История создания и развития систем программирования

Лекция1

*Машинные языки

Каждый компьютер имеет свой определенный машинный язык. С помощью машинного языка программист мог задавать команды, оперируя с ячейками памяти, полностью используя возможности машины. В команде сообщается информация о местонахождении операндов и типе выполняемой операции.

x := sin(y*Pi) + 1;

11011001 11101011 11011100 00001101 11010000 10010111
01000000 00000000 11011001 11111110 11011001 11101000
11011110 11000001 11011101 00011101 10011000 10010111
01000000 00000000

D9 EB

DC 0D D0 97 40 00

D9 FE

D9 E8

DE C1

DD 1D 98 97 40 00

*Ассемблеры

Используют мнемонические команды взамен машинных команд.

В программировании на Ассемблере стало возможным использование меток, что облегчало в значительной мере отладку программ.

Появилась возможность разработки целой серии вычислительных машин с одинаковой или сходной системой команд, что можно назвать первым подобием переносимости кода.

Ассемблер нес в себе еще одну революционность для программирования: появилось два представления программ на нем: в кодах и в откомпилированном виде.

```
begin
   WriteLn ('Hello, world!');
end.
```

```
mov ax,cs
mov ds,ax
mov ah,9
mov dx, offset Hello
int 21h
xor ax,ax
int 21h
 Hello:
  db "Hello World!",13,10,"$"
```

*Языки высокого уровня

Языки высокого уровня имитируют естественные языки, используя некоторые слова разговорного языка и общепринятые математические символы.

Первый язык программирования высокого уровня под названием Фортран был разработан в 1954 году Джоном Бэкусом. Самым главным и принципиальным отличием Фортрана от Ассемблера была концепция подпрограмм.

В 1960 году был создан язык программирования Cobol. Он задумывался как язык для создания коммерческих приложений. Его отличительной особенностью является возможность эффективной работы с большими массивами данных.

В 1963 году был создан язык программирования BASIC. Язык задумывался в первую очередь как средство обучения и как первый изучаемый язык программирования.

В 1964 году был создан 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.

*Структурное программирование

Структурное программирование — методология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры блоков.

В 1970 году Никлаусом Виртом был создан язык программирования Pascal. Язык замечателен тем, что это первый широко распространенный язык для структурного программирования. В этом языке также внедрена строгая проверка типов, что позволило выявлять многие ошибки на этапе компиляции.

В 1972 году был создан язык программирования C. Он создавался как язык для разработки операционной системы UNIX. C 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).

*Объектно-ориентированное программирование

В рамках данного подхода программа представляет собой описание объектов, их свойств (или атрибутов), совокупностей (или классов), отношений между ними, способов их взаимодействия и операций над объектами (или методов).

Использование ранее разработанных (возможно, другими коллективами программистов) библиотек объектов и методов позволяет значительно сэкономить трудозатраты при производстве программного обеспечения, в особенности типичного.

Наиболее известным примером объектно-ориентированного языка программирования является язык C++. Первый коммерческий транслятор впервые был реализован в 1993-м году, и сам язык был назван C++. Его прямым потомком и логическим продолжением является язык C#.

*Языки сценариев или скрипты

В рамках данного подхода программа представляет собой совокупность возможных сценариев обработки данных, выбор которых инициируется наступлением того или иного события (щелчок по кнопке мыши, попадание курсора в определенную позицию и т.д.).

Характерными особенностями данных языков являются, во-первых, их интерпретируемость (компиляция либо невозможна, либо нежелательна), во-вторых, простота синтаксиса, а в-третьих, легкая расширяемость. Таким образом, они идеально подходят для использования в часто изменяемых программах, очень небольших программах или в случаях, когда для выполнения операторов языка затрачивается время, несопоставимое со временем их разбора.

*Функциональные языки и языки логического программирования

Все языки, о которых шла речь ранее, имеют одно общее свойство: они императивны. Это означает, что программы на них, в конечном итоге, представляют собой пошаговое описание решения той или иной задачи. Можно попытаться описывать лишь постановку проблемы, а решать задачу поручить компилятору. Существует два основных подхода, развивающие эту идею: функциональное и логическое программирование.

Основная идея, лежащая в основе функционального программирования, — это представление программы в виде математических функций.

Программы на языках логического программирования выражены как формулы математической логики, а компилятор пытается получить следствия из них;

*Система программирования

Система программирования — это система для разработки новых программ на конкретном языке программирования. Обычно в ее состав входят:

- компилятор или интерпретатор;
- интегрированная среда разработки;
- средства создания и редактирования текстов программ;
- обширные библиотеки стандартных программ и функций;
- отладочные программы;
- "дружественная" к пользователю диалоговая среда;
- многооконный режим работы;
- графические библиотеки; утилиты для работы с библиотеками
- встроенный ассемблер;

*Трансляция

Для перевода кода с одного языка программирования на другой требуется специальная программа — транслятор.

Выделяют два основных способа трансляции — *компиляция* программы или ее *интерпретация*.

При компиляции весь *исходный программный код* сразу переводится в машинный. Создается отдельный *исполняемый файл*, выполнение которого обеспечивается операционной системой.

При интерпретации выполнение кода происходит последовательно (строка за строкой). Операционная система взаимодействует с интерпретатором, а не исходным кодом.

Выполнение откомпилированной программы происходит быстрее, т.к. она представляет собой готовый машинный код.

*Домашнее задание

1. Составить опорный конспект лекции по теме «История создания и развития систем программирования» на основе презентации.
2. Подготовить мини-доклады о языках программирования не упомянутых в лекции.