АЛЧНИ АЛГОРИТМИ

Задача 1. Налага се да пътуваме с кола от град A до град B. Разстоянието между A и B е d. По протежение на шосето има бензиностанции на разстояния $d_1 < d_2 < d_3 < \ldots < d_n$ от A. Резервоарът на колата побира достатъчно гориво за изминаване на разстояние m и е пълен в началото на пътуването. (Разстоянието m е достатъчно голямо, т.е. пътуването е възможно.) Предложете алгоритъм, който за време $\Theta(n)$ съставя възможно най-кратък списък от бензиностанции, на които да спрем за зареждане.

Решение: Задачата се решава чрез алчен алгоритъм: спираме за зареждане на бензин възможно най-късно, т.е. на последната бензиностанция, до която можем да достигнем с текущия запас от гориво.

```
minStop(d[1...n]: array of integers; distAB, m: integers)
// distAB е разстоянието от A до B, означено с d в условието
Last ← 0
for k ← 2 to n
   if d[k] - Last > m
      print k-1  // зареждаме на бензиностанция № k-1
      Last ← d[k-1]
if distAB - Last > m
   print n  // зареждаме на бензиностанция № n
```

Алгоритъмът се изпълнява за време $\Theta(n)$, тъй като еднократно обхожда масив от n елемента.

Задача 2. Дадено е множество от мероприятия, всяко от които се характеризира с начален час и краен час. Съставете алгоритъм, който избира възможно най-голям брой мероприятия, така че никои две от тях да не се застъпват (допуска се краят на едното да съвпада с началото на следващото).

Решение: Трябва да съставим алгоритъма внимателно, иначе той ще се окаже некоректен.

П р и м е р : Нека мероприятията се провеждат в следните времеви интервали: [0 ; 13], [2 ; 5], [4 ; 6], [5 ; 9], [8 ; 14], [11 ; 12], [15 ; 20]. Алчната стратегия се състои в това, да вземем първото възможно мероприятие (за да можем след него да вземем максимален брой). Трябва да уточним обаче в какъв ред разглеждаме мероприятията, т.е. най-напред трябва да извършим подходящо сортиране.

Ако ги взимаме в хронологичен ред по началния час, няма да се получи оптимален резултат. В конкретния пример този алгоритъм води до следното множество: [0 ; 13], [15 ; 20], тоест общо две мероприятия. Това не е най-големият възможен брой.

Ако сортираме мероприятията по дължина (т.е. по разликата на крайния и началния час) и взимаме първо най-късите, ще получим следното множество: [11 ; 12], [4 ; 6], [15 ; 20], т.е. три мероприятия, което пак не е оптимално.

Правилно е да сортираме мероприятията по техните краища и после вече да прилагаме алчна стратегия (т.е. всеки път да добавяме към нашия списък първото мероприятие, което не се застъпва с никое от досега избраните).

В конкретния пример резултатът е [2 ; 5], [5 ; 9], [11 ; 12], [15 ; 20], т.е. четири мероприятия, което е максималният възможен брой.

Коректност на алгоритъма: Първо ще докажем, че най-рано завършващото мероприятие непременно е в поне едно от оптималните решения. Избираме произволно оптимално множество. Нека първото (по краен час) мероприятие в това множество не е първото от всички мероприятия. Тогава заменяме първото мероприятие на множеството с най-рано завършващото от всички мероприятия. След тази замяна крайният час на първото мероприятие на множеството ще стане по-ранен, значи, това мероприятие ще продължава да не се застъпва с останалите мероприятия в множеството (които не се застъпват едно с друго). Следователно новото множество пак е оптимално, защото то съдържа същия (максималния) брой две по две незастъпващи се мероприятия. Обаче новото оптимално множество вече съдържа най-рано завършващото от всички мероприятия.

Ето защо няма да сбъркаме, ако, конструирайки оптимално множество стъпка по стъпка, вземем най-рано завършващото мероприятие на първата стъпка от алгоритъма. Тогава всички други мероприятия, които ще вземем, или завършват преди неговото начало, или започват след неговия край (за да не се застъпват с него). Първият вариант е невъзможен: иначе най-рано завършващото мероприятие не би било такова. Остава другият вариант. Затова отсяваме оставащите мероприятия, като премахваме всички, започващи преди края на най-рано завършващото мероприятие. От останалите трябва да вземем колкото може повече, т.е. решаваме същата задача за останалите мероприятия. По същата логика няма да сбъркаме (т.е. ще стигнем до оптимално решение), ако пак вземем най-рано завършващото от останалите мероприятия и т.н.

Този алчен алгоритъм изисква време $\Theta(n \, . \log n)$ заради сортирането, където n е броят на мероприятията. (Вторият етап — построяването на максимално множество, след като масивът е сортиран — се изпълнява за линейно време.)

Задача 3. Даден е свързан граф. Във всеки негов връх има плод с определен брой калории. Една маймунка яде плодовете в произволен ред. След изяждане на плод съответният връх се изтрива заедно с излизащите ребра. Ако графът се разпадне на няколко компоненти, маймунката избира една от тях и продължава да яде от нея, а другите компоненти се изтриват (така графът остава свързан). Маймунката иска да изяде максимум калории и прилага алчен алгоритъм: всеки път изяжда най-калоричния плод, а при разпадане на графа избира компонентата с най-калоричния плод. Верен ли е този алгоритъм?

Решение: Алчният алгоритъм, прилаган от маймунката, не е коректен.

Контрапример: графът 20 — 30 — 18 — 19

Маймунката първо изяжда плода с 30 калории.

Графът се разпада на две компоненти: 20 18 — 19

Маймунката избира компонентата с най-калоричния плод, т.е. 20, следователно компонентата с 18 и 19 калории отпада: 20

Сега маймунката изяжда плода с 20 калории и не остават повече плодове. Така маймунката е изяла общо 30 + 20 = 50 калории. Това не е максимумът: тя би могла да изяде всичките плодове, т.е. 20 + 30 + 18 + 19 = 87 > 50 калории, като ги яде например отляво надясно.