ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ С++

ВВЕДЕНИЕ

Традиционная технология программирования складывалась в условиях, когда основными потребителями программ были научные учреждения, вычислительные ресурсы были ограничены, а проблемы сопровождения, по существу, неизвестны. Основными критериями качества программы считалась ее узко понимаемая эффективность и компактность. Со временем сложность программ возросла настолько, что на их разработку уходили годы труда большого коллектива, а в результате программы появлялись с большим опозданием и содержали много ошибок.

Кризис программного обеспечения привел к необходимости создания нового способа разработки программ, который снижал бы общие затраты на протяжении всего цикла – от замысла до завершения эксплуатации. Такая технология появилась в начале 70-х годов XX в. и была названа *структурным программированием*. В его основе лежит сочетание теории программирования и личного опыта высококвалифицированных программистов, а также учет современных требований к программам и промышленного характера их производства.

Структурное программирование – это технология создания программ, позволяющая путем соблюдения определенных правил уменьшить время разработки и количество ошибок, а также облегчить возможность модификации программы. Структурный подход охватывает все стадии разработки проектов: спецификацию, проектирование, собственно программирование и тестирование.

С увеличением объема программы становится все более сложным удерживать в памяти все детали. Естественным способом борьбы со сложностью любой задачи является ее разбиение на части. В С+ + задача может быть разделена на более простые подзадачи с помощью *функций*, после чего программу можно рассматривать на уровне взаимодействия функций.

Использование функций является первым шагом к повышению степени абстракции программы и ведет к упрощению структуры.

Следующим шагом в повышении уровня абстракции программы является группировка функций в отдельные файлы (модули), компилируемые раздельно. Получившиеся в результате компиляции объектные модули объединяются в исполняемую программу с помощью компоновщика.

Следующий шаг – описание собственных типов данных, позволяющих структурировать и группировать информацию, представляя ее в более естественном виде. Для этого данные объединяются с обрабатывающими их алгоритмами. Данный подход получил название *объектно-ориентированное программирование* (ООП); он позволяет разрабатывать крупные программные системы.

Бурное развитие вычислительной техники, потребность в эффективных средствах разработки программного обеспечения привели к появлению на рынке программ целого ряда систем программирования, ориентированных на быструю разработку приложений, среди которых следует отметить *Microsoft Visual Basic, Borland Delphi, Borland C++ Builder, Microsoft Visual Studio.* В основе систем быстрой разработки лежит технология визуального проектирования и событийно-управляемого программирования, суть которой заключается в том, что среда разработки берет на себя большую часть работы по генерации программного кода, оставляя программисту работу по конструированию диалоговых окон и написания функций обработки событий, возникающих в программе. Понятно, что такие системы резко повышают производительность работы программиста.

Представленная учебная программа позволяет слушателю пройти все технологии программирования на языке C++ в их естественном историческом развитии, поскольку каждая технология строится основе предыдущих, либо пропустить какие-то этапы, если слушатель знаком с основами данных технологий.

Часть 1. Структурное программирование

Состав языка. Алфавит языка. Идентификаторы. Зарезервированные слова. Знаки операций. Константы. Комментарии. Основные типы данных. Структура программы. Переменные и выражения.Операции. Операторы. Функции ввода-вывода. Указатели и ссылки. Массивы. Строки.Типы данных, определенные пользователем. Структуры.

Часть 2. Модульное программирование

Объявление и определение функции. Глобальные переменные. Возвращаемое значение. Параметры функции. Передача массивов в качестве параметров. Передача имен функций в качестве параметров. Параметры со значениями по умолчанию. Перегрузка функций. Функции стандартной библиотеки. Функции ввода-вывода. Функции для работы с символами и строками. Математические функции.

Директивы препроцессора. Области действия и пространства имен.

Часть 3. Объектно-ориентированное программирование

Классы. Описание класса. Описание объектов. Конструкторы. Конструкторы копирования. Статические элементы класса. Дружественные функции и классы. Дружественные функции. Дружественный класс. Деструкторы. Перегрузка операций. Наследование. Ключи доступа. Простое наследование. Виртуальные методы и механизм позднего связывания. Обработка исключительных ситуаций.Синтаксис исключений. Перехват исключений. Иерархии исключений.

Часть 4. Визуальное проектирование и событийно-управляемое программирование

Основы программирования в среде *C++ Builder*. Среда разработки *C++ Builder*. Структура проекта. Разработка интерфейса программы. Формы. Компоненты. Свойства компонентов. События и функции обработки

событий. Запуск и сохранение проекта. Доработка проекта. Обработка исключительных ситуаций.

Создание отчуждаемого приложения. Название программы. Значок приложения. Компоновка программы. Графика. Холст. Карандаш и кисть. Графические примитивы. Иллюстрации. Битовые образы. Мультимедиа. Воспроизведение видео: компонент *Animate*. Воспроизведение звука: компонент *MediaPlayer*.

Базы данных (БД). Структура БД. Псевдоним БД. Компоненты доступа и манипулирования данными. Создание БД. Доступ к БД. Отображение данных. Манипулирование данными. Выбор информации из БД. Перенос программы управления БД на другой компьютер.

Консольные приложения. Функции ввода/вывода в текстовом режиме.

Создание установочного диска. Программа *InstallShield Express*. Выбор устанавливаемых компонентов. Конфигурирование системы пользователя. Настройка диалогов. Системные требования. Создание образа установочной дискеты.