

Име: _____, ФН: _____, Спец.: _____, Група: _____

Задача	1	2	3	4	5	6	Общо
получени точки							
максимум точки	20	20	20	30	20	20	130

Забележка: За отлична оценка са достатъчни 100 точки!

Задача 1 Граф $G(V, E)$ има n върха. Някои от ребрата му са ориентирани, други не. Ориентираните ребра в G не образуват цикли.

Можем ли да зададем ориентация на неориентираните ребра, без да се появят ориентирани цикли в G ? Предложете бърз алгоритъм, който задава такава ориентация!

Задача 2 В компютърна мрежа с n възела (компютри, рутери и пр.) има m директни връзки (кабели) между устройствата. Информацията между свързани възли протича мигновено, но престоива в приемащия възел, преди да бъде препредадена към друг възел. За възел i времето за престой е $f(i) \in \mathbb{R}^+$. Предложете бърз алгоритъм, който намира маршрут с най-малко време за обмен на информацията между възли i_0 и j_0 . Приемете, че информацията престоива и в краищата на маршрута.

Задача 3 Ребрата в неориентиран теглови граф са представени с тройки от вида (u, v, w) , където u и v са краищата на реброто, а w е теглото му.

Графът има 6 върха, обозначени с буквите a, b, c, d, e и f , а ребрата му са:

$(a, b, 1), (a, c, 8), (a, e, 8), (a, f, 4), (b, c, 4), (b, d, 2), (b, f, 2), (c, d, 1), (c, e, 8), (d, e, 4), (d, f, 2), (e, f, 1)$

Алгоритъмът на Прим–Ярник започва от връх a и намира минимално покриващо дърво.

(а – 8 т.) Опишете един възможен ред на включване на ребра в дървото.

(б – 4 т.) Изчислете теглото на минималното покриващо дърво.

(в – 8 т.) Ако има различни решения, оценете броя им.

Задача 4 Крадец влиза в картинна галерия с n картини, с цени съответно a_1, a_2, \dots, a_n лева. Картините са подредени последователно и крадецът знае за повреда в алармената система, която се активира, ако бъдат откраднати две съседни картини, но не се активира, ако бъде открадната картина, без да се пипат съседните.

Предложете бърз алгоритъм, изчисляващ цената на най-скъпата колекция, която крадецът може да откъдне безнаказано (20 т.).

Разширете алгоритъма така, че да изчисли кои картини да вземе крадецът (10 т.).

Задача 5 В неориентиран свързан граф $G(V, E)$ алгоритъмът BFS започва обхождане от връх s и построява $2l + 1$ слоя (върховете от i -тия слой са на разстояние i ребра от s).

Докажете, че в G има независимо множество (антиклика) с поне $l + 1$ върха.

Упътване: Използвайте факта, че при обхождане на неориентиран граф с алгоритъма BFS краищата на всяко ребро попадат в съседни слоеве или в един и същ слой.

Задача 6 Докажете, че задачата за разпознаване дали даден граф G_1 съдържа подграф, изоморфен на даден граф G_2 , е NP-пълна.

Упътване: Опитайте се да сведете задачата Клика (CLIQUE) към задачата от условието.

Решения:

Задача 1. Правим топологично сортиране на подграф на $G(V, E)$ със същите върхове, съдържащ само ориентираните ребра от E . Нека в списъка l на топологичната наредба ребрата са насочени надясно. Даваме същата ориентация и на неориентираните ребра:

Ако $\{u, v\}$ е неориентирано ребро и u е преди v в l , ориентацията на реброто ще е от u към v .

Ако $\{u, v\}$ е неориентирано ребро и u е след v в l , ориентацията на реброто ще е от v към u .

Задача 2. Разглеждаме ориентиран граф $G(V, E)$, чиито върхове са възлите на дадената компютърна мрежа. Ако $p = (v_1, v_2, \dots, v_k)$ е път в графа, тогава времето за преминаване на информация по този път (ако броим престоите в началния и крайния възел) е равно на $time(p) = \sum_{i=1}^k f(v_i)$.

За всяка директна връзка между два възела u и v има две ориентирани ребра в графа — едното е (u, v) с тегло $w(u, v) = f(v)$, а другото е (v, u) с тегло $w(v, u) = f(u)$. В така получения теглови граф сумарното тегло на ребрата по пътя p е равно на $w(p) = time(p) - f(v_1)$, тоест $time(p) = f(v_1) + w(p)$.

Ако в графа $G(V, E)$ с теглова функция w пуснем алгоритъма на Дейкстра и намерим най-кратък път p_0 от връх i_0 до връх j_0 , по него ще се достига и минималното време за пренос на информация. То ще е равно на $time(p_0) = f(i_0) + w(p_0)$.

Задача 3. За решаването на задачата е добре да си начертаем графа.

(а) $(a, b, 1), (b, d, 2), (c, d, 1), (b, f, 2), (e, f, 1)$.

(б) Теглото на минималното покриващо дърво е 7.

(в) Има всичко три минимални покриващи дървета, които се различават по начина на избор на ребрата с тегло 2:

$(a, b, 1), (b, d, 2), (c, d, 1), (b, f, 2), (e, f, 1)$;

$(a, b, 1), (b, d, 2), (c, d, 1), (d, f, 2), (e, f, 1)$;

$(a, b, 1), (b, f, 2), (c, d, 1), (d, f, 2), (e, f, 1)$.

Задача 4 се решава с помощта на *динамично програмиране*:

$THIEF(a[1 \dots n]$: array of positive integers)

1 $dyn[1 \dots n]$: array of non-negative integers;

2 // $dyn[k]$ = максималната печалба, която крадецът щеше да получи,

3 // ако в галерията бяха само първите k картини.

4 $dyn[1] \leftarrow a[1]$

5 $dyn[2] \leftarrow \max(a[1], a[2])$

6 **for** $k \leftarrow 3$ **to** n

7 $dyn[k] \leftarrow \max(dyn[k-1], dyn[k-2] + a[k])$

8 **return** $dyn[n]$

Идея на алгоритъма: При изчисляването на $dyn[k]$ има две възможности: крадецът да вземе или да не вземе k -ата картина. Ако той реши да не вземе k -ата картина, тогава максималната му печалба е колкото от първите $k-1$ картини, т.е. $dyn[k-1]$. Ако крадецът вземе k -ата картина, то максималната му печалба е колкото от първите $k-2$ картини, т.е. $dyn[k-2]$, плюс стойността на k -ата картина, т.е. $a[k]$. Тук имаме $k-2$, а не $k-1$, тъй като, щом крадецът е решил да вземе k -ата картина, той трябва да се откаже от $(k-1)$ -ата.

Количеството допълнителна памет, използвано от алгоритъма, може да се оптимизира: елементите на масива dyn не са нужни през цялото време на изпълнение на алгоритъма; достатъчно е да пазим три последователни елемента, а именно $dyn[k-2]$, $dyn[k-1]$ и $dyn[k]$. Това намалява на порядък сложността по памет.

Ако допълнително поискаме от алгоритъма да намира оптималната колекция от картини, а не само максималната възможна печалба, то трябва да внесем промени в кода:

```
THIEF( $a[1 \dots n]$ : array of positive integers)
1   $dyn[1 \dots n]$ : array of non-negative integers;
2  //  $dyn[k]$  = максималната печалба, която крадецът щеше да получи,
3  //      ако в галерията бяха само първите  $k$  картини.
4   $taken[1 \dots n]$ : array of boolean;
5  //  $taken[k] = \text{true} \Leftrightarrow k$ -тата картина би трябвало да бъде взета,
6  //      ако в галерията бяха само първите  $k$  картини.
7   $dyn[1] \leftarrow a[1]$ 
8   $taken[1] \leftarrow \text{true}$ 
9   $dyn[2] \leftarrow \max(a[1], a[2])$ 
10  $taken[2] \leftarrow (a[2] > a[1])$ 
11 for  $k \leftarrow 3$  to  $n$ 
12      $dyn[k] \leftarrow \max(dyn[k-1], dyn[k-2] + a[k])$ 
13      $taken[k] \leftarrow (dyn[k-2] + a[k] > dyn[k-1])$ 
14  $k \leftarrow n$ 
15 while  $k > 0$  do
16     if  $taken[k]$ 
17         print  $k$  // Крадецът взема  $k$ -тата картина.
18          $k \leftarrow k - 2$ 
19     else  $k \leftarrow k - 1$ 
20 return  $dyn[n]$ 
```

Оптимизацията по памет не може да се приложи към масива *taken*: той е нужен през цялото време на изпълнение на алгоритъма.

Задача 5. Номерираме словете с целите числа от 0 до $2l$ и взимаме по един връх от всеки слой с четен номер, т.е. от словете $0, 2, 4, \dots, 2l$. Тези върхове са точно $l + 1$ на брой и образуват независимо множество (антиклика), защото никои два от тях не са свързани с ребро. Последното твърдение е в сила, понеже никои два от взетите върхове не са нито в един и същи слой, нито в съседни слоеве.

Задача 6. Нека наречем задачата от условието SubIso (от Subgraph Isomorphism).

Задачата Клика (CLIQUE) изглежда така: Даден е неориентиран граф $G(V, E)$ и число k . Има ли клика с поне k върха в G ?

Свеждаме задачата Клика до задачата SubIso така:

Нека G_1 е неориентираният граф $G(V, E)$ от задачата Клика, а G_2 е пълен граф с k върха.

— Ако G_1 съдържа подграф, изоморфен на G_2 , този подграф е търсената клика в G .

— Ако пък G_1 не съдържа подграф, изоморфен на G_2 , той не съдържа клика с поне k върха.

Следователно задачата Клика се свежда към задачата SubIso. Тази редукция е полиномиална, понеже пълният граф може да бъде построен за полиномиално време.

SubIso е от класа NP, защото можем бързо да проверим дали биекция f от подмножество върхове на G_1 към върховете на G_2 е изоморфизъм (т.е. запазва ребрата).

От сводимостта и наличието на сертификат (биекцията f) следва, че SubIso е NP-пълна.