**ПРАКТИЧЕСКОЕ ЗАДАНИЕ 2**

**Классы. Библиотеки FCL. Классы как типы и объекты этих типов**

Задание 1.



Листинг программы:

namespace job1

{ public class A

{

public int a;

public int b;

public double FuncCalc(int b)

{

return (Math.Pow(b, 10));

}

public double ZnachVyr(int b)

{

return (4 / (a + 2)) \* b;

}

public A(int a, int b)

{

this.a = a;

this.b = b;

}

static void Main(string[] vs)

{

Console.WriteLine("Enter numbers with enter: ");

A a = new(int.Parse(Console.ReadLine()), int.Parse(Console.ReadLine()));

Console.WriteLine($"Подсчёт степени { a.FuncCalc(a.b)}");

Console.WriteLine($"Подсчёт выражения {a.ZnachVyr(a.b)}");

}

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| А =4;  B= 5; | Подсчёт степени 9765625  Подсчёт выражения 0 |

Анализ результатов:

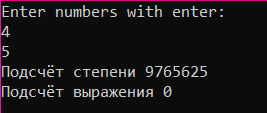


Рисунок 1.1 – Результат работы программы

Задание 2. Построить иерархию классов в соответствии с вариантом задания:

Млекопитающее, парнокопытное, птица, животное.

Листинг программы:

using System;

namespace job2

{

internal class Program

{

public class Mlekop : Animal

{

string title;

public Mlekop() : base()

{

this.title = string.Empty;

Input();

}

void Input()

{

Console.WriteLine("Введите название млекопитающего : ");

title = Console.ReadLine();

}

public override string Vivod()

{

return base.Vivod() + "\n" +

" Имя - "+ title +" ";

}

}

static void Main(string[] args)

{

Animal ziv = new Mlekop();

ziv.Vvod();

Console.WriteLine(ziv.Vivod());

}

}

public class Parnok : Animal

{

string title;

public Parnok() : base()

{

this.title = String.Empty;

Input();

}

void Input()

{

Console.WriteLine("Введите название животного ");

title = Console.ReadLine();

}

public override string Vivod()

{

return base.Vivod() + " Название парнокопытного - " + title + " ";

}

}

public class Bird : Animal

{

string title;

public Bird () : base()

{

this.title = String.Empty;

Input();

}

void Input()

{

Console.WriteLine("Введите название птицы: ");

title = Console.ReadLine();

}

public override string Vivod()

{

return base.Vivod() + " Название птицы- " + title+ " ";

}

}

public class Animal

{

int years;

string vid\_animal;

public Animal()

{

this.vid\_animal = String.Empty;

this.years = 0;

//Vvod();

}

public void Vvod()

{

Console.WriteLine("Введите разновидность животного: ");

vid\_animal = Console.ReadLine();

Console.WriteLine("Введите количество лет: ");

years = int.Parse(Console.ReadLine());

}

public virtual string Vivod()

{

return "Разновидность животного - " + vid\_animal + "\n" +

"Количество лет - " + years + "\n";

}

}

}

Таблица 1.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Введите название млекопитающего :  Медведь  Введите разновидность животного:  Млекопитающее  Введите количество лет:  12 | Разновидность животного - Млекопитающее  Количество лет - 12  Имя - Медведь |

Анализ результатов:

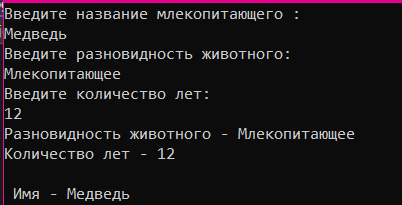


Рисунок 1.2 – Результат работы программы

Задание 3.

Описать класс «файл», содержащий сведения об имени, дате создания и длине

файла. Предусмотреть инициализацию с проверкой допустимости значений

полей. Описать метод добавления информации в конец файла и свойства для

получения состояния файла.

Написать программу, демонстрирующую все разработанные элементы класса.

Листинг программы:

try

{

File im = new File("n\_file", "12345", "15.05.2022");

Console.WriteLine(im.name);

Console.WriteLine(im.LenghtText);

im.text = im.AppendFile("13245", im.text);

Console.WriteLine(im.text);

Console.WriteLine(im.LenghtText);

Console.WriteLine(im.Status);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

class File

{

public string name;

public string text;

public string date;

public string status;

public string Status { get { return status; } }

public int LenghtText { get { return (text.Length); } }

public File(string name = "File1", string text = "", string date = "00.00.0000", string status = "")

{

try

{

if (string.IsNullOrEmpty(name)) throw new Exception("Name can not be empty");

this.name = name;

this.text = text;

if (text.Length != 0) { status = "Файл заполнен"; } else { status = "Файл пустой"; }

this.date = date;

this.status = status;

}

catch (Exception ex) { Console.WriteLine(ex.Message); }

}

public string AppendFile(string userText, string text)

{

text = text + userText;

if (text.Length == 0) { status = "Файл пустой"; } else { status = "Файл заполнен"; }

return text;

}

}

Таблица 1.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | n\_file  5  1234513245  10  Файл заполнен |

Анализ результатов:

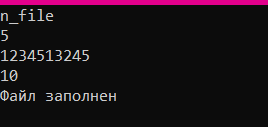


Рисунок 1.3 – Результат работы программы