управления памятью (размещает программы в памяти);
- система управления программами (осуществляет запуск и выполнение программ);
- система связи с драйверами устройств (отвечает за взаимодействие с внешними устройствами);
- система обработки ошибок;
- служба времени (предоставляет всем программам информацию о системном времени).

Модуль расширения BIOS придает гибкость операционной системе, позволяя добавлять драйверы, обслуживающие дополнительные устройства.

Драйверы требуются в тех случаях, когда обмен информацией с устройствами должен происходить иначе, чем определено в BIOS. Драйверы устройств - это программы, управляющие работой внешних (периферийных) устройств на физическом уровне. Они дополняют систему ввода-вывода ОС и обеспечивают обслуживание новых устройств или нестандартное использование имеющихся. Они передают или принимают данные от аппаратуры и делают пользовательские программы независимыми от ее особенностей.

Драйверы загружаются в память компьютера при загрузке операционной системы; необходимость и порядок их загрузки указываются в специальных файлах конфигурации. Такая схема облегчает подключение к машине новых устройств и позволяет делать это, не затрагивая системные файлы ОС.

Командный процессор - это программа, функции которой заключаются в следующем: прием и синтаксический разбор команд, полученных с клавиатуры или из командного файла;

- исполнение внутренних команд операционной системы;
- загрузка и исполнение внешних команд (реализованных в виде самостоятельных программ) операционной системы и прикладных программ пользователя (файлы с расширением COM, EXE или BAT).

Некоторые стандартные команды (TYPE, DIR и другие) командный процессор выполняет сам. Такие команды называются внутренними (как правило, это основные команды работы с файлами и каталогами). Для выполнения внешних команд пользователя командный процессор ищет на дисках программу с соответствующим именем и расширением COM, EXE (например, FORMAT.COM), и если находит ее, то загружает в память и передает ей управление. По окончании работы программы командный процессор удаляет ее из памяти. Таким образом, **внешние команды ОС** - это программы, поставляемые вместе с операционной системой в виде отдельных файлов.

В функции командного процессора входит также исполнение командных файлов (это текстовые файлы с набором команд и расширением BAT). Когда в качестве команды задается имя такого файла, командный процессор начинает последовательно читать и интерпретировать содержащиеся в нем строки, каждая из которых может содержать одну команду, метку или комментарий. Если в очередной строке стоит команда, осуществляющая вызов какой-то программы, выполнение командного файла приостанавливается и начинается работа вызванной программы. После ее завершения происходит выполнение следующей команды командного файла.

В 1983 году компания Microsoft приступила к разработке операционной системы с графическим интерфейсом. Графический интерфейс - это оболочка, позволяющая

использовать для выполнения программ графические элементы, которые можно видеть на экране монитора.

Первой операционной системой с поддержкой графического интерфейса пользователя (GUI, Graphical User Interface - полное название графического интерфейса) стала операционная система Macintosh (сокращенно - Mac), разработанная для компьютеров Apple PC, в 1983 году

Компания Microsoft представила свою "операционную систему" Windows 1.0 на выставке компьютерных технологий в Лас-Вегасе лишь 20 ноября 1985 года