



Задача сортировки чисел в порядке возрастания:



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n >$



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n > 1$
 - ▶ Выход: перестановка входной последовательности < $a'_1, a'_2, ..., a'_n >$, такая что $a'_1 ≤ a'_2 ≤ ... ≤ a'_n$



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n >$
 - ▶ Выход: перестановка входной последовательности $< a_1', a_2', ..., a_n' >$, такая что $a_1' ≤ a_2' ≤ ... ≤ a_n'$
- Пример экземпляра задачи сортировки:



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n >$
 - ▶ Выход: перестановка входной последовательности $< a_1', a_2', ..., a_n' >$, такая что $a_1' ≤ a_2' ≤ ... ≤ a_n'$
- Пример экземпляра задачи сортировки:
 - ightharpoonup Вход: <10, 8, 2, 12, 5, 100>



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n >$
 - ▶ Выход: перестановка входной последовательности < $a'_1, a'_2, ..., a'_n >$, такая что $a'_1 ≤ a'_2 ≤ ... ≤ a'_n$
- Пример экземпляра задачи сортировки:
 - ightharpoonup Вход: < 10, 8, 2, 12, 5, 100 >
 - ► Выход: < 2, 5, 8, 10, 12, 100 >



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n >$
 - ▶ Выход: перестановка входной последовательности $< a_1', a_2', ..., a_n' >$, такая что $a_1' ≤ a_2' ≤ ... ≤ a_n'$
- Пример экземпляра задачи сортировки:
 - **Вход**: < 10, 8, 2, 12, 5, 100 >
 - ▶ Выход: < 2, 5, 8, 10, 12, 100 >
- В этой лекции будет изучен алгоритм сортировки вставками



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n >$
 - ▶ Выход: перестановка входной последовательности $< a_1', a_2', ..., a_n' >$, такая что $a_1' \le a_2' \le ... \le a_n'$
- Пример экземпляра задачи сортировки:
 - **Вход**: < 10, 8, 2, 12, 5, 100 >
 - ightharpoonup Выход: < 2, 5, 8, 10, 12, 100 >
- В этой лекции будет изучен алгоритм сортировки вставками
- ▶ На следующей неделе другие алгоритмы сортировки, основанные на сравнении



- Задача сортировки чисел в порядке возрастания:
 - ▶ Вход: последовательность из n чисел $< a_1, a_2, ..., a_n >$
 - ▶ Выход: перестановка входной последовательности $< a_1', a_2', ..., a_n' >$, такая что $a_1' ≤ a_2' ≤ ... ≤ a_n'$
- Пример экземпляра задачи сортировки:
 - **Вход**: < 10, 8, 2, 12, 5, 100 >
 - ightharpoonup Выход: < 2, 5, 8, 10, 12, 100 >
- В этой лекции будет изучен алгоритм сортировки вставками
- На следующей неделе другие алгоритмы сортировки, основанные на сравнении
 - Могут быть применены к сортировке любых сравнимых между собой элементов



Пример:

<u>5</u> 2 4 6 1 3

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

```
<u>5</u> 2 4 6 1 3
```

```
1: for j \leftarrow 2 to n do
```

- 2: $i \leftarrow j-1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



```
Пример:
```

```
2 <u>5</u> 4 6 1 3
```

```
1: \mathbf{for} j \leftarrow 2 \mathbf{to} n \mathbf{do}

2: i \leftarrow j - 1

3: \mathbf{while} \ i > 0 \ \text{and} \ \mathbf{A}[i] > \mathbf{A}[i+1] \ \mathbf{do}

4: \mathbf{swap}(A[i], A[i+1])

5: \mathbf{temp} \leftarrow A[i]

6: \mathbf{A}[i] \leftarrow A[i+1]

7: A[i+1] \leftarrow \mathbf{temp}

8: i \leftarrow i-1
```



Пример:

<u>2</u> <u>5</u> 4 6 1 3

```
1: for j \leftarrow 2 to n do
```

- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

<u>2</u> <u>5</u> 4 6 1 3

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

```
<u>2</u> 4 <u>5</u> 6 1 3
```

```
1: for j \leftarrow 2 to n do

2: i \leftarrow j - 1

3: while i > 0 and A[i] > A[i+1] do

4: swap(A[i], A[i+1])

5: i \leftarrow i - 1
```



Пример:

<u>**2**</u> 4 <u>5</u> 6 1 3

```
1: for j \leftarrow 2 to n do
```

- 2: $i \leftarrow j 1$
- 3: while i > 0 and A[i] > A[i+1] do
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

<u>2 4 5</u> 6 1 3

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

<u>2 4 **5** 6</u> 1 3

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

<u>2 4 5 6</u> 1 3

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

<u>1 2 4 5 6</u> 3

```
1: for j \leftarrow 2 to n do
```

- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

1 2 4 5 6 3

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Пример:

1 2 3 4 5 6

```
1: for j \leftarrow 2 to n do
```

- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$



Оценим время однократного выполнения каждой строки псевдокода алгоритма сортировки вставками



Оценим время однократного выполнения каждой строки псевдокода алгоритма сортировки вставками

InsertionSort(A)

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$

Время

 c_1



Оценим время однократного выполнения каждой строки псевдокода алгоритма сортировки вставками

InsertionSort(A)

- 1: for $j \leftarrow 2$ to n do
- 2: $i \leftarrow j 1$
- 3: **while** i > 0 and A[i] > A[i+1] **do**
- 4: swap(A[i], A[i+1])
- 5: $i \leftarrow i 1$

Время

 c_1

 c_2



5: $i \leftarrow i - 1$

Оценка времени работы сортировки вставками

Оценим время однократного выполнения каждой строки псевдокода алгоритма сортировки вставками

InsertionSort(A) Время 1: $\mathbf{for}\,j \leftarrow 2\,\mathbf{to}\,n\,\mathbf{do}$ 2: $i \leftarrow j - 1$ 3: $\mathbf{while}\,i > 0\,\mathbf{and}\,A[i] > A[i+1]\,\mathbf{do}$ 4: $\mathbf{swap}(A[i],A[i+1])$



Оценим время однократного выполнения каждой строки псевдокода алгоритма сортировки вставками

Inse	rtionSort(A)	Время
1: 1	for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$
2:	$i \leftarrow j-1$	$oldsymbol{c}_2$
3:	while $i > 0$ and $A[i] > A[i+1]$ do	c_3
4:	swap(A[i],A[i+1])	$\textit{\textbf{c}}_4$
5.	$i \leftarrow i - 1$	



Оценим время однократного выполнения каждой строки псевдокода алгоритма сортировки вставками

InsertionSort(A)	Время
1: for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$
2: $i \leftarrow j-1$	$oldsymbol{c}_2$
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3
4: $\operatorname{swap}(A[i], A[i+1])$	$oldsymbol{c}_4$
5: $i \leftarrow i - 1$	c_5



Оценим общее число выполнений каждой строки

InsertionSort(A)	Время	Число раз
1: $\mathbf{for} j \leftarrow 2 \mathbf{\textit{to}} n \mathbf{do}$	$oldsymbol{c}_1$	n
2: $i \leftarrow j - 1$	c_2	
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3	
4: $\operatorname{swap}(A[i], A[i+1])$	$\textit{\textbf{c}}_4$	
5: $i \leftarrow i - 1$	C 5	



Оценим общее число выполнений каждой строки

InsertionSort(A)		Время	Число раз
1: f	or $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n n — 1
2:	$i \leftarrow j-1$	$oldsymbol{c}_2$	H = 1
3:	while $i > 0$ and $A[i] > A[i+1]$ do	c_3	
4:	$swap(\boldsymbol{A}[\boldsymbol{i}], \boldsymbol{A}[\boldsymbol{i}+1])$	c_4	
5:	$i \leftarrow i - 1$	C 5	



Оценим общее число выполнений каждой строки t_j — число проверок условия while на j-ой итерации цикла for

InsertionSort(A)	Время	Число раз
1: $\mathbf{for} j \leftarrow 2 \mathbf{to} n \mathbf{do}$	$oldsymbol{c}_1$	$egin{array}{c} oldsymbol{n} \ oldsymbol{n} - 1 \end{array}$
2: $i \leftarrow j - 1$	c_2	H - 1
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3	
4: $\operatorname{swap}(A[i], A[i+1])$	\boldsymbol{c}_4	
5: $i \leftarrow i - 1$	c_5	



Оценим общее число выполнений каждой строки t_j — число проверок условия while на j-ой итерации цикла for

InsertionSort(A)	Время	Число раз
1: $\mathbf{for} j \leftarrow 2 \mathbf{to} n \mathbf{do}$	$oldsymbol{c}_1$	$oldsymbol{n} - 1$
$2: i \leftarrow j-1$	c_2	$\sum_{j=2}^{n} t_j$
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\angle j=2$ 4
4: $\operatorname{swap}(A[i], A[i+1])$	\boldsymbol{c}_4	
5: $i \leftarrow i - 1$	c_5	



Оценим общее число выполнений каждой строки t_j — число проверок условия while на j-ой итерации цикла for

InsertionSort(A)		Время	Число раз
1: 1	for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n
	$i \leftarrow j-1$	\boldsymbol{c}_2	n-1
3:	while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4:	swap(A[i],A[i+1])	$\textit{\textbf{c}}_4$	$\sum_{j=2}^{n} (t_j - 1)$
5:	$i \leftarrow i - 1$	C ₅	



Оценим общее число выполнений каждой строки t_i — число проверок условия while на j-ой итерации цикла for

InsertionSort(A)		Время	Число раз
1: f c	or $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n
	$i \leftarrow j - 1$	$oldsymbol{c}_2$	n-1
3:	while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4:	swap(A[i], A[i+1])	\boldsymbol{c}_4	$\sum_{j=2}^{n} (t_j - 1)$
5:	$i \leftarrow i - 1$	$oldsymbol{c}_5$	$\sum_{i=2}^{n} (t_i - 1)$



Оценим общее число выполнений каждой строки t_i — число проверок условия while на j-ой итерации цикла for

InsertionSort(A)		Время	Число раз
1: 1	for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n
	$i \leftarrow j - 1$	${m c}_2$	n-1
3:	while $i > 0$ and $A[i] > A[i+1]$ do	$oldsymbol{c}_3$	$\sum_{j=2}^{n} t_j$
4:	swap(A[i], A[i+1])	\boldsymbol{c}_4	$\sum_{j=2}^{n} (t_j - 1)$
5:	$i \leftarrow i - 1$	c_5	$\sum_{i=2}^{n} (t_i - 1)$



0 - 0	J	-p	P 0. 0 0 . D.	C C P	. –
T(n) =					

InsertionSort(A)	Время	Число раз
1: for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n T
2: $i \leftarrow j - 1$	\boldsymbol{c}_2	n-1
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4: $\operatorname{swap}(A[i], A[i+1])$	${m c}_4$	$\sum_{j=2}^{n} (t_j - 1)$
5: $i \leftarrow i - 1$	c_5	$\sum_{i=2}^{n} (t_i - 1)$



$$T(n) = c_1 n + c_2 n$$

InsertionSort(A)		Время	Число раз
1: f c	or $j \leftarrow 2$ to n do	c_1	<i>n</i>
	$i \leftarrow j - 1$	$oldsymbol{c}_2$	n-1
3:	while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4:	swap(A[i], A[i+1])	\boldsymbol{c}_4	$\sum_{j=2}^{n} (t_j - 1)$
5:	$i \leftarrow i - 1$	c_5	$\sum_{i=2}^{n} (t_i - 1)$



$$T(n) = c_1 n + c_2 (n-1) +$$

InsertionSort(A)	Время	Число раз
1: for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n
2: $i \leftarrow j - 1$	c_2	$\frac{n-1}{\sum_{n=1}^{n}}$
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4: $swap(A[i], A[i+1])$	$oldsymbol{c}_4$	$\sum_{j=2}^{n} (t_j - 1)$
5: $i \leftarrow i - 1$	c_5	$\sum_{i=2}^{n} (t_i - 1)$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_3 \sum_{j=2}^{n}$$

InsertionSort(A)	Время	Число раз
1: for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n
$2: i \leftarrow j-1$	c_2	n-1
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4: $\operatorname{swap}(A[i], A[i+1])$	\boldsymbol{c}_4	$\sum_{j=2}^{n} (t_j - 1)$
5: $i \leftarrow i - 1$	$oldsymbol{c}_5$	$\sum_{i=2}^{n} (t_i - 1)$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_4 \sum_{j=2}^{n} (t_j - 1)$$

InsertionSort(A)		Время	Число раз
1: for $j \leftarrow 2$ to n do		$oldsymbol{c}_1$	n 1
	$i \leftarrow j - 1$	c_2	n-1
3:	while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4:	swap(A[i], A[i+1])	c_4	$\sum_{j=2}^{n} (t_j - 1)$
5:	$i \leftarrow i - 1$	c_5	$\sum_{i=2}^{n} (t_i - 1)$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

InsertionSort(A)	Время	_I Число раз
1: for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n
2: $i \leftarrow j - 1$	$oldsymbol{c}_2$	n-1
3: while $i > 0$ and $A[i] > A[i + 1]$	$oldsymbol{c}_1$ do	$\sum_{j=2}^{n} t_j$
4: $swap(A[i], A[i+1])$	$oldsymbol{c}_4$	$\sum_{j=2}^{n} (t_j - 1)$
5: $i \leftarrow i - 1$	$oldsymbol{c}_5$	$\sum_{i=2}^{n} (t_i - 1)$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

InsertionSort(A)	Время	Число раз
1: for $j \leftarrow 2$ to n do	$oldsymbol{c}_1$	n
2: $i \leftarrow j - 1$	$oldsymbol{c}_2$	n-1
3: while $i > 0$ and $A[i] > A[i+1]$ do	c_3	$\sum_{j=2}^{n} t_j$
4: $\operatorname{swap}(A[i], A[i+1])$	\boldsymbol{c}_4	$\sum_{j=2}^{n} (t_j - 1)$
5: $i \leftarrow i - 1$	c_5	$\sum_{i=2}^{n} (t_i - 1)$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$



- ▶ Время работы сортировки вставками в общем случае :
 - $T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j 1) + c_5 \sum_{j=2}^{n} (t_j 1)$
- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке
 - $t_j = 1$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке
 - $t_j = 1$
 - $T(n) = c_1 n + c_2 (n-1) + c_3 (n-1) = (c_1 + c_2 + c_3) n (c_2 + c_3) = \Theta(n)$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке
 - ▶ $t_i = 1$
 - $T(n) = c_1 n + c_2 (n-1) + c_3 (n-1) = (c_1 + c_2 + c_3) n (c_2 + c_3) = \Theta(n)$
- Худший случай: элементы входной последовательности отсортированы в порядке, обратном требуемому



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке
 - ▶ $t_i = 1$
 - $T(n) = c_1 n + c_2 (n-1) + c_3 (n-1) = (c_1 + c_2 + c_3) n (c_2 + c_3) = \Theta(n)$
- Худший случай: элементы входной последовательности отсортированы в порядке, обратном требуемому
 - $ightharpoonup t_j = j$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке
 - ▶ $t_i = 1$
 - $T(n) = c_1 n + c_2 (n-1) + c_3 (n-1) = (c_1 + c_2 + c_3) n (c_2 + c_3) = \Theta(n)$
- Худший случай: элементы входной последовательности отсортированы в порядке, обратном требуемому
 - $ightharpoonup t_i = j$
 - ▶ По формуле суммы арифметической прогрессии

$$\sum_{j=2}^{n} (j) = \frac{n(n+1)}{2} - 1, \qquad \sum_{j=2}^{n} (j-1) = \frac{n(n-1)}{2}$$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке
 - $t_j = 1$
 - $T(n) = c_1 n + c_2 (n-1) + c_3 (n-1) = (c_1 + c_2 + c_3) n (c_2 + c_3) = \Theta(n)$
- Худший случай: элементы входной последовательности отсортированы в порядке, обратном требуемому
 - $ightharpoonup t_i = j$
 - ▶ По формуле суммы арифметической прогрессии

$$\sum_{i=2}^{n} (j) = \frac{n(n+1)}{2} - 1, \qquad \sum_{i=2}^{n} (j-1) = \frac{n(n-1)}{2}$$

$$T(n) = (\frac{c_3}{2} + \frac{c_4}{2} + \frac{c_5}{2})n^2 + (c_1 + c_2 + \frac{c_3}{2} - \frac{c_4}{2} - \frac{c_5}{2})n - (c_2 + c_3)$$



$$T(n) = c_1 n + c_2 (n-1) + c_3 \sum_{j=2}^{n} t_j + c_4 \sum_{j=2}^{n} (t_j - 1) + c_5 \sum_{j=2}^{n} (t_j - 1)$$

- Лучший случай: элементы входной последовательности отсортированы в требуемом порядке
 - $ightharpoonup t_i = 1$
 - $T(n) = c_1 n + c_2 (n-1) + c_3 (n-1) = (c_1 + c_2 + c_3) n (c_2 + c_3) = \Theta(n)$
- Худший случай: элементы входной последовательности отсортированы в порядке, обратном требуемому
 - $t_i = j$
 - ▶ По формуле суммы арифметической прогрессии $\sum_{i=2}^{n} (j) = \frac{n(n+1)}{2} 1, \qquad \sum_{i=2}^{n} (j-1) = \frac{n(n-1)}{2}$
 - $T(n) = (\frac{c_3}{2} + \frac{c_4}{2} + \frac{c_5}{2})n^2 + (c_1 + c_2 + \frac{c_3}{2} \frac{c_4}{2} \frac{c_5}{2})n (c_2 + c_3)$
- ▶ В общем случае, время работы сортировки вставками оценивается сверху как $O(n^2)$



При оценке времени работы алгоритмов часто используется оценка сверху



- При оценке времени работы алгоритмов часто используется оценка сверху
 - ▶ Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки



- При оценке времени работы алгоритмов часто используется оценка сверху
 - ▶ Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - В некоторых алгоритмах худший случай достигается часто



- При оценке времени работы алгоритмов часто используется оценка сверху
 - Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - ▶ В некоторых алгоритмах худший случай достигается часто
 - ▶ Среднее время работы алгоритмов во многих случаях не лучше, чем худшее



- При оценке времени работы алгоритмов часто используется оценка сверху
 - Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - В некоторых алгоритмах худший случай достигается часто
 - ▶ Среднее время работы алгоритмов во многих случаях не лучше, чем худшее

InsertionSort(A)

```
1: for j \leftarrow 2 to n do
2: i \leftarrow j - 1
3: while i > 0 and A[i] > A[i+1] do
4: swap(A[i], A[i+1])
5: i \leftarrow i - 1
6: A[i+1] \leftarrow key
```



- При оценке времени работы алгоритмов часто используется оценка сверху
 - Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - В некоторых алгоритмах худший случай достигается часто
 - ▶ Среднее время работы алгоритмов во многих случаях не лучше, чем худшее

```
InsertionSort(A) T_{for}(n) = O(n)

1: for j \leftarrow 2 to n do

2: i \leftarrow j - 1

3: while i > 0 and A[i] > A[i + 1] do

4: swap(A[i], A[i + 1])

5: i \leftarrow i - 1

6: A[i + 1] \leftarrow key
```



- При оценке времени работы алгоритмов часто используется оценка сверху
 - Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - В некоторых алгоритмах худший случай достигается часто
 - ▶ Среднее время работы алгоритмов во многих случаях не лучше, чем худшее

```
InsertionSort(A) T_{\text{for}}(n) = O(n)
1: for j \leftarrow 2 to n do
2: i \leftarrow j - 1
3: while i > 0 and A[i] > A[i+1] do
4: swap(A[i], A[i+1])
5: i \leftarrow i - 1
6: A[i+1] \leftarrow key
```



- При оценке времени работы алгоритмов часто используется оценка сверху
 - Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - В некоторых алгоритмах худший случай достигается часто
 - ▶ Среднее время работы алгоритмов во многих случаях не лучше, чем худшее

```
InsertionSort(A)
```

```
1: for j \leftarrow 2 to n do
2: i \leftarrow j - 1
3: while i > 0 and A[i] > A[i+1] do
4: swap(A[i], A[i+1])
5: i \leftarrow i - 1
6: A[i+1] \leftarrow key
```

```
T_{	ext{for}}(n) = O(n)
T_{	ext{while}}(n) = O(n)
T(n) = T_{	ext{for}}(n) \times T_{	ext{while}}(n) = O(n^2)
```



- При оценке времени работы алгоритмов часто используется оценка сверху
 - Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - В некоторых алгоритмах худший случай достигается часто
 - ▶ Среднее время работы алгоритмов во многих случаях не лучше, чем худшее

InsertionSort(A)

```
1: for j \leftarrow 2 to n do
```

2:
$$i \leftarrow j - 1$$

3: **while**
$$i > 0$$
 and $A[i] > A[i+1]$ **do**

4:
$$swap(A[i], A[i+1])$$

5:
$$i \leftarrow i - 1$$

6:
$$A[i+1] \leftarrow key$$

$$egin{aligned} T_{\mathsf{for}}(n) &= \mathsf{O}(n) \ T_{\mathsf{while}}(n) &= \mathsf{O}(n) \ T(n) &= T_{\mathsf{for}}(n) imes T_{\mathsf{while}}(n) = \mathsf{O}(n^2) \end{aligned}$$



- При оценке времени работы алгоритмов часто используется оценка сверху
 - ▶ Верхняя оценка дает гарантии на время выполнения алгоритма, не требуя уточнения условий выполнения оценки
 - В некоторых алгоритмах худший случай достигается часто
 - ▶ Среднее время работы алгоритмов во многих случаях не лучше, чем худшее

```
InsertionSort(A)
```

```
1: for j \leftarrow 2 to n do 2: i \leftarrow i - 1
```

3: **while**
$$i > 0$$
 and $A[i] > A[i+1]$ **do**

4:
$$swap(A[i], A[i+1])$$

5: $i \leftarrow i - 1$

6: $A[i+1] \leftarrow key$

$$egin{aligned} T_{\mathsf{for}}(n) &= O(n) \ T_{\mathsf{while}}(n) &= O(n) \ T(n) &= T_{\mathsf{for}}(n) imes T_{\mathsf{while}}(n) = O(n^2) \end{aligned}$$

Спасибо за внимание!