Лекция 1.

Раздел 1. Математические основы функционального и логического программирования

Тема 1. Общая характеристика декларативных языков программирования.
Тема 2. Математические модели, лежащие в основе императивных и декларативных ЯП. Семантический разрыв. Машина Тьюринга (Поста) - математическая модель языков программирования с императивной семантикой
Тема 3. Обзор основных понятий логики высказываний (ЛВ) и логики предикатов первого порядка (ЛП1). Аппликативные вычислительные системы (АВС)

План лекции.

1. Введение.
2. Классификация языков программирования.
3. Математические модели, лежащие в основе императивных и декларативных ЯП.
4. Логика высказываний, логика предикатов. Аппликативные модели.

Основная часть.

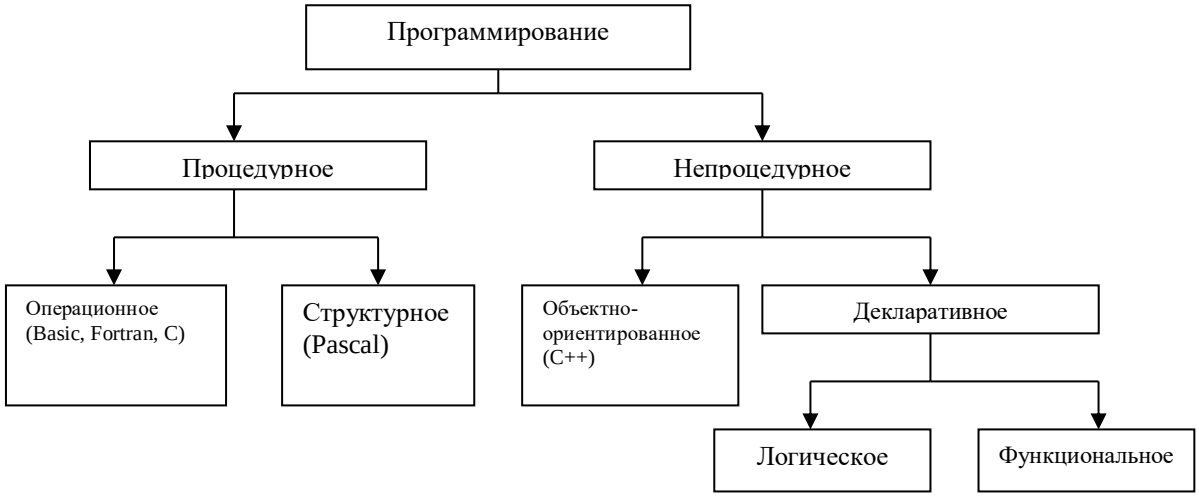
В настоящее время существуют четыре парадигмы, которые рассматриваются и как подходы к разработке программ на современных языках программирования, и как методы изучения информатики: это процедурная, объектно-ориентированная, функциональная и логическая парадигмы.

Основные стили программирования

Название стиля	*Основополагающие категории*
Логико-ориентированный	Цели, часто выраженные исчислениями предикатов
Ориентированный на правила	Правила «Если-то»
Ориентированный на ограничения	Инвариантные отношения
Процедурно-ориентированный	Алгоритмы, абстрактные термины
Объектно-ориентированный	Классы и объекты

Есть еще одна классификация языков программирования

Классификация языков программирования



Процедурное (структурное, директивное) программирование. В 60-70-х гг. XX столетия развивалась методика, которая получила название «структурное программирование».

При структурном (процедурном) программировании описывается процесс получения результата определенной последовательностью операторов (команд). Данные называются операндами (бывают простыми и сложными), разнородные данные образуют структуру.

Элементами структур данных является информация о характеристиках (атрибутах) объектов (имя, цена, количество, возраст и т.п.). В структурах данных элементы связаны между собой отношениями, которые могут иметь разный смысл (входить в состав, подчиняться и прочее). Связь между элементами всегда имеет один смысл: «перейти к выполнению команды».

Принципы структурной методики программирования.

При построении алгоритма используется пять базовых алгоритмических структур:

- следование – это однозначная линейка последовательных действий;

- ветвление – разделение алгоритма на два пути (две ветви) по некоторому условию с дальнейшим выходом на общее продолжение;

- цикл – повторение некоторой группы действий по условию;

- рекурсия – некоторая группа действий, при описании которой происходит ссылка на себя как на составную часть;

- подпрограмма – повторяющаяся часть совокупности действий, выделенная в отдельно описанный алгоритм.

Сложный алгоритм состоит из множества соединенных между собой базовых структур, для которых существует два способа соединения элементов алгоритма: последовательный и вложенный.

Метод последовательной детализации – это построение алгоритма «сверху вниз».

В эту группу входят языки Algol, Fortran, Pascal, Basic.

Преимущества структурных языков:

- простота в составлении программ (три базовые структуры – следование, ветвление, цикл);

- большую программу могут писать одновременно несколько исполнителей, что сокращает сроки окончания работы;

- простота проектирования и последующего изменения программы;

- возможность применения структурных программ для решения простых расчетных задач.

Объектно-ориентированное программирование. В этой парадигме считаются объекты (данные), которые могут активно взаимодействовать друг с другом с помощью механизма передачи сообщений. Каждые данные можно обрабатывать как единое целое. Программирование сводится к описанию объектов и структурированию исходных данных и результатов в такие объекты.

Объектно-ориентированное программирование является в настоящее время наиболее популярной технологией программирования. К этой группе относятся языки Visual Basic, Java, Object Pascal, C++. Объектно-ориентированное программирование является развитием технологии алгоритмического программирования, которое представляет собой алгоритм в виде последовательности различных алгоритмических структур (линейная, ветвление, цикл).

По своей сути объектно-ориентированное программирование – это создание приложений из объектов. Одни объекты приходится полностью создавать самостоятельно, тогда как другие можно позаимствовать в готовом варианте из разнообразных библиотек.

Главное место в технологии объектно-ориентированного программирования занимает событие. В качестве событий может рассматриваться щелчок кнопкой мыши на объекте (клик), нажатие определенной клавиши, открытие документа и т.д. В качестве реакции на событие вызывается определенная процедура, которая может изменять свойства объекта, вызывать его методы и т.д.

В объектно-ориентированном программировании у объектов есть возможность наследования свойств других объектов, которые являются родителями.

Преимущества объектно-ориентированного подхода состоят в следующем:

- простота построения новых классов, использование механизма наследования;

- полиморфизм – возможность работать с объектами, абстрагируясь от их класса;

- описание утилит и методов класса на любом языке программирования;

- инкапсуляция – структура объекта включает в себя как данные, так и методы работы с ним;

Объектно-ориентированные языки применяются для разработки операционных систем, трансляторов, баз данных, других прикладных и системных программ. В настоящее время они очень распространены.

Декларативное программирование как научное направление, связанное с попытками формализовать мышление человека, *имеет длительную историю.* Связь между логикой и программированием впервые проявилась в процессе формализации математики. Оказалось, что между вычислениями и доказательствами существует взаимодействие, состоящее в том, что всякое доказательство задает построение или вычисление того объекта, существование которого доказывается. С другой стороны, развитие самого программирования и усложнение реальных программ привели к необходимости формально выразить и доказать их свойства. Для этого используется математическая логика.

В качестве основной методологии разработки программных средств для японского проекта ЭВМ пятого поколения было избрано логическое программирование, ярким представителем которого является ***язык Пролог.*** Пролог основывается на таком разделе математической логики, как исчисление ***предикатов.*** Точнее, его базис составляет процедура доказательства теорем методом резолюции для хорновских дизъюнктов.

Функциональное программирование. Функциональное программирование возникло в 1962 г. вместе с созданием Дж. Маккарти языка программирования LISP. В основе функционального программирования находится λ-исчисление. Подавляющее большинство программ искусственного интеллекта составлено на языке LISP.

Язык LISP – один из первых языков обработки данных в символьной форме. В LISP и программа, и обрабатываемые ей данные представляются в одной и той же форме – форме списка.

Используемый в LISP функциональный подход к программированию основывается на той простой идее, что вся обработка информации и получение искомого результата могут быть представлены в виде вложенных и/или рекурсивных вызовов функций, выполняющих некоторые действия, так что значение одной функции используется как аргумент другой. Значение этой функции становится аргументом следующей и т.д. пока не будет получен результат – решение задачи.

Программы строятся из логически расчлененных определений функций. Определения состоят из управляющих структур, организующих вычисления, и из вложенных вызовов функций. Основными методами функционального подхода

(программирования) являются композиция и рекурсия. Все это представляет собой реализацию идей теории рекурсивных функций (вызывающих самих себя).

Имеется большое число систем программирования на LISP, реализованных для компьютеров различных типов. Как правило, это интерпретирующие системы, работающие в интерактивном (диалоговом) режиме. Соответствующие описания и команды вводятся с клавиатуры после приглашения («_»), затем прочитывается результат.

Преимущества функционального программирования:

- LISP может обрабатывать данные в символьной форме в виде списков;

- программа может обрабатывать и преобразовывать другие программы и даже самих себя.

Применение функциональных языков связано с разработкой искусственного интеллекта, с помощью рекурсивных функций описывают интеллектуальную деятельность людей.

Математические модели, лежащие в основе императивных и декларативных ЯП. ***Машина Тьюринга*** - это строгое математическое построение, математический аппарат (аналогичный, например, аппарату дифференциальных уравнений), созданный для решения определенных задач. Этот математический аппарат был назван “машиной” по той причине, что по описанию его составляющих частей и функционированию он похож на вычислительную машину. Принципиальное отличие машины Тьюринга от вычислительных машин состоит в том, что ее запоминающее устройство представляет собой бесконечную ленту: у реальных вычислительных машин запоминающее устройство может быть как угодно большим, но обязательно конечным. Машину Тьюринга нельзя реализовать именно из-за бесконечности ее ленты. В этом смысле она мощнее любой вычислительной машины.

В каждой машине Тьюринга есть две части:

1) неограниченная в обе стороны лента, разделенная на ячейки;

2) автомат (головка для считывания/записи, управляемая программой).

Машина Поста - абстрактная вычислительная машина, предложенная Эмилем Постом в 1936 году, создана независимо от машины Тьюринга, но сообщение о машине Поста опубликовано на несколько месяцев позднее. Отличается от машины Тьюринга большей простотой, при том обе машины алгоритмически «эквивалентны» и обе разработаны для формализации понятия алгоритма и решения задач об алгоритмической разрешимости, то есть, демонстрации алгоритмического решения задач в форме последовательности команд для машины Поста.

Машина Поста - это абстрактная (т.е. не существующая в арсенале действующей техники), но очень простая вычислительная машина. Она способна выполнять лишь самые элементарные действия, и потому ее описание и составление простейших программ может быть доступно ученикам начальной школы. Тем не менее на машине Поста можно запрограммировать - в известном смысле - любые алгоритмы. Изучение машины Поста можно рассматривать как начальный этап обучения теории алгоритмов и программированию. Разработка программ для машин Поста - достаточно эффективный этап в обучении алгоритмизации, т.к. в процессе написания этих программ учатся разбивать интуитивно понятные вычислительные процедуры на элементарные действия.

Машина Поста состоит из ленты и каретки (называемой также считывающей и записывающей головкой).

Обзор основных понятий логики высказываний (ЛВ) и логики предикатов первого порядка (ЛП1). Аппликативные вычислительные системы (АВС). Высказывание – повествовательное предложение, про которое можно сказать истинно оно или ложно. Соединяя различные высказывания союзами ”и”, ”или”, ”если..., то...”, ”не” (другими словами, используя различные логические операции), можно строить новые высказывания. Об истинности полученных высказываний можно судить по истинности исходных высказываний. Рассмотрим основные логические операции и их таблицы истинности, при этом высказывания будем обозначать большими латинскими буквами.

1. Конъюнкция: A ∧ B (читается "A и B"). Конъюнкция истинна тогда и только тогда, когда истины оба высказывания A и B.

2. Дизъюнкция: A∨B (читается "A или B"). Дизъюнкция истинна тогда и только тогда, когда истинно хотя бы одно из высказываний A или B. Другими словами, дизъюнкция ложна тогда и только тогда, когда ложны оба высказывания A и B.

3. Импликация: A → B (читается "если A, то B"). Импликация ложна тогда и только тогда, когда истинно A и ложно B.

4. Отрицание: ¬A (читается "не A"). Отрицание истинно тогда и только тогда, когда исходное выказывание A ложно. Истинностные значения логических операций отражены в следующих таблицах, где истина обозначена как 1, ложь – как 0.

Язык логики высказываний:

1. пропозициональные переменные (высказывания), которые будем обозначать большими латинскими буквами (возможно с индексами) A,B,C ...,

2. логические символы: ∧,∨,→,¬, 3.вспомогательныесимволы: открывающаяся скобка (,закрывающаяся скобка), запятая.

Формулы логики высказываний. Понятие формулы логики высказываний вводится индуктивно:

1. пропозициональная переменная есть формула,

2. если A, B – формулы, то (A∧B), (A∨B), (A → B), ¬A – формулы.

Соглашение. При записи формул внешние скобки опускают. Кроме того, опускают некоторые внутренние скобки, считая, что ∧ выполняется раньше ∨, которая выполняется раньше →.

Подформула. Введем индуктивно понятие подформулы:

1. подформулой пропозициональной переменной является она сама,

2. если формула имеет вид ¬A, то ее подформулами являются она сама, формула A и все подформулы формулы A; если формула имеет вид (A∧B), (A∨B) или (A → B), то ее подформулами являются она сама, формулы A и B и все подформулы формул A и B.

Формула логики высказываний называется тождественно истинной, если она принимает значения истина при любых значениях входящих в нее пропозициональных переменных. Формула логики высказываний называется выполнимой, если она принимает значения истина хотя бы при одном наборе значений входящих в нее пропозициональных переменных. Формула логики высказываний называется тождественно ложной, если она принимает значения ложь при любых значениях входящих в нее пропозициональных переменных. Определить тождественную истинность, тождественную ложность, выполнимость формулы можно записав ее нормальную форму. Элементарной конъюнкцией называется произвольная конъюнкция пропозициональных переменных или их отрицаний. Элементарной дизъюнкцией называется произвольная дизъюнкция пропозициональных переменных или их отрицаний. Дизъюнктивной нормальной формой (ДНФ) называется произвольная дизъюнкция элементарных конъюнкций. Конъюнктивной нормальной формой (КНФ) называется произвольная конъюнкция элементарных дизъюнкций. ДНФ (КНФ) называется совершенной и обозначается СДНФ (СКНФ), если каждая переменная, входящая в нее, входит с отрицанием или без в каждую элементарную конъюнкцию (элементарную дизъюнкцию) ровно один раз. Для каждой формулы существуют эквивалентные ей ДНФ и КНФ.

Легко доказываются следующие теоремы:

Теорема 1. Элементарная дизъюнкция тождественно истинна тогда и только тогда, когда она содержит некоторую пропозициональную переменную и ее отрицание.

Теорема 2. Элементарная конъюнкция тождественно ложна тогда и только тогда, когда она содержит некоторую пропозициональную переменную и ее отрицание.

Теорема 3. Формула тождественно истинна тогда и только тогда, когда в ее конъюнктивной нормальной форме каждая элементарная дизъюнкция содержит некоторую пропозициональную переменную и ее отрицание.

Теорема 4. Формула тождественно ложна тогда и только тогда, когда в ее дизъюнктивной нормальной форме каждая элементарная конъюнкция содержит некоторую пропозициональную переменную и ее отрицание.

Основные понятия логики предикатов. Пусть задано некоторое множество M и A: $M^n \to \{0,1\}$ – функция, которая принимает значение истина или ложь на каждом наборе значений переменных, то есть функция, которая при подстановке переменных становится высказыванием (истинным или ложным). Такие функции называют предикатами от соответствующего числа переменных. Если n = 0, то предикат является высказыванием. Пусть A: $M^n \to \{0,1\}$ – предикат от n переменных, тогда $\exists x_i A(x_1,x_2,...,x_n)$ и $\forall x_i A(x_1,x_2,...,x_n)$ – предикаты от $n-1$ переменной, где $\exists$ и $\forall$ – кванторы существования и всеобщности соответственно. Если A(x) – предикат от одной переменной, заданный на множестве M, то $\exists x A(x)$ и $\forall x A(x)$ – высказывания, причем $\forall x A(x)$ – истинно тогда и только тогда, когда A(x) истинно для любого $x \in M$ и $\exists x A(x)$ – истинно тогда и только тогда, когда A(x) истинно при некотором $x \in M$. Переменная, не связанная никаким квантором, называется свободной. Переменная, связанная либо квантором всеобщности, либо квантором существования, называется связанной. Язык логики предикатов:

1) предметные переменные. Будем их обозначать малыми латинскими буквами, возможно с индексами: x,y,z,...,x1,x2,x3,...,

2) логические символы $\wedge,\vee,\to,\neg,\forall,\exists$,

3) вспомогательные символы: запятая, открывающаяся скобка (, закрывающаяся скобка),

4) символы сигнатуры Σ, то есть а) предикатные символы: $A_n(x_1,x_2,...,x_n),B_k(y_1,y_2,...,y_k),...$, б) функциональные символы: $f_n(x_1,x_2,...,x_n),g_k(y_1,y_2,...,y_k),...$ Для каждого символа как предикатного, так и функционального указывается его местность, то есть число переменных, от которых он зависит. Нульместный предикатный символ называется символом высказываний, нульместный функциональный символ – символом константы. Интерпретация. Интерпретацией сигнатуры Σ на множестве M называется сопоставление каждому символу из Σ некоторой функции $f : M^n \to M$ (для функционального символа) или $A : M^n \to \{0,1\}$ (для предикатного символа). Модель. Моделью сигнатуры Σ называется набор $\langle M, f_1,f_2,...,f_n, A_1,A_2,...,A_k\rangle$, где M – непустое множество (называемое универсумом модели), а

f1,f2,...,fn,A1,A2,...,Ak – интерпретация сигнатуры Σ на этом множестве. Терм. Понятие терма вводится индуктивно:

1) каждая предметная переменная – терм,

2) каждый символ константы – терм,

3) если t1,t2,...,tn – термы и f – n–местный функциональный символ, то f(t1,t2,...,tn) также терм. Атомарная формула. Атомарной формулой называется выражение A(t1,t2,...,tn), где A – n–местный предикатный символ, а t1,t2,...,tn – термы. Формула. Свободное и связанное вхождение переменной в формулу. Понятие формулы вводится индуктивно. Вместе с определением формулы, дадим определение свободной и связанной переменной:

1) атомарная формула есть формула, каждая предметная переменная, входящая в атомарную формулу, входит в нее свободно,

2) пусть A – формула, тогда ¬A – формула, переменные, которые были в A свободны, в ¬A также свободны, которые были в A связанны, в ¬A также связанны,

3) пусть A и B – формулы, причем переменные, которые входят в одну из формул свободно, не могут входить в другую связанно. Тогда (A∧B),(A∨B),(A → B) –формулы, причем свободные переменные совпадают со свободными переменными, а связанные – со связанными переменными формул A и B,

4) Пусть A(x) – формула, содержащая переменную x свободно, тогда ∀xA(x), ∃xA(x) – формулы содержащие переменную x связанно. Формула называется замкнутой, если она не содержит свободных переменных. Формула A называется тождественно истинной, если она принимает значение истина на любой модели M, сигнатура которой содержит сигнатуру формулы, при любых значениях входящих в нее свободных переменных.

Формула A называется выполнимой, если она принимает значение истина на некоторой (хотя бы одной) модели M, сигнатура которой содержит сигнатуру формулы, при некоторых значениях входящих в нее свободных переменных. Формула A называется тождественно ложной, если она принимает значение ложь на любой модели M, сигнатура которой содержит сигнатуру формулы, при любых значениях входящих в нее свободных переменных.

Аппликативные вычислительные системы (ABC). **Аппликативное программирование** - один из видов декларативного программирования, в котором написание программы состоит в систематическом осуществлении применения одного объекта к другому. Результатом такого применения является объект, который может участвовать в применениях как в роли функции, так и в роли аргумента и так далее. Это делает запись программы математически ясной. Тот факт, что функция обозначается

выражением, свидетельствует о возможности использования значений-функций - функциональных объектов - на равных правах с прочими объектами, которые можно передавать как аргументы, либо возвращать как результат вычисления других функций.

Модели аппликативного программирования основываются, как правило, на комбинаторной логике или λ-исчислении. В комбинаторной логике единственный метаоператор - аппликация, обеспечивающая применение одного объекта к другому, в λ-исчислении, кроме аппликации, есть метаоператор λ-абстракции, с помощью которого возможно построение функций по выражениям, которые, в свою очередь, можно применять к другим объектам.

Аппликативный язык программирования - язык программирования, который предназначен для поддержки разработки программ способом получения результата вычисления функции, зависящей от комбинации переменных. Примерами аппликативных языков программирования служат функциональные языки Лисп и ML.