УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите приведенную лекцию в своей рабочей тетради, законспектируйте основные тезисы.

Результаты работы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 30.01.2023.

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лекция № 32
Тема: «Объектно-ориентированное программирование»

Процедурное программирование - есть отражение фон Неймановской архитектуры компьютера. Программа, написанная на процедурном языке, представляет собой последовательность команд, определяющих алгоритм решения задачи. Основная команда присвоение, с помощью которой определяется и меняется память компьютера. Программа производит преобразование содержимого памяти, изменяя его от исходного состояния к результирующему.

Алгоритм — набор инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за конечное число действий. Объектно-ориентированное программирование (ООП) — это методика разработки программ, в основе которой лежит понятие объект.

Объект — это некоторая структура, соответствующая объекту реального мира, его поведению. Задача, решаемая с использованием методики ООП, описывается в терминах объектов и операций над ними, а программа при таком подходе представляет собой набор объектов и связей между ними. Системы объектно-ориентированного программирования (ООП) дают возможность визуализировать процесс создания графического интерфейса

разрабатываемого приложения Взаимодействие программных объектов между собой и их изменения описываются с помощью программного кода.

Создание программного кода в ООП базируется на использовании алгоритмических структур различных типов (линейной, ветвления, цикла), исполнителями которых выступают программные объекты.

Класс — разновидность абстрактного типа данных в объектно-ориентированном программировании (ООП), характеризуемый способом своего построения.

Другие абстрактные типы данных — метаклассы, интерфейсы, структуры, перечисления, — характеризуются какими-то своими, другими особенностями. Наряду с понятием «объекта» класс является ключевым понятием в ООП. Программные объекты обладают свойствами, могут использовать методы и реагируют на события Классы объектов являются «шаблонами», определяющими наборы свойств, методов и событий, по которым создаются объекты.

Основными классами объектов являются классы, реализующие графический интерфейс проектов Объект в программировании — некоторая сущность в виртуальном пространстве, обладающая определѐнным состоянием и поведением, имеющая заданные значения свойств (атрибутов) и операций над ними. Как правило, при рассмотрении объектов выделяется то, что объекты принадлежат одному или нескольким классам, которые определяют поведение (являются моделью) объекта. Термины «экземпляр класса» и «объект» взаимозаменяемы. Объект, созданный по шаблону класса объектов, является экземпляром класса и наследует весь набор свойств, методов и событий данного класса Каждый экземпляр класса имеет уникальное для данного класса имя.

Различные экземпляры класса обладают одинаковым набором свойств, однако значения свойств у них могут отличаться Каждый объект обладает определенным набором свойств, первоначальные значения которых можно установить.

Значения свойств объектов можно изменять и в программном коде: Объект.Свойство := ЗначениеСвойства.

Метод в объектно-ориентированном программировании — это функция или процедура, принадлежащая какому-то классу или объекту. Как и процедура в процедурном программировании, метод состоит из некоторого количества операторов для выполнения какого-то действия и имеет набор входных аргументов.

Различают простые методы и статические методы (методы класса): простые методы имеют доступ к данным объекта (конкретного экземпляра данного класса), статические методы не имеют доступа к данным объекта и для их использования не нужно создавать экземпляры (данного класса). Методы предоставляют интерфейс, при помощи которого осуществляется доступ к данным объекта некоторого класса, тем самым, обеспечивая инкапсуляцию данных.

В зависимости от того, какой уровень доступа предоставляет тот или иной метод, выделяют:

– открытый (public) интерфейс — общий интерфейс для всех пользователей данного класса;

– защищенный (protected) интерфейс — внутренний интерфейс для всех наследников данного класса;

– закрытый (private) интерфейс — интерфейс, доступный только изнутри данного класса.

Такое разделение интерфейсов позволяет сохранять неизменным открытый интерфейс, но изменять внутреннюю реализацию. Обратиться к методу объекта можно также с использованием точечной нотации: Объект.

Метод Событие в объектно-ориентированном программировании — это сообщение, которое возникает в различных точках исполняемого кода при выполнении определенных условий.

События предназначены для того, чтобы иметь возможность предусмотреть реакцию программного обеспечения. Для решения

поставленной задачи создаются обработчики событий: как только программа попадает в заданное состояние, происходит событие, посылается сообщение, а обработчик перехватывает это сообщение.

В общем случае в обработчик не передается ничего, либо передается ссылка на объект, инициировавший (породивший) обрабатываемое событие. В особых случаях в обработчик передаются значения некоторых переменных или ссылки на какие-то другие объекты, чтобы обработка данного события могла учесть контекст возникновения события.

Самое простое событие — это событие, сообщающее о начале или о завершении некоторой процедуры. Событие, по сути, сообщает об изменении состояния некоторого объекта. Наиболее наглядно события представлены в пользовательском интерфейсе, когда каждое действие пользователя порождает цепочку событий, которые, затем обрабатываются в приложении. Событие может создаваться пользователем (щелчок мышью или нажатие клавиши) или быть результатом воздействия других объектов.

В качестве реакции на событие вызывается определенная процедура, которая может изменять свойства объекта или вызывать его методы Графический интерфейс. Визуальное программирование позволяет делать графический интерфейс разрабатываемых приложений на основе форм и управляющих элементов. В роли основных объектов при визуальном программировании выступают формы (Forms). Форма представляет собой окно, на котором размещаются управляющие элементы. Управляющие элементы — это командные кнопки (CommandButton), переключатели, или «флажки» (Checkbox), поля выбора, или «радиокнопки» (OptionsButton), списки (ListBox), текстовые поля (TextBox) и др. Все основанные на объектах языки (C#, Java, C++, Smalltalk, Visual Basic и т.п.) должны отвечать трем основным принципам объектно-ориентированного программирования (ООП):

- Инкапсуляция;
- Наследование;
- Полиморфизм;

Инкапсуляция — это механизм программирования, объединяющий вместе код и данные, которыми он манипулирует, исключая как вмешательство извне, так и неправильное использование данных. В объектно-ориентированном языке данные и код могут быть объединены в совершенно автономный черный ящик.

Внутри такого ящика находятся все необходимые данные и код. Когда код и данные связываются вместе подобным образом, создается объект. Иными словами, объект — это элемент, поддерживающий инкапсуляцию.

Следующий принцип ООП — наследование — касается способности языка позволять строить новые определения классов на основе определений существующих классов. По сути, наследование позволяет расширять поведение базового (или родительского) класса, наследуя основную функциональность в производном подклассе (также именуемом дочерним классом) Т.е. наследование представляет собой процесс, в ходе которого один объект приобретает свойства другого объекта. Это очень важный процесс, поскольку он обеспечивает принцип иерархической классификации.

Полиморфизм - это свойство, которое позволяет одно и тоже имя использовать для решения нескольких технически разных задач. В общем смысле, концепцией полиморфизма является идея "один интерфейс, множество методов". Это означает, что можно создать общий интерфейс для группы близких по смыслу действий. Преимуществом полиморфизма является то, что он помогает снижать сложность программ, разрешая использование одного интерфейса для единого класса действий. Выбор конкретного действия, в зависимости от ситуации, возлагается на компилятор.