Примерни решения на домашно № 1, 20 март 2021 г.

Задача 1: Докажете или опровергайте, че при $n, m \in \mathbb{N}^+$ и $n \geq m$ е в сила

$$\binom{n}{m} = O\left(\frac{n^m}{m^2}\right)$$

Решение: Твърдението е вярно. На лекции видяхме дефиниция на Тита-нотация от функция на две променливи:

$$\Theta(g(n,m)) \stackrel{\text{def}}{=} \big\{ f(n,m) \mid \exists c_1, c_2 > 0\ \exists n_0, m_0\ \forall n \geq n_0\ \forall m \geq m_0 : \\ 0 \leq c_1 \cdot g(n,m) \leq f(n,m) \leq c_2 \cdot g(n,m) \big\}$$

Очевидният смислен начин за дефиниране на О-голямо от функция на две променливи е:

$$O(g(n,m)) \stackrel{\text{def}}{=} \big\{ f(n,m) \mid \exists c > 0\ \exists n_0, m_0\ \forall n \geq n_0\ \forall m \geq m_0 : \\ 0 \leq f(n,m) \leq c \cdot g(n,m) \big\}$$

Имайки предвид и ограничението $n \geq m$, ясно е, че $\binom{n}{m} = O\left(\frac{n^m}{m^2}\right)$ е кратък запис за

$$\exists c > 0\ \exists n_0, m_0\ \forall n \geq n_0\ \forall m \geq m_0 : n \geq m\ \rightarrow\ 0 \leq \binom{n}{m} \leq c \cdot \frac{n^m}{m^2} \tag{1}$$

Лявото неравенство $0 \leq \binom{n}{m}$ е очевидно, понеже биномният коефициент е положителен, ако долният индекс не надхвърля горния. Да разгледаме $\binom{n}{m} \leq c \cdot \frac{n^m}{m^2}$. Твърди се, че има положителна константа c и стойности m_0 и n_0 съответно на m и n, такива че за всички стойности отвъд тези, за които $n \geq m$, е вярно, че

$$\frac{n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot (n-m+1)}{m!} \leq c \cdot \frac{n^m}{m^2} \tag{2}$$

Наистина, ако вземем $c = 1$ и $n_0 = m_0 = 4$, неравенство (2) е сила, понеже

- произведението от m множителя $n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot (n-m+1)$ от числителя вляво е по-малко от произведението от m множителя $\underbrace{n \cdot n \cdot \dots \cdot n}_{m}$ от числителя вдясно,
- а $m!$ от знаменателя вляво е по-голямо от m^2 от знаменателя вдясно.

Задача 2: Подредете по асимптотично нарастване следните 13 функции:

$$f_1(n) = n^n \qquad f_2(n) = \binom{3n}{2n} \qquad f_3(n) = \ln n \qquad f_4(n) = \sum_{k=1}^{\infty} \frac{1}{k!}$$

$$f_5(n) = \sum_{k=0}^{n} \binom{n}{k}^2 \qquad f_6(n) = 3 + \cos n \qquad f_7(n) = (\lg \lg n)^{\lg \lg n} \qquad f_8(n) = n + 5 \sin n$$

$$f_9(n) = \sum_{k=2}^{n} \frac{1}{k} \qquad f_{10}(n) = n^{3n} \qquad f_{11}(n) = \sum_{k=2}^{100} \frac{1}{k} \qquad f_{12}(n) = \sum_{k=1}^{n^2} k$$

$$f_{13}(n) = (\sqrt{n!})^{\sqrt{n!}}$$

Решение: Да разгледаме f_4. Известно е, че редът $\sum_{k=1}^{\infty} \frac{1}{k!}$ е сходящ. Тогава $f_4(n) = \Theta(1)$. Сега да разгледаме f_6. Тъй като $-1 \leq \cos n \leq 1$, вярно е, че $2 \leq f_6(n) \leq 4$. Но тогава $f_6(n) = \Theta(1)$. Сега да разгледаме f_{11}. Сумата $\sum_{k=2}^{100} \frac{1}{k}$ е константа, така че $f_{11}(n) = \Theta(1)$. Тогава $f_4(n) \asymp f_6(n) \asymp f_{11}(n)$. Тези три функции са първият клас на еквивалентност.

Да разгледаме f_9. Съгласно казаното на лекции, $f_9(n) \asymp \lg n$. Да сравним f_9 с f_3. Тъй като $\ln n \asymp \lg n$, виждаме, че $f_3(n) \asymp f_9(n)$. Тези две функции са вторият клас на еквивалентност.

Очевидно е, че $\Theta(1) \prec \lg n$. Следователно,

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n)$$

Да сравним f_3 с f_7. Твърдим, че $f_3(n) \prec f_7(n)$. За да покажем това, логаритмуваме f_3 и f_7. Очевидно, $\lg f_3(n) \asymp \lg \lg n$ и $\lg f_7(n) = \lg \left((\lg \lg n)^{\lg \lg n} \right)$, а $\lg \left((\lg \lg n)^{\lg \lg n} \right) = (\lg \lg n)(\lg \lg \lg n)$. Тъй като $\lg \lg \lg n$ е неограничено растяща, вярно е, че $\lg \lg n \prec (\lg \lg n)(\lg \lg \lg n)$. Съгласно изучаваното на лекции, от това, че образът на f_3 при логаритмична трансформация расте по-бавно, в асимптотичния смисъл, от образа на f_7 при логаритмична трансформация, следва, че $f_3(n) \prec f_7(n)$. Дотук получихме

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n)$$

Да сравним f_7 с f_8. Твърдим, че $f_7(n) \prec f_8(n)$. Наистина, тъй като $-1 \leq \sin n \leq 1$, вярно е, че $n - 5 \leq f_8(n) \leq n + 5$. Тогава очевидно $f_8(n) \asymp n$. Твърдим, че

$$(\lg \lg n)^{\lg \lg n} \prec n \tag{3}$$

Ако логаритмуваме (1), вляво получаваме $(\lg \lg n)(\lg \lg \lg n)$, а вдясно получаваме $\lg n$. Но $(\lg \lg n)(\lg \lg \lg n) \prec (\lg \lg n)(\lg \lg n) = (\lg \lg n)^2$, а $(\lg \lg n)^2 \prec \lg n$ по същата причина, по която $(\lg n)^2 \prec n$: всяка полилогаритмична функция расте по-бавно, в асимптотичния смисъл, от всяка полиномиална. Показахме, че след логаритмуване на (3), лявата страна расте по-бавно, в асимптотичния смисъл, от дясната. Тогава, съгласно изучаваното на лекции, лявата страна на (3) расте по-бавно от дясната, в асимптотичния смисъл. Дотук получихме

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n) \prec f_8(n)$$

Да сравним f_8 с f_{12}. Да разгледаме $f_{12}(n)$. Известно е, че сумата $\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$. Замествайки n с n^2, получаваме $\sum_{k=1}^{n^2} k = \frac{n^2(n^2+1)}{2} \asymp n^4$. Тогава $f_{12}(n) \asymp n^4$. Както вече видяхме, $f_8(n) \asymp n$. Веднага следва, че $f_8(n) \prec f_{12}(n)$. Дотук получихме

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n) \prec f_8(n) \prec f_{12}(n)$$

Да сравним f_{12} с f_5. От курса Дискретни Структури знаем, че $\sum_{k=0}^{n} \binom{n}{k}^2 = \binom{2n}{n}$, така че $f_5(n) = \binom{2n}{n}$. На лекции видяхме, че $\binom{n}{n/2} \asymp \frac{1}{\sqrt{n}} 2^n$, така че $\binom{2n}{n} \asymp \frac{1}{\sqrt{n}} 2^{2n} = \frac{1}{\sqrt{n}} 4^n$. Ерго, $f_5(n) \asymp \frac{1}{\sqrt{n}} 4^n$. Но функцията $\frac{1}{\sqrt{n}} 4^n$ расте по-бързо, в асимптотичния смисъл, от всяка полиномиална функция. Имайки предвид, че $f_{12}(n) \asymp n^4$, веднага следва, че $f_{12}(n) \prec f_5(n)$. Дотук получихме

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n) \prec f_8(n) \prec f_{12}(n) \prec f_5(n)$$

Да сравним f_5 с f_2. Както вече видяхме, $f_5(n) \asymp \frac{1}{\sqrt{n}} 4^n$. Да разгледаме $f_2(n)$.

$$f_2(n) = \binom{3n}{2n} = \frac{(3n)!}{(2n)!n!} \asymp \frac{\sqrt{n}\frac{(3n)^{3n}}{e^{3n}}}{\sqrt{n}\frac{(2n)^{2n}}{e^{2n}} \cdot \sqrt{n}\frac{n^n}{e^n}} = \frac{1}{\sqrt{n}} \frac{3^{3n} n^{3n}}{2^{2n} n^{2n} n^n} = \frac{1}{\sqrt{n}} \frac{3^{3n}}{2^{2n}} =$$
$$\frac{1}{\sqrt{n}} 3^n \left(\frac{3}{2}\right)^{2n} = \frac{1}{\sqrt{n}} 3^n \left(\frac{9}{4}\right)^n = \frac{1}{\sqrt{n}} \left(\frac{27}{4}\right)^n$$

И така, $f_2(n) \asymp \frac{1}{\sqrt{n}} \left(\frac{27}{4}\right)^n$. Тъй като $\frac{27}{4} > 4$, веднага следва, че $f_5(n) \prec f_2(n)$. Дотук получихме

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n) \prec f_8(n) \prec f_{12}(n) \prec f_5(n) \prec f_2(n)$$

Да сравним f_2 с f_1, имайки предвид, че $f_2(n) \asymp \frac{1}{\sqrt{n}} \left(\frac{27}{4}\right)^n$ и $f_1(n) = n^n$. При логаритмуване се получава съответно $n \lg \left(\frac{27}{4}\right) - \lg \sqrt{n}$ и $n \lg n$. Но $n \lg \left(\frac{27}{4}\right) - \lg \sqrt{n} \prec n \lg n$, така че $f_2(n) \prec f_1(n)$. Дотук получихме

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n) \prec f_8(n) \prec f_{12}(n) \prec f_5(n) \prec f_2(n) \prec f_1(n)$$

Да сравним f_1 с f_{10}. Твърдим, че $\lim_{n\to\infty} \frac{f_{10}(n)}{f_1(n)} = \infty$. Наистина,

$$\lim_{n\to\infty} \frac{n^{3n}}{n^n} = \lim_{n\to\infty} n^{2n} = \infty$$

Тогава $f_1(n) \prec f_{10}(n)$. Дотук получихме

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n) \prec f_8(n) \prec f_{12}(n) \prec f_5(n) \prec f_2(n) \prec f_1(n) \prec f_{10}(n)$$

Да сравним f_{10} с f_{13}. Логаритмуваме и двете функции и сравняваме съответните образи $\lg\left(n^{3n}\right)$ и $\lg\left(\left(\sqrt{n!}\right)^{\sqrt{n!}}\right)$:

$$3n \lg n \quad \text{vs} \quad \sqrt{n!} \lg \sqrt{n!} \qquad \leftrightarrow \qquad 3 \cdot n \lg n \quad \text{vs} \quad \sqrt{n!} \cdot \frac{1}{2} \cdot n \lg n$$

Очевидно $3 \prec \frac{1}{2}\sqrt{n!}$, така че $\lg f_{10}(n) \prec \lg f_{13}(n)$, така че $f_{10}(n) \prec f_{13}(n)$. Получихме крайната наредба

$$f_4(n) \asymp f_6(n) \asymp f_{11}(n) \prec f_3(n) \asymp f_9(n) \prec f_7(n) \prec f_8(n) \prec f_{12}(n) \prec f_5(n) \prec f_2(n) \prec f_1(n) \prec f_{10}(n) \prec f_{13}(n)$$

Задача 3: Разгледайте функцията `foo`, написана на `C`.

```
1  int foo(int a, int b) {
2    int s,t;
3
4    a = abs(a);
5    b = abs(b);
6
7    s = 0;
8    t = a;
9
10   do {
11     s = s + b;
12     t --;
13   }
14   while(t > 0);
15
16   return s;
17 }
```

foo.c

5 т. • Какво връща тя?

25 т. • Докажете това колкото можете по-формално и прецизно.

Решение: Програмата връща

- $|\mathtt{b}|$, ако $\mathtt{a} = 0$,
- $|\mathtt{a} \cdot \mathtt{b}|$, ако $\mathtt{a} \neq 0$.

Ще направим доказателство за коректност. То ще ползва "`a`" и "`b`". За да няма двусмислици, казваме явно, че под "`a`" и "`b`" разбираме стойностите им <u>от входа</u>. Първо допускаме, че $\mathtt{a} = 0$. Тогава `a` остава 0 след ред 4. Тялото на цикъла се изпълнява точно веднъж. Заради присвояванията на редове 7 и 8, при влизането в цикъла,

$\mathtt{s} = 0$ и $\mathtt{t} = \mathtt{a}$. На ред 11, в $\mathtt{s}$ се записва $0 + |\mathtt{b}| = |\mathtt{b}|$. На ред 12, $\mathtt{t}$ получава стойност -1, цикълът не се изпълнява повече и алгоритъмт връща $\mathtt{s} = |\mathtt{b}|$. ✓

Сега допускаме, че $\mathtt{a} \neq 0$. Следното твърдение е инварианта на цикъла на редове 10–14:

> При всяко достигане на ред 14:
>
> $$\mathtt{s} = (|\mathtt{a}| - \mathtt{t})|\mathtt{b}|$$

База: При първото достигане на ред 14 е вярно, че $\mathtt{s} = |\mathtt{b}|$, а $\mathtt{t} = |\mathtt{a}| - 1$. Тъй като $\mathtt{a} \neq 0$ и $\mathtt{a}$ е цяло число, вярно е, че $|\mathtt{a}| \geq 1$. Тогава $\mathtt{t} = |\mathtt{a}| - 1$, така че изразът $\boxed{(|\mathtt{a}| - \mathtt{t})|\mathtt{b}|}$ от инвариантата е $(|\mathtt{a}| - (|\mathtt{a}| - 1))|\mathtt{b}|$, което е $|\mathtt{b}|$. ✓

Поддръжка: Да разгледаме някое достигане на ред 14, което не е последното. Допускаме, че $\mathtt{s} = (|\mathtt{a}| - \mathtt{t})|\mathtt{b}|$. Следва изпълнение на ред 11, при което $\mathtt{s}$ вече съдържа $(|\mathtt{a}| - \mathtt{t})|\mathtt{b}| + |\mathtt{b}| = (|\mathtt{a}| - \mathtt{t} + 1)|\mathtt{b}| = (|\mathtt{a}| - (\mathtt{t} - 1))|\mathtt{b}|$. След това $\mathtt{t}$ се декрементира на ред 12. По отношение на новата стойност на $\mathtt{t}$ е вярно, че $\mathtt{s} = (|\mathtt{a}| - \mathtt{t})|\mathtt{b}|$. И така, при следващото достигане на ред 14, инвариантата остава в сила.

Терминация: При последното достигане на ред 14 е вярно, че $\mathtt{t} \leq 0$. Тъй като $\mathtt{t}$ бе инициализирана с цяла положителна стойност, а именно $|\mathtt{a}|$, и се декрементира с единица при всяко изпълнение на тялото на цикъла, очевидно е, че $\mathtt{t} = 0$ в този момент. Заместваайки $\mathtt{t}$ с 0 в инвариантата, получаваме, че $\mathtt{s} = |\mathtt{a}| \cdot |\mathtt{b}|$, което е същото като $\mathtt{s} = |\mathtt{a} \cdot \mathtt{b}|$. Заключаваме, че на ред 16 програмата връща $|\mathtt{a} \cdot \mathtt{b}|$, което и трябваше да покажем. ✓

Задача 4: Алгоритъмът Insertion Sort, който разгледахме на лекции, във вътрешния си цикъл отмества подмасива на $A[1, \ldots, i]$, състоящ се от елементите, по-големи от key, на една позиция вдясно, по този начин освобождавайки място за слагане на key на правилното му място.

6 т.
- Напишете на псевдокод друга версия на алгоритъма със същия външен цикъл, но друг вътрешен цикъл. Тази модификация на Insertion Sort не трябва да използва променлива, за да съхранява временно елемента, който е на i-та позиция в началото на изпълнението на външния цикъл, а трябва да слага този елемент на правилното му място в $A[1, \ldots, i]$ само с използване на размени на елементи. С други думи, вътрешният цикъл трябва да работи с примитива $\boxed{\text{swap}(A[x], A[y])}$, където това е функция, работеща във време $O(1)$ и разменяща елементите на позиции x и y в A.

20 т.
- Докажете коректността на тази модифицирана версия на Insertion Sort колкото можете по-формално и прецизно.

Решение: Ето една възможност за псевдокода.

Insertion Sort, swap($A[1, 2, \ldots n]$: array of integers)

```
1 for i ← 2 to n
2     j ← i
3     while j > 1 and A[j − 1] > A[j] do
4         swap(A[j − 1], A[j])
5         j--
```

Ще докажем коректността му. Първо ще докажем нещо за вътрешния цикъл.

Лема 1 *По отношение на едно изпълнение на външния цикъл (редове 1–5) на* Insertion Sort, swap*, ако преди началото на вътрешния цикъл (редове 3–5) наречем с "A'" масива и допуснем, че $A'[1, \ldots, i-1]$ е сортиран и $j = i$, то ефектът от изпълнението на вътрешния цикъл е следният:*

- *текущият $A[1, \ldots, j-1]$ е същият като $A'[1, \ldots, j-1]$ и се състои точно от елементите на $A'[1, \ldots, i-1]$, по-малки или равни на $A'[i]$;*
- *текущият елемент $A[j]$ е същият като $A'[i]$;*
- *текущият $A[j+1, \ldots, i]$ е същият като $A'[j, \ldots, i-1]$ и се състои точно от елементите на $A'[1, \ldots, i-1]$, по-големи от $A'[i]$.*

Доказателство: Следното твърдение е инварианта за вътрешния цикъл.

> По отношение на едно изпълнение на външния цикъл, ако наречем с "A'" масива в началото на изпълнение на външния цикъл, при всяко достигане на ред 3 е вярно, че
>
> - текущият $A[1, \ldots, j-1]$ е същият като $A'[1, \ldots, j-1]$;
> - текущият елемент $A[j]$ е същият като $A'[i]$;
> - текущият $A[j+1, \ldots, i]$ е същият като $A'[j, \ldots, i-1]$ и всеки негов елемент е по-голям от $A'[i]$.

База: При първото достигане, $j = i$ по допускане. Първата част от инвариантата е "текущият $A[1, \ldots, i-1]$ е същият като $A'[1, \ldots, i-1]$", което е тривиално вярно. Втората част от инвариантата е "текущият елемент $A[i]$ е същият като $A'[i]$", което е тривиално вярно. Масивът $A[j+1, \ldots, i]$ е празен при $i = j$, също така и $A'[j, \ldots, i-1]$ е празен при $i = j$, откъдето следва, че третата част от инвариантата е вярна в празния смисъл. ✓

Поддръжка: Да допуснем, че инвариантата е вярна за някое достигане на ред 3, което не е последното. Щом не е последното, то е вярно, че $j > 1$ и $A[j-1] > A[j]$. Да видим как се запазва първата част на инвариантата. След swap-а на ред 4 вече не е вярно, че текущият $A[1, \ldots, j-1]$ е същият като $A'[1, \ldots, j-1]$. Това, което е вярно, е че текущият $A[1, \ldots, j-2]$ е същият като $A'[1, \ldots, j-2]$. Но след декрементацията на j на ред 5, изразено в новата стойност на j, отново е вярно, че $A[1, \ldots, j-1]$ е същият като $A'[1, \ldots, j-1]$.

Да видим как се запазва втората част от инвариантата. След swap-а ред 4 вече не е вярно, че $A[j] = A'[i]$. Това, което е вярно, е че $A[j-1] = A'[i]$. Но след декрементацията на j на ред 5, изразено в новата стойност на j, отново е вярно, че $A[j] = A'[i]$.

Да видим как се запазва третата част от инвариантата. След swap-а ред 4 продължава да е вярно, че текущият $A[j+1, \ldots, i]$ е същият като $A'[j, \ldots, i-1]$ и всеки негов елемент е по-голям от $A'[i]$. Нещо повече, сега вече е вярно, че $A[j, \ldots, i]$ е същият като $A'[j-1, \ldots, i-1]$ и всеки негов елемент е по-голям от $A'[i]$. Но след декрементацията на j на ред 5, изразено в новата стойност на j, отново е вярно, че текущият $A[j+1, \ldots, i]$ е същият като $A'[j, \ldots, i-1]$ и всеки негов елемент е по-голям от $A'[i]$. С това доказателството на поддръжката приключи.

Терминация: При последното достигане на ред 3 е вярно, че $j \leq 1$ или $A[j-1] \leq A[j]$. Първо да допуснем, че $j \leq 1$. Тъй като j се декрементира със стъпка 1, виждаме, че всъщност $j = 1$. Замествайки в инвариантата j с 1, получаваме

- текущият $A[1, \ldots, 0]$ е същият като $A'[1, \ldots, 0]$;
- текущият елемент $A[1]$ е същият като $A'[i]$;
- текущият $A[2, \ldots, i]$ е същият като $A'[1, \ldots, i-1]$ и всеки негов елемент е по-голям от $A'[i]$.

Масивът $A[1, \ldots, 0]$ е празен, така че в празния смисъл е вярно, че се състои точно от елементите на $A'[1, \ldots, i-1]$, по-малки или равни на $A'[i]$. Ерго, първото твърдение на лемата е вярно.

Второто твърдение тук е точно като второто твърдение на лемата, така че и второто твърдение на лемата е вярно.

Да разгледаме третото твърдение тук. Щом $A[2, \ldots, i] = A'[1, \ldots, i-1]$ и всеки елемент на $A[2, \ldots, i]$ е по-голям от $A'[i]$ и $A[1] = A'[i]$, то всички елементи на $A'[1, \ldots, i-1]$, по-големи от $A'[i]$, се намират именно в $A[2, \ldots, i]$. Тогава третото твърдение тук е същото като третото твърдение на лемата.

Сега да допуснем, че $A[j-1] \leq A[j]$. От първото твърдение на инвариантата знаем, че текущият $A[1, \ldots, j-1]$ е същият като $A'[1, \ldots, j-1]$. От второто знаем, че $A[j] = A'[i]$. Щом $A[j-1] \leq A'[i]$ и освен това $A'[1, \ldots, i-1]$ е сортиран, веднага следва, че текущият $A[1, \ldots, j-1]$ се състои точно от елементите на $A'[1, \ldots, i-1]$, по-малки или равни на $A'[i]$. Първото твърдение на лемата е вярно.

Второто твърдение на лемата е същото като второто твърдение на инвариантата.

Това, че текущият $A[j+1, \ldots, i]$ е същият като $A'[j, \ldots, i-1]$ и всеки негов елемент е по-голям от $A'[i]$, знаем от инвариантата. Имайки предвид, че елементите на $A[1, \ldots, j-1]$ са по-малки или равни на $A'[i]$ и $A[j] = A'[i]$, получаваме и това, че текущият $A[j+1, \ldots, i]$ съдържа точно тези елементи от $A'[1, \ldots, i-1]$, които са по-големи от $A'[i]$. И третото твърдение на лемата "излезе". $\square$

Сега сме готови да докажем коректността на алгоритъма.

Теорема 1 *Алгоритъмът* INSERTION SORT, SWAP *е коректен сортиращ алгоритъм.*

Доказателство: Следното твърдение е инварианта за външния цикъл.

> При всяко достигане на ред 1, текущият $A[1, \ldots, i-1]$ се състои точно от елементите на входния $A[1, \ldots, i-1]$, но в сортиран вид.

База: При първото достигане, $i = 2$. Твърдението е тривиално вярно.

Поддръжка: Да допуснем, че изпълнението е на ред 1 и предстои поне още едно изпълнение на тялото на цикъла. На ред 2, j става i. Следва изпълнението на вътрешния цикъл. След приключването му, съгласно Лема 1,

- текущият $A[1, \ldots, j-1]$ е същият като $A'[1, \ldots, j-1]$ и се състои точно от елементите на $A'[1, \ldots, i-1]$, по-малки или равни на $A'[i]$;
- текущият елемент $A[j]$ е същият като $A'[i]$;
- текущият $A[j+1, \ldots, i]$ е същият като $A'[j, \ldots, i-1]$ и се състои точно от елементите на $A'[1, \ldots, i-1]$, по-големи от $A'[i]$.

Но от това следва, че текущият $A[1, \ldots, i]$ се състои от елементите на входния $A[1, \ldots, i]$, само че в сортиран вид. Следва инкрементация на i с единица. Спрямо новата стойност на i е вярно, че текущият $A[1, \ldots, i-1]$ се състои от елементите на входния $A[1, \ldots, i-1]$, само че в сортиран вид. Което е точно инвариантата.

Терминация: При терминиране на алгоритъма, $i = n + 1$. Замествайки i с $n + 1$ в инвариантата, получаваме "текущият $A[1, \ldots, n]$ се състои точно от елементите на входния $A[1, \ldots, n]$, но в сортиран вид." $\square$