1. **Классы**

Задание 1. Создать класс А с целочисленными полями а и b и двумя методами согласно варианту. Внутри класса реализовать конструктор для инициализации a и b. Создать объект класса и продемонстрировать работу со всеми элементами класса.

Метод вычисления значения выражения 1/a + 1/sqrt(b), метод вычисления a6

Листинг программы:

namespace Task\_1

{

class Task\_1

{

public static void Main(string[] args)

{

Console.Write("Введите первое значение (a): ");

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

Console.Write("Введите второе значение (b): ");

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

ClassA ca = new ClassA(numA, numB);

Console.Write($"Выражение 1/a + 1/sqrt(b)= {ca.Function()}");

Console.Write($"\nВыражение a^6= {ca.FunctionPow()}");

}

}

class ClassA

{

public int numA;

public int numB;

public ClassA(int numA, int numB)

{

this.numA = numA;

this.numB = numB;

}

public double Function()

{

return 1 / numA + 1 / Math.Sqrt(numB);

}

public double FunctionPow()

{

return Math.Pow(numA, 6);

}

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 2 5 | 0,4472… 64 |

Анализ результатов:

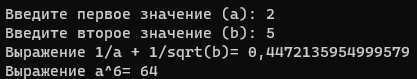


Рисунок 1.1 – Результат работы программы

Источник: собственная разработка

Задание 2. Построить иерархию классов в соответствии с вариантом задания: Двигатель, двигатель внутреннего сгорания, дизель, реактивный двигатель.

Листинг программы:

namespace Task\_2

{

class Task\_2

{

public static void Main(string[] args)

{

Console.WriteLine("Выберете тип двигателя: 1-двигатель внутреннего сгорания / 2-дизельный двигатель / 3-реактивный двигатель");

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

if (num == 1)

{

var PetrolEngine = CreatePetrolEngine();

Console.WriteLine(PetrolEngine.Output());

}

else if (num == 2)

{

var DieselEngine = CreateDieselEngine();

Console.WriteLine(DieselEngine.Output());

}

else if (num == 3)

{

var jetEngine = CreateJetEngine();

Console.WriteLine(jetEngine.Output());

}

else

{

Console.WriteLine("Некорректный ввод!");

}

}

private static void Input(out string name, out int power, out double volume)

{

Console.Write("Введите наименование двигателя: ");

name = Console.ReadLine();

Console.Write("Введите мощность двигателя: ");

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

Console.Write("Введите обьем двигателя: ");

volume = double.Parse(Console.ReadLine());

}

private static JetEngine CreateJetEngine()

{

string name;

int power;

double volume;

Input(out name, out power, out volume);

Console.Write("Введите тип реактивного двигателя: ");

JetEngineType jetEngineType = JetEngineType.Rocket;

while (!Enum.TryParse<JetEngineType>(Console.ReadLine(), true, out jetEngineType))

{

Console.Write("Введите тип реактивного двигателя:");

}

return new JetEngine(name, power, volume, jetEngineType);

}

private static DieselEngine CreateDieselEngine()

{

string name;

int power;

double volume;

Input(out name, out power, out volume);

Console.Write("Введите тип камеры сгорания: ");

DieselEngineType DieselEngineType = DieselEngineType.Separate;

while (!Enum.TryParse<DieselEngineType>(Console.ReadLine(), true, out DieselEngineType))

{

Console.Write("Введите тип камеры сгорания:");

}

return new DieselEngine(name, power, volume, DieselEngineType);

}

private static PetrolEngine CreatePetrolEngine()

{

string name;

int power;

double volume;

Input(out name, out power, out volume);

Console.Write("Введите тип камеры сгорания: ");

PetrolEngineType PetrolEngineType = PetrolEngineType.Single;

while (!Enum.TryParse<PetrolEngineType>(Console.ReadLine(), true, out PetrolEngineType))

{

Console.Write("Введите тип камеры сгорания:");

}

return new PetrolEngine(name, power, volume, PetrolEngineType);

}

}

public class Engine

{

string nameEngine;

int powerEngine;

double volumeEngine;

public Engine(string name, int power, double volume)

{

this.nameEngine = name;

this.powerEngine = power;

this.volumeEngine = volume;

}

public virtual string Output()

{

return $"Название двигателя: {nameEngine}, " +

$"Обьем двигателя: {volumeEngine}, " +

$"Мощность двигателя: {powerEngine}";

}

}

}

Таблица 2.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1 RB34 260 2,0 manu Multi | RB34 260 2 Multi |

Анализ результатов:

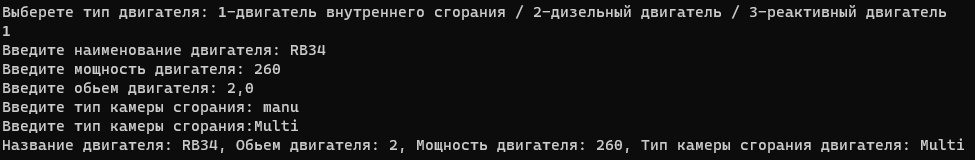


Рисунок 2.1 – Результат работы программы

Источник: собственная разработка

Задание 3. Описать класс для работы со строкой, позволяющей хранить только двоичное число и выполнять с ним арифметические операции. Предусмотреть инициализацию с проверкой допустимости значений. Написать программу, демонстрирующую все разработанные элементы класса.

Листинг программы:

using System.Collections.Generic;

namespace Task\_3

{

class Task\_3

{

public static void Main(string[] args)

{

Console.Write("Введите 1-е двоичное число: ");

string bynNumOne = Console.ReadLine();

Console.Write("Введите 2-е двоичное число: ");

string bynNumTwo = Console.ReadLine();

BynaryNumberSystem bns = new BynaryNumberSystem();

Console.WriteLine($"Cложение: {bns.Sum(bynNumOne, bynNumTwo)}");

Console.WriteLine($"Вычитание: {bns.Sub(bynNumOne, bynNumTwo)}");

Console.WriteLine($"Умножение: {bns.Mul(bynNumOne, bynNumTwo)}");

Console.WriteLine($"Деление: {bns.Div(bynNumOne, bynNumTwo)}");

}

}

class BynaryNumberSystem

{

public bool CheckBynary(string bynary)

{

for (int i = 0; i < bynary.Length; i++)

{

if (bynary[i] != '1' && bynary[i] != '0')

{

return false;

}

}

return true;

}

public string Sum(string bynOne, string bynTwo)

{

if (CheckBynary(bynOne) && CheckBynary(bynTwo))

{

return Convert.ToString(Convert.ToInt32(bynOne, 2) + Convert.ToInt32(bynTwo, 2), 2);

}

else

{

return "Значение введено неверно";

}

}

public string Sub(string bynOne, string bynTwo)

{

if (CheckBynary(bynOne) && CheckBynary(bynTwo))

{

return Convert.ToString(Convert.ToInt32(bynOne, 2) - Convert.ToInt32(bynTwo, 2), 2);

}

else

{

return "Значение введено неверно";

}

}

public string Mul(string bynOne, string bynTwo)

{

if (CheckBynary(bynOne) && CheckBynary(bynTwo))

{

return Convert.ToString(Convert.ToInt32(bynOne, 2) \* Convert.ToInt32(bynTwo, 2), 2);

}

else

{

return "Значение введено неверно";

}

}

public string Div(string bynOne, string bynTwo)

{

if (CheckBynary(bynOne) && CheckBynary(bynTwo))

{

return Convert.ToString(Convert.ToInt32(bynOne, 2) / Convert.ToInt32(bynTwo, 2), 2);

}

else

{

return "Значение введено неверно";

}

}

}

}

Таблица 3.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 111110 101010 | 1101000 10100 101000101100 1 |

Анализ результатов:

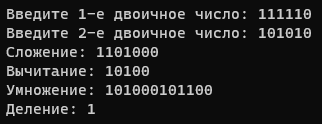


Рисунок 3.1 – Результат работы программы

Источник: собственная разработка