УДК 371.38 : 004.047 **DOI** https://doi.org/10.32782/3041-2021/2024-2-9

ЗМЕНШЕННЯ ОБЧИСЛЮВАЛЬНОЇ СКЛАДНОСТІ АЛГОРИТМУ ВИЗНАЧЕННЯ ДОСКОНАЛИХ ЧИСЕЛ ТА ВИВЧЕННЯ ЦЬОГО ПОНЯТТЯ У СЕРЕДНІЙ ШКОЛІ

Шпортько О. В.
кандидат технічних наук, доцент,
доцент кафедри інформаційних систем та обчислювальних методів
Приватного вищого навчального закладу «Міжнародний економіко-гуманітарний університет імені академіка Степана Дем'янчука»
ORCID ID: *0000-0002-4013-3057*

Шпортько Л. В.
викладач вищої категорії циклової комісії інформатики і комп'ютерної техніки
ДВНЗ «Рівненський коледж економіки та бізнесу»
ORCID ID: *0000-0003-4953-783X*

Гаврилюк В. І.
кандидат технічних наук,
доцент кафедри комп'ютерних наук та прикладної математики
Національного університету водного господарства та природокористування
ORCID ID: *0000-0003-3377-6465*

Іскра А. В.
здобувач третього (освітньо-наукового) рівня вищої освіти
Приватного вищого навчального закладу «Міжнародний економіко-гуманітарний університет імені академіка Степана Дем'янчука»

Панчук О. Я.
здобувач третього (освітньо-наукового) рівня вищої освіти
Приватного вищого навчального закладу «Міжнародний економіко-гуманітарний університет імені академіка Степана Дем'янчука»

В статті обґрунтована доцільність вивчення поняття асимптотичної обчислювальної складності на уроках інформатики в середній школі, починаючи з теми «Вкладені цикли». Доведено парність досконалих чисел та показано, що квадрати натуральних чисел не можуть бути досконалими числами. Встановлено, що для визначення досконалості числа N його нетривіальні дільники варто шукати на інтервалі від двох до квадратного кореня з N-1 та додавати до суми не лише знайдений дільник, а й отриману частку. Це дало змогу прискорити обчислення та не перевіряти відмінності частки від дільника. Наведені в статті вдосконалення алгоритму пошуку досконалих чисел на заданому інтервалі дають змогу зменшити його асимптотичну обчислювальну складність з $O(N^2)$ до $O(N\sqrt{N})$. В процесі виведення теорем про парність та властивості досконалих чисел використано дедуктивний метод. Оцінку складності алгоритмів визначення досконалих чисел на заданому інтервалі проведено методами теорії алгоритмів.

В роботі описано реалізації розроблених алгоритмів мовою програмування C#. Правильність теоретичних викладок та коректність наведених фрагментів програм підтверджена експериментально за допомогою віддаленого незалежного обчислювального середовища на сайті https://www.e-olymp.com. В дослідженні показано, що кардинальне зменшення обчислювальної складності алгоритму можливо насамперед за рахунок зменшення асимптотичної обчислювальної складності. Поняття асимптотичної обчислювальної складності в контексті часу виконання доцільно формувати в учнів на уроках інформатики не тільки спеціалізованих, а й загальноосвітніх шкіл, починаючи з 7 класу, в процесі вивчення вкладених циклів, наприклад, за допомогою задачі визначення досконалих чисел на заданому інтервалі. Це формуватиме в учнів навики написання не лише правильних, а й ефективних програм.

Ключові слова: *досконалі числа, асимптотична обчислювальна складність, оптимізація алгоритмів.*

Shportko O. V., Shportko L. V., Gavrilyuk V. I., Iskra A. V., Panchuk O. Ya. Reducing the Computational Complexity of the Algorithm for Determining Perfect Numbers and Studying this Concept in Secondary School

Introduction. *Today, the concept of computational complexity of the algorithm is considered in the course of high school computer science in Ukraine only on an intuitive level, although this concept teaches future programmers to pay attention not only to the correctness of the problem, but also its effectiveness.*

Purpose. *Demonstration of the possibility of studying the concept of computational complexity of the algorithm in high school on the example of the problem of determining perfect numbers in a given interval. Proving the parity of perfect numbers and determining the minimum non-trivial divisors for finding them.*

Methods. *The deductive method is used in the process of deriving the theorems on parity and properties of perfect number. Estimation of complexity of algorithms of definition of perfect numbers on the set interval is carried out by methods of the theory of algorithms.*

Results. *The publication proves the theorem of the parity of perfect numbers. It is shown that the squares of natural numbers cannot be perfect numbers. It is established that to determine the perfection of the number N its non-trivial divisors should be found in the interval from two to the square root of N-1 and add to the sum not only the found divisor but also the obtained fraction. The improvements in the algorithm for finding perfect numbers in a given interval presented in the article make it possible to reduce its asymptotic computational complexity from $O(N^2)$ to $O(N\sqrt{N})$. The correctness of the theoretical calculations and the correctness of the fragments of programs was confirmed experimentally using the remote independent computing environment at https://www.e-olymp.com.*

Originality. *The theorems of the parity of perfect numbers and on the imperfection of squares of natural numbers are proved for the first time. To determine the perfection of the number N, the search range of non-trivial divisors was reduced to the square root of N-1, which allowed to speed up the calculations and not to check the differences of the fraction from the divisor.*

Conclusion. *A drastic reduction in the computational complexity of the algorithm is possible primarily by reducing the asymptotic computational complexity. The concept of asymptotic computational complexity in the context of execution time should be formed in students during computer science lessons in all secondary schools, starting from 7th form, in the process of studying nested cycles, for example, by determining perfect numbers at a given interval. This will develop students' skills in writing not only correct but also effective programs.*

Key words: *perfect numbers, asymptotic computational complexity, algorithm optimization.*

На сьогодні поняття *обчислювальної складності алгоритму* [1] розглядається в курсі інформатики середньої школи лише на інтуїтивному рівні [2; 3]. І даремно, адже саме показники обчислювальної складності «дають уявлення про час виконання алгоритмів» [4, с. 57], дозволяють зрозуміти, чому один алгоритм буде працювати значно швидше від іншого, привчають майбутніх програмістів звертати увагу не лише на правильність розв'язку поставленої задачі, а й на його ефективність, не лише на час виконання програми, а й на обсяги використовуваної пам'яті. На нашу думку, поняття *асимптотичної складності алгоритму* [5] доцільно вводити не лише для порівняння ефективності різних алгоритмів сортування у 9 класі навчальних закладів з поглибленим вивченням інформатики [6, с. 165-172], а й навіть при вивченні вкладених циклів у 7 класі загальноосвітніх шкіл [2]. Покажемо, наприклад, як можна використати асимптотичну складність для аналізу ефективності алгоритмів визначення досконалих чисел на заданому інтервалі.

Як відомо, натуральне число називається *досконалим*, якщо воно рівне сумі своїх дільників за винятком самого себе. Наприклад, число 6 – досконале, оскільки 6=1+2+3, а число 8 – не досконале, адже 8≠1+2+4 [7, с. 136]. Два перших досконалих числа (6 і 28) були відомі задовго до Евкліда, але саме він не лише знайшов два наступні досконалі числа (496 і 8128), а й довів досконаліcть чисел, які можна подати у вигляді

$$N = 2^{p-1}(2^p-1), \qquad (1)$$

де 2^p-1 – просте число [8]. На сьогодні відомо понад 50 досконалих чисел і їх подальші пошуки тривають за допомогою проекту розподілених обчислень GIMPS [9].

Метою цієї статті є доведення парності всіх досконалих чисел та проведення аналізу ефективності послідовності вдосконалень алгоритму визначення таких чисел на заданому інтервалі з обчисленням їх асимптотичної обчислюваль-

ної складності та демонстрацією результатів виконання їх реалізацій у незалежному обчислювальному середовищі. Цю послідовність алгоритмів можна використати при вивченні вкладених циклів у 7 класі для формування в учнів поняття обчислювальної складності.

Задачу, яка розв'язується у цій статті, коротко сформулюємо так, як вона подана на сайті https://www.e-olymp.com під назвою «Досконалі числа» (№ 848): «Число називається досконалим, якщо воно дорівнює сумі всіх своїх дільників, менших за нього. Потрібно знайти всі досконалі числа від *m* до *n*». Ліміт часу для виконання програмою кожного тесту – 5 с, максимально можливий обсяг використання пам'яті – 64 Мб.

Зрозуміло, що для визначення досконалості кожного натурального числа *num* (скорочення від *number*) з введеного інтервалу [*m*; *n*] можна скористатися очевидним алгоритмом [10], який для чергового *num* послідовно перебирає натуральні числа з інтервалу [1; *num*-1], визначає серед них дільники числа *num* та обчислює їх суму. Якщо знайдена сума дорівнює числу *num*, то це число досконале. Наведемо програму для розв'язання поставленої задачі мовою програмування C#, оскільки на сьогодні вона є однією з основних мов програмування прикладних додатків:

```
1 int[] borders = Console.ReadLine().Trim().
      Split(' ').Select(int.Parse).ToArray();
2 int m = borders[0], n = borders[1], sumaD,
num, div;
3 bool available = false;//досконалі числа
поки не знайдені
4 for (num = m; num <= n; num++)//цикл по
числах діапазону
5  {sumaD = 0;//початкове значення суми
дільників
6  for (div = 1; div <= num-1; div++)
7   if (num % div == 0)//знайшли черговий
дільник
8   sumaD += div;
9  if (num == sumaD)//знайшли досконале
число
10  {Console.WriteLine(num.ToString());
11   available = true; }}
12 if (!available)
13 Console.WriteLine(«Absent»);
```

Результати тестування цієї та наступних програм подамо з незалежного обчислювального середовища на сайті https://www.e-olymp.com (рис. 1). На цьому сайті до кожної задачі програміст може завантажити розв'язки у вигляді текстів різноманітних програм, написаних навіть на різних мовах програмування. Після цього сервер сайту компілює отримані програми і подає їм на вхід виконання різні тести, невідомі програмістам та звіряє отримані результати на виході з очікуваними. Задача вважається розв'язаною лише тоді, коли відповідна програма пройшла всі тести за відведені проміжки часу та не перевищила допустимі обсяги використання оперативної пам'яті. Переваги від використання такого віддаленого незалежного обчислювального середовища очевидні:

– учень-програміст може писати тексти програм для розв'язування однієї задачі на різних мовах програмування і порівнювати їх ефективність;

– учень може розв'язувати задачу різними методами та порівнювати їх ефективність;

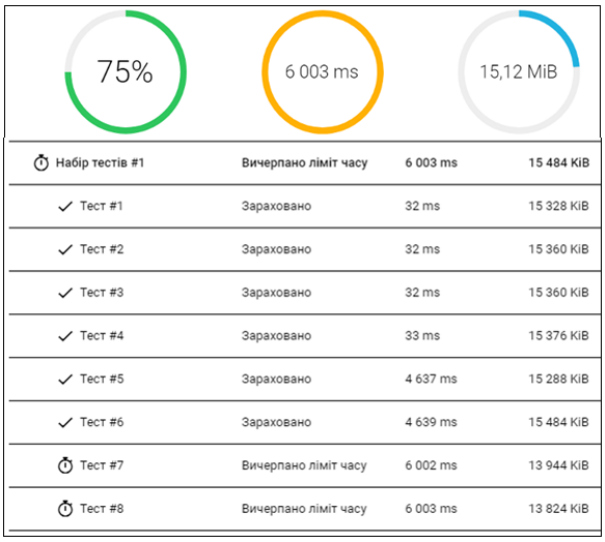
– учні можуть змагатися між собою, намагаючись написати найефективнішу програму як в класі, так і дистанційно;

– учні не можуть адаптувати програми під конкретні вхідні дані, оскільки завдання тестів їм невідомі, вони починають усвідомлювати значення ефективності програмного коду та критерії його оцінювання.

Результати тестування наведеної вище програми на згаданому сайті https://www.e-olymp.com наведені на рис. 1. У першому стовпці цих результатів після номера тесту наведено результат його проходження, у другому – час виконання тесту, у третьому – обсяг використаної пам'яті.

Бачимо, що дана програма, незважаючи на правильність реалізованого алгоритму, не змогла пройти 25% тестів, оскільки вичерпала ліміт часу. Виконуючи 25% інших тестів програма вичерпала майже весь відведений час (понад 4 с). При цьому обсяг використання оперативної пам'яті практично не змінюється і коливається в межах 25% від максимально допустимого.

Після отримання таких результатів варто наголосити учням, що дієздатний алгоритм



Набір тестів #1	Вичерпано ліміт часу	6 003 ms	15 484 KiB
✓ Тест #1	Зараховано	32 ms	15 328 KiB
✓ Тест #2	Зараховано	32 ms	15 360 KiB
✓ Тест #3	Зараховано	32 ms	15 360 KiB
✓ Тест #4	Зараховано	33 ms	15 376 KiB
✓ Тест #5	Зараховано	4 637 ms	15 288 KiB
✓ Тест #6	Зараховано	4 639 ms	15 484 KiB
Тест #7	Вичерпано ліміт часу	6 002 ms	13 944 KiB
Тест #8	Вичерпано ліміт часу	6 003 ms	13 824 KiB

Рис. 1. Результати тестування програми першого варіанту алгоритму визначення досконалих чисел

і відповідна програма – це ще не все. Потрібно, щоб програма виконувалася за відведений час і економно використовувала оперативну пам'ять. Скільки операцій визначення дільника виконує дана програма, якщо m=1, а n – як завгодно велике (позначимо його через N)? Використовуючи формулу суми n перших членів арифметичної прогресії, ця кількість рівна $N\times(N\text{-}1)/2$. І ось тут варто ввести поняття асимптотичної обчислювальної складності і наголосити, що це величина порядку $O(N^2)$.

Як зменшити обчислювальну складність цього алгоритму? Потрібно зауважити, що *частка* від ділення числа N на його дільник не може бути меншим 2 (адже частку 1 забезпечує сам дільник N, який недопустимий за умовою задачі), тому дільники числа, менші за N, доцільно шукати не в інтервалі [1; N-1], а в [1; N/2] [11, с. 47]. Тоді шостий рядок наведеної вище програми запишеться у вигляді:

```
6  for (div = 1; div <= num/2; div++),
```

а на сайті https://www.e-olymp.com отримаємо результати тестування, наведені на рис. 2.

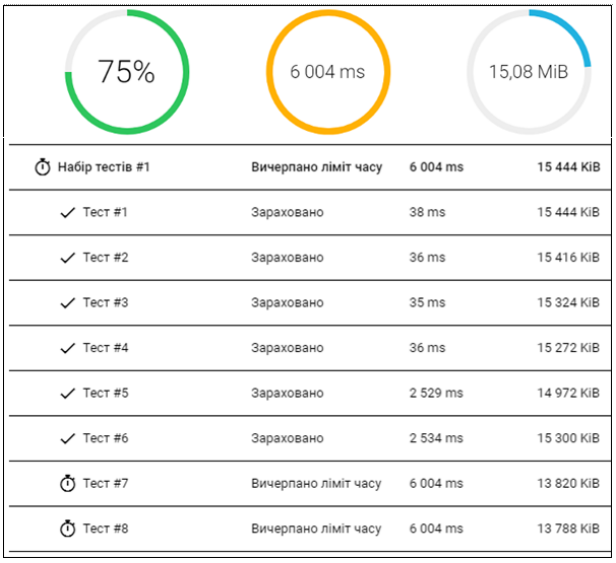
Кількість операцій визначення дільника становить тепер $N^2/4$, але *порядок* асимптотичної обчислювальної складності складає все ще $O(N^2)$. Час виконання тестів для невеликих N майже не змінився, для середніх – зменшився приблизно на 45%, а для великих – все ще не вкладається у 5 с. Тобто зменшення вдвічі кількості операцій визначення дільника пропорційно вплинуло на час виконання програми.

Для кардинального зменшення обчислювальної складності алгоритму зауважимо, що в суму для визначення досконалості чергового числа $N > 1$ завжди ввійде тривіальний дільник 1. Для всіх інших нетривіальних дільників a числа N (N % a = 0) завжди можна визначити також дільник

$$b = N / a \qquad (2)$$

цього ж числа N. Не зменшуючи загальності, для визначення унікальності дільників числа N покладемо

$$2 \le a \le b. \qquad (3)$$



Набір тестів #1	Вичерпано ліміт часу	6 004 ms	15 444 KiB
✓ Тест #1	Зараховано	38 ms	15 444 KiB
✓ Тест #2	Зараховано	36 ms	15 416 KiB
✓ Тест #3	Зараховано	35 ms	15 324 KiB
✓ Тест #4	Зараховано	36 ms	15 272 KiB
✓ Тест #5	Зараховано	2 529 ms	14 972 KiB
✓ Тест #6	Зараховано	2 534 ms	15 300 KiB
Тест #7	Вичерпано ліміт часу	6 004 ms	13 820 KiB
Тест #8	Вичерпано ліміт часу	6 004 ms	13 788 KiB

Рис. 2. Результати тестування програми другого варіанту алгоритму визначення досконалих чисел

Оскільки

$$a \times b = N, \qquad (4)$$

то, враховуючи (3), отримаємо, що $a^2 \le N$, тобто

$$2 \le^* a \le \sqrt{N}\ . \qquad (5)$$

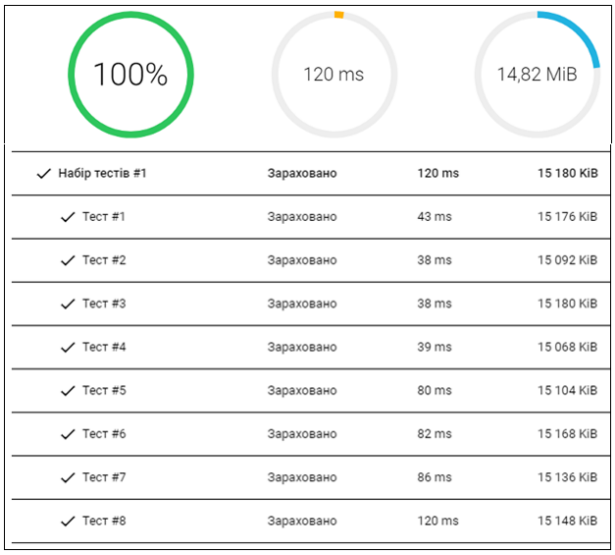
Отже, нетривіальні дільники числа N варто шукати навіть не в інтервалі [2; N/2], а в [2; $\sqrt{N}$] [11, с. 47-48], але тоді до суми потрібно додавати не лише кожен знайдений дільник a, а й його частку від ділення N на цей дільник, тобто дільник b (2). Причому додавання дільника b потрібно виконувати при $a \ne b$, оскільки, за умовою задачі, у сумі

дільники мають бути унікальні (це некоректно зроблено в [11, с. 48]). Цим самим вдасться уникнути подвійного додавання гранично можливого дільника

$$a = b = \sqrt{N} \qquad (6)$$

(далі буде показано, що цей дільник можна ігнорувати). З врахуванням (5) рядки початкової версії програми, позначені маркерами, перепишуться у вигляді:

```
4   for (num = m; num <= n; num++)//цикл по числах діапазону
5.1 {if (num>1) sumaD = 1;
4.2   else sumaD = 0;//початкове значення суми дільників
5.3 sqrt = (int)Math.Sqrt(num);
6   for (a = 2; a <= sqrt; a++)
7   if (num % a == 0)
8.1 {sumaD += a; b=num/a;
8.2 if (b != a) sumaD += b; }
```



✓ Набір тестів #1	Зараховано	120 ms	15 180 KiB
✓ Тест #1	Зараховано	43 ms	15 176 KiB
✓ Тест #2	Зараховано	38 ms	15 092 KiB
✓ Тест #3	Зараховано	38 ms	15 180 KiB
✓ Тест #4	Зараховано	39 ms	15 068 KiB
✓ Тест #5	Зараховано	80 ms	15 104 KiB
✓ Тест #6	Зараховано	82 ms	15 168 KiB
✓ Тест #7	Зараховано	86 ms	15 136 KiB
✓ Тест #8	Зараховано	120 ms	15 148 KiB

Рис. 3. Результати тестування програми третього варіанту алгоритму визначення досконалих чисел зі зменшеною асимптотичною складністю

Результати тестування програми з такими змінами наведено на рис. 3. Порядок асимптотичної обчислювальної складності з цими модифікаціями знизився до $O(N\sqrt{N})$ і тому середній час виконання тестів зменшився в десятки разів, що дало змогу розв'язати поставлену задачу.

Для подальшого зменшення асимптотичної складності алгоритму визначення досконалих чисел доведено наступні дві теореми.

Теорема 1. Всі досконалі числа парні.

Доведення. Скористаємося методом доведення від супротивного. Нехай існує непарне досконале число N. Тоді, аналізуючи всеможливі розклади цього числа виду (4), приходимо до висновку, що всі дільники a, b в таких розкладах мають бути непарними. У суму дільників непарного досконалого числа N завжди має входити 1, всеможливі пари непарних дільників a, b ($a < b$, сума яких завжди парна), та, можливо, число $\sqrt{N}$. Ми дійшли до суперечності: виходячи з суми дільників, $\sqrt{N}$ має бути парним, а, виходячи з (4) – непарним. **Тому $\sqrt{N}$ не може входити у суму дільників непарного досконалого числа *N***. Отже, виходить, що непарне досконале число N має лише різні **непарні** дільники, включаючи 1, а це суперечить формулі Евкліда (1).

Теорема 2. Квадратний корінь досконалого числа не може бути натуральним дільником цього числа. Або, що те саме: досконале число не може бути квадратом іншого натурального числа.

Доведення. Враховуючи твердження попередньої теореми, покажемо, що корінь парного досконалого числа не може бути його натуральним дільником. Знову скористаємося методом доведення від супротивного. Нехай таке парне досконале N існує. Тоді $\sqrt{N}$ – теж парне. Нехай $\sqrt{N}=2k$, де k – натуральне. Тоді $N=4k^2$. Якщо k – просте, то непарними дільниками числа N будуть числа 1, k, k^2 (3^1 непарних чисел). Якщо в розкладі на прості множники числа k буде два простих непарних числа p та q, то непарними дільниками числа N будуть числа 1, p, p^2, q, qp, qp^2, q^2, q^2p, q^2p^2 (3^2 непарних чисел). Аналогічно, якщо в розкладі на прості множники числа k буде s непарних чисел, то непарними дільниками числа N будуть 3^s непарних чисел. Тобто сума дільників числа N завжди буде непарною, що суперечить твердженню про парність та досконалість N.

Отже, досконалі числа N на заданому інтервалі доцільно шукати лише серед парних чисел, а їх нетривіальні дільники – в інтервалі $[2; \sqrt{N-1}]$. Тоді рядки початкової версії про-

грами, позначені маркерами, перепишуться у вигляді:

```
4.1 if (m % 2 == 1) m++;//переходимо на перше парне
4.2 for (num = m; num <= n; num+=2)
5.1 {if (num > 1) sumaD = 1;
5.2   else sumaD = 0;//початкове значення суми дільників
5.3  sqrt = (int)Math.Sqrt(num-1);
6  for (a = 2; a <= sqrt; a++)
7   if (num % a == 0)
8    sumaD += a + num/a;
```

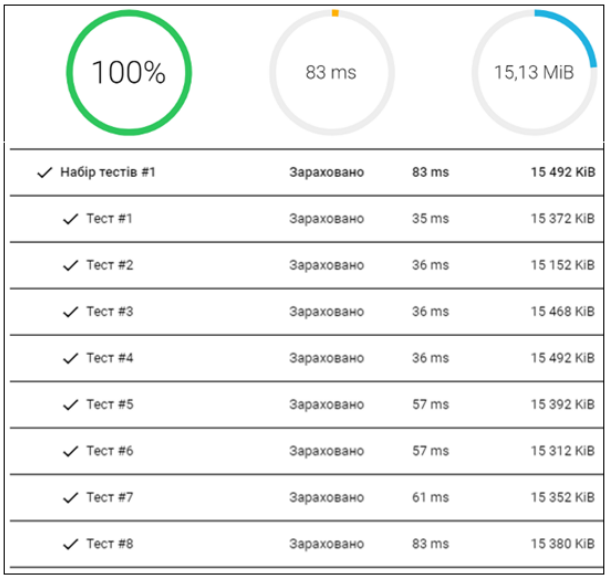


Рис. 4. Результати тестування програми четвертого варіанту алгоритму визначення досконалих чисел після врахування їх властивостей

Врахування доведених властивостей досконалих чисел дає змогу зменшити середній час проходження тестів ще на 30% (рис. 4). Саме розв'язування цієї та аналогічних задач дасть змогу учням чітко усвідомити зміст та значення поняття асимптотичної складності алгоритму.

За результатами дослідження зробимо такі **висновки**:

1. Кардинальне зменшення обчислювальної складності алгоритму можливо саме за рахунок зменшення асимптотичної обчислювальної складності.

2. Поняття асимптотичної обчислювальної складності в контексті часу виконання [12] варто формувати в учнів на уроках інформатики всіх загальноосвітніх шкіл, починаючи з 7 класу, в процесі вивчення вкладених циклів. Це формуватиме навики написання не лише правильних, а й ефективних (в контексті часу виконання та використання обчислювальних ресурсів) програм.

3. Пошук досконалих чисел на вказаному інтервалі доцільно виконувати лише серед парних натуральних чисел, аналізуючи їх нетривіальні дільники до квадратного кореня з цих чисел, зменшених на одиницю. При цьому до суми дільників потрібно додавати не лише знайдені дільники, а й отримані частки. Такі вдосконалення відносно варіанту послідовного перебору натуральних чисел інтервалу та їх дільників дають змогу зменшити асимптотичну обчислювальну складність з $O(N^2)$ до $O(N\sqrt{N})$.

Список використаних джерел

1. Обчислювальна складність. URL: https://znaimo.com.ua/Обчислювальна_складність#link0 (дата звернення: 21.12.2020).
2. Програма курсу «Інформатика». 5 – 9 класи загальноосвітніх навчальних закладів. URL: https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-5-9-klas/onovlennya-12-2017/programa-informatika-5-9-traven-2015.pdf (дата звернення: 21.12.2020).
3. Програма курсу «Інформатика». 8–9 класи загальноосвітніх навчальних закладів з поглибленим вивченням інформатики. URL: https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-5-9-klas/informatika.pdf (дата звернення: 21.12.2020).
4. Шинкаренко В. І. Особливості практичного застосування показників обчислювальної складності алгоритмів. *Проблеми програмування*. 2008. № 2-3. С. 57-63.
5. Оцінка складності алгоритмів, або Що таке O(log n). URL: https://echo.lviv.ua/dev/53 (дата звернення: 21.12.2020).
6. Руденко В. Д., Речич Н. В., Потієнко В. О. Інформатика для загальноосвітніх навчальних закладів з поглибленим вивченням інформатики : підруч. для 9 кл. загальноосвіт. навч. закл. Харків : Вид-во «Ранок». 2017. 240 с.

7. Захарова О. Вивчаємо математику: Досконалі числа. 2014. URL: http://matematikav6.blogspot.com/2014/09/blog-post_66.html (дата звернення: 21.12.2020).
8. Perfect number. URL: https://en.wikipedia.org/wiki/Perfect_number (дата звернення: 21.12.2020).
9. Great Internet Mersenne Prime Search. URL: https://www.mersenne.org/ (дата звернення: 21.12.2020).
10. Теорія чисел – математичні основи розв'язування олімпіадних задач. URL: https://www.e-olymp.com/uk/blogs/posts/53 (дата звернення: 21.12.2020).
11. Креневич А. П., Обвінцев О. В. С у задачах і прикладах : навчальний посібник із дисципліни «Інформатика та програмування». Київ: Видавничо-поліграфічний центр «Київський університет», 2011. 208 с.
12. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein Introduction to Algorithms, Third Edition. Cambridge: MIT Press, 2009. 1320 p.

References

1. Obchysliuvalna skladnist [Computational complexity]. Retrieved from: https://znaimo.com.ua/Обчислювальна_складність#link0 [in Ukrainian].
2. Prohrama kursu «Informatyka». 5 – 9 klasy zahalnoosvitnikh navchalnykh zakladiv ["Informatics" course program. 5-9 classes of general educational institutions]. Retrieved from: https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-5-9-klas/onovlennya-12-2017/programa-informatika-5-9-traven-2015.pdf [in Ukrainian].
3. Prohrama kursu «Informatyka». 8 – 9 klasy zahalnoosvitnikh navchalnykh zakladiv z pohlyblenym vyvchenniam informatyky ["Informatics" course program. 8-9 grades of general educational institutions with in-depth study of informatics]. Retrieved from: https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-5-9-klas/informatika.pdf [in Ukrainian].
4. Shynkarenko V. I. (2008). Osoblyvosti praktychnoho zastosuvannia pokaznykiv obchysliuvalnoi skladnosti alhorytmiv [Peculiarities of practical application of indicators of computational complexity of algorithms]. *Problemy Prohramuvannia (Programming problems)*. 2-3, 57-63. [in Ukrainian].
5. Otsinka skladnosti alhorytmiv, abo Shcho take O(log n) [Estimating the complexity of algorithms, or What is O(log n)]. Retrieved from: https://echo.lviv.ua/dev/53 [in Ukrainian].
6. Rudenko V. D., Rechych N. V., Potiienko V. O. (2017). Informatyka dlia zahalnoosvitnikh navchalnykh zakladiv z pohlyblenym vyvchenniam informatyky : pidruch. dlia 9 kl. zahalnoosvit. navch. zakl. [Informatics for general educational institutions with in-depth study of informatics: a textbook for 9th grade of general educational institutions]. Kharkiv: «Ranok» Publishing House. 240 p. [in Ukrainian].
7. Zakharova O. (2014). Vyvchaiemo matematyku: Doskonali chysla [We study mathematics: Perfect numbers]. Retrieved from: http://matematikav6.blogspot.com/2014/09/blog-post_66.html [in Ukrainian].
8. Wikipedia (2024). Perfect number. Retrieved from: https://en.wikipedia.org/wiki/Perfect_number
9. Great Internet Mersenne Prime Search (2024). Welcome to GIMPS, the Great Internet Mersenne Prime Search. Retrieved from: https://www.mersenne.org/
10. Teoriia chysel – matematychni osnovy rozviazuvannia olimpiadnykh zadach [Number theory – mathematical foundations of solving Olympiad problems]. Retrieved from: https://www.e-olymp.com/uk/blogs/posts/53 [in Ukrainian].
11. Krenevych A. P., Obvintsev O. V. (2011). C u zadachakh i prykladakh : navchalnyi posibnyk iz dystsypliny "Informatyka ta prohramuvannia" [C in problems and examples: a study guide on the discipline "Informatics and programming".]. Kyiv: Publishing and Printing Center «Kyiv University». 208 p. [in Ukrainian].
12. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (2009). Introduction to Algorithms, Third Edition. Cambridge: MIT Press. 1320 p.