Дистанционная подготовка к Всероссийской олимпиаде по информатике

Преподаватель:
к.ф.-м.н., заведующий кафедрой ВТиКГ ДВГУПС,
преподаватель программы IT-школа Samsung,
Пономарчук Юлия Викторовна
E-mail: yulia.ponomarchuk@gmail.com

Занятие 2.

Основы оценки сложности алгоритмов.

Поиск НОД и НОК.

Системы счисления

Знакомство с понятием сложности алгоритма

При сравнении производительности различных алгоритмов решения задачи следует учитывать, что *скорость выполнения* компьютерных программ зависит от:

- вычислительной производительности процессора ЭВМ,
- объема кэш-памяти, оперативной памяти,
- скорости доступа к запоминающим устройствам и т.п.

Поэтому время работы программы измеряется ***количеством операций***

- Главную роль играет характер *возрастания числа элементарных операций при увеличении объема данных* в задаче.
- Кроме того, обычно учитывают *объем памяти*, которая используется во время работы программы
- *Эффективным решением* олимпиадной задачи является такое, что:
    - укладывается в ограничения по времени работы программы
    - укладывается в ограничения ресурсов памяти, требующейся для выполнения программы
    - написано за кратчайшее время

Сложность алгоритма

- **Сложность алгоритма** – функция *$F_A(n)$*, определенная как наибольшее количество элементарных действий при решении задачи с объемом входных данных *n* с помощью алгоритма *A*
- Например:
  - на вход подается *n* чисел, следует вычислить их сумму
  - тогда сложность алгоритма определяется *количеством операций сложения* и линейно зависит от *n*

Важные определения

- Функция *G(n)* называется ***оценкой сверху для функции*** *F(n),* если существует положительное число *c* и натуральное *m*, для которых $F(n) \le cG(n)$ при $n > m$
  - *Обозначение:* $F(n) = O\big(G(n)\big)$
  - Читается: «*O* большое от…»
- Функция *G(n)* называется ***оценкой снизу для функции*** *F(n),* если существует положительное число *d* и натуральное *m*, для которых $dG(n) \le F(n)$ при $n > m$
  - *Обозначение:* $F(n) = \Omega\big(G(n)\big)$
  - Читается: «омега от…»
- Функция *G(n)* называется ***оценкой для функции*** *F(n),* или *F(n)* является функцией порядка *G(n),* если существуют положительные числа c_1, c_2 и натуральное *m*, для которых

$$\mathrm{c}_1 G(n) \le F(n) \le c_2 G(n) \text{ при } n > m$$

  - *Обозначение:* $F(n) = \Theta\big(G(n)\big)$
  - Читается: «*тэта* от…»

Важные определения

- Функция *G(n)* называется ***оценкой сверху для функции*** *F(n),* если существует положительное число *c* и натуральное *m*, для которых $F(n) \leq cG(n)$ при $n > m$
  - *Обозначение:* $F(n) = O\big(G(n)\big)$
  - Читается: «*O* большое от…»
- Функция *G(n)* называется ***оценкой снизу для функции*** *F(n),* если существует положительное число *d* и натуральное *m*, для которых $dG(n) \leq F(n)$ при $n > m$
  - *Обозначение:* $F(n) = \Omega\big(G(n)\big)$
  - Читается: «омега от…»
- Функция *G(n)* называется ***оценкой для функции*** *F(n),* или *F(n)* является функцией порядка *G(n),* если существуют положительные числа c_1, c_2 и натуральное *m*, для которых

$$\mathrm{c}_1 G(n) \leq F(n) \leq c_2 G(n) \text{ при } n > m$$

  - *Обозначение:* $F(n) = \Theta\big(G(n)\big)$
  - Читается: «*тэта* от…»

- Функция *G(n)* называется ***оценкой сверху для функции*** *F(n),* если существует положительное число *c* и натуральное *m*, для которых $F(n) \leq cG(n)$ при $n > m$
  - *Обозначение:* $F(n) = O\big(G(n)\big)$
  - Читается: «*O* большое от…»
- Функция *G(n)* называется ***оценкой снизу для функции*** *F(n),* если существует положительное число *d* и натуральное *m*, для которых $dG(n) \leq F(n)$ при $n > m$
  - *Обозначение:* $F(n) = \Omega\big(G(n)\big)$
  - Читается: «омега от…»
- Функция *G(n)* называется ***оценкой для функции*** *F(n),* или *F(n)* является функцией порядка *G(n),* если существуют положительные числа c_1, c_2 и натуральное *m*, для которых

$$c_1 G(n) \leq F(n) \leq c_2 G(n) \text{ при } n > m$$

  - *Обозначение:* $F(n) = \Theta\big(G(n)\big)$
  - Читается: «*тэта* от…»

Характер возрастания сложности

- Функция *G(n)* называется ***оценкой сверху для функции*** *F(n),* если существует положительное число *c* и натуральное *m*, для которых $F(n) \leq cG(n)$ при $n > m$
  - *Обозначение:* $F(n) = O\big(G(n)\big)$
  - Читается: «*O* большое от…»
- Функция *G(n)* называется ***оценкой снизу для функции*** *F(n),* если существует положительное число *d* и натуральное *m*, для которых $dG(n) \leq F(n)$ при $n > m$
  - *Обозначение:* $F(n) = \Omega\big(G(n)\big)$
  - Читается: «омега от…»
- Функция *G(n)* называется ***оценкой для функции*** *F(n),* или *F(n)* является функцией порядка *G(n),* если существуют положительные числа c_1, c_2 и натуральное *m*, для которых

$$c_1 G(n) \leq F(n) \leq c_2 G(n) \text{ при } n > m$$

  - *Обозначение:* $F(n) = \Theta\big(G(n)\big)$
  - Читается: «*тэта* от…»

Классификация алгоритмов по сложности

Порядок сложности	Во сколько раз увеличивается максимальный размер задач при увеличении скорости работы компьютера в 100 раз (с 10^6 до 10^8 операций в секунду)
n	100
n^2	≈ 10
n^3	≈ 5
2^n	≈ 1,3
$n!$	1,2

Примеры задач

- Функция *G(n)* называется ***оценкой сверху для функции*** *F(n),* если существует положительное число *c* и натуральное *m,* для которых $F(n) \le cG(n)$ при $n > m$
  - *Обозначение:* $F(n) = O(G(n))$
  - Читается: «*О* большое от…»
- Функция *G(n)* называется ***оценкой снизу для функции*** *F(n),* если существует положительное число *d* и натуральное *m,* для которых $dG(n) \le F(n)$ при $n > m$
  - *Обозначение:* $F(n) = \Omega(G(n))$
  - Читается: «омега от…»
- Функция *G(n)* называется ***оценкой для функции*** *F(n),* или *F(n)* является функцией порядка *G(n),* если существуют положительные числа c_1, c_2 и натуральное *m,* для которых

$$c_1G(n) \le F(n) \le c_2G(n) \text{ при } n > m$$

```
function isPrime(n : integer) : boolean;
  var k, t : integer; {очередной делитель, верхний предел}
begin
  if not odd(n) then begin
    isPrime := (n = 2); exit
  end;
  k := 3; {первый делитель 3}
  t := round(sqrt(n)); {round - для гарантии}
  while (k <= t) and (n mod k <> 0) do inc(k, 2);
  {(k > t) or (n mod k = 0)}
  isPrime := k > t {если k > t, то число простое,
                     иначе составное}
end;
```

Примеры задач

- Функция *G(n)* называется ***оценкой сверху для функции*** *F(n)*, если существует положительное число *c* и натуральное *m*, для которых $F(n) \leq cG(n)$ при $n > m$
  - *Обозначение:* $F(n) = O\big(G(n)\big)$
  - Читается: «*O* большое от…»
- Функция *G(n)* называется ***оценкой снизу для функции*** *F(n)*, если существует положительное число *d* и натуральное *m*, для которых $dG(n) \leq F(n)$ при $n > m$
  - *Обозначение:* $F(n) = \Omega\big(G(n)\big)$
  - Читается: «омега от…»
- Функция *G(n)* называется ***оценкой для функции*** *F(n)*, или *F(n)*

```
while b > 0 do begin
  c := a mod b; a := b; b := c
end; {a - искомое}
```

и существуют положительные
ых
) при $n > m$
  - *Обозначение:* $F(n) = \Theta(G(n))$
  - Читается: «*тэта* от…»

Бинарный алгоритм Евклида

Бинарный алгоритм Евклида основан на соотношениях *(a>b)*:

$$\text{НОД}(2a, 2b) = 2 \cdot \text{НОД}(a, b),$$
$$\text{НОД}(a, b) = \text{НОД}(b, a), \text{ если } a < b,$$
$$\text{НОД}(2a, b) = \text{НОД}(a, b), \text{ если } b \text{ нечетно},$$
$$\text{НОД}(a, 2b) = \text{НОД}(a, b), \text{ если } a \text{ нечетно},$$
$$\text{НОД}(a, b) = \text{НОД}(a-b, b), \text{ если } a > b,$$
$$\text{НОД}(a, a) = a,$$
$$\text{НОД}(a, 1) = 1.$$

Бинарный алгоритм Евклида выполняется примерно на 60% быстрее традиционного.

Поиск наименьшего общего кратного

- Функция $G(n)$ называется ***оценкой сверху для функции*** $F(n)$, если существует положительное число c и натуральное m, для которых $F(n) \leq cG(n)$ при $n > m$
  - *Обозначение:* $F(n) = O(G(n))$
  - Читается: «*O* большое от…»
- Функция $G(n)$ называется ***оценкой снизу для функции*** $F(n)$, если существует положительное число d и натуральное m, для которых $dG(n) \leq F(n)$ при $n > m$
  - *Обозначение:* $F(n) = \Omega(G(n))$
  - Читается: «омега от…»
- Функция $G(n)$ называется ***оценкой для функции*** $F(n)$, или $F(n)$ является функцией порядка $G(n)$, если существуют положительные числа c_1, c_2 и натуральное m, для которых

  $$c_1 G(n) \leq F(n) \leq c_2 G(n) \text{ при } n > m$$
  - *Обозначение:* $F(n) = \Theta(G(n))$
  - Читается: «*тэта* от…»

Арифметика остатков

b=a mod p -> остаток от деления a на p равен b

Пример: 5 mod 3=8 mod 3=2

Свойства арифметических операций

(a+b) mod p = ((a mod p) + (b mod p)) mod p

(a-b) mod p = (p + (a mod p) – (b mod p)) mod p, при a>b

-a mod p= (p-a) mod p

(a*b) mod p = ((a mod p) * (b mod p)) mod p

Аналогичной формулы для деления нет!
Поскольку обратный элемент по умножению существует только когда НОД (a, p)=1

Операцию деления можно выполнить только найдя обратный элемент по модулю p с помощью расширенного алгоритма

Позиционные системы счисления (ПСС)

Система счисления – система записи чисел с помощью определенного набора цифр

Цифры – символы, с помощью которых записываются числа в системе счисления

Алфавит – совокупность символов, используемых для записи чисел

Основание системы счисления – количество цифр, используемых для записи чисел

Позиционная система счисления – система счисления, в которой количественный эквивалент цифры зависит от ее положения в записи числа

$$5047 = 5*1000 + 0*100 + 4*10 + 7*1$$

Позиционные системы счисления (ПСС)

Базис ПСС – последовательность чисел, каждое из которых задает «вес» соответствующего разряда

Традиционная ПСС – система счисления, базис которой образуют члены геометрической прогрессии,

знаменатель P>1 – натуральное число,

значения цифр – целые числа

$$\ldots, P^{-3}, P^{-2}, P^{-1}, 1, P^{1}, P^{2}, P^{3}, \ldots$$

P – **основание** P-ричной системы счисления

<u>Примеры нетрадиционных СС</u>

Фибоначчиева система счисления

Алфавит: цифры 0 и 1

Базис: последовательность чисел Фибоначчи 1, 2, 3, 5, 8, 13, 21, 34, …

Факториальная система счисления

Алфавит: количество цифр увеличивается с ростом номера разряда

В двоичной, восьмеричной и шестнадцатеричной системах счисления

10	2	8	16
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8

10	2	8	16
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

Арифметические операции в ПСС

- Сложение
- Вычитание
- Умножение
- Деление

(действуют обычные правила выполнения операций «в столбик», подробнее рассмотрим в следующей лекции)

Перевод целых чисел из Р-СС в 10-СС

$\mathbf{a_n \ldots a_2 a_1 a_0\ _P = a_n{*}P^n + \ldots + a_2{*}P^2 + a_1{*}P^1 + a_0{*}P^0}$

$B0F9_{16} = [11_{10}][0_{10}][15_{10}][9_{10}] =$

$= 11_{10}{*}16_{10}^{\ 3} + {*}\ 15_{10}\ 16_{10}^{\ 1} + 9_{10}\ {*}16_{10}^{\ 0} = 45305_{10}$

Схема Горнера:

$$a_n P^n + a_{n-1} P^{n-1} + \ldots + a_1 P^1 + a_0 =$$

$$= \left(a_n P^{n-1} + a_{n-1} P^{n-2} + \ldots + a_1 \right) P + a_0 =$$

$$= \left(\ldots \left(\left(\left(a_n P + a_{n-1} \right) P + a_{n-2} \right) P + a_{n-3} \right) P + \ldots + a_1 \right) P + a_0$$