ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПАРАДИГМЫ И МЕТОДОЛОГИИ ПРОГРАММИРОВАНИЯ

О.Ю. Лягинова, М.Г. Можаева
кафедра математики и информатики ЧГУ

План

1. Понятие «парадигма программирования»
2. Императивное программирование
3. Декларативное программирование
4. Объектно-ориентированное программирование

Литература

Литература

1. Немнюгин С.А. Turbo Pascal : Программирование на языке высокого уровня : учебник для вузов / С. А. Немнюгин - 2-е изд. - СПб. : Питер, 2007. - 543 с.
2. Крылов Е.В. Техника разработки программ : учебник для вузов : В 2-х книгах. Кн.1 : Программирование на языке высокого уровня / Е. В. Крылов, В. А. Острейковский, Н. Г. Типикин ; Крылов Е.В., Острейковский В.А., Типикин Н.Г. - Москва : Высшая школа, 2007. - 376 с.
3. Парфилова Н.И. Программирование. Структурирование программ и данных : учебник для студ. учреждений высш. проф. образования / Н.И. Парфилова, А.Н. Пылькин, Б.Г. Трусов ; под ред. Б.Г. Трусова. — М. : Издательский центр «Академия», 2012. — 240 с. — (Сер. Бакалавриат).
4. http://ru.wikibooks.org/wiki/Основы_функционального_программирования/Вводная_лекция

План

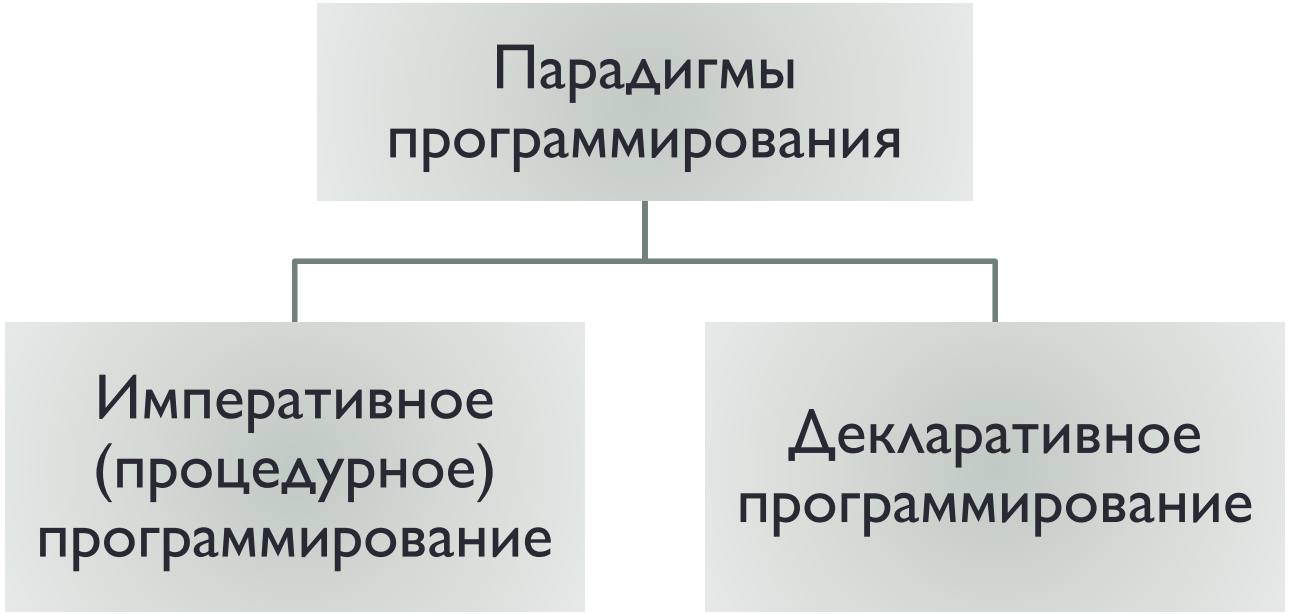
Парадигмы программирования

- Структура языка программирования и технология разработки программ определяются выбранной и положенной в основу этого языка парадигмой программирования.

Парадигма (от греч. *paradeigma* — образец, пример для подражания) — схема/модель постановки проблем и их решения, методы исследования.

- Языки программирования классифицируют по поддерживаемым парадигмам программирования.

Парадигмы программирования



Требует явно описывать действия и процесс изменения состояния исполнителя.

Требует задать описание цели, а также понятия и факты, необходимые для её достижения.

План

Императивное программирование

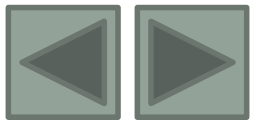
Императивное программирование (от греч. *imper* — действие) предполагает, что программа явно описывает алгоритм решения конкретной задачи (действия исполнителя), т.е. описывает как решать поставленную задачу.

Императивное программирование

- В основе императивного программирования находятся два основных понятия:

Алгоритм
(отсюда название алгоритмические языки)

Принстонская архитектура ЭВМ



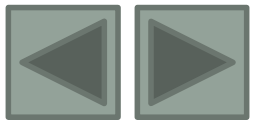
Императивное программирование

Выполнение действий над данными определённого типа

- Используются допустимые для данного типа операции и функции:
  - арифметические;
  - логические;
  - символьные;
  - строковые.

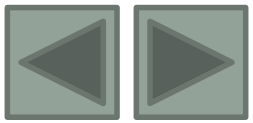
Управление ходом исполнения алгоритма

- Используются операторы, реализующие основные управляющие структуры:
  - следование;
  - ветвление;
  - цикл;
  - вызов подпрограммы.



Императивное программирование

- Традиционная область применения алгоритмических языков – вычислительные задачи и обработка данных различного типа: арифметических; логических; символьных.
- Языки программирования высокого уровня, поддерживающие императивный стиль — это известные традиционные языки, например Паскаль, Бейсик, Си.
- Базовая технология программирования, положенная в их основу — **структурное программирование**.



Структурное программирование

Структурное программирование – методология и технология разработки программных средств, основанная на:

трёх базовых конструкциях:
- следование;
- ветвление;
- цикл

принципах разработки:
- **программирование «сверху-вниз»** (нисходящее программирование);
- **модульное программирование** с иерархическим упорядочением связей между модулями/подпрограммами

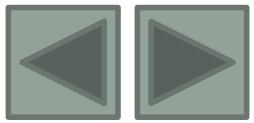
План

Декларативное программирование

Декларативное программирование (от лат. *declaratio* — объявление) — это попытка сначала реализовать программу для решения целого класса задач — «решатель», тогда для решения конкретной задачи этого класса достаточно декларировать в терминах данного языка только её условие (исходные данные и необходимый вид результата), не описывая алгоритм решения, так чтобы «решатель» (интерпретатор) смог выполнить процесс получения результата, реализуя известный ему способ решения.

Декларативное программирование

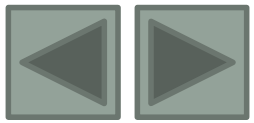
- Языки, поддерживающие декларативный стиль программирования, позволяют описывать «что» должно быть решено, а «как» уже известно «решателю».
- Декларативная программа не содержит алгоритма.



Декларативные языки

Наиболее известными декларативными языками являются:

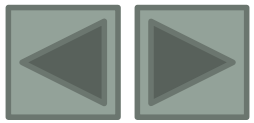
- языки функционального программирования (Lisp, Haskell);
- язык логического программирования (Prolog);
- SQL (англ. *structured query language* — «язык структурированных запросов»).



Функциональное программирование

Функциональное программирование — программирование в терминах композиции функций.

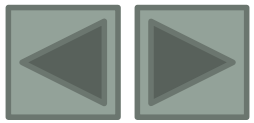
- Единственным действием функциональной программы является вызов функции.
- Функциональная программа представляет собой композицию вложенных друг в друга функций.
- Целью программы является вычисление значения исходной функции.
- Основной способ решения задач — рекурсия.



Функциональное программирование

- В функциональных языках программирования отсутствуют передача управления, оператор присваивания, ветвления и циклы, характерные для традиционных языков.
- Для представления программ и данных используется единая структура — символьное выражение, которое в памяти представляется в виде списка.
- Например, определение N-ого числа Фибоначчи (1, 1, 2, 3, 5, 8, 13, …) на Haskell.

```
fib 1 = 1
fib 2 = 1
fib n = fib (n – 2) + fib (n – 1)
```



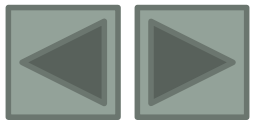
Задачи, решаемые функциональным программированием

- построение математического описания функций;
- описание динамических структур данных;
- доказательство наличия некоторого свойства программы;
- эквивалентная трансформация программ;
- получение остаточной процедуры и др.

Логическое программирование

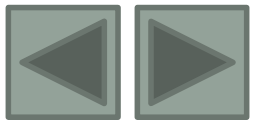
Логическое программирование — программирование в терминах фактов и правил вывода, с использованием языка, основанного на формальных логических исчислениях.

- Логическая программа представляет собой описание некоторой предметной области/задачи с помощью набора фактов, логических утверждений/правил и предположения, нуждающегося в доказательстве.
- Решение поставленной задачи — получение новых знаний осуществляется в виде запроса, представляющего собой логическую формулу.



Логическое программирование

- На языке Prolog описывается база данных, содержащая сведения об автомобилях: марка машины, год выпуска, цвет, цена. Программа позволяет строить запросы к базе.
- **Predicates** //утверждения об объекте
  car(string, integer, string, integer)
  Clauses //факты
  car(volvo, 1990, red, 1800).
  car(toyota, 1988, black, 2000).
  car(ford, 1994, white, 3000).
- **Goal** car(X, Y, _ , _), Y < 1992. //цель
  X = “volvo” Y = 1990
  X = "toyota" Y = 1988



Задачи, решаемые логическим программированием

- задачи искусственного интеллекта;
- задачи технологии знаний;
- экспертные системы и др.

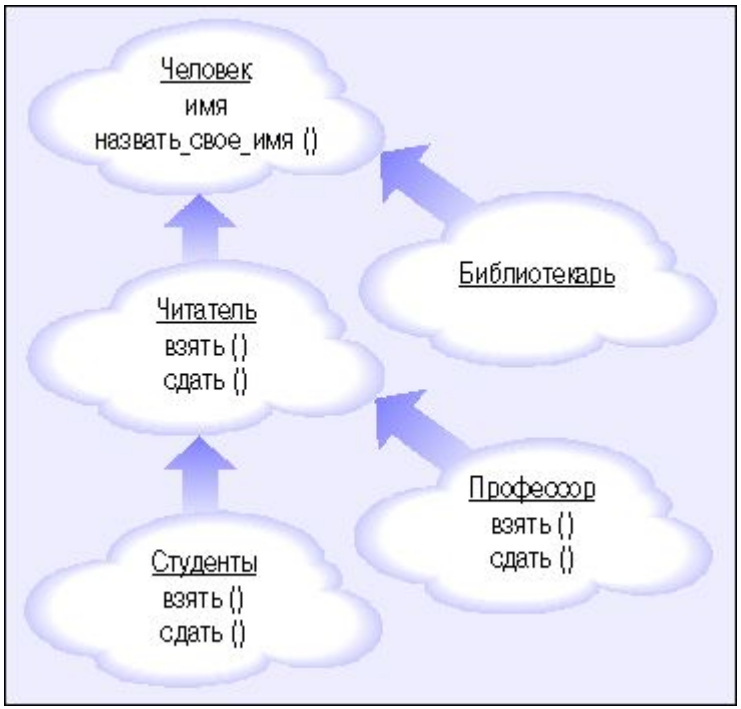
План

Объектно-ориентированное программирование

- Объектно-ориентированные языки программирования могут быть отнесены к императивным языкам, т.к. их вычислительная модель имеет процедурный характер, при этом содержат значительную декларативную компоненту — описание классов.

Понятия объектно-ориентированного программирования

- **Объектно-ориентированное программирование** (ООП) — методология программирования, в которой основными являются понятия «класс» и «объект».



- **Класс** является описываемой на языке терминологии исходного кода моделью ещё не существующей сущности (объекта). Фактически он описывает устройство объекта.
- **Объект** — это экземпляр класса.

Принципы объектно-ориентированного программирования

1. **Наследование** – описание нового класса на основе уже существующего с частично или полностью заимствующейся функциональностью.
2. **Инкапсуляция** – объединение данных и методов, работающие с ними, в классе и сокрытие деталей реализации от пользователя.
3. **Полиморфизм** – возможность объектов с одинаковым описанием иметь различную реализацию.

План

Суть метода «сверху вниз»

Сначала пишется текст основной программы, в которую вместо каждого логического фрагмента вставляется вызов подпрограммы, выполняющей данный фрагмент. Вместо настоящих, работающих подпрограмм, в программу вставляются «заглушки», которые ничего не делают.

Полученная программа проверяется и отлаживается. После того, как программист убедится, что общая структура программы верна, подпрограммы-заглушки последовательно заменяются на реально работающие, причём разработка каждой подпрограммы ведётся тем же методом, что и основной программы.

Разработка заканчивается, когда не останется ни одной «заглушки», которая не была бы удалена.

Рекурсивная функция

- **Рекурсивная функция** (от лат. *recursio* — возвращение) — это числовая функция числового аргумента, которая в своей записи содержит себя же.
- **Факториал числа *n*** (лат. *factorialis* — действующий, производящий, умножающий; обозначается ***n*!**) — произведение всех натуральных чисел от 1 до *n* включительно:

$$n! = 1 \cdot 2 \cdot \ldots \cdot n = \prod_{i=1}^{n} i.$$

- **Рекуррентная формула:**

$$n! = \begin{cases} 1 & n = 0, \\ n \cdot (n-1)! & n > 0. \end{cases}$$

Остаточная процедура

Если даны следующие объекты:

- $P(x_1, x_2, \ldots, x_n)$ — некоторая процедура.
- $x_1 = a_1,\ x_2 = a_2$ — известные значения параметров.
- $x_3, \ldots, x_n$ — неизвестные значения параметров.

Требуется получить остаточную процедуру $P_1(x_3, \ldots, x_n)$.

ЧЕРЕПОВЕЦКИЙ
ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ

Спасибо за внимание!

Выход