Эволюция методологий программирования

Хаотическое программирование

На заре программирования каждая программа разрабатывалась вручную одним разработчиком. Для этого периода, когда программы были штучным продуктом, характерен стиль программирования, который впоследствии получил название “хаотическое” программирование. Ни набор операторов, ни порядок их применения никак не регламентировался.

Главный недостаток разработанных таким образом программ – большие трудности его сопровождения. Программы, имеют структуру “итальянского спагетти” из-за частого применения в таких программах оператора перехода **`goto`**. Как правило, в таких программах может разобраться только автор, они плохо пригодны для тиражирования и превращения в товар.

Знаменательной датой в истории программирования стал 1968 год. Тогда создавшееся в программировании положение обсуждалось на международной конференции, получившей название “Кризис программного обеспечения”. Чарльз Хоар и Николаус Вирт впервые указали на необходимость придания языку программирования свойств, которые превратили бы его в “инструмент надежного создания сложных программ”.

Структурное программирование

Несколько позже автор структурного программирования голландский ученый **Дийкстра** (в некоторых источниках Дейкстра) выпустил работу под названием “заметки по структурному программированию”, в которой доказывал, что большинство программ сложны и неуправляемы из-за отсутствия в них четкой математической структуры.

Интуитивному и бессознательному (“хаотическому”) программированию он последовательно противопоставлял логически строгую методологию, предполагавшую ко всему прочему личную дисциплинированность и ответственность программиста. Одним из его кардинальных предложений было признание оператора перехода **`goto`** недопустимым в программировании.

Цель структурного программирования – разработка программы, которой присуща определенная структура, основанная на применении принципов структурного программирования.

Перечислим эти принципы:

1. Каждый программный модуль (блок, функция, процедура) должен иметь только один вход и один выход. Это позволяет максимально упростить стыковку модулей в программе.
2. В программах рекомендуется применять 4 типа конструкций:
   а) последовательность (модулей, операторов);
   б) разветвление (условный оператор);
   в) цикл;
   г) выбор из нескольких альтернатив (или переключатель).
3. Разработку программ рекомендуется вести сверху – вниз или по нисходящей стратегии.

Объектно-ориентированное программирование

Объектно-ориентированное программирование (ООП) – методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования

Если говорить образно, то объекты – это существительные, свойства объекта – это прилагательные, а методы объекта – это глаголы.

Программные **объекты** обладают **свойствами** и могут использовать **методы** обработки данных.

Классы объектов являются “шаблонами” (чертежами), определяющими наборы свойств, методов и событий, по которым создаются объекты. Основными классами объектов являются объекты, реализующие графический интерфейс проектов.

Объект, созданный по “шаблону” класса объектов, является **экземпляром класса** и наследует весь набор свойств, методов и событий данного класса. Каждый экземпляр класса объектов имеет уникальное для данного класса имя.

Основные понятия ООП:

- **Инкапсуляция** – способ спрятать сложную логику внутри класса, предоставив программисту лаконичный и понятный интерфейс для взаимодействия с сущностью.
- **Наследование** – способ легко и просто расширить существующий класс, дополнив его функциональностью.
- **Полиморфизм** – принцип «один интерфейс – множество реализаций». Например, метод `print` может вывести текст на экран, распечатать его на бумаге или вовсе записать в файл.