Примерни решение на поправителния изпит по ДАА на 21 август 2025 г.

Зад. 1 Има кутия с $n_w \geq 1$ бели камъчета и $n_b \geq 1$ черни камъчета. Двама играчи, да кажем X и Y, играят, редувайки се, по следните правила. Първо играе X. Всеки играч, когато е на ход, изважда на сляпо (тоест, без да гледа в кутията) две камъчета от кутията и тези камъчета повече не участват в играта. Ако извадените камъчета са от един и същи цвят, то в кутията се добавя черно камъче, в противен случай в кутията се добавя бяло камъче. Допуснете, че отстрани има неограничено количество бели и черни камъчета, които може да бъдат слагани в кутията в процеса на играта. Ако и само ако в даден момент в кутията остане точно едно камъче, играта спира.

Преди началото на играта, X и Y знаят числата n_w и n_b и X казва или "черно", или "бяло", имайки предвид цвета на камъчето в кутията в края на играта. Ако играта спре (понеже в кутията има само едно камъче), то, ако X е познал цвета на последното камъче в кутията, то X печели, в противен случай, X губи.

1 т. **а)** Докажете, че играта винаги спира; тоест, за всеки n_w и n_b и за всяка редица от ходове се стига до момент, в който в кутията има само едно камъче.

19 т. **б)** Докажете, че съществува печеливша стратегия за X.

Решение: Това, че играта спира, е почти очевидно. В началото в кутията има $n_w + n_b \geq 2$ камъчета, а на всеки ход от кутията се изваждат две камъчета и се връща едно камъче, тоест, броят на камъчетата намалява точно с едно. Следователно, след $n_w + n_b - 1$ хода, в кутията ще има точно едно камъче и играта ще спре.

Печеливша стратегия за играча, който назовава/познава цвета на последното камъче, има, но тя се базира само на числата n_w и n_b. Забележете, че никой от играчите няма контрол над това, какви камъчета да извади, понеже ваденето е на сляпо.

Ключовото наблюдение е това: след всеки ход, или броят на белите камъчета намалява с две, или се запазва. Наистина,

- ако от кутията бъдат извадени различни по цвят камъчета (което означава, че едното е бяло, а другото е черно), в кутията се добавя бяло камъче, така че броят на белите камъчета в кутията не се променя;
- ако от кутията бъдат извадени две черни камъчета, в кутията се добавя черно камъче, така че броят на белите камъчета в кутията не се променя;
- ако от кутията бъдат извадени две бели камъчета, в кутията се добавя черно камъче, така че броят на белите камъчета в кутията намалява с две.

От това следва, че четността на броя на белите камъчета в кутията не се променя в процеса на играта. Това може да бъде доказано формално с инвариант, но е прекалено очевидно, за да го правим.

Заключаваме, че ако n_w е четно, то последното камъче е черно, в противен случай, последното камъче е бяло.

Стратегията на X е тази: ако n_w е четно, той казва "черно" в началото, в противен случай, той казва "бяло" в началото.

Зад. 2 Нека $G = (V, E)$ е граф. *Съчетание* в G е всяко $E' \subseteq E$, такова че нито две ребра от E' нямат общ край. По правило, задачата Съчетание е оптимизационна (по-точно, максимизационна): да се намери съчетание с максимална мощност. Тук ще я разгледаме като задача за разпознаване и също ще разгледаме нейна ограничена версия, пак във вариант за разпознаване.

Задача Съчетание, накратко задача M:

екземпляр Граф $G = (V, E)$, число k.
въпрос Дали G има съчетание с мощност $\geq k$?

Задача Двуделно Съчетание, накратко задача BM:

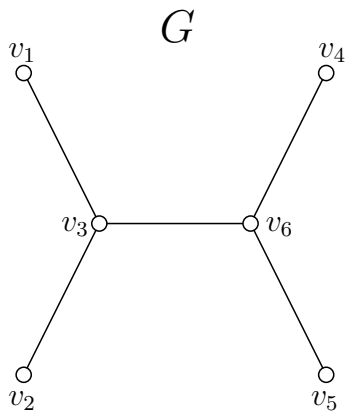
екземпляр Двуделен граф $G = (V, E)$, число k.
въпрос Дали G има съчетание с мощност $\geq k$?

Професор Дълбоков предлага следната полиномиална редукция $M \leq_p BM$. При даден екземпляр $(G = (\{v_1, \ldots, v_n\}, E), k)$ на M конструираме екземпляр $(G' = (V', E'), k')$ на BM така.

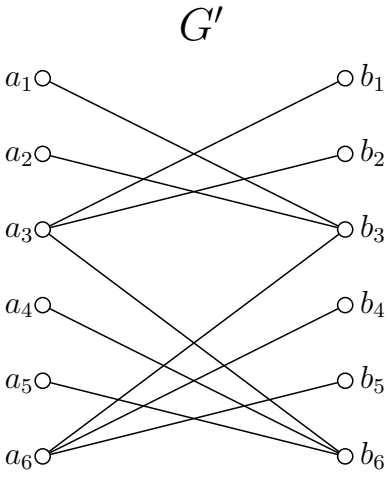
- Конструираме две множества от нови върхове $U_1' = \{a_1, \ldots, a_n\}$ и $U_2' = \{b_1, \ldots, b_n\}$, такива че $U_1' \cap U_2' = \emptyset$. После конструираме $V' = U_1' \cup U_2'$. Множествата U_1' и U_2' са дяловете на двуделния граф G'.
- $E' = \{(a_i, b_j) \mid (v_i, v_j) \in E\}$.
- $k' = k$.

Професорът казва, че редукцията е коректна, понеже конструкцията може да се реализира в полиномиално време. Какво бихте казали за предложената редукция?

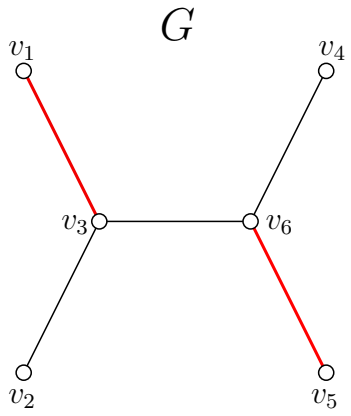
Решение: Редукцията на професора е некоректна. За да бъде коректна, трябва всеки ДА-екземпляр на задачата M да се изобразява в ДА-екземпляр на задачата BM и всеки НЕ-екземпляр на задачата M да се изобразява в НЕ-екземпляр на задачата BM. Но предложената редукция няма това свойство. Ето контрапример. Нека G е този граф:



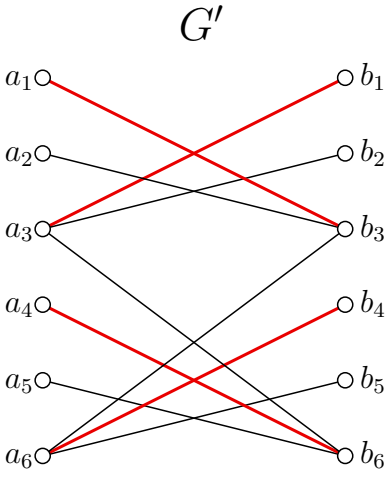
Тогава редукцията конструира този граф G':



Очевидно е, че максималната мощност на съчетание в G е 2:



Следователно, $(G, 4)$ е НЕ-екземпляр на задачата M. Обаче $(G', 4)$ е ДА-екземпляр на задачата BM:



Зад. 3 За числена редица $(a_1, a_2, \ldots, a_k)$ казваме, че е *алтерниращa*, ако

$$\forall i \in \{1, \ldots, k-2\} \ : \text{ или } a_i < a_{i+1} > a_{i+2}, \text{ или } a_i > a_{i+1} < a_{i+2}$$

Дадена е непразна числена редица $X = (x_1, x_2, \ldots, x_n)$. *Поредица в* X е всяка редица $(x_{i_1}, x_{i_2}, \ldots, x_{i_t})$, такава че $1 \leq i_1 < i_2 < \cdots < i_t \leq n$. Задачата е да се намери максималната дължината на алтерниращa поредица в X. Предложете колкото е възможно по-ефикасен алгоритъм, който решава тази задача. Дайте кратка обосновка за коректност и кратък анализ на сложността.

Решение: Задачата прилича на задачата Longest Increasing Sequence, която разгледахме на лекции, и има подобно решение.

Забелязваме, че поредиците с дължина 1 или 2 са алтерниращи съгласно формалната дефиниция на "алтерниращa редица". Причината е, че ако k е дължината на поредицата и $k = 1$ или $k = 2$, то множеството $\{1, \ldots, k-2\}$ е празното множество. Знаем, че всеки предикат е истина, ако променливата му взема стойности от празното множество. Накратко, при $k = 1$ или $k = 2$, всички наредени тройки от съседни елементи имат желаното свойство, понеже такива няма.

Конституентите са непразните префикси на X, а именно (x_1), (x_1, x_2), $\ldots$, $(x_1, x_2, \ldots, x_n)$. На пръв поглед, може да решим задачата, като кажем, че $M(i)$ е оптималната дължина на алтерниращa поредица, завършваща на x_i, и да се опитаме да изчислим $M(i)$ чрез някакви $M(j)$ за $j < i$. Но не е толкова просто! Ако искаме да разширим алтерниращa поредица A чрез добавяне x_i в десния край, не е достатъчно да знаем последния елемент на A и да го сравним с x_i. Има значение дали в A последният елемент е по-голям от предпоследния, или предпоследният е по-голям от последния.

Това съображение ни мотивира да въведем една полезна дефиниция.

Определение 1 *Ако $t \geq 2$, то алтерниращата редица $(x_{i_1}, x_{i_2}, \ldots, x_{i_t})$ е или* качваща*, в случай че $x_{i_{t-1}} < x_{i_t}$, или* слизаща*, в случай че $x_{i_{t-1}} > x_{i_t}$. Ако обаче $t = 1$, редицата е едновременно качваща и слизаща.*

Има смисъл за всеки префикс $(x_1, x_2, \ldots, x_i)$ да изчислим дължината както на максимална качваща поредица, завършваща на x_i, така и дължината на максимална слизаща поредица, завършваща на x_i. Ако разполагаме с тази информация за префиксите (x_1), (x_1, x_2), $\ldots$, $(x_1, x_2, \ldots, x_{i-1})$, не е трудно да извършим изчислението за префикса $(x_1, x_2, \ldots, x_i)$.

Определение 2 *B $(x_1, x_2, \ldots, x_i)$, $M_a(i)$ е дължината на максимална качваща поредица, завършваща на x_i, а $M_d(i)$ е дължината на максимална слизаща поредица, завършваща на x_i.*

Ползвайки Определение 2, рекурсивната декомпозиция е следната:

$$M_a(i) = 1 + \max\{M_d(j) \mid j < i \wedge x_i > x_j\}$$
$$M_d(i) = 1 + \max\{M_a(j) \mid j < i \wedge x_i < x_j\}$$

Ободновката е елементарна:

- ако x_i се явява край на качваща алтерниращa поредица $B = (x_{t_1}, \ldots, x_{t_k}, x_i)$, то нейният префикс $B' = (x_{t_1}, \ldots, x_{t_k})$ се явява слизаща алтерниращa поредица, като освен това $x_{t_k} < x_i$;
- ако x_i се явява край на слизаща алтерниращa поредица $B = (x_{t_1}, \ldots, x_{t_k}, x_i)$, то нейният префикс $B' = (x_{t_1}, \ldots, x_{t_k})$ се явява качваща алтерниращa поредица, като освен това $x_{t_k} > x_i$.

Принципът на оптималността е в сила: във всяка от тези две възможности има смисъл да разгледаме само префикси B' с максимална дължина.

В сила са $M_a(1) = 1$ и $M_d(1) = 1$, понеже (x_1) е едновременно качваща и слизаща поредица, но началните условия на рекурсията **не са** просто $M_a(1) = 1$ и $M_d(1) = 1$. Подобно на решението на

задачата Longest Increasing Sequence, всяко x_i може да се отговаря на начално условие: ако $x_1, \ldots, x_{i-1}$ са по-големи от x_i, то единствената алтернираща качваща редица, завършваща на x_i, е (x_i); аналогично, ако $x_1, \ldots, x_{i-1}$ са по-малки от x_i, то единствената алтернираща слизаща редица, завършваща на x_i, е (x_i). Накратко,

$$\forall i \in \{1, \ldots, n\} : M_a(i) = 1, \text{ ако } x_i = \min\{1, \ldots, i\}$$
$$\forall i \in \{1, \ldots, n\} : M_d(i) = 1, \text{ ако } x_i = \max\{1, \ldots, i\}$$

Следният алгоритъм реализира тази идея. "LAS" идва от Longest Alternating Sequence.

```
LAS((x_1, x_2, ..., x_n): числена редица)
 1  създай масиви M_a[1 .. n], M_d[1 .. n]
 2  for i ← 1 to n
 3      M_a[i] ← 1, M_d[i] ← 1
 4  maxlength ← 1
 5  for i ← 2 to n
 6      for j ← 1 to i − 1
 7          if x_j < x_i and M_a[i] < M_d[j] + 1
 8              M_a[i] ← M_d[j] + 1
 9          if x_j > x_i and M_d[i] < M_a[j] + 1
10              M_d[i] ← M_a[j] + 1
11      maxlength ← max(maxlength, M_a[i], M_d[i])
12  return maxlength
```

Коректността следва директно от коректността на рекурсивната декомпозиция. Сложността по време е квадратична, а сложността по памет е линейна.

LAS е добре известен пример за прилагане на схемата **Динамично Програмиране**. Но този алгоритъм може да се подобри значително както по отношение на сложността по време, така и по отношение на сложността по памет. LAS изчислява и съхранява две стойности **за всяко** x_i, а именно $M_a[i]$ и $M_d[i]$. Подобрението се състои в това, да изчисляваме и съхраняваме само две стойности, да ги наречем asc и desc, по време на **цялата работа** на алгоритъма. Алгоритъмът е следният.

```
LAS-fast((x_1, x_2, ..., x_n): числена редица)
 1  asc ← 1, desc ← 1
 2  for i ← 2 to n
 3      if x_{i−1} < x_i
 4          asc ← desc + 1
 5      if x_{i−1} > x_i
 6          desc ← asc + 1
 7  return max{asc, desc}
```

Коректността на LAS-fast следва (почти) директно от Теорема 1.

Теорема 1 *Нека $n \geq 2$. За всяко $i \in \{2, \ldots, n\}$ е в сила*

$$M_d(i) = M_d(i-1), M_a(i) = M_d(i-1) + 1, \quad \text{ако } x_{i-1} < x_i \tag{1}$$
$$M_a(i) = M_a(i-1), M_d(i) = M_a(i-1) + 1, \quad \text{ако } x_{i-1} > x_i \tag{2}$$
$$M_d(i) = M_d(i-1), M_a(i) = M_a(i-1), \quad \text{ако } x_{i-1} = x_i \tag{3}$$

Доказателство: По индукция по i.

База: Базата е $i = 2$. Знаем, че $M_a(1) = M_d(1) = 1$.

- Ще докажем (1). Ако $x_1 < x_2$, то (x_1, x_2) е растяща редица и максимална слизаща поредица в нея е всяка от (x_1), (x_2), а максималната качваща поредица в нея е (x_1, x_2), така че $M_d(2) = 1$ и $M_a(2) = 2$, така че наистина $M_d(2) = M_d(1)$ и $M_a(2) = M_d(1) + 1$.

- Ще докажем (2). Ако $x_1 > x_2$, то (x_1, x_2) е намаляваща редица и максималната слизаща поредица в нея е (x_1, x_2), а максимална качваща поредица в нея е всяка от (x_1), (x_2), така че $M_d(2) = 2$ и $M_a(2) = 1$, така че наистина $M_a(2) = M_a(1)$ и $M_d(2) = M_a(1) + 1$.

- Ще докажем (3). Ако $x_1 = x_2$, то максимална слизаща поредица е всяка от (x_1), (x_2) и максимална качваща поредица е всяка от (x_1), (x_2), така че $M_d(2) = 1$ и $M_a(2) = 1$, така че наистина $M_d(2) = M_d(1)$ и $M_a(2) = M_a(1)$.

Индуктивно предположение: Нека твърдението е в сила за някое $i \in \{2, \ldots, n-1\}$.

Индуктивна стъпка: Разглеждаме стойност на аргумента $i + 1$.

- Нека $x_i < x_{i+1}$. Ще докажем (1) за стойност на аргумента $i + 1$.

  Очевидно е, че дължината на максималната слизаща алтернираща поредица в $(x_1, \ldots, x_i, x_{i+1})$ е равна на дължината на максималната слизаща алтернираща поредица в $(x_1, \ldots, x_i)$. Ерго, $M_d(i + 1) = M_d(i)$.

  Остава да покажем, че $M_a(i+1) = M_d(i)+1$. Ясно е, че $M_a(i+1) \geq 2$, понеже $x_i < x_{i+1}$. Нека A е максимална качваща поредица в $(x_1, \ldots, x_i, x_{i+1})$. Да кажем, че $A = (x_{t_1}, \ldots, x_{t_k})$. Но $k = M_a(i + 1)$, така че $k \geq 2$. БОО, нека A е максимална качваща поредица, чийто последен елемент е възможно най-вдясно; тоест, t_k е максималният индекс от $\{2, \ldots, i + 1\}$, такъв че x_{t_k} е последният елемент на максимална качваща поредица в $(x_1, \ldots, x_i, x_{i+1})$.

  Ако $t_k = i + 1$, директно заключаваме, че $M_a(i + 1) = M_d(i) + 1$. Да допуснем, че $t_k < i + 1$. В този случай x_{t_k} е по-голям от всеки елемент вдясно от него, тоест, $x_{t_k} > x_{t_k+1}$, $\ldots$, $x_{t_k} > x_i$, $x_{t_k} > x_{i+1}$. Ключовоно наблюдение е, че x_{t_k} не може да е x_i, защото според текущите допскания $x_{t_k} > x_{i+1}$ и $x_{i+1} > x_i$. Ерго, $t_k < i$. Но тогава добавянето на x_i и x_{i+1} в десния край на A дава алтернираща качваща поредица в $(x_1, \ldots, x_i, x_{i+1})$ с дължина $k + 2$, което противоречи на допускането, че A е максимална алтернираща качваща поредица в $(x_1, \ldots, x_i, x_{i+1})$.

- Нека $x_i > x_{i+1}$. Доказваме (2) за стойност на аргумента $i + 1$ по напълно аналогичен начин.

- Нека $x_i = x_{i+1}$. Очевидно, $M_d(i + 1) = M_d(i)$ и $M_a(i + 1) = M_a(i)$, така че (3) е в сила за стойност на аргумента $i + 1$. □

Следствие 1 *При всяко достигане на ред 2 в* LAS-FAST*, asc съдържа дължината на максимална алтернираща качваща поредица в $(x_1, \ldots, x_{i-1})$, а desc съдържа дължината на максимална алтернираща слизаща поредица в $(x_1, \ldots, x_{i-1})$.* □

Заключаваме, че на ред 7, алгоритъмът LAS-FAST връща дължината на максимална алтернираща поредица B в $(x_1, \ldots, x_n)$, тъй като B е или качваща, или слизаща.

Очевидно, сложността по време на LAS-FAST е $\Theta(n)$, а сложността по памет на LAS-FAST е $\Theta(1)$.