**Выполнили студент 215 группы**

**Марков К.А.**

**Практическое занятие 9**

**Тема: «Абстрактные классы»**

**Цель работы:** Ознакомиться с а) разработкой абстрактных классов; б) наследованием от абстрактного класса; в) переопределения абстрактных методов классом-наследником.

**Задание:**

**Вариант 7:** Разработать программу, используя абстрактный класс и образуя его потомки:

1. написать программу для расчета объема и площади поверхности следующих геометрических фигур: шар, цилиндр;

б) написать программу, которая организует поиск по списку квартир дома по номеру квартиры/фамилии квартиросъемщика;

а) abstract class Figures

{

// Абстрактный метод для расчета объема

public abstract double Volume();

// Абстрактный метод для расчета площади поверхности

public abstract double SurfaceArea();

}

class Sphere : Figures

{

private double radius;

public Sphere(double radius)

{

this.radius = radius;

}

public override double Volume()

{

return (4.0 / 3.0) \* Math.PI \* Math.Pow(radius, 3);

}

public override double SurfaceArea()

{

return 4 \* Math.PI \* Math.Pow(radius, 2);

}

}

class Cylinder : Figures

{

private double radius;

private double height;

public Cylinder(double radius, double height)

{

this.radius = radius;

this.height = height;

}

public override double Volume()

{

return Math.PI \* Math.Pow(radius, 2) \* height;

}

public override double SurfaceArea()

{

return (2 \* Math.PI \* radius \* height) + (2 \* Math.PI \* Math.Pow(radius, 2));

}

}

static void Main(string[] args)

{

Sphere sphere = new Sphere(3);

Cylinder cylinder = new Cylinder(2, 5);

Console.WriteLine("Шар:");

Console.WriteLine("Объем: " + sphere.Volume());

Console.WriteLine("Площадь поверхности: " + sphere.SurfaceArea());

Console.WriteLine("\nЦилиндр:");

Console.WriteLine("Объем: " + cylinder.Volume());

Console.WriteLine("Площадь поверхности: " + cylinder.SurfaceArea());

Console.ReadKey();

}

Б) abstract class Flat

{

public int ApartmentNumber { get; set; }

public string TenantName { get; set; }

public Flat(int apartmentNumber, string tenantName)

{

ApartmentNumber = apartmentNumber;

TenantName = tenantName;

}

// Абстрактный метод для вывода информации о квартире

public abstract void DisplayInfo();

}

// Класс для представления квартиры с арендатором

class RentedFlat : Flat

{

public RentedFlat(int apartmentNumber, string tenantName) : base(apartmentNumber, tenantName)

{

}

// Реализация метода для вывода информации о квартире

public override void DisplayInfo()

{

Console.WriteLine($"Квартира №{ApartmentNumber} снимается арендатором {TenantName}");

}

}

// Класс для представления пустой квартиры

class EmptyFlat : Flat

{

public EmptyFlat(int apartmentNumber) : base(apartmentNumber, "Пусто")

{

}

// Реализация метода для вывода информации о квартире

public override void DisplayInfo()

{

Console.WriteLine($"Квартира №{ApartmentNumber} свободна");

}

}

// Класс, организующий поиск квартир по номеру или фамилии

class ApartmentSearch

{

private List<Flat> flats = new List<Flat>();

// Метод для добавления квартиры

public void AddFlat(Flat flat)

{

flats.Add(flat);

}

// Метод для поиска квартиры по номеру

public Flat SearchByApartmentNumber(int apartmentNumber)

{

foreach (var flat in flats)

{

if (flat.ApartmentNumber == apartmentNumber)

{

return flat;

}

}

return null;

}

// Метод для поиска квартиры по фамилии арендатора

public Flat SearchByTenantName(string tenantName)

{

foreach (var flat in flats)

{

if (flat.TenantName == tenantName)

{

return flat;

}

}

return null;

}

}

static void Main(string[] args)

{

// Создаем экземпляр класса для поиска квартир

ApartmentSearch search = new ApartmentSearch();

// Предлагаем пользователю ввести данные о квартирах

Console.WriteLine("Введите данные о квартирах или нажмите Enter для завершения ввода:");

// Цикл для ввода данных о квартирах

while (true)

{

// Запрашиваем номер квартиры

Console.Write("Номер квартиры: ");

string input = Console.ReadLine();

// Если ввода нет или пользователь нажал Enter, завершаем ввод данных

if (string.IsNullOrEmpty(input))

break;

int apartmentNumber;

// Пытаемся преобразовать введенные данные в число (номер квартиры)

if (!int.TryParse(input, out apartmentNumber))

{

Console.WriteLine("Некорректный ввод. Пожалуйста, введите целое число для номера квартиры.");

continue;

}

// Запрашиваем фамилию арендатора (если есть)

Console.Write("Фамилия арендатора (если есть): ");

string tenantName = Console.ReadLine();

// Запрашиваем информацию о статусе аренды квартиры

Console.Write("Является ли квартира арендованной? (yes/no): ");

string isRentedInput = Console.ReadLine().ToLower();

// Проверяем, является ли ответ пользователя "yes", если да, то квартира арендованная, иначе - свободная

bool isRented = isRentedInput == "yes";

// Создаем объект в зависимости от статуса аренды квартиры и добавляем его в список квартир

if (isRented)

{

search.AddFlat(new RentedFlat(apartmentNumber, tenantName));

}

else

{

search.AddFlat(new EmptyFlat(apartmentNumber));

}

}

// Выводим информацию о квартире, если пользователь хочет ее найти

Console.WriteLine("\nПоиск квартиры:");

// Запрашиваем номер квартиры для поиска

Console.Write("Введите номер квартиры для поиска: ");

int searchApartmentNumber;

// Пытаемся преобразовать введенные данные в число (номер квартиры)

if (int.TryParse(Console.ReadLine(), out searchApartmentNumber))

{

// Ищем квартиру по номеру

Flat foundFlat = search.SearchByApartmentNumber(searchApartmentNumber);

// Если квартира найдена, выводим информацию о ней

if (foundFlat != null)

{

Console.WriteLine("Найдена квартира:");

foundFlat.DisplayInfo();

}

else

{

// Если квартира не найдена, сообщаем об этом

Console.WriteLine("Квартира с таким номером не найдена.");

}

}

else

{

// В случае некорректного ввода номера квартиры сообщаем об этом

Console.WriteLine("Некорректный ввод. Пожалуйста, введите целое число для номера квартиры.");

}

Console.ReadKey();

}

**Вывод:** Ознакомился с а) разработкой абстрактных классов; б) наследованием от абстрактного класса; в) переопределения абстрактных методов классом-наследником.