НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет прикладної математики
Кафедра прикладної математики

Звіт
із лабораторної роботи №5
із дисципліни «Розподілені і хмарні обчислення»

Виконав:
студент групи КМ-03
Шаповалов Г. Г.

Керівник:
Лiскін В. О.

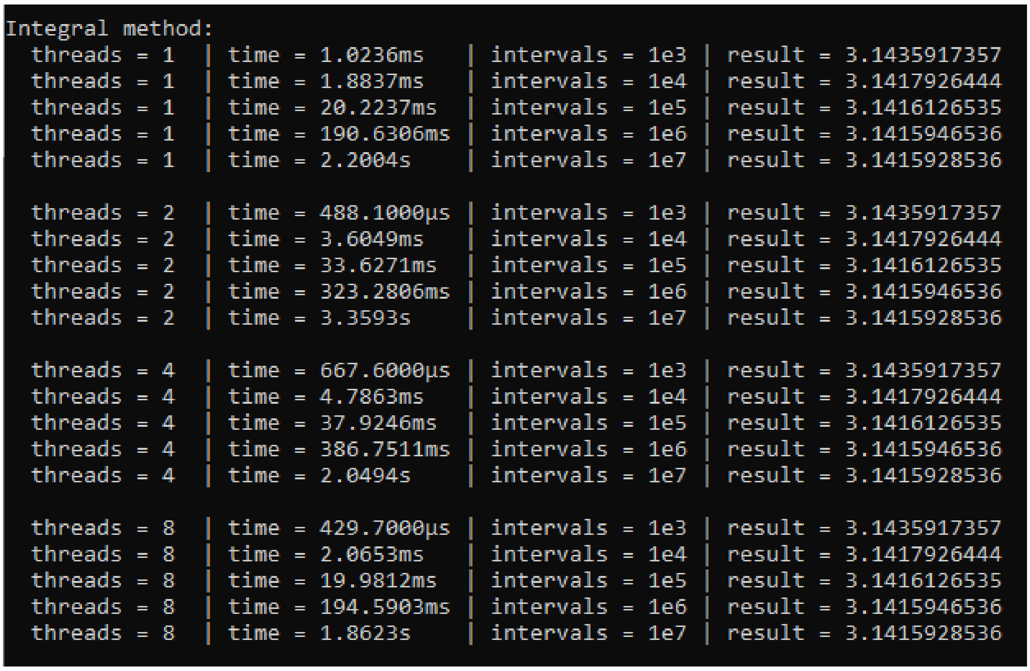
Київ — 2023

Мета роботи

Розпаралелити метод обчислення константи PI

Опис програми

Завдання 1 – Знайти значення PI інтегральним методом:



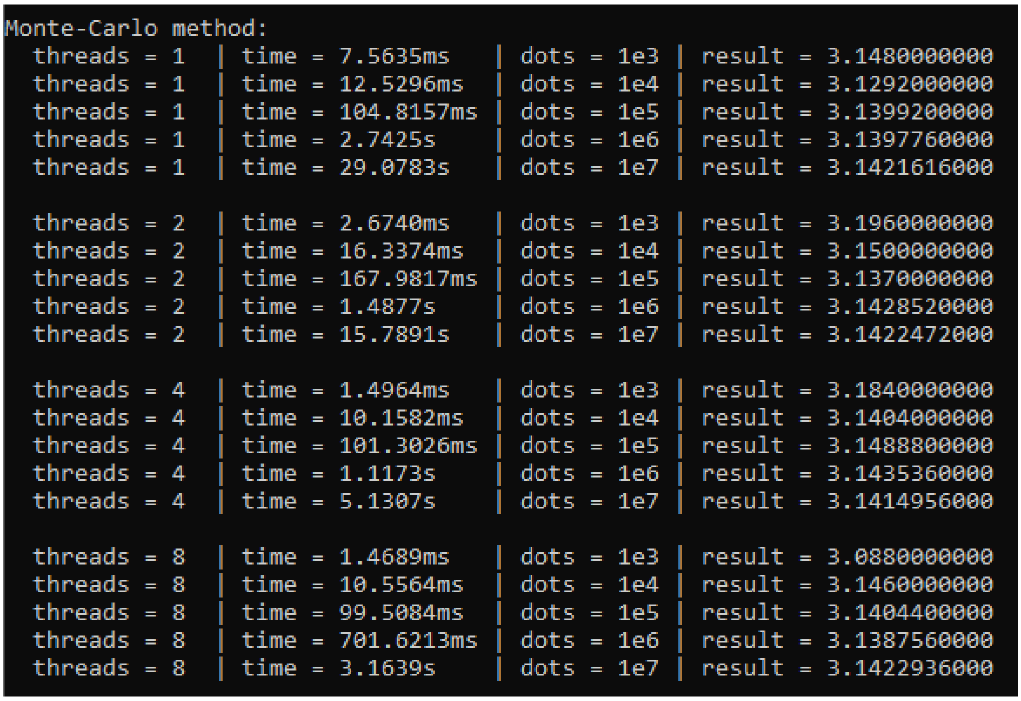
```
Integral method:
  threads = 1  | time = 1.0236ms    | intervals = 1e3 | result = 3.1435917357
  threads = 1  | time = 1.8837ms    | intervals = 1e4 | result = 3.1417926444
  threads = 1  | time = 20.2237ms   | intervals = 1e5 | result = 3.1416126535
  threads = 1  | time = 190.6306ms  | intervals = 1e6 | result = 3.1415946536
  threads = 1  | time = 2.2004s     | intervals = 1e7 | result = 3.1415928536

  threads = 2  | time = 488.1000µs  | intervals = 1e3 | result = 3.1435917357
  threads = 2  | time = 3.6049ms    | intervals = 1e4 | result = 3.1417926444
  threads = 2  | time = 33.6271ms   | intervals = 1e5 | result = 3.1416126535
  threads = 2  | time = 323.2806ms  | intervals = 1e6 | result = 3.1415946536
  threads = 2  | time = 3.3593s     | intervals = 1e7 | result = 3.1415928536

  threads = 4  | time = 667.6000µs  | intervals = 1e3 | result = 3.1435917357
  threads = 4  | time = 4.7863ms    | intervals = 1e4 | result = 3.1417926444
  threads = 4  | time = 37.9246ms   | intervals = 1e5 | result = 3.1416126535
  threads = 4  | time = 386.7511ms  | intervals = 1e6 | result = 3.1415946536
  threads = 4  | time = 2.0494s     | intervals = 1e7 | result = 3.1415928536

  threads = 8  | time = 429.7000µs  | intervals = 1e3 | result = 3.1435917357
  threads = 8  | time = 2.0653ms    | intervals = 1e4 | result = 3.1417926444
  threads = 8  | time = 19.9812ms   | intervals = 1e5 | result = 3.1416126535
  threads = 8  | time = 194.5903ms  | intervals = 1e6 | result = 3.1415946536
  threads = 8  | time = 1.8623s     | intervals = 1e7 | result = 3.1415928536
```

Завдання 2 - Знайти значення PI методом Монте-Карло:



```
Monte-Carlo method:
  threads = 1  | time = 7.5635ms    | dots = 1e3 | result = 3.1480000000
  threads = 1  | time = 12.5296ms   | dots = 1e4 | result = 3.1292000000
  threads = 1  | time = 104.8157ms  | dots = 1e5 | result = 3.1399200000
  threads = 1  | time = 2.7425s     | dots = 1e6 | result = 3.1397760000
  threads = 1  | time = 29.0783s    | dots = 1e7 | result = 3.1421616000

  threads = 2  | time = 2.6740ms    | dots = 1e3 | result = 3.1960000000
  threads = 2  | time = 16.3374ms   | dots = 1e4 | result = 3.1500000000
  threads = 2  | time = 167.9817ms  | dots = 1e5 | result = 3.1370000000
  threads = 2  | time = 1.4877s     | dots = 1e6 | result = 3.1428520000
  threads = 2  | time = 15.7891s    | dots = 1e7 | result = 3.1422472000

  threads = 4  | time = 1.4964ms    | dots = 1e3 | result = 3.1840000000
  threads = 4  | time = 10.1582ms   | dots = 1e4 | result = 3.1404000000
  threads = 4  | time = 101.3026ms  | dots = 1e5 | result = 3.1488800000
  threads = 4  | time = 1.1173s     | dots = 1e6 | result = 3.1435360000
  threads = 4  | time = 5.1307s     | dots = 1e7 | result = 3.1414956000

  threads = 8  | time = 1.4689ms    | dots = 1e3 | result = 3.0880000000
  threads = 8  | time = 10.5564ms   | dots = 1e4 | result = 3.1460000000
  threads = 8  | time = 99.5084ms   | dots = 1e5 | result = 3.1404400000
  threads = 8  | time = 701.6213ms  | dots = 1e6 | result = 3.1387560000
  threads = 8  | time = 3.1639s     | dots = 1e7 | result = 3.1422936000
```

Висновки: Інтегральний метод виявився точнішим за Монте-Карло

Лістинг програми:

```
use std::time::Instant;
use lab_5::constcalc::{picalc, mc_picalc};
use rayon::ThreadPoolBuilder;

fn main() {
    task_intergral();
    task_monte_carlo();
}

fn format_dots(n: i32) -> String {
    let mut dots = n.to_string();
    let len = dots.len();
    if len > 3 {
        let e = len - 1;
        dots = format!("1e{}", e);
    }
    dots
}

fn task_intergral() {
    println!("\nIntegral method:");
    for &threads in [1, 2, 4, 8].iter() {
        let pool =
ThreadPoolBuilder::new().num_threads(threads).build().unwrap();
        for &n in [1e3 as i32, 1e4 as i32, 1e5 as i32, 1e6 as i32, 1e7 as
i32].iter() {
            let start = Instant::now();
            pool.install(|| {
                let pi = picalc(n);
                let duration = start.elapsed();
                println!("  threads = {:<2} | time = {:<10} | intervals = {:<2} |
result = {:.10}",
                        threads, format!("{:.4?}", duration), format_dots(n),
pi);
            });
        }
        println!();
    }
}

fn task_monte_carlo() {
    println!("\n\n\nMonte-Carlo method:");
    for &threads in [1, 2, 4, 8].iter() {
        let pool =
ThreadPoolBuilder::new().num_threads(threads).build().unwrap();
        for &n in [1e3 as i32, 1e4 as i32, 1e5 as i32, 1e6 as i32, 1e7 as
i32].iter() {
            let start = Instant::now();
            pool.install(|| {
                let pi = mc_picalc(n);
                let duration = start.elapsed();
                println!("  threads = {:<2} | time = {:<10} | dots = {:<2} |
result = {:.10}",
                        threads, format!("{:.4?}", duration), format_dots(n),
pi);
            });
        }
        println!();
    }
}
```