**20 ПАРАЛЛЕЛЬНОЕ ПРОГРАММИРОВАНИЕ**

Задание 1. Создайте и реализуйте метод решения задачи и выполните его в объектах класса Task используя три варианта создания объектов класса. Дано четырехзначное число. Найти число, полученное при прочтении его цифр справа налево.

Листинг программы:

namespace ReverseNumber

{

class Task

{

private int number;

public Task(int number)

{

this.number = number;

}

public int ReverseNumber()

{

int reversedNumber = 0;

int tempNumber = number;

while (tempNumber > 0)

{

int digit = tempNumber % 10;

reversedNumber = reversedNumber \* 10 + digit;

tempNumber /= 10;

}

return reversedNumber;

}

static void Main(string[] args)

{

Task task1 = new Task(1234);

Console.WriteLine("Reversed number (Variant 1): " + task1.ReverseNumber());

Task task2 = new Task(0);

task2.number = 5678;

Console.WriteLine("Reversed number (Variant 2): " + task2.ReverseNumber());

Task task3 = Task.CreateTask(9876);

Console.WriteLine("Reversed number (Variant 3): " + task3.ReverseNumber());

}

public static Task CreateTask(int number)

{

return new Task(number);

}

Таблица 20.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Вводные данные | Результат |

Анализ результатов:

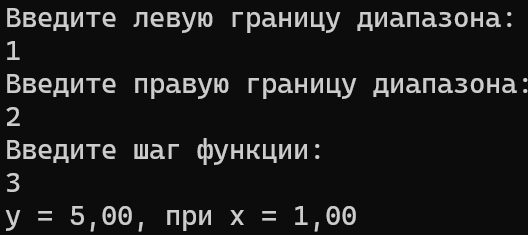
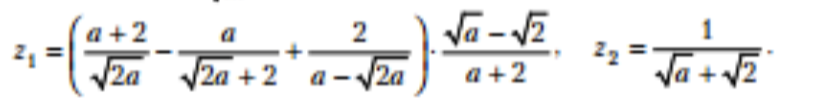


Рисунок 20.1 – Результат работы программы

Задание 2. Создайте массив из 2 задач (объектов класс Task) в каждом объекте выполните вычисление значения функций и выполните условия: 1. Дождитесь выполнения всех задач; 2. Дождитесь выполнения хот бы одной задачи Замедлить выполнение задачи можно с помощью Thread.Sleep(n) в методе, выполняемом задачей; где n – время в миллисекундах.



Листинг программы:

class TaskClass

{

private double a;

public double z1 { get; private set; }

public double z2 { get; private set; }

public TaskClass(double a)

{

this.a = a;

}

public void ComputeZ1()

{

Thread.Sleep(2000);

double numerator1 = (1 + a + Math.Pow(a, 2)) / (2 \* a + Math.Pow(a, 2)) + 2;

double numerator2 = (1 - a + Math.Pow(a, 2)) / (2 \* a - Math.Pow(a, 2));

double part1 = Math.Pow(numerator1 - numerator2, -1);

z1 = part1 \* (5 - 2 \* Math.Pow(a, 2));

}

public void ComputeZ2()

{

Thread.Sleep(3000);

z2 = (4 - Math.Pow(a, 2)) / 2;

}

public void Compute()

{

ComputeZ1();

ComputeZ2();

}

static async System.Threading.Tasks.Task Main(string[] args)

{

var tasks = new TaskClass[2];

tasks[0] = new TaskClass(1);

tasks[1] = new TaskClass(2);

var task1 = System.Threading.Tasks.Task.Run(() => tasks[0].Compute());

var task2 = System.Threading.Tasks.Task.Run(() => tasks[1].Compute());

await System.Threading.Tasks.Task.WhenAll(task1, task2);

Console.WriteLine("Results after waiting for all tasks:");

foreach (var task in tasks)

Таблица 20.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Массив данных | Результат |

Анализ результатов:

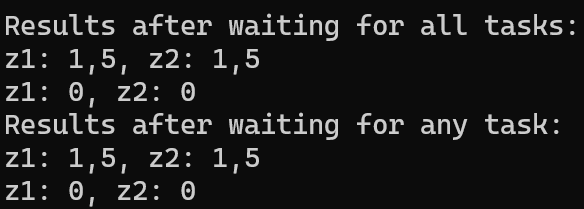


Рисунок 20.2 – Результат работы программы