Языки программирования высокого уровня

Материалы курса
«Языки и системы программирования»
Тема 3

Залогова Любовь Алексеевна

Описание языка программирования

Цель: формализовать описание языка программирования для построения процедур трансляции

Описание языка программирования состоит из четырёх компонентов:

- описание лексики,
- описание синтаксиса,
- описание семантики,
- описание прагматики.

Описание языка программирования

Лексика *(словарь языка, алфавит языка)* – набор основных и специальных символов. К специальным символам относятся знаки операций и разделители (+ - * = : и др.).

Литера – один знак (буква цифра = ! < и др.).

Символ – неделимая с точки зрения языка конструкция.

Примеры символов языка С:

while do for else int float return = != < <= >= <> ( )

+ ++ - -- ;

Описание языка программирования

В исходной программе смежные символы могут быть разделены произвольным количеством пробелов, но внутри символа пробелы не допускаются, например,

```
for (i=1;  i <= n; ++i)
   p = p*x;
return (p);
```

Описание языка программирования

Синтаксис ЯП – совокупность правил для построения правильных предложений языка (набор требований, которым должна удовлетворять программа).

Семантика ЯП – правила истолкования предложений языка.

Например, a = a + 25 имеет следующее истолкование: содержимое ячейки с именем a увеличивается на 25.

.

Описание языка программирования

Прагматика ЯП – методология программирования, т.е. описание методов и приёмов, позволяющих исходя из постановки задачи составить программу её решения.

Описание прагматики (применения) языка отвечает на вопрос: «Как писать программы на данном языке программирования?». Описание прагматики не формализуется, это - передача опыта.

Структура компилятора

Компилятор - это программа, которая переводит программу на языке высокого уровня в эквивалентную программу на другом (объектном) языке.

Структура компилятора

Работа компилятора включает два основных этапа:

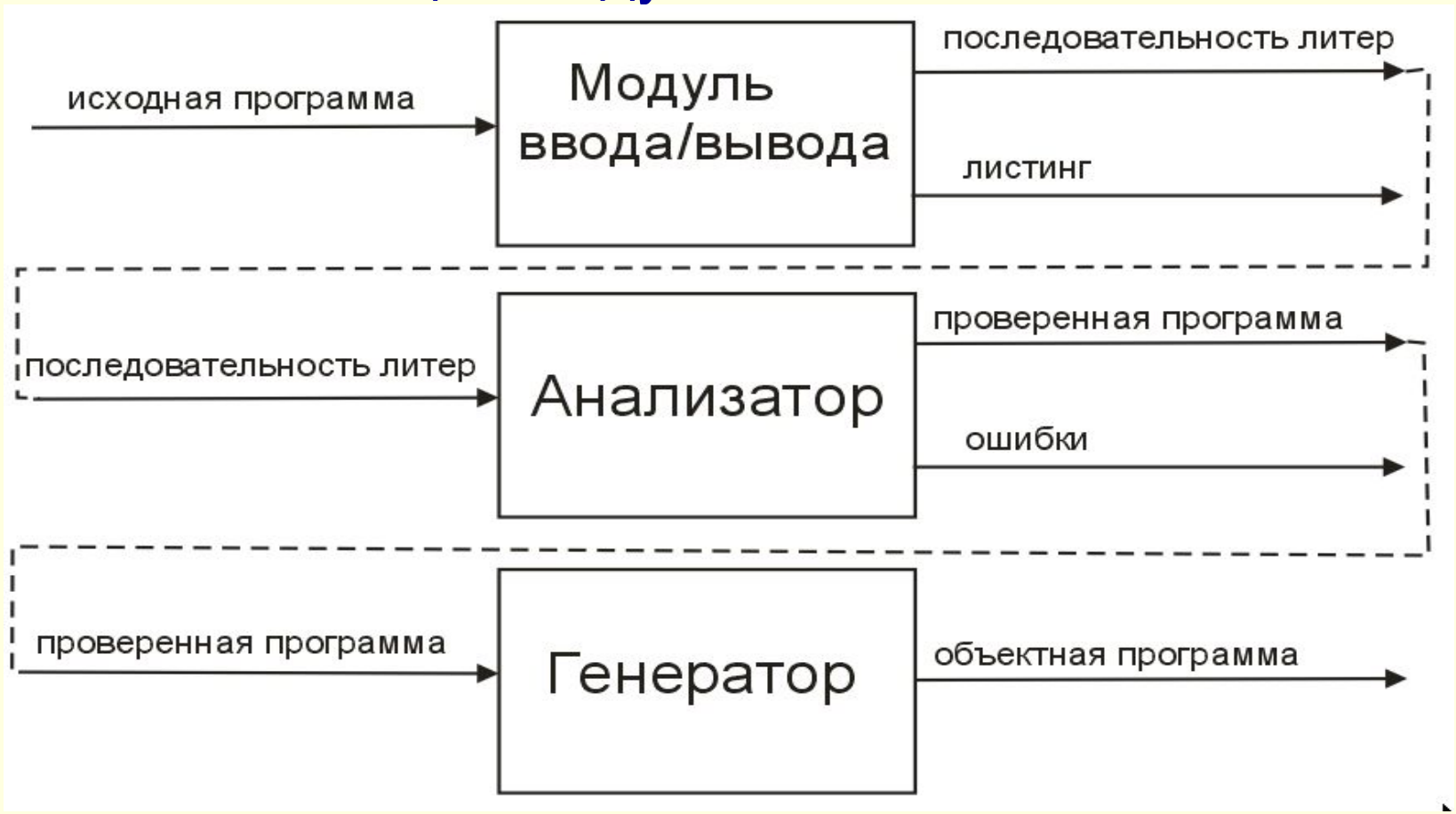
1. **Анализ** - определение синтаксической правильности исходной программы.

2. **Синтез** - генерация объектной программы;

этот этап выполняется для программ, не содержащих ошибок.

Структура компилятора

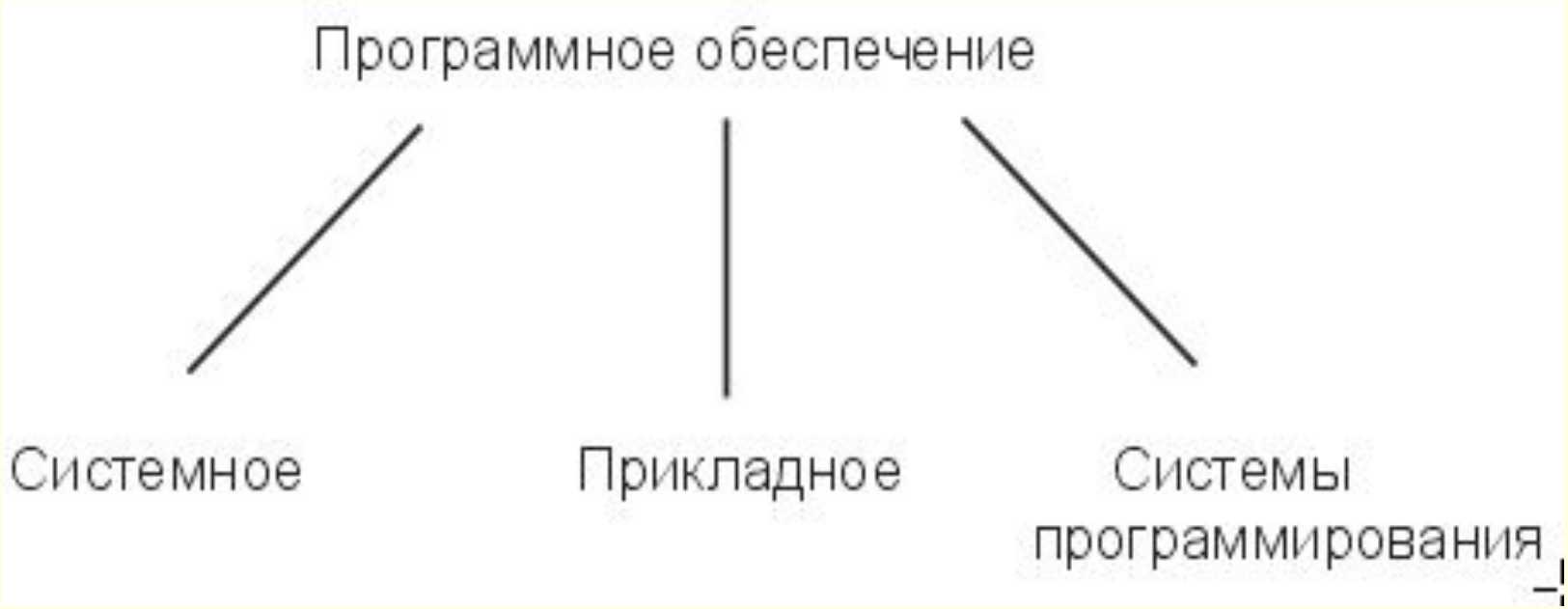
Таким образом, компилятор разбивается на составляющие модули:



Классификация программного обеспечения

Программное обеспечение(ПО) – совокупность программ, описаний и инструкций по их применению, позволяющих использовать компьютер как универсальную систему для хранения, обработки и обмена информацией.

Классификация ПО:



Классификация программного обеспечения

Системное ПО обеспечивает организацию вычислительного процесса и контроль за ходом его выполнения независимо от конкретной решаемой задачи.

Основная часть системного ПО – ***операционная система (ОС).***

ОС – набор программ, предназначенный для

- *управления устройствами компьютера*
- *управления файлами*
- *организации диалога с пользователем*
- *выполнения различных сервисных функций (обслуживание дисков: копирование, форматирование, сжатие файлов).*

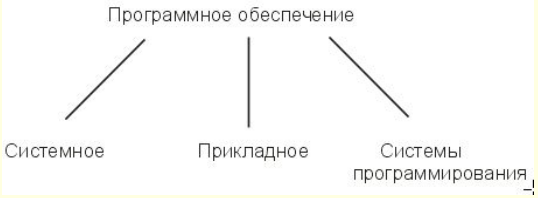
Классификация программного обеспечения

Прикладное ПО – программы, с помощью которых пользователь решает задачи, не прибегая к программированию.

Примеры прикладного ПО:

- графические редакторы (Adobe Photoshop, CorelDRAW),
- текстовые редакторы (Microsoft Word),
- табличные процессоры (Microsoft Excel),
- издательские системы (PageMaker, QuarkXPress),
- программы монтажа видеофильмов (Adobe Premiere),
- игры и др.

Классификация программного обеспечения



Система программирования (СП) – это программное обеспечение, позволяющее разрабатывать и исполнять на компьютере программы, написанные на языке более высокого уровня, чем язык машинных команд.

Наиболее известные и широко используемые СП: Microsoft Visual Studio (C++, C#, F#, Basic – в версии 2010), Borland Delphi и др.

Компоненты СП:

Парадигмы программирования

Виды парадигм программирования:

- процедурная,
- объектно-ориентированная,
- функциональная,
- логическая.

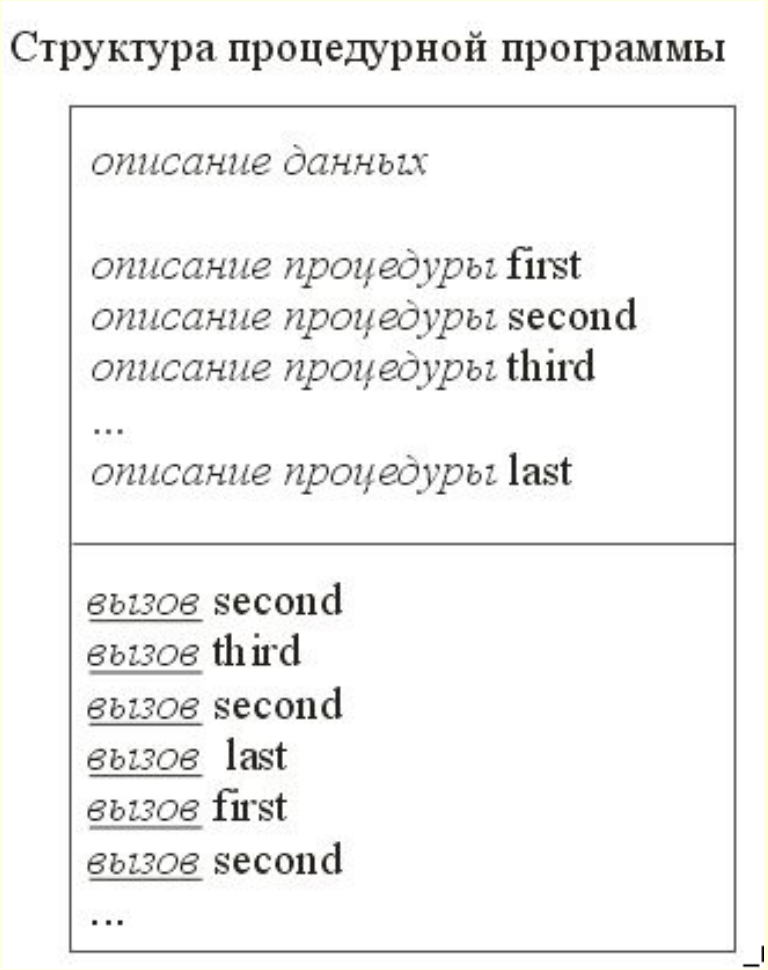
Каждая из парадигм используется для решения задач определённого класса.

Парадигмы программирования

Процедурные языки дают программисту возможность разбить программу на несколько процедур (подпрограмм).

Процедура - именованная последовательность действий для решения некоторой подзадачи поставленной задачи.

Языки процедурного программирования: ***Фортран, Паскаль, С и др.***



Парадигмы программирования

Объектно-ориентированная парадигма
базируется на понятии – объект.

==========

Объект объединяет (инкапсулирует) данные и действия (методы) по их обработке (Рис.*).

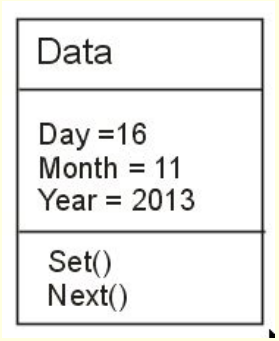


Рис.*

==========

Объекты взаимодействуют между собой, посылая и получая ***сообщения*** (Рис.**).

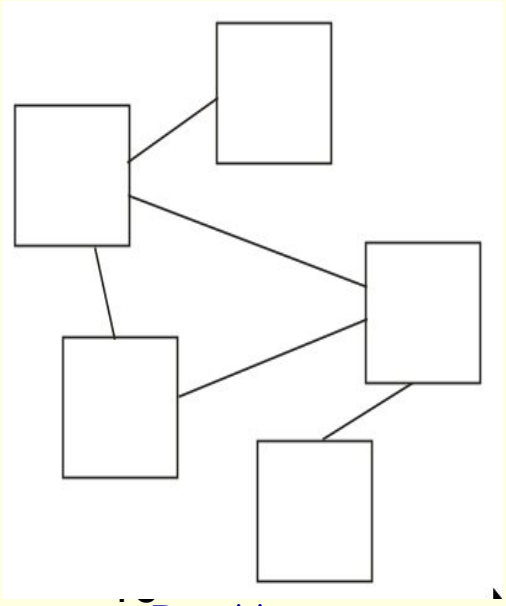


Рис.**

==========

Взаимодействуя между собой, объекты управляют ходом выполнения программы.

==========

Языки ООП: C++, C#, Java и др.

Парадигмы программирования

Идея логической парадигмы заключается в том, чтобы

1. описать совокупность утверждений на формальном языке
2. воспользоваться системой логического вывода для получения решения.

==========

Программист сообщает что известно и задаёт вопросы.

==========

Программа больше является описанием того, что нужно сделать, чем того, как это сделать.

==========

Язык логического программирования: Prolog и др.

Парадигмы программирования

Задача, предложенная в книге Н. Нильсона по искусственному интеллекту, решается в рамках ***логической парадигмы***.

Тони, Майкл и Джон – члены альпинистского клуба. Каждый член этого клуба является *горнолыжником*, *скалолазом* или *и тем и другим*.

- Все скалолазы не любят дождь.
- Все горнолыжники любят снег.
- Джон любит снег.
- Майкл любит всё, что не любит Тони, и не любит всё, что любит Тони.
- Тони любит снег и дождь.

Есть ли член альпинистского клуба, который является скалолазом и не является горнолыжником? Кто он?

Парадигмы программирования

Функциональное программирование – это способ составления программ, в которых единственным действием является *вызов функции* (функция понимается в математическом смысле).

Структура функциональной программы:

последовательность определений функций,

последовательность вызовов функций.

Пример. Базовая функция – ***макс (x, y).***

Вычисление наибольшего из 3 чисел:

макс (макс (x, y), z)

Вычисление наибольшего из 4 чисел:

макс(макс (x, y), макс (v, w)).

Языки функционального программирования - Lisp, F#, Haskell.