Основные понятия объектно-ориентированного программирования. Классы, объекты, методы, свойства

Структурное программирование – написание программы с использованием нескольких основных структурных единиц:

- **линейная структура,**
- **ветвление,**
- **циклы.**

Для укрупнения структурных единиц программы используются **подпрограммы (в Pascal процедуры и функции).**

Написание и отладка программы происходит сверху вниз.

К середине 90-х годов сформировался новый подход в программировании – объектно-ориентированный.

Программа рассматривается не только как последовательность выполняемых команд, но и как набор объектов, у каждого из которых имеются свои заранее определенные свойства.

С каждым **объектом** программы предусмотрены заранее определенные действия – **методы.**

Объектно-ориентированное программирование (ООП) — это методика разработки программ, в основе которой лежит понятие объект.

Объект — это некоторая структура, соответствующая объекту реального мира, его поведению.

Задача, решаемая с использованием методики ООП, описывается в терминах объектов и операций над ними, а программа при таком подходе представляет собой набор объектов и связей между ними.

Преимущество объектного подхода в программировании:

логическая простота построения программы.

Ко всему прочему, программист практически не ограничен в своих действиях.

Он может разрабатывать объекты любого назначения и с любыми свойствами.

Основные понятия ООП:

Объект – это понятие, сочетающее в себе совокупность данных и действий над ними.

Класс – это сложная структура, включающая, помимо описания данных, описание процедур и функций, которые могут быть выполнены над представителем класса – объектом.

Методы класса (процедуры и функции, объявление которых включено в описание класса) выполняют действия над объектами класса.

Свойства объекта - это характеристики состояния объекта. Каждое свойство объекта имеет свое значение. Любое свойство объекта – характеристика объекта, задаваемое в классе объектов.

- **Инкапсуляция** - скрытие внутренней структуры объекта за интерфейсом. То есть извне из всего объекта виден один интерфейс.
- **Наследование** - при создании нового класса объектов программист задает тип элементов этого класса (свойства) и функции (методы), выполняемые над объектами этого класса.
- **Полиморфизм** - объединение различных функций (методов) с разными входными параметрами под одним именем.

ПОКОЛЕНИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

Поколения	Языки программирования	Характеристики
Первое	Машинные	Ориентированы на использование конкретной ЭВМ, сложны в исполнении, требуют хорошего знания архитектуры ЭВМ
Второе	Ассемблеры Макроассемблеры	Более удобны для использования, но по – прежнему машино-зависимы
Третье	Языки высокого уровня	Мобильные, человеко-ориентированные, простые в освоении
Четвертое	Непроцедурные, объекто-ориентированные, языки запросов	Ориентированы на непрофессионального пользователя
Пятое	Языки искусственного интеллекта, экспертных систем и баз знаний	Ориентированы на повышение интеллектуального уровня ЭВМ и интерфейса с языками

Классификация ЯП

Фактор	Характеристика	Группы	Примеры ЯП
Уровень ЯП	Степень близости ЯП к архитектуре ПК	Низкий	Автокод, ассемблер
		Высокий	Fortran, Pascal, ADA, Basic, C и др.
		Сверхвысокий	Сетл
Специализация ЯП	Потенциальная или реальная область применения	Общего назначения (универсальные)	Fortran (инженерные расчёты), Cobol (Коммерческие задачи), Refal, Lisp(символьная обработка), Modula, ADA(программирование в реальном времени)
		Специализированные	
Алгоритмичность (процедурность)	Возможность абстрагироваться от деталей алгоритма решения задачи. Алгоритмичность тем выше, чем точнее приходится	Процедурные	Ассемблер, Fortran, Basic, Pascal, ADA
		Непроцедурные	Prolog, Langin