Концепция разработки программного модуля

МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ

- ***Модульное программирование*** — это организация программы как совокупности независимых блоков, называемых модулями, структура и поведение которых подчиняются определенным правилам

Понятия и положения модульного программирования:

- большие задачи разбиваются на ряд более мелких, функционально самостоятельных подзадач — модулей, которые связаны между собой только по входным и выходным данным;
- модуль представляет собой «черный ящик» с одним входом и одним выходом. Это позволяет безболезненно производить модернизацию программы в процессе ее эксплуатации, облегчает ее сопровождение, а также позволяет разрабатывать части программного проекта на разных языках программирования;

Понятия и положения модульного программирования:

- в каждом модуле должны осуществляться ясные задачи. Процесс декомпозиции нужно продолжать до тех пор, пока не будет ясного понимания назначения всех модулей и их оптимального сочетания;
- исходный текст модуля должен иметь заголовок и интерфейсную часть, где отражаются назначение модуля и все его внешние связи;
- в ходе разработки модулей программы следует предусматривать специальные блоки операций, учитывающие реакцию на возможные ошибки в данных или в действиях пользователя

Отличие от процедур и функций

- Традиционные правила сферы действия глобальных и локальных переменных для модулей не работают. Эта языковая конструкция разработана так, чтобы исключить влияние глобальных переменных, объявленных в главной программе, на внутренние описания модуля. Поэтому, если возникает необходимость ввести доступные для всех блоков программы глобальные описания то следует создать модуль глобальных объявлений и включить его в список импорта всех модулей, где нужны его описания

СТРУКТУРА МОДУЛЯ В *OBJECT PASCAL*

- Первая строка модуля начинается с ключевого слова:
- **unit <идентификатор_модуля>;**
- Для правильной работы среды программирования это имя должно совпадать с именем дискового файла, в который помещается исходный текст модуля.

СТРУКТУРА МОДУЛЯ В *OBJECT PASCAL*

- ***{Интерфейсный раздел}*** **interface**
- где описывается взаимодействие данного модуля с другими пользовательскими и стандартными модулями, а также с главной программой.
- Здесь содержатся объявления всех глобальных объектов модуля, которые должны стать доступными основной программе и/или другим модулям. При объявлении глобальных подпрограмм в интерфейсной части указывается только их заголовок.

СТРУКТУРА МОДУЛЯ В *OBJECT PASCAL*

- Связь модуля с другими модулями устанавливается специальным предложением:
- ***{Список импорта интерфейсного раздела}* uses <список_модулей>**
- В этом списке через запятые перечисляются идентификаторы модулей, информация интерфейсных частей которых должна быть доступна в данном модуле.
- *{Список экспорта интерфейсного раздела}* const type var
- procedure function

СТРУКТУРА МОДУЛЯ В *OBJECT PASCAL*

- Список экспорта состоит из подразделов описания констант, типов, переменных, заголовков процедур и функций, которые определены в данном модуле, но использовать которые разрешено во всех других модулях и программах, включающих имя данного модуля в своей строке uses.
- Для процедур и функций здесь описываются только заголовки, но с обязательным полным описанием формальных параметров.

- ***{Раздел реализации)* implementation**
- В этом разделе указывается реализационная (личная) часть описаний данного модуля, которая недоступна для других модулей и программ.

- ***{Список импорта раздела реализации)*** **uses**
- В этом списке через запятые перечисляются идентификаторы модулей, информация интерфейсных частей которых должна быть доступна в данном модуле.
- Здесь целесообразно описывать идентификаторы всех необходимых модулей, информация из которых не используется в описаниях раздела interface данного модуля.

- ***{Подразделы внутренних для модуля описаний}*** label const type var
- procedure function
- В этих подразделах описываются метки, константы, типы, переменные, процедуры и функции, которые описывают алгоритмические действия, выполняемые данным модулем, и которые являются «личной собственностью» исключительно только данного модуля. Эти описания недоступны ни одному другому модулю.

- Исполняемая часть содержит описания подпрограмм, объявленных в интерфейсной части. Описанию подпрограммы должен предшествовать заголовок, в котором можно опускать список формальных параметров и тип результата для функции. Если заголовки указаны с параметрами, то их список должен быть идентичен такому же списку для соответствующей процедуры или функции в разделе interface.

- ***{Раздел инициализации}* initialization**
- В этом разделе между ключевыми словами initialization и finalization располагаются операторы начальных установок, необходимых для запуска корректной работы модуля.
- Эти операторы исполняются до передачи управления основной программе и обычно используются для подготовки ее работы.
- Операторы разделов инициализации модулей, используемых в программе, выполняются при начальном запуске программы в том же порядке, в каком идентификаторы модулей описаны в предложениях uses файла проекта.
- Если операторы инициализации не требуются, то зарезервированное слово initialization может быть опущено.

- ***{Раздел завершения)* finalization**
- Раздел завершения finalization является необязательным и может присутствовать только вместе с разделом инициализации initialization.
- В разделе завершения располагается список операторов, которые будут выполняться при завершении модуля, что обычно происходит при окончании работы приложения.
- Разделы finalization модулей приложения выполняются в порядке, противоположном выполнению разделов initialization этих модулей.
- Раздел завершения используется, как правило, для освобождения ресурсов, которые выделяются приложению в разделе инициализации. Это гарантирует корректное завершение приложения, что особенно это важно, когда приложение заканчивается по возникновению исключительных ситуаций.