|  |  |  |  |
| --- | --- | --- | --- |
| **МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**  **КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка**  **ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**  **Кафедра програмних систем і технологій**  **Дисципліна**  **«Кросплатформне програмування»**  **Лабораторна робота 1** | | | |
| **Виконав:** | Чушенко Ярослав Володимирович | **Перевірив**: | Петрівський В.Я |
| Група | ІПЗ-41 | Дата перевірки |  |
| Форма навчання | денна | Оцінка |  |
| Спеціальність | 121 |
| 2023 | | | |

**Варіант – 54**Репозиторій на Github: [Посилання](https://github.com/PingvinAustr/CrossplatformKNU)  
 **Умова завдання:**Описано рекурсивну функцію з трьома параметрами F(a, b, c):

якщо a ≤ 0 або b ≤ 0 або c ≤ 0, то F(a, b, c) = 1

якщо a > 20 чи b > 20 чи c > 20, то F(a, b, c) = F(20, 20, 20)

якщо a < b та b < c, то F(a, b, c) = F(a, b, c-1) + F(a, b-1, c-1) - F(a, b-1) , c)

інакше F(a, b, c) = F(a-1, b, c) + F(a-1, b-1, c) + F(a-1, b, c-1) - F(a -1, b-1, c-1)

Однак, якщо вказану функцію реалізувати безпосередньо, то навіть для невеликих значень a, b та c (наприклад, a = 15, b = 15, c = 15), програма працюватиме кілька годин! Необхідно реалізувати ефективний алгоритм обчислення функції F, який встигне знайти будь-яке значення менш ніж за одну секунду!

*Вхідні дані:*

Вхідний файл INPUT.TXT містить три цілі числа a, b, c - параметри функції F (-104 ≤ a, b, c ≤ 104).

*Вихідні дані:*

У вихідний файл OUTPUT.TXT виведіть значення функції F(a, b, c).

*Приклади:*

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 1 1 | 2 |
| 2 | 2 2 2 | 4 |
| 3 | 10 4 6 | 523 |
| 4 | 50 50 50 | 1048576 |

**Виконання роботи:**

1. **Код:**
2. using System.Diagnostics;
3. class Program
4. {
5. static Dictionary<(int, int, int), int> memo = new Dictionary<(int, int, int), int>();
6. static int F(int a, int b, int c)
7. {
8. if (memo.TryGetValue((a, b, c), out int value))
9. return value;
10. if (a <= 0 || b <= 0 || c <= 0)
11. return 1;
12. if (a > 20 || b > 20 || c > 20)
13. return F(20, 20, 20);
14. if (a < b && b < c)
15. value = F(a, b, c - 1) + F(a, b - 1, c - 1) - F(a, b - 1, c);
16. else
17. value = F(a - 1, b, c) + F(a - 1, b - 1, c) + F(a - 1, b, c - 1) - F(a - 1, b - 1, c - 1);
18. memo[(a, b, c)] = value;
19. return value;
20. }
21. static void Main()
22. {
23. var input = File.ReadAllText("..//..//..//INPUT.TXT").Split();
24. int a = int.Parse(input[0]);
25. int b = int.Parse(input[1]);
26. int c = int.Parse(input[2]);
27. Console.WriteLine($"Received following values from input.txt: a={a}, b={b}, c={c}\n\nCalculating...");
28. Stopwatch stopwatch = new Stopwatch();
29. stopwatch.Start();
30. int result = F(a, b, c);
31. stopwatch.Stop();
32. File.WriteAllText("..//..//..//OUTPUT.TXT", result.ToString());
33. Console.WriteLine($"\nResult:{result}");
34. Console.WriteLine($"Execution Time: {stopwatch.Elapsed.TotalMilliseconds} ms");
35. }
36. }

**Опис алгоритму виконання:**

*Базові випадки:*

Якщо будь-яке з a, b або c менше або дорівнює 0, функція повертає 1.

Якщо будь-яке з a, b або c більше 20, функція повертає результат F(20, 20, 20).

*Рекурсивні випадки:*

Якщо a < b і b < c, функція повертає F(a, b, c-1) + F(a, b-1, c-1) - F(a, b-1, c).

В іншому випадку функція повертає F(a-1, b, c) + F(a-1, b-1, c) + F(a-1, b, c-1) - F(a-1, b-1, c-1).

В якості методу оптимізації була використана мемоїзація:

Мемоїзація - це метод оптимізації, який використовується в основному для прискорення роботи комп'ютерних програм шляхом збереження результатів «дорогих» викликів функцій та їх повторного використання при повторному обчисленні тих самих вхідних даних.

Для зберігання раніше вже обчислених результатів функції F(a, b, c) використовується словник «memo».

Ключами словника є кортежі (a, b, c), що представляють вхідні параметри функції.

Значеннями словника є відповідні результати роботи функції.

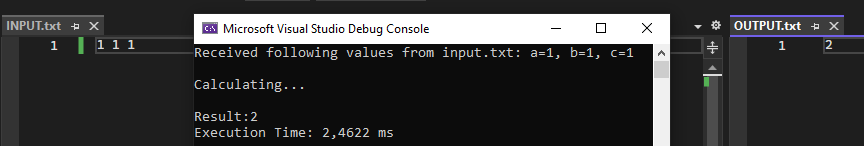
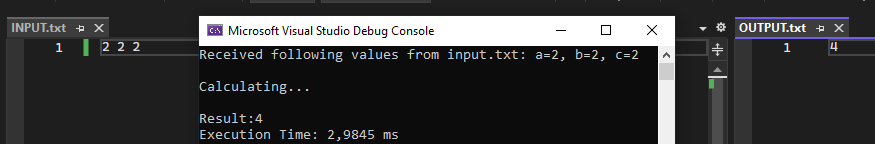
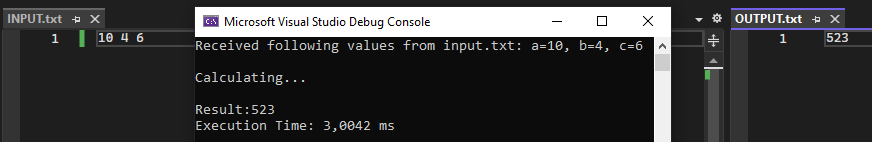
Перед виконанням будь-яких обчислень алгоритм перевіряє, чи результат для поточних вхідних параметрів (a, b, c) вже є у словнику “memo”.

Якщо результат є, то він повертається одразу без подальших обчислень.

Якщо результат відсутній, алгоритм обчислює його за рекурсивними правилами, зберігає у словнику нотаток і повертає його.

Тобто, завдяки використанню мемоїзації алгоритм уникає повторного обчислення функції F(a, b, c) для тих самих вхідних параметрів, тим самим значно зменшуючи кількість рекурсивних викликів і підвищуючи загальну ефективність алгоритму. Ця оптимізація гарантує, що алгоритм може обчислити будь-яке значення функції менш ніж за одну секунду, досягаючи бажаної продуктивності.

**Результати виконання програми:**

Для більшої наочності в програму було додано обчислення часу, що витрачається на виконання обрахунку.  
  
  
1. Input = 1, 1, 1.   
Output = 2  
Час виконання = 2,46 мс  
  
  
2. Input = 2, 2, 2  
Output = 4  
Час виконання = 2,98 мс  
  
  
  
3. Input = 10, 4, 6  
Output = 523  
Час виконання = 3 мс  
  
  
4. Input = 50, 50, 50  
Output = 1048576  
Час виконання = 3,6 мс  
