Az objektumorientált programozás (OOP) egy programozási paradigma, amelyet az adatok és az azokon végrehajtandó műveletek összekapcsolására és szervezésére használnak. Az OOP azt a modellt használja, hogy az alkalmazásokat objektumokra osztja, amelyek adatait és műveleteit egyesíti. Az objektumok az osztályok példányai, és az osztályok olyan sablonok, amelyek meghatározzák az adott objektumok tulajdonságait (attribútumokat) és viselkedését (metódusokat).

Az OOP négy fő alapelvére gyakran hivatkoznak:

1. \*\*Öröklődés: \*\* Az osztályok létrehozhatnak új osztályokat, amelyek öröklik az ősosztály (szülőosztály) tulajdonságait és metódusait. Ez lehetővé teszi az osztályok közötti hierarchikus struktúra kialakítását és a kód újra felhasználását.

2. \*\*Polimorfizmus: \*\* Ez azt jelenti, hogy ugyanazon metódus neve különböző osztályokban eltérő módon viselkedhet. Ez lehetővé teszi az azonos nevű metódusok felülírását az örökölt osztályokban.

3. \*\*Absztrakció: \*\* Az absztrakció az adatok és az azokon végrehajtandó műveletek közötti határokat határozza meg. Az osztályok az adatokat reprezentálják, és azokhoz kapcsolódó metódusokon keresztül lehet velük interakcióba lépni, anélkül, hogy részleteiben bemutatná az adatok belső működését.

4. \*\*Encapsulation (kapszulázás):\*\* Ez az elv arra ösztönzi a fejlesztőket, hogy az adatokat és a hozzájuk kapcsolódó műveleteket egyetlen egységbe, azaz az osztályba zárják. Az adatok védelmét is szolgálja, mivel az osztályok bizonyos részeit privát (nem elérhető) módon lehet definiálni, így csak az osztályon belül lehet hozzájuk férni.

Az objektumorientált programozás rugalmas és strukturált módon segíti a fejlesztőket az alkalmazások felépítésében és karbantartásában. Olyan fogalmakat és eszközöket nyújt, amelyek lehetővé teszik a komplex problémák felbontását kisebb, könnyebben kezelhető részekre. Ezáltal az OOP segíti a fejlesztőket az újra felhasználható, karbantartható és kiterjeszthető kód írásában.

Az objektumorientált programozás (OOP) egy programozási módszer, amely az alkalmazásokat objektumokra osztja. Ezek az objektumok adatokat és műveleteket egyesítenek. Az osztályok meghatározzák az objektumok tulajdonságait és viselkedését. Az OOP négy fő elve az öröklődés, polimorfizmus, absztrakció és kapszulázás. Ennek segítségével könnyebben kezelhető, karbantartható és újra felhasználható kódot lehet létrehozni.

using System;

// Példa osztály létrehozása: Autó

public class Auto

{

// Adattagok (mezők)

private string gyarto;

private string modell;

private int evjarat;

// Konstruktor az osztályhoz

public Auto(string gyarto, string modell, int evjarat)

{

this.gyarto = gyarto;

this.modell = modell;

this.evjarat = evjarat;

}

// Metódus adataink kiírására

public void AutoAdatai()

{

Console.WriteLine($"Gyártó: {gyarto}, Modell: {modell}, Évjárat: {evjarat}");

}

}

class Program

{

static void Main(string[] args)

{

// Példa az Auto osztály használatára: példány létrehozása

Auto auto1 = new Auto("Toyota", "Corolla", 2020);

Auto auto2 = new Auto("Ford", "Focus", 2018);

// A példányok metódusainak hívása

Console.WriteLine("Első autó adatai:");

auto1.AutoAdatai();

Console.WriteLine("\nMásodik autó adatai:");

auto2.AutoAdatai();

// Objektumorientált programozás alapelvének követése: adatok rejtése

// Az adattagok privátak, ezért nem közvetlenül érhetőek el a következő módon:

// auto1.gyarto = "BMW"; // Nem lehetséges, mert a gyarto privát

// Viszont az osztályon belül módosíthatóak vagy elérhetőek a privát adattagok a megfelelő metódusokon keresztül.

}

}