namespace oop16

{

class Program

{

public static Task task1;

public static bool task2 = false;

public static CancellationTokenSource cancelTokenSource;

public static CancellationToken token;

static BlockingCollection<string> block;

static void Main(string[] args)

{

WriteLine("Задание 1.");

int r = 450, c = 450;

int[,] matr1 = new int[r, c];

int[,] matr2 = new int[r, c];

int[,] matr3 = new int[matr1.GetLength(0), matr2.GetLength(1)];

WriteLine("Замеры генерации двух массивов...");

Stopwatch st = new Stopwatch();

for (int i = 0; i < 5; i++)

{

st.Restart();

Task1(ref matr1, ref matr2, r, c);

st.Stop();

WriteLine($"Генерация {i + 1}\tВремя: {st.ElapsedMilliseconds}мс");

}

task1 = new Task(() => Umn(matr1, matr2, ref matr3));

task1.Start();

task1.Wait();

WriteLine();

WriteLine("Задание 2.");

cancelTokenSource = new CancellationTokenSource();

token = cancelTokenSource.Token;

task2 = true;

task1 = new Task(() => Umn(matr1, matr2, ref matr3));

task1.Start();

task1.Wait();4

if (token.IsCancellationRequested)

WriteLine("Task отменен");

task1.Dispose();

WriteLine();

WriteLine("Задание 3.");

var task11 = Task.Factory.StartNew<int>(() => Task11(15));

var task22 = Task.Factory.StartNew<int>(() => Task22(15));

task1 = new Task(() => Task4(task11.Result, task22.Result));

task1.Start();

task1.Wait();

task11.Dispose();

task22.Dispose();

WriteLine();

WriteLine("Задание 4.");

Task task4 = new Task(() =>

{

WriteLine($"Выполняется задача {Task.CurrentId}");

});

Task task44 = task4.ContinueWith(Task44);

task4.Start();

task44.Wait();

task4.Dispose();

task44.Dispose();

var task444 = Task.Factory.StartNew<int>(() => Task11(20));

task444.Wait();

var awaiter = task444.GetAwaiter();

if (awaiter.IsCompleted)

{

Task task4444 = new Task(() =>

{

WriteLine("GetAwaiter...");

});

task4444.Start();

task4444.Wait();

task4444.Dispose();

}

task444.Dispose();

WriteLine();

WriteLine("Задание 5.");

int[] arr1 = new int[1000000];

int[] arr2 = new int[1000000];

Random random = new Random();

st.Restart();

for (int i = 0; i < arr1.Length; i++)

{

arr1[i] = random.Next(0, 300);

arr2[i] = random.Next(0, 300);

}

st.Stop();

WriteLine($"Генарация двух массивов в for: {st.ElapsedMilliseconds}мс");

int[] arr3 = new int[1000000];

int[] arr4 = new int[1000000];

st.Restart();

Parallel.For(0, arr3.Length, i => { arr3[i] = random.Next(0, 300); arr4[i] = random.Next(0, 300); });

st.Stop();

WriteLine($"Генарация двух массивов в ParallelFor: {st.ElapsedMilliseconds}мс");

int[] arr5 = new int[1000000];

int[] arr6 = new int[1000000];

st.Restart();

Parallel.ForEach<int>(arr5, i => { arr5[i] = random.Next(0, 300); });

st.Stop();

long t1 = st.ElapsedMilliseconds;

st.Restart();

Parallel.ForEach<int>(arr6, i => { arr6[i] = random.Next(0, 300); });

st.Stop();

WriteLine($"Генерация двух массивов в ParallelForEach: {st.ElapsedMilliseconds + t1}мс");

WriteLine();

WriteLine("Задание 6.");

Parallel.Invoke(Task6, Task66);

WriteLine();

WriteLine("Задание 7.");

block = new BlockingCollection<string>(5);

Task Sup = new Task(Supl);

Task Con = new Task(Cons);

Sup.Start();

Con.Start();

Task.WaitAll(Sup, Con);

Sup.Dispose();

Con.Dispose();

WriteLine();

WriteLine("Задание 8.");

Async();

WriteLine("Введите что-нибудь во время выполнения асинхронного метода....");

var p = ReadLine();

}

static void Task1(ref int[,] matr1, ref int[,] matr2, int r, int c)

{

Random rand = new Random();

for (int i = 0; i < r; i++)

{

for (int j = 0; j < c; j++)

{

matr1[i, j] = rand.Next(0, 100);

matr2[i, j] = rand.Next(0, 100);

}

}

}

static void Umn(int[,] matr1, int[,] matr2, ref int[,] matr3)

{

WriteLine("Id задачи: {0}", task1.Id);

for (int i = 0; i < matr1.GetLength(0); i++)

{

for (int j = 0; j < matr2.GetLength(1); j++)

{

for (int q = 0; q < matr2.GetLength(0); q++)

matr3[i, j] += matr1[i, q] \* matr2[i, q];

}

if (i == matr1.GetLength(0) / 2)

WriteLine($"Статус задачи: {task1.Status}; статус завершения: {task1.IsCompleted}");

if (i == matr1.GetLength(0) / 5 && task2)

{

cancelTokenSource.Cancel();

if (token.IsCancellationRequested)

{

return;

}

}

}

}

public static int Task11(int i)

{

i += DateTime.Now.Day;

return i;

}

public static int Task22(int i)

{

i -= DateTime.Now.Day;

return i;

}

public static void Task4(int a, int b)

{

int r = a + b;

WriteLine($"Результат = {r}");

}

static void Task44(Task t)

{

WriteLine($"Продолжение выполнения задачи {t.Id}");

Thread.Sleep(2000);

}

static void Task6()

{

WriteLine("Метод Task6 запущен");

for (int i = 0; i < 6; i++)

{

Thread.Sleep(500);

WriteLine("Task6 = " + i);

}

WriteLine("Task6 завершен");

}

static void Task66()

{

WriteLine("Метод Task66 запущен");

for (int i = 0; i < 6; i++)

{

Thread.Sleep(600);

WriteLine("Task66 = " + i);

}

WriteLine("Task66 завершен");

}

static void Supl()

{

Random r = new Random();

int x;

List<string> products = new List<string>() { "Роза", "Ромашка", "Фиалка", "Маргаритки", "Фикус" };

for (int i = 0; i < 5; i++)

{

x = r.Next(0, products.Count - 1);

WriteLine("Добавлен цветок: {0}", products[x]);

block.Add(products[x]);

products.RemoveAt(x);

Thread.Sleep(r.Next(1, 3));

}

block.CompleteAdding();

}

static void Cons()

{

string str;

while (!block.IsAddingCompleted)

{

if (block.TryTake(out str))

WriteLine("Был полит цветок: {0}", str);

else

WriteLine("Полив завершен");

}

}

static void Meth()

{

for (int i = 0; i < 100; i++)

{

if (i % 3 == 0)

{

Write(i + ", ");

Thread.Sleep(1000); } }}

static async void Async()

{

WriteLine("Начало асинхронного метода");

await Task.Run(() => Meth());

WriteLine("Конец асинхронного метода");}}}