

ОПТИМІЗАЦІЯ РОЗВ'ЯЗАННЯ ЗАДАЧ З ПРОГРАМУВАННЯ ЗАСОБАМИ МАТЕМАТИКИ

Присяжнюк Т.А.

У статті висвітлено основні шляхи підготовки школярів до участі в олімпіадах з програмування. Наведено приклади застосування алгоритму Евкліда та функції Ейлера для розв'язування задач та їх реалізація на мові програмування Pascal.

В статье освещены основные пути подготовки школьников к участию в олимпиадах с программирования. Показаны примеры применения алгоритма Эвклида и функции Эйлера для решения задач, а также их реализация на языке программирования Pascal.

In the article the basic ways of preparation of students are lighted up to participating in olympiads from programming. The examples of application of algorithm of Evklida and functions of Euler are shown for the decision of tasks, and also their realization in programming of Pascal language.

Ключові слова: підготовка, прямий перебір, оптимізація, алгоритм Евкліда, функція Ейлера.

У наш час, коли відбувається постійне оновлення інформаційного суспільства, все більшої уваги науковців і практиків привертає проблема кваліфікованої підготовки школярів до олімпіад з інформатики різних рівнів, оскільки розвиток інформаційних технологій та стрімке зростання обсягів інформації відкривають нові можливості для учнів.

Підготовка до олімпіад – це тривалий трудомісткий процес, який потребує від учнів концентрації уваги, розвитку логічного мислення, а також вміння оптимізувати будь-яку задачу. Обов'язковою умовою, при проведенні змагань, є виконання задач за певний проміжок часу (тобто ліміт часу обмежений). Для прикладу наведемо сайт www.e-olimp.com, який використовується для підготовки та проведення змагань різних рівнів [1, 2]. Для кожної задачі, яка подана до загального переліку завдань, встановлюється визначений ліміт часу.

Важливим аспектом підготовки школярів до змагань є уміння та навички побудови оптимальних алгоритмів розв'язання задач. Одним із перспективних шляхів оптимізації є використання математичних знань.

Для заданої задачі шукаємо інваріантну їй задачу, розв'язання якої буде значно простіше від початкової. Так, наприклад, як відомо з математики, виходячи з властивостей степенів $a^m \cdot a^n = a^{m+n}$, $a^m : a^n = a^{m-n}$ і т.д., операцію множення степенів можемо замінити додаванням показників, ділення степенів – відніманням показників, тощо. Таким чином задача істотно спрощується, відповідно спрощується і її дослідження. Аналогічний підхід до аналізу задач дозволить провести оптимізацію алгоритму її розв'язання.

Наведемо один із методів оптимізації алгоритму на прикладі розв'язання такої задачі.

Задача. Скільки існує правильних нескоротних дробів на проміжку (0;1), знаменник яких не перевищує натуральне число N ?

Спочатку нагадаємо, які дробі називаються правильними та нескоротними. Дріб, чисельник якого менший знаменника, називається правильним, а нескоротним – чисельник і знаменник яких не мають спільних дільників.

Перший спосіб розв'язку даної задачі школярами можна спрогнозувати: вони запропонують прямий перебір. Очевидно, що всі дробі, в яких чисельник – 1, а знаменник – від 2 до N , – будуть правильними нескоротними, тому початкове значення кількості нескоротних дробів становитиме $N-1$. Для всіх інших дробів, у яких чисельник не дорівнює 1, будемо розглядати ті випадки, в яких знаменник знаходиться на проміжку від 3 до N , а чисельник – від 2, але менший знаменника. Для кожної пари чисельник-знаменник з'ясуємо: скільки буде спільних множників (не враховуючи 1); якщо такі відсутні, то кількість нескоротних дробів збільшуємо на 1, інакше переходимо до наступного дробу.

Примітка: якщо $N=1$, то зрозуміло, що кількість таких дробів буде рівна нулю.

Реалізуємо даний алгоритм розв'язання задачі мовою програмування Паскаль.

```
program prjamui_perebir;
```

```
var n, i, j, k : int64;
```

```
l, flag : integer;
```

begin

read(n);

if n=1 then k:=0 else

begin

k:=n-1; i:=2;

while i<=n-1 do

begin

j:=i+1;

while j<=n do

begin

flag:=0;

if (i mod 2=0) and (j mod 2=0) then flag:=1

else

for l:=3 to trunc(j/2) do

if (j mod l =0) and (i mod l =0) then

begin

flag:=1; break

end;

if flag=0 then inc(k);

inc(j)

end;

inc(i)

end

end;

```
writeln(k)
```

```
end.
```

Даний алгоритм розв'язання є правильним, але перевищує ліміт часу, оскільки задовольняє лише 52 % тестів, запропонованих на вищевказаному сайті, який ми використовуємо для підготовки та проведення змагань різних рівнів з інформатики.

Наступним етапом розв'язання цієї задачі є її оптимізація.

Зрозуміло, щоб спростити наш алгоритм, досить для кожної пари чисельник-знаменник знайти *НСД* (найбільший спільний дільник) за відомим *алгоритмом Евкліда* [3;4], і якщо він рівний 1, то збільшуватимемо лічильник кількості дробів на 1.

Подамо реалізацію запропонованого методу на мові програмування Паскаль.

```
program nsd;
```

```
var n, i, j, k : int64;
```

```
l, flag : integer;
```

```
function nod (a,b:int64):int64;
```

```
var x,y :int64;
```

```
begin
```

```
  x:=a; y:=b;
```

```
  while x<>y do
```

```
    if x>y then x:=x-y else y:=y-x;
```

```
  nod:=x
```

```
end;
```

```
begin
```

```
  read(n);
```

```
  if n=1 then k:=0 else
```

```
    begin
```

```

k:=n-1; i:=2;

while i<=n-1 do

begin

    j:=i+1;

    while j<=n do

begin

    if nod(i,j)=1 then inc(k);

    inc(j)

end;

    inc(i)

end

end;

writeln(k)

end.

```

Даний розв'язок є правильним, але він знову перевищує визначений ліміт часу умовою задачі (60 % тестів, наведених на сайті www.e-olimp.com [1]).

Щоб вирішити задачу на всі 100 % необхідно звернутись до відомої *функції Ейлера* [5–8], яка дозволяє, не перебираючи всі можливі варіанти, для заданого числа N обчислити кількість чисел менших ніж N і взаємпростих із ним.

Знаходження *функції Ейлера*, яка вивчається на факультативах з математики, здійснюється наступним чином. Якщо натуральне число p є простим, то кількість чисел взаємпростих із ним буде обчислюватись за формулою: $\varphi(p) = p - 1$. Інакше, будь-яке натуральне число можна подати в канонічному вигляді $p = p_1^{\alpha} p_2^{\beta} \dots p_n^{\gamma}$, (де p_1, p_2, \dots, p_n – прості числа), для якого кількість чисел, взаємпростих з даним числом p , буде обчислюватися за формулою:

$$\varphi(p) = (p_1^{\alpha} - p_1^{\alpha-1}) (p_2^{\beta} - p_2^{\beta-1}) \dots (p_n^{\gamma} - p_n^{\gamma-1}).$$

Покажемо алгоритм розв'язання задачі з використанням функції Ейлера.

```
var n, i, j, k : longint;
```

```
l, flag : integer;
```

```
function ejler (a:longint):longint;
```

```
var x,i1,j1,j2,k1,zz,o :longint;
```

```
f1,ej : longint;
```

```
begin
```

```
  x:=a; ej:=1; i1:=2; j2:=2; k1:=0;
```

```
  zz:=trunc(sqrt(x));
```

```
  while j2<=zz do
```

```
    begin
```

```
      if x mod j2 =0 then begin k1:=1; break; end;
```

```
      inc(j2)
```

```
    end;
```

```
  if k1=0 then ej:=x-1 else
```

```
  while x>1 do
```

```
    begin
```

```
      j1:=0;
```

```
      if (x mod i1=0) then
```

```
        begin
```

```
          while (x mod i1 = 0) do
```

```
            begin
```

```
              x:=x div i1; inc(j1)
```

```
            end;
```

```

    o:=i1;

    for f1:=2 to j1 do

        o:=o*i1;

        ej:=ej*(o-(o div i1))

    end;

    inc( i1)

    end;

    ejler:=ej

end;

begin

    read(n);

    if n=1 then k:=0 else

        begin

            i:=2;

            while i<=n do

                begin

                    k:=k+ejler(i);

                    inc(i)

                end

            end;

            writeln(k)

        end.

```

Отже, як бачимо, використання деяких елементів математики сприяє значному спрощенню алгоритму розв'язання задачі, а саме його оптимізації по часу. Тому при

підготовці школярів до олімпіад з програмування поряд із умінням програмувати важливим є володіння математичними знаннями, а також уміння логічно поєднувати обидва компоненти.

Література:

1. www.e-olimp.com
2. Жуковський С.С. “Е-olimp” – система автоматичної перевірки задач та проведення олімпіад з інформатики в інтернеті /Комп’ютер у школі та сім’ї. – №1 (65). – 2008. – С.48-50.
3. http://ru.wikipedia.org/wiki/Алгоритм_Евклида
4. <http://borlpasc.narod.ru/docym/prac/algorev.htm>
5. http://algolist.manual.ru/maths/count_fast/phi_n.php
6. http://e-maxx.ru/algo/euler_function
7. <http://dic.academic.ru/dic.nsf/ruwiki/102639>
8. http://ru.wikipedia.org/wiki/Функция_Эйлера