МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

***Циклова комісія програмування***

**Робочий план**

з курсу «Об’єктно-орієнтоване програмування»

Практична робота №11

Солом’яного Ярослава Сергійовича

*ПІБ студента*

студента групи **2П-18**

|  |  |  |
| --- | --- | --- |
| **Вид роботи** | **Дата** | **Оцінка** |
| *Пром. контроль* |  |  |
| *Залік* |  |  |

Викладач Марченко С. В.

Черкаси-2021

Завдання 1. Запишіть перелік з 20 URL-посилань та використовуючи засоби для пінгу з простору імен System.Net.NetworkInformation перевірте в паралельному режимі їх IP-статус (онлайн/оффлайн). Запустіть код кілька разів та переконайтесь у невпорядкованості отриманих результатів.

Лістинг 1. Код для виконання завдання 1.

using System;

using System.Globalization;

using System.Text;

namespace ex\_1\_\_Practice\_11

{

class Program

{

static void Main(string[] args)

{

String[] urls = {

"www.vk.com",

"www.facebook.com",

"www.cyberforum.ru",

"www.sinoptik.ua",

"www.repl.it",

"www.allo.ua",

"www.mail.ru",

"www.yandex.ru",

"www.google.com",

"www.flyuia.com",

"www.emailondeck.com",

"www.csbc.edu.ua",

"www.instagram.com"

};

System.Threading.Tasks.Parallel.ForEach(urls, url =>

{

var ping = new System.Net.NetworkInformation.Ping();

var result = ping.Send(url);

if (result.Status == System.Net.NetworkInformation.IPStatus.Success)

{

Console.WriteLine(string.Format("{0} is online", url));

}

else

{

Console.WriteLine(string.Format("{0} is offline",url));

}

}

);

Console.OutputEncoding = System.Text.Encoding.UTF8;

PrintTimeStamp("Солом'яний Ярослав Сергійович, 2П-18");

}

public static void PrintTimeStamp(string name)

{

Console.OutputEncoding = Encoding.UTF8;

DateTime localDate = DateTime.Now;

var culture = new CultureInfo("ru-RU");

Console.WriteLine("Дата та час компіляції: {0}", localDate.ToString(culture));

Console.WriteLine("Автор: {0}", name);

}

}

}

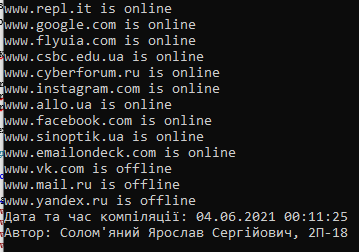


Рис 1. Результат виконання завдання 1

Завдання 2. Розгляньте та реалізуйте програмне забезпечення, представлене в статті. Детально опишіть початковий алгоритм роботи та використаний спосіб прискорення обчислень.

Лістинг 2. Код для виконання завдання №2.

**public** **struct** State

{

    public double a, h;

    public int steps;

    public Func<double, double> f;

    public List<double> sum;

    public State(double a, double h, int steps, Func<double, double> f)

    {

        this.a = a;

        this.h = h;

        this.steps = steps;

        this.f = f;

        sum = new List<double>();

    }

}

public void SumTask(object o)

{

    State s = (State)o;

    double sum = 0;

    for (int i = 1; i <= s.steps; i++)

    {

        sum += s.f(s.a + i \* s.h);

    }

    s.sum.Add(sum);

}

public double TasTrapezoidalRule(

  double a, double b, int cores, int n,

  Func<double, double> f)

{

    double h = (b - a) / (n + 1);

    double sum = 0.5 \* (f(a) + f(b));

    int steps = n / cores;

    State[] states = new State[cores];

    Task[] tasks = new Task[cores];

    for (int i = 0; i < cores; i++)

    {

        states[i] = new State(a + i \* h \* steps, h, steps, f);

        tasks[i] = new Task(SumTask, states[i]);

        tasks[i].Start();

    }

    Task.WaitAll(tasks);

    for (int i = 0; i < cores; i++)

    {

        sum += states[i].sum[0];

    }

    return h \* sum;

}

public double TasTrapezoidalRuleWithCorrection(

  double a, double b, int cores, int n,

  Func<double, double> f, Func<double, double> fp)

{

    double h = (b - a) / (n + 1);

    double sum = 0.5 \* (f(a) + f(b));

    int steps = n / cores;

    State[] states = new State[cores];

    Task[] tasks = new Task[cores];

    for (int i = 0; i < cores; i++)

    {

        states[i] = new State(a + i \* h \* steps, h, steps, f);

        tasks[i] = new Task(SumTask, states[i]);

        tasks[i].Start();

    }

    Task.WaitAll(tasks);

    for (int i = 0; i < cores; i++)

    {

        sum += states[i].sum[0];

    }

    return h \* (sum + h \* (fp(a) - fp(b)) / 12.0);

}

Пришвидчене обчислення реалізується за допомогою багатопоточності

Завдання 4. Протестуйте продуктивність виконання LINQ та PLINQ запитів на прикладі знаходження простих чисел. Реалізуйте процес пошуку простих чисел за допомогою LINQ-запиту для діапазонів чисел від 2 до 1000, до 10 000, до 100 000 та до 1 000 000. Доповніть запит інструментами для забезпечення паралелізму різного ступеня (AsParallel, WithDegreeOfParallelism()). Відобразіть у звіті код для LINQ- та PLINQ-запитів, детальну характеристику процесора і таблицю з відповідним часом виконання:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ступінь паралелізму | n | 1000 | 10000 | 100000 | 1000000 |
| N = 1 | | 00:00:00.07 | 00:00:00.07 | 00:00:00.13 | 00:00:00.97 |
| N = 2 | | 00:00:00.10 | 00:00:00.08 | 00:00:00.12 | 00:00:00.60 |
| N = 3 | | 00:00:00.14 | 00:00:00.09 | 00:00:00.12 | 00:00:00.48 |
| N = 4 | | 00:00:00.09 | 00:00:00.10 | 00:00:00.11 | 00:00:00.48 |
| N = 6 | | 00:00:00.09 | 00:00:00.09 | 00:00:00.12 | 00:00:00.40 |
| N = 8 | | 00:00:00.12 | 00:00:00.11 | 00:00:00.12 | 00:00:00.41 |
| N = 12 | | 00:00:00.10 | 00:00:00.10 | 00:00:00.12 | 00:00:00.40 |
| N =16 | | 00:00:00.11 | 00:00:00.10 | 00:00:00.12 | 00:00:00.38 |
| N = 32 | | 00:00:00.12 | 00:00:00.14 | 00:00:00.13 | 00:00:00.37 |

Лістинг 3. Код для виконання завдання

using System;

using System.Diagnostics;

using System.Globalization;

using System.Linq;

using System.Text;

namespace Ex\_4\_Practice\_11

{

class Program

{

static void Main(string[] args)

{

DateTime StartTime = DateTime.Now;

var ParallelQuery = from n in Enumerable.Range(2, 1000000)

.AsParallel().WithDegreeOfParallelism(32)

where Enumerable.Range(0, (int)Math.Sqrt(n) / 2 + 1).Select(e => e == 0 ? 2 : e + 1).All(i => n % i > 0)

select n;

int[] primes = ParallelQuery.ToArray();

TimeSpan ts = DateTime.Now.Subtract(StartTime);

string elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",

ts.Hours, ts.Minutes, ts.Seconds,

ts.Milliseconds / 10);

Console.WriteLine(elapsedTime, "RunTime");

PrintTimeStamp("Солом'яний Ярослав, студент групи 2П-18");

}

public static void PrintTimeStamp(string name)

{

Console.OutputEncoding = Encoding.UTF8;

DateTime localDate = DateTime.Now;

var culture = new CultureInfo("ru-RU");

Console.WriteLine("Дата та час компіляції: {0}", localDate.ToString(culture));

Console.WriteLine("Автор: {0}", name);

}

}

}

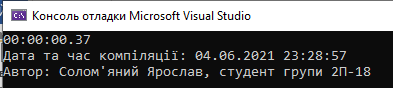


Рис 2.Результат виконання завдання 4.

Завдання 5. Розгляньте цикл статей Parallel LINQ in Depth. Детально опишіть та запустіть усі запропоновані підзадачі.

Нічого не зрозуміло, але зрозумів частину, що вказує як саме описуються методи які вживаються в

Parallel LINQ за допомогою сайту: <https://www.csharpcodi.com/vs2/?source=1376%2FCodeSnippets%2FTutorial.Shared%2FParallelLinq%2FParallelEnumerableX.QuerySettings.cs#>

Код є аналогічний представленому, тому немає сенсу копіпастить..