Лекция №1.
Общие сведения.

План:

- Что такое «структуры данных и алгоритмы»?
- Зачем нужны структуры данных и алгоритмы?
- Какую пользу принесет их изучение?

Зачем нужны структуры данных и алгоритмы?

Структурой данных называется способ организации данных в памяти компьютера (или на диске).

Примерами структур данных являются массивы, связанные списки, стеки, двоичные деревья, хеш-таблицы и т. д. Алгоритмы обеспечивают выполнение различных операций с этими структурами — например, поиск определенного элемента данных или сортировку данных.

Какие задачи помогает решить хорошее знание структур данных и алгоритмов?

Ситуации, в которых они могут пригодиться, можно приблизительно разделить на три категории:

- Хранение реальных данных.
- Инструментарий программиста.
- Моделирование.

Хранение реальных данных

Под реальными данными подразумеваются данные, описывающие физические сущности, внешние по отношению к компьютеру.

Например, запись в базе данных персонала описывает конкретного человека; запись в складской базе данных описывает деталь для автомобиля или бакалейный товар, а запись в базе данных финансовых операций — выписку счета за электричество.

Хранение реальных данных

«Физическим» примером хранения реальных данных служит картотека. Карточки могут использоваться для разных целей. Если записать на них имена, адреса и телефонные номера, мы получим адресную книгу. Если же на карточках записаны названия, местонахождение и стоимость предметов, то результатом будет инвентарная опись.

Характеристики структур данных

Структура данных	Достоинства	Недостатки
Массив	Быстрая вставка, очень быстрый доступ (если известен индекс)	Медленный поиск, медленное удаление, фиксированный размер
Упорядоченный массив	Поиск выполняется быстрее, чем в обычном массиве	Медленная вставка и удаление, фиксированный размер
Стек	Доступ в порядке LIFO («последним пришел, первым вышел»)	Медленный доступ к другим элементам
Очередь	Доступ в порядке FIFO («первым пришел, первым вышел»»)	Медленный доступ к другим элементам
Связанный список	Быстрая вставка, быстрое удаление	Медленный поиск
Двоичное дерево	Быстрый поиск, вставка, удаление (если дерево остается сбалансированным)	Сложный алгоритм удаления
Красно-черное дерево	Быстрый поиск, вставка, удаление. Дерево всегда сбалансировано	Сложность
Дерево 2-3-4	Быстрый поиск, вставка, удаление. Дерево всегда сбалансировано. Аналогичные деревья хорошо подходят для хранения данных на диске	Сложность
Хеш-таблица	Очень быстрый доступ (если известен ключ)	Медленное удаление, медленный доступ (если ключ неизвестен), неэффективное использование памяти
Куча	Быстрая вставка, удаление, доступ к наибольшему элементу	Медленный доступ к другим элементам
Граф	Моделирование реальных ситуаций	Некоторые алгоритмы медленны и сложны

Алгоритмы

Для большинства структур данных необходимо знать, как выполняются операции:

- Вставки нового элемента данных.
- Поиска заданного элемента.
- Удаления заданного элемента.

База данных

Термином *база данных* обозначается совокупность всех данных, используемых в некоторой ситуации. Предполагается, что все элементы базы данных имеют одинаковый формат. Например, в адресной книге, составленной из карточек, все карточки совместно образуют базу данных. Иногда в этом же значении используется термин *файл*.

Запись

Записями называются логические блоки, на которые делится база данных. Запись определяет формат хранения информации. В нашей аналогии с картотекой каждая отдельная карточка представляет запись. Запись включает всю информацию о некоторой сущности в ситуации, в которой существует много похожих сущностей. Запись может описывать человека в базе данных отдела кадров, деталь машины в базе данных складского учета, рецепт в файле поваренной книги и т. д.

Поле

Запись обычно делится на несколько полей. Поле содержит данные определенного типа. В адресной картотеке имя человека, его адрес или номер телефона является отдельным полем.

Классификация языков программирования

Процедурные языки программирования

Программа состоит из последовательности императивных команд (явно, задающих какие преобразования выполнять над данными). Данные хранятся в виде переменных.

Классификация языков программирования

Логические языки программирования

Языки программирования данного типа основываются на формальной логике и булевой алгебре. Программа не содержит в себе явных алгоритмов. Задаётся описание условий задачи и логических соотношений, по которым система программирования строит дерево вывода и находит решения задачи.

Классификация языков программирования

Функциональные языки программирования

Функциональное программирование основывается на использование списков и функций. Переменные могут отсутствовать вообще.

Примером процедурного языка является язык программирования Паскаль. Язык Пролог является логическим языком программирования, а язык Лисп есть функциональный язык программирования.

Классификация языков программирования

Объектно-ориентированное программирование

Переменные и функции группируются в объекты и классы. Благодаря этому достигается более высокий уровень структуризации и абстракции программы. Одни объекты (классы) могут порождаться от других объектов (классов).

СПАСИБО!