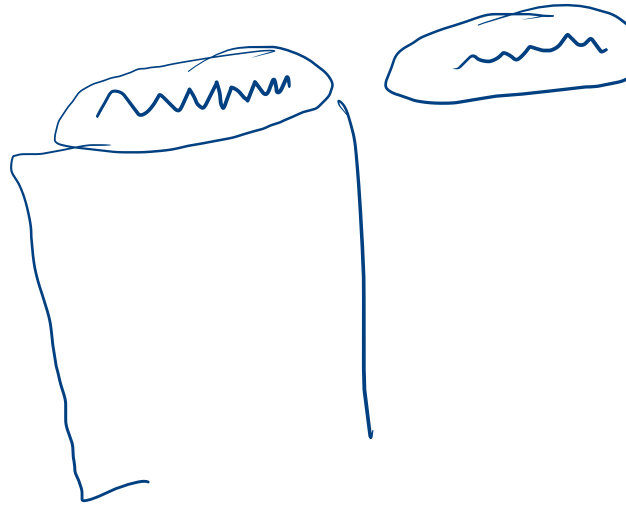
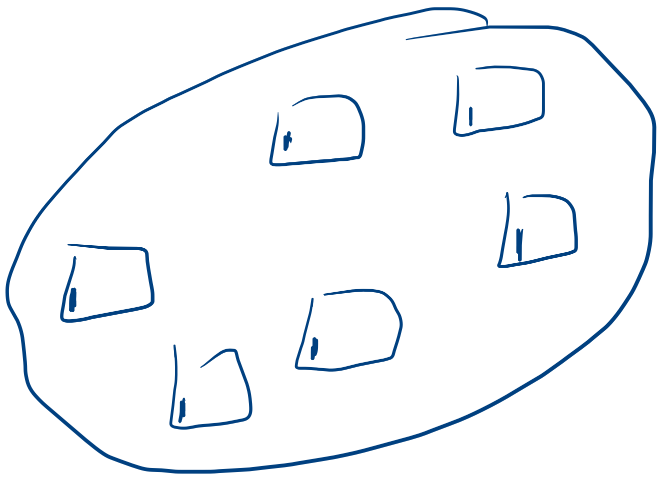
О курсе и алгоритмах



Алгоритм: вход(значения) → вычислительная процедура → выход(значения)

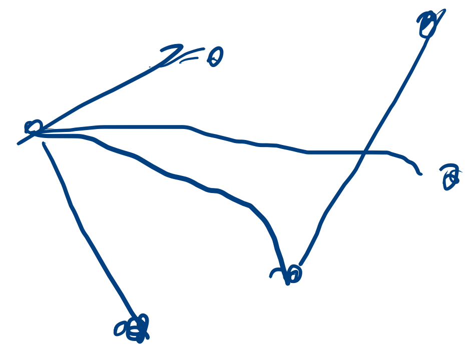
Используются для решения задач:

Google Maps — построение маршрута = графы
Биоинформатика — обработка геномных последовательностей = строки
Интернет — поиск страниц = хеш-таблицы
и много дугих...



В курсе изучим:

- Сортировки
- Элементарные структуры данных(стеки, очереди, списки)
- Кучи и двоичные деревья поиска
- Графы и алгоритмы на них
- Хеш-таблицы
- Алгоритмы для работы со строками



Зачем оценивать время работы алгоритма?

Задача: $\underbrace{\text{отсортировать массив чисел}}$

A

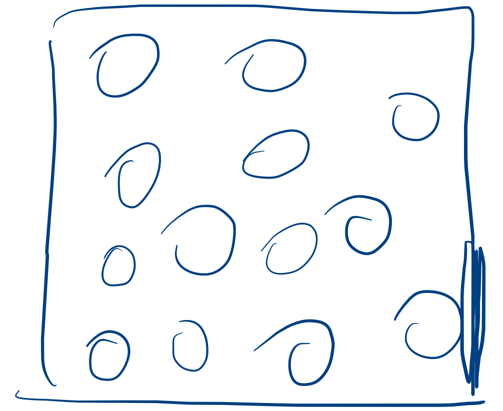
10^9 ком/сек

n^2 опер

B

10^7 ком/с

$n \log n$ опер



7	5	1	13	19

1 5 7 13 19

Массив 10^4 элем $n = 10^7$

$$\frac{(10^7)^2}{10^9} = 10^5 \text{ сек} > \text{сутки}$$

$$\frac{\cancel{10^7} \cdot \log 10^7}{\cancel{10^7}} \approx \underbrace{23,25 \text{ сек}}$$

Вывод: скорость компьютера менее важный показатель, чем эффективность алгоритма

Сортировка вставкой

Пример:

7 4 1 3 2 5

4 7 1 3 2 5

1 4 7 3 2 5

1 3 4 7 2 5

key = 3

i = j - 1 = 3

j = 4

7 4 1 3 2 5

key = 4

Оценка времени работы:

INSERTION-SORT(A)

for $j = 2$ to $A.length$
    $key = A[j]$
    // Вставка $A[j]$ в отсортированную последовательность $A[1..j-1]$
    $i = j - 1$
    while $i > 0$ и $A[i] > key$
        $A[i+1] = A[i]$
        $i = i - 1$
    $A[i+1] = key$

n

n

$O(n)$

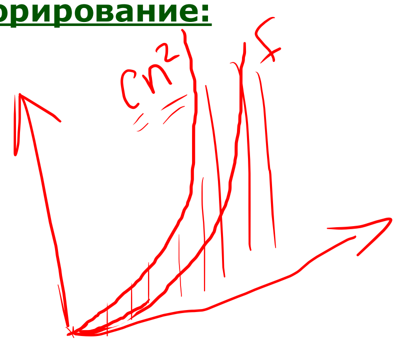
Худший случай: массив упорядочен в порядке обратном искомому

Лудший случай: массив упорядочен

$$\sum_{j=2}^{n} j-1 = \sum_{j=1}^{n-1} j = \frac{(n-1)n}{2} = \frac{n^2-n}{2} = O(n^2)$$

$g = O(n\log n)$

Мажорирование:



Асимптотические обозначения

1. Точная оценка.

```
INSERTION-SORT(A)
for j = 2 to A.length
    key = A[j]
    // Вставка A[j] в отсортированную
        последовательность A[1..j − 1].
    i = j − 1
    while i > 0 и A[i] > key
        A[i + 1] = A[i]
        i = i − 1
    A[i + 1] = key
```

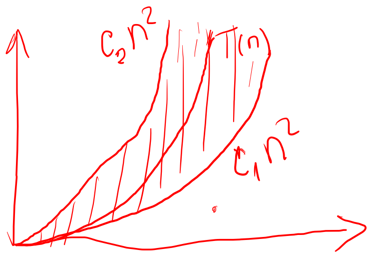
T(n) — время работы алгоритма размера вх. данных

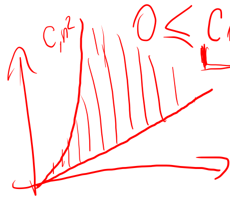
$T(n) = \Theta(h(n))$

$n,\ n^2,\ n\log n,\ n^3$

$\Theta(h(n)) = \begin{cases} T(n) : \exists\, c_1 > 0,\ c_2 > 0,\ n_0 > 0 \\ \forall n > n_0 : \end{cases}$

$0 \le c_1 h(n) \le T(n) \le c_2 h(n)$





$T(n) = \Theta(n^2)$

2. Верхняя оценка.

$O(h(n)) = \begin{cases} T(n),\ \exists\, c > 0,\ n_0 > 0 \\ \forall n > n_0 \quad 0 \le T(n) \le c \cdot h(n) \end{cases}$

$h(n)$ - верхняя асимптотич. оценка

3. Нижняя оценка.

$\Omega(h(n)) = \begin{cases} T(n) : \quad \exists\, c > 0,\ n_0 > 0 \\ \forall n > n_0 \quad 0 \le c\,h(n) \le T(n) \end{cases}$

4. Теорема о связи. Θ, Ω, O

$T(n) = \Theta(g(n)) \iff \begin{cases} T(n) = O(g(n)) \\ T(n) = \Omega(g(n)) \end{cases}$

Как проверять предположения об оценках

Пример доказательства, что функция является точной асимптотической оценкой

$$T(n) = \frac{1}{2}n^2 - 3n = \Theta(n^2)$$

ищем $C_1 > 0$, $C_2 > 0$, n_0

$$0 \le C_1 \cdot n^2 \le \frac{1}{2}n^2 - 3n \le C_2 \cdot n^2$$

$$C_1 \le \frac{1}{2} - \frac{3}{n} \le C_2$$

$$n \uparrow \Rightarrow \frac{1}{2} - \frac{3}{n} \uparrow$$

$\frac{1}{2} - \frac{3}{n} > 0$ верно при $n \ge 7$

$\frac{1}{2} - \frac{3}{n} \ge \frac{1}{14}$ при $n \ge 7$ $\quad C_1 = \frac{1}{14}$

при $n \ge 1$ $\quad C_2 = \frac{1}{2}$

$C_1 = \frac{1}{14}$ $\quad C_2 = \frac{1}{2}$ $\quad n_0 = \max(7, 1) = 7$

Свойства O

$$\exists\, T_1(n) = O(g_1(n)) \qquad T_2(n) = O(g_2(n))$$

1) Правило сумм $T_1(n) + T_2(n) = O(\max(g_1(n), g_2(n)))$

2) Правило произведений $T_1(n) \cdot T_2(n) = O(g_1(n) \cdot g_2(n))$

3) Умножение на константу $c\, T_1(n) = O(g_1(n))$

4) Прибавление константы $T_1(n) + c = O(g_1(n)$

Док-во:

$$T_1(n) = O(g_1(n)) \Rightarrow \exists\, c_1, n_1 : \ \forall n > n_1 \quad T_1(n) \le c_1 \cdot g_1(n)$$ по опр

$$T_2(n) = O(g_2(n)) \Rightarrow \exists\, c_2, n_2 : \ \forall n > n_2 \quad T_2(n) \le c_2 \cdot g_2(n)$$

Надо найти c_3 и n_3

$$\forall n > n_3 \quad (T_1 + T_2)(n) \le c_3 \cdot \max(g_1(n), g_2(n))$$

$$\forall n_3 > \max(n_1, n_2) \quad T_1(n) + T_2(n) \le c_1 g_1(n) + c_2 g_2(n) \le$$

$$\le c_1 \cdot \max(g_1(n), g_2(n)) + c_2 \cdot \max(g_1(n), g_2(n)) \le$$

$$\le \underbrace{(c_1 + c_2)}_{= c_3} \max(g_1(n), g_2(n))$$

$$n_3 = \max(n_1, n_2) \qquad c_3 = c_1 + c_2.$$

Корректность сортировки вставками

Один из вариантов — с помощью инварианта цикла

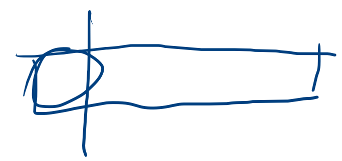
Опр Инвариант цикла — истинное утверждение, которое сохраняется перед и после каждой итерации цикла алгоритма

- Инициализация — верно до начала работы
- Сохранение — верно после каждой итерации
- Завершение — верно после выполнения алгоритма

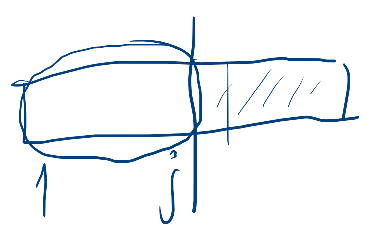
Инвариант: Массив A[1...j-1] содержит те же элементы, что были изначально, но в отсортированном виде

Инициализация:

$j = 2 \quad A[1, j-1] = A[1]$



Сохранение:



Завершение:

$j = n+1$