Члены класса, атрибуты и методы, играют важнейшую роль в объектно-ориентированном программировании. Они представляют данные и поведение класса.

Атрибуты — это переменные, которые содержат данные и определяют состояние объекта. Они могут быть разных типов, например целочисленные, с плавающей запятой, строковые или определённые пользователем объекты. Они объявлены внутри класса, и к ним можно получить доступ и изменить их с помощью экземпляров объекта или экземпляров класса. Например, класс “Person” может иметь такие атрибуты: “Name”, “Age”, “Gender”.

Во многих современных языках программирования атрибутам, так же, как и другим членам класса, можно задавать различный уровень доступа к данному члену класса. Инструмент определения доступа к членам класса необходим в процессе инкапсулирования класса [1]. Основные уровни доступа это:

Public – член класса доступен на любом уровне – им может манипулировать сам класс, имеющий данный член; наследники класса; пользователи класса.

Protected – доступен для изменения самим классом, наследниками.

Private – доступен только для самого класса [2].

**Например, при разработке своего проекта VR-симуляции я реализовал класс ItemsGrabber, который имеет различные уровни доступа к членам класса. Класс отвечает за подбор предметов, имеет public метод GrabItem, private атрибуты класса для реализации поведения.**

**Реализация на языке C# следующая:**

**public class ItemsGrabber**

**{**

**private const int grabMagnitude = 100;**

**public void Grab(Item item)**

**{**

**if (item.magnitude < grabMagnitude) return;**

**Item.grab();**

**}**

**}**

Методы представляют поведение класса. Это функции, которые могут манипулировать атрибутами объекта или выполнять некоторые другие действия. Методы объявляются внутри класса, и доступ к ним можно получить с помощью экземпляров объекта. В том же примере класса “Person” методами могут быть “changeName”, “calculateAge” и “printInfo”.

Сигнатура функции относится к списку параметров и их типов данных для функции, а также к типу возвращаемого значения функции. Например, функция сложения двух целых чисел может иметь сигнатуру: int Add (int num1, int num2);

Тело функции относится к реализации функции и содержит фактический код функции для выполнения своей задачи. Для приведенной выше сигнатуры тело функции может быть:

int Add(int num1, int num2) { return num1 + num2; }

Статические члены класса — это атрибуты или методы, связанные с классом, а не с экземпляром этого класса. Они объявляются с помощью ключевого слова «static», и к ним можно получить доступ без необходимости создания экземпляра объекта класса. Статические члены полезны для создания общих данных между всеми объектами класса или для реализации служебных функций, которым не требуется доступ к каким-либо данным, относящимся к конкретному экземпляру.

**Например, при разработке своего проекта VR-симуляции я реализовал класс WidgetCreator, который имел статический член класса GlobalWidgetCount, это целочисленный атрибут, который инкрементировался при создании виджета из любого экземпляра класса и декрементировался при уничтожении виджета из любого экземпляра класса.**

**Реализация на языке программирования C# следующая:**

**public class WidgetCreator**

**{**

**private static int GlobalWidgetCount = 0;**

**public static int GetGlobalWidgetCount() { return GlobalWidgetCount; }**

**public Widget CreateWidget() { GlobalWidgetCount++; return new Widget(); }**

**public void DeleteWidget(Widget\* widget) {widget.Dispose(); delete widget; GlobalWidgetCount—;}**

**}**

Конструкторы — это специальные типы методов, которые вызываются при создании объекта класса. Они используются для инициализации атрибутов объекта. Конструкторы могут быть двух типов: с параметрами и без них.

Конструкторы без параметров являются конструкторами по умолчанию. Они не принимают никаких аргументов и имеют пустую реализацию. Например, рассмотрим следующий код:

class Person

{

public string Name;

public int Age;

public Person() { }

}

Приведенный выше код объявляет класс “Person” без какого-либо параметризованного конструктора. Инициализировать атрибуты “Name” и “Age” в таком случае необходимо через обращение к атрибутам экземпляра класса. Например:

Person worker = new Person();

Worker.Name = “Oleg”;

Worker.Age = 22;

Конструкторы с параметрами принимают один или несколько аргументов, которые используются для инициализации атрибутов объекта. Например, рассмотрим следующий код:

class Person

{

public string Name;

public int Age;

public Person(string name, int age)

{

this.Name = name;

this.Age = age;

}

}

Приведенный выше код объявляет класс “Person” с параметризованным конструктором, который принимает два аргумента: “name” и “age”. Ключевое слово this используется для ссылки на текущий создаваемый объект.

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рисунок 1. Описание структуры класса, его членов, использование класса.

В заключение, понимание членов класса, включая атрибуты и методы, сигнатуру и тело функции, статические члены класса и конструкторы с параметрами и без них, необходимо для создания объектно-ориентированных программ. Эти концепции позволяют создавать хорошо спроектированный и организованный код, эффективный и удобный в сопровождении.

Источники:

1 – w3schools, C++ Encapsulation, <https://www.w3schools.com/cpp/cpp_encapsulation.asp>

2 – Microsoft, Модификаторы доступа (Руководство по программированию в C#)), <https://learn.microsoft.com/ru-ru/dotnet/csharp/programming-guide/classes-and-structs/access-modifiers>