Лекция 1. Введение. Понятие вычислений. Декларативные и императивные знания. Язык программирования.

Компьютер умеет делать хорошо только 2 вещи:

- считать или выполнять вычисления
- сохранять результаты вычислений

Вычислительные возможности человека всегда были ограничены скоростью вычислений и скоростью записи результатов, как правило на бумажке.

Все знания могут быть отнесены только к двум типам:

- декларативные
- императивные

Декларативные знания – набор фактов. недостаток – невозможно понять как получить тот или иной факт. Пример про квадратный корень.

Императивные знания – набор рецептов или howtos отвечающих на вопрос как получить (вывести), необходимую информацию.

Воплощением императивных знаний является алгоритм.

Алгоритм – конечная последовательность простых (элементарных, понятных для выполнения) инструкций или шагов, для получения информации (результата).

Другими словами, алгоримт это конечный список инструкций, описывающих вычисления над предоставленными входными данными. Эти вычисления определяют последовательность четко определнных состояний связывающих входные данные с конечным значеним – результатом вычислений.

Некоторые сравнивают алгоритм с рецептом приготовления супа (ингридиенты – входные данные, последовательность операций – инструкции/команды)

Отметим, что алгоритмы:

- последовательны, порядок исполения важен
- содержат ряд проверок / тестов состояний
- содержат переходы между инструкциями которые связаны с результатами проверок

Первые компьютеры – с фиксированными програмами, например только для вычисления траектории движения снаряда

Manchester Mark I – первый компьютер с загружаемой программой. Ядром компьютера становится interpretator – программа или аппаратный компонент, который умеет выполнять происзвольный набор инструкций

И программа и инструкции хранятся в памяти. Обычно интерпретатор берет команды последовательно и их исполняет, но ингда встречаются команды (иногда опирающиеся на результаты проверок) которые заставляют интерпретатор пропустить несколько команд или наоборот вернуться на несколько шагов назад. Control flow

Язык программирования – такой набор инструкций и правил записи, на котором записываются программы/инструкции/рецепты (точнее формальная система для записи программ, предназначенных для выполнения на компьютере)

1936, Turing Машина Тьюринга абстрактный исполнитель (абстрактная вычислительная машина). Была предложена Аланом Тьюрингом в 1936 годудля формализации понятия алгоритма.

Структура: бесконечная лента, переходы, чтение, запись, элементарные вычисления

Тезис Тьюринга-Чарча: если функция вычислима, то машину тьюринга можно запрограммировать чтобы её вычислить

если -> не все задачи имеют вычислимые решения.

halting problem – проблема остановки, неформально: есть описание процедуры и входные данные. требуется определить

остановится ли программа с эти данными. Неразрешима на машине Тьюринга

Полнота по тьюрингу – (например для языка программирования) обозначает, что на языке программирования (вычилителе) можно записать алгоритм для вычисления любой вычислимой функции. Т.е. в некотором смысле декларируется эквивалентность языка программирования универсальной машине тьюринга.

Современные языки как правило обладают полнотой по тьюрингу, а значит могут использоваться для создания алгоритмов.

Среди тысяч языков программирования нет лучшего, каждый хорош для свлих целей

В структуре языка как правило:

- набор примитивынх конструкций (лексемы)
- синтаксис – описывает какие строки (наборы слов) чвляются корректными с точки зрения языка
- семантика

Переменные:

- в императивном программировании – именуемая область памяти
- в функциональном – просто имя, с которым может быть связано значение, или место.

переменная обладает

- время жизни
- область видимости

О Парадигмах

Парадигма программирования – набор идей и понятий [wiki] определяющих стиль написания программы и организацию и структуризацию вычислений. Единства в точном определении этого понятия нет.

Большинство языков мультипарадигменные, парадигма – стиль

Декларативное VS Императивное

императивное	декларативное
в противоположности к декларативному КАК	в противоположности к императивному ЧТО либо программа написана на Ф,Л

C/C++
assembler
java
python
...
	• DSL:
	  • make
	  • yacc (генератор парсеров)
	  • SQL
	  • Markup: HTML,XML, XAML,...
	• функциональное программирование
	  • Scheme
	  • haskel
	  • lisp
	• Логическое программирование
	  • prolog – автоматическое доказательство, логический вывод из набора фактов

сложно написать программу, просто ее исполнить	просто написать программу, сложно исполнить

Пример

Декларация: корень из x, такой Y>0 что Y*Y=x

Алгоритм:

приближение	частное x/y	среднее
1	2/1=2	(2+1)/2=1.5
1.5	2/1.5=1.33	(1.5+1.33)/2=1.42
1.42	2/1.42=1.41	(1.42+1.41)/2=...

Парадигмы (стили програмирования)

- Объектно-ориентированное: инкапсуляция, наследование, полиморфизм (без наследования, Объектное)
- Функциональное
- Обобщенное программирование
- Логическое
- Распределенное
- (*) Рефлексивное

Контрольные вопросы

- все ли функции являются вычислимыми?
- приведите пример декларативного языка?
- что такое язык программирования?
- должен ли алгоритм быть конечным?
- может ли переменная быть константной?

Дополнительные материалы

- Сравнение особенностей языков программирования в википедии
- C. Strachey. Fundamental Concepts in Programming Languages
- P.V. Roy, S.Haridi Concepts, Techniques, and Models of Computer Programming (Introduction only)