Эффективность

Эффективность системы обычно оценивается отдельно по **каждому ресурсу вычислительной машины**.

Часто используют следующие критерии:

- **время ответа или выполнения**
- **объем оперативной памяти**
- **объем внешней памяти**
- **количество обслуживаемых внешних устройств**

и др.

Часто оценки становятся взаимозависимыми, например:

- время выполнения программы можно уменьшить за счет повышения объема оперативной памяти.
- при экономии памяти может увеличиться время выполнения;
- повысить эффективность работы с файлами можно за счет оперативной памяти;
- повысить эффективность оперативной памяти можно за счет внешней памяти.

Рекомендуется повышать эффективность,
но не за счет технологичности
разрабатываемого программного обеспечения.

Способы экономии памяти.

- ✓ Избегать дублирования данных;
- ✓ Использовать динамическую память, т.е. управлять памятью.

Способы уменьшения времени выполнения.

- В первую очередь необходимо **обращать внимание на циклические структуры**.
  (обращать внимание на управление циклом и на тело цикла)

- Выбрать **другой метод** реализации операции и др.

Эффективность программы **зависит от компилятора**.

Например,

Машинно-зависимый компилятор может выполнять оптимизацию кодов **на уровне машинных команд**. В частности, исключить лишние пересылки или использовать другие более эффективные команды.

Машинно-независимый компилятор может выполнить оптимизацию **на уровне исходного кода.**

Например, вынести неизменяемое выражение из цикла и т.п.

Критерии качества программных продуктов

Критерии качества:

- правильность
- универсальность
- надежность
- проверяемость
- точность результатов
- защищенность
- эффективность
- адаптируемость
- повторная входимость
- реентерабельность

Абстрактные структуры данных

- ✓ Такие структуры предназначены для удобного хранения и доступа к информации.

- ✓ Они позволяют организовать эффективный интерфейс для выполнения различных операций с хранимыми объектами.

Статические структуры представляют собой структурированное множество примитивных, базовых, структур. **Память для них выделяется один раз**, объем остается неизменным до уничтожения структуры. ***Динамические структуры*** представляют связные данные.

Элемент динамической структуры состоит из двух частей:

- Информационная часть (поля) – для пользователя;
- Служебная часть (поля связок), в которой содержатся указатели связывающие элементы структуры – для программиста.

Недостатки: дополнительная память и менее эффективный по времени доступ.

Поэтому связное представление **практически никогда** не применяется в задачах, где логическая структура данных имеет вид вектора или массива - с доступом по номеру элемента,

но часто применяется в задачах, где логическая структура требует другой исходной информации доступа (таблицы, списки, деревья и т.д.).

Стек - список с переменной длиной, в котором включение и исключение элементов выполняются только с одной стороны списка, называемого вершиной стека. Основные операции: - включение нового элемента;- исключение элемента из стека;- определение текущего числа элементов в стеке; - очистка стека и др.

Очередь - список с переменной длиной, в котором включение элементов выполняется только с одной стороны списка (конец очереди), а исключение - с другой стороны (начало очереди).

Дек - очередь с двумя концами, где включение и исключение элементов может осуществляться с любого из двух концов списка.