



XII конкурс по програмиране на PC Magazine Bulgaria и Мусала Софт



Стартира 3-ти кръг на конкурса по програмиране
на PC Magazine Bulgaria и Мусала Софт под патронажа на БАСКОМ



Стартира традиционният Коледен турнир по програмиране на Българската асоциация на софтуерните компании (БАСКОМ), който се провежда в рамките на третия кръг на конкурса по програмиране, организиран от PC Magazine Bulgaria и Мусала Софт.

БАСКОМ е създадена през 2001 г. и обединява 40 водещи български софтуерни фирми. Организацията подпомага насърчаването на конку-

рентните предимства и дългосрочните традиции в разработването на софтуер, както и приемането на добрите бизнеспрактики с цел повишаване конкурентноспособността на софтуерните компании в страната.

БАСКОМ е съорганизатор и спонсор на Коледния кръг от конкурса по програмиране и традиционно осигурява парични награди за победителите.



МАГИЧЕСКА СУМА

Един хубав слънчев ден Мечо Пух, както си вървял из гората, срещнал златната рибка. Те решили да играят следната игра: Мечо Пух си измисля три желаня, а златната рибка ги изпълнява, като добавя някакво допълнително условие към тях.

Първото желание на Мечо Пух било да има N склада, пълни с мед. Златната рибка изпълнила това желание, като във всеки склад сложила определен брой гърнета с мед, не непременно еднакъв за всички складове.

Второто желание на Пух било да си избере двойки складове, които да се съберат магически. Под магическо събиране на два склада Мечо Пух разбира броят на гърнетата във всеки от двата склада да се увеличи с броя на гърнетата в другия. Златната рибка обаче му казала, че ще го направи, но за всяко събиране той трябва да направи разликата между броя на гърнетата в двата склада, вдигната на квадрат, коремни преси.

Пух бързо осъзнал факта, че в зависимост от това в какъв ред се изберат двойките складове ще трябва да направи различен брой коремни преси. Третото желание на Пух било той да избере в какъв ред да се извършват събиранията на двойките складове. Златната рибка се съгласила, но Мечо Пух трябвало да направи 5 лицеви опори.

Като се замислил повече, Мечо Пух разбрал, че намирането на възможно най-добрия ред за магическо събиране на двойките гърнета изобщо не е проста задача. Помогнете му, като напишете програма **MAGICSUM**, която намира наредбата с възможно най-малко коремни преси.

Входните данни се четат от файл **MAGICSUM.IN**, като на първия ред е записан броят складове N ($N \leq 32$). След това следват N естествени числа, ненадвишаващи 100 000, които задават броя на гърнетата във всеки склад. Следва числото M , задаващо броя на двойките, а на следващите M реда има по две различни числа в интервала $[1, N]$ - складовете, които ще бъдат магически събирани. Два склада не могат да бъдат събирани повече от веднъж.

Програмата трябва да отпечата в **MAGICSUM.OUT** двойките, които трябва да бъдат магически събрани в определения от програмата ред.

В 40% от тестовете нито един от складовете няма да бъде събиран повече от пет пъти. За всеки тест максимален брой точки ще получи състезателят, чиято програма е намерила наредба с най-малко коремни преси. Всички останали решения ще получат точки пропорционално на броя коремни преси, намерен от тяхната програма. Програма-

та на участника трябва да завърши един тест за по-малко от 10 секунди.

ПРИМЕР

MERGE.IN	MERGE.OUT
4	1 3
1	3 4
8	1 2
4	2 3
6	
4	
1 2	
1 3	
2 3	
3 4	

ПОЯСНЕНИЕ: Първо се събират складове 1 и 3, което означава, че Пух трябва да направи $(1-4)^2 = 9$ коремни преси. Броят на гърнетата в складове 1 и 3 вече е 5. Следва събиране на складове 3 и 4 с цена $(5-6)^2 = 1$ преса. В склад 3 и склад 4 вече има по 11 гърнета. След събиране на складовете 1 и 2 Пух трябва да направи $(5-8)^2 = 9$ преси. Броят на гърнетата в склад 1 и склад 2 вече е 13. За събирането на складове 2 и 3 Мечо Пух трябва да направи $(13-11)^2 = 4$ коремни преси. Общо 23 коремни преси. Това е най-доброто възможно решение на примера.

Решенията на задачата може да изпратите чрез системата за up-load директно на сайта <http://konkurs.musala.com> или на адрес konkurs@musala.com до 10 януари 2007 г.

Анализ на задача 1 – НАМЕРИ МЕДА

Задачата, както забелязаха повечето от състезателите, може да се раздели на две абсолютно независими части.

Първата част е да се намери времето за претърсване на всяко гърне. Т.е. за дадена редица $a_1, a_2, a_3, \dots, a_m$ да се намери най-дългата нарастваща подредица от не непременно съседни елементи. Това е известен проблем и за неговото ефективно решаване се изисква използването на динамично оптимизиране. Ще изложим накратко два алгоритъма с различна алгоритмична сложност.

Алгоритъм 1: Нека $f[k]$ е дължината на най-дългата нарастваща редица с последен елемент a_k . Тогава $f[k] = \max(f[i], 0) + 1$, където $1 \leq i < k$ и $a_i < a_k$. Т.е. най-доброто решение с последен елемент a_k се получава от най-дългата редица, която можем да продължим, увеличена с 1. Пресмятаме последователно всички стойности на $f[i]$ и отговорът на задачата е равен на $\max(f[i])$, където $1 \leq i \leq m$. Този алгоритъм има алгоритмична сложност $O(m^2)$ и излиза от времевото ограничение за по-големите тестове.

Алгоритъм 2: Обхождаме елементите на редицата един по един. Нека сме стигнали до елемент a_j , $g[k]$ е равна на най-малкия елемент, на който може да завършва нарастваща редица с дължина k до момента, а len е дължината на най-дългата нарастваща редица до текущия елемент. В началото len е равно на 0. На всяка стъпка намираме минималното j , така че $g[j] \geq a_i$ и приравняваме $g[j] = a_i$. В случай че не съществува такава j (т.е. $a_i > g[j]$: за всяко $j = 1 \dots len$), то увеличаваме len с едно и присвояваме $g[len] = a_i$. След като сме обходили всички елементи, отговорът на задачата е равен на len .

Забележете, че така получената редица е строго растяща. Затова на всяка стъпка можем да намерим индекс j чрез двоично търсене с алгоритмична сложност $O(\log_2 m)$ и да получим алгоритъм с обща сложност $O(m \cdot \log_2 m)$. Доказателството за верността на алгоритъма оставам за читателя като леко упражнение.

Като приложим алгоритъм за намиране на най-дългата нарастваща редица за всяко гърне, получаваме алгоритмична сложност $O(n \cdot m \cdot \log_2 m)$, където n е броят на гърнетата. Вече сме пресметнали необходимите ни стойности, за да преминем към втората част.

Втората част е по дадени цени на гърнетата наредени в редица $b_1, b_2, b_3, \dots, b_n$, да се намери оптимален ред на взимане на гърнетата. Един очевиден алгоритъм за решаване на задачата е пробването на всички възможни наредби. Този алгоритъм има алгоритмична сложност $(n!)$ и очевидно е твърде бавен. С този подход може да се реши само един от десетте тестови примера.

Другият подход отново се основава на динамичното оптимизиране. А именно, нека $d[i][j]$ е равно на времето, което ни е нужно да вземем гърнетата с номера от i до j по оптимален начин. При $i > j$, $d[i][j] = 0$, а при $i = j \Rightarrow d[i][j] = b_i = b_j$. Забележете, че взимането на някое гърне от редицата разбива задачата на две абсолютно независими части – да намерим оптималния ред в лявата и в дясната част. Затова $d[i][j] = b_i + b_{i+1} + \dots + b_{j-1} + b_j + \min(d[i][k-1] + d[k+1][j])$, където k е в интервала $[i, j]$. Отговорът на задачата е $d[1][n]$. Тъй като за всяка двойка (i, j) , обхождаме числата от i до j , то този алгоритъм има алгоритмична

М.	Име	Град	Общо	Време
1	Свилен Марчев	София	100	8.523
2	Владимир Недев	Варна	100	15.704
3	Борис Даскалов	София	100	26.646
4	Божидар Пенчев	Плевен	100	28.360
5	Васил Люцканов	София	100	34.308
6	Васил Поповски	София	100	38.655
7	Йордан Маджунков	Плевен	100	44.314
8	Ростислав Руменов	Шумен	100	48.950
9	Радослав Герганов	София	100	57.283
10	Емил Ибришимов	Стара Загора	100	66.388
11	Ангел Джигаров	Бургас	100	69.194
12	Иван Иванов	София	100	70.954
13	Йосиф Йосифов	Велико Търново	100	76.079
14	Любомир Господинов	София	100	76.441
15	Веселин Георгиев	София	90	52.817
16	Ивелина Захариева	София	90	59.716
17	Иван Ваклинов	София	90	72.076
18	Ивайло Бояджиев	Кюстендил	80	39.828
19	Илиян Илиев	София	70	22.793
20	Васил Жеков	София	60	9.914
21	Александър Георгиев	София	60	12.938
22	Калоян Радев	Добрич	60	19.348
23	Руско Атанасов	София	50	1.151
24	Петър Иванов	Шумен	50	1.623
25	Борислав Станимиров	София	50	1.732
26	Александър Александров	София	50	1.742
27	Даниел Попов	Ямбол	50	1.972
28	Калоян Донев	Добрич	50	1.993
29	Ангел Евров	София	50	2.342
30	Момчил Иванов	Ямбол	50	2.343
31	Антони Средков	Бургас	50	2.715
32	Иван Василев	Плевен	50	3.085
33	Деян Дойчев	Стара Загора	50	5.508
34	Кирил Арабаджийски	София	50	5.868
35	Веселин Митров	София	50	6.479
36	Невен Николов	Бургас	40	4.065
37	Лъчезар Янков	София	10	0.070
38	Радослав Г. Рачев	Варна	10	0.070
39	Ивайло Михайлов	София	10	0.070
40	Свилен Димитров	Дулово	10	0.080
41	Нено Галев	София	10	0.080
42	Красимир Георгиев	Разград	10	0.080
43	Георги Димитров	Сливен	10	0.080
44	Христо Илиев	Русе	10	0.110
45	*Момчил Томов	Плевен	5	0.080
46	*Тодор Балабанов	София	5	0.280
47	Пламен Томов	София	0	0.000
47	Павел Владов	Горна Оряховица	0	0.000
47	Николина Арабаджиева	Самоков	0	0.000
47	Радослав Х. Рачев	Шумен	0	0.000
47	Любомир Русев	Габрово	0	0.000
47	Климент Ненов	Нови Пазар	0	0.000
47	Христо Борисов	Бургас	0	0.000
47	*Севгин Гюджан	Варна	0	0.000
47	Надя Станева	София	0	0.000
47	Борислав Кирилов	София	0	0.000
47	Орлин Колев	София	0	0.000
47	Тодор Цонков	Враца	0	0.000
47	Васил Люнчев	Смолян	0	0.000
47	Борис Пелтеков	София	0	0.000
47	Иван Бобев	Сливен	0	0.000
47	*Антон Керезов	Пловдив	0	0.000
47	Велислав Николов	София	0	0.000
47	Ангел Владов	Горна Оряховица	0	0.000
47	Иван Тодоров	Бургас	0	0.000
47	Владимир Ангелов	Плевен	0	0.000
47	Господин Бондуков	Бургас	0	0.000
47	*Лазарин Лазаров	София	0	0.000

* Точките на участника са намалени с 50%, защото програмата му чете или пише в грешен файл или от/на стандартния вход/изход.

сложност $O(n^3)$. Този алгоритъм винаги намира правилното решение, но е достатъчно бърз за пет от десетте тестови примера. Много от състезателите, които имат 50 точки, са реализирали именно него.

Най-интересният момент в задачата е оптимизирането на този алгоритъм. Една разпространена техника при динамичното оптимизиране е съкращаване на вътрешния цикъл. Тази техника е подходяща и при тази задача. Нека $p[i][j]$ е равно на k , при което се получава минимална сума $d[i][k-1] + d[k+1][j]$. Очевидно $p[i][i] = i$. Тогава след задълбочено наблюдение забелязваме, че $p[i][j-1] \leq p[i][j] \leq p[i+1][j]$. Този факт се доказва чрез пълна математическа индукция по дължината на интервала и чрез допускане на противното.

Това неравенство ни позволява, когато търсим оптималното k за интервала $[i, j]$, да не обхождаме всички числа между i и j , а само онези от тях, които го удовлетворяват. Така за интервала $[1, x]$ ще обходим индексите от $p[1][x-1]$ до $p[2][x]$. За интервала $[2, x+1]$ ще обходим индексите от $p[2][x]$ до $p[3][x+1]$. За интервала $[3, x+2]$ от $p[3][x+1]$ до $[4][x+2]$ и така нататък. Неформално казано - всеки път ще продължаваме оттам откъдето сме приключили предишния път. Т.е. ще пресметнем всички интервали с дължина x чрез едно обхождане на числата от 1 до n или с $O(n)$ операции. В крайна сметка, за всички възможни $n-1$ дължини ще направим същото нещо и ще получим алгоритъм с алгоритмична сложност $O(n^2)$, който, комбиниран с втория изложен алгоритъм за намиране на най-дълга нарастваща редица, има сложност $(n^2 + n \cdot \log_2 m)$, минава на десет от десетте тестови примера и получава пълен брой точки.

Този алгоритъм или близък до този е реализиран от всички състезатели и пълен брой точки на задачата, като изключително ефективната реализация на Свилен Марчев го изведе една крачка пред останалите и му донесе първото място.

Класиране на първи кръг

На първо място се класира Свилен Марчев от София. Той е възпитаник на 12 СОУ „Цар Иван Асен II“ в столицата, активен участник е в състезания по информатика и се състезава за втора година в конкурса по програмиране на Мусала Софт и PC Magazine. Свилен е тренирал Тае Куон До, а сега се занимава активно с тенис на маса. Любимият му автор е Паулу Коелю и е прочел почти всичките му книги. Свилен спечели Bluetooth устройството, Oxford картинен речник и абонамент за PC Magazine Bulgaria. Първенецът получи и грамота от Мусала Софт.

На второ място застана Владимир Недев, а на трето Борис Даскалов. И двамата са студенти във Факултета по математика и информатика към СУ „Климент Охридски“, съответно в трети и във втори курс. За Владимир наградата бе Европейски езиков сертификат TELC - английски език, подготовка за ниво A2, и абонамент за PC Magazine, а за Борис - Европейски езиков сертификат TELC - английски език, подготовка за ниво B2, и абонамент за PC Magazine.



Награждаването на победителите се състоя на 14 ноември в офиса на фирма Мусала Софт. Участниците бяха наградени от Тервел Няголов – главен редактор на PC Magazine Bulgaria, и Делян Лилев – изпълнителен директор на Мусала Софт.

