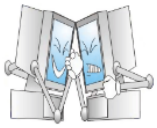


Назив проблема: Игра

Аутор: Душан Здравковић Анализа: Андреј Ивашковић

Тагови: сортирање, „ад-хок“

Решење и анализа:

Напомена. У наредном образложењу се претпоставља да су низови индексирани од нуле.

Формулишимо ову игру на другачији начин: за дати низ природних бројева одредити који играч има победничку стратегију уколико играч који направи празан низ побеђује, а дозвољени потези су:

1. брисање почетног елемента низа и свих елемената који су једнаки почетном;
2. брисање последњег елемента и свих елемената који су једнаки последњем елементу низа.

20 поена

Посматрајмо најпре први „подзадатак“, у ком не постоји више од два елемента у низу. Очигледно, уколико имамо само један елемент, побеђује А (први играч). Уколико постоје два елемента, разликујемо случајеве:

- елементи су једнаки: побеђује А у првом потезу;
- елементи су различити: ма како А играо, остаје само једна вредност у низу.

За овакво решење (исправно имплементирано) предвиђено је 20 поена.

50 поена

Један могући приступ јесте решавање овог задатка „грубом силом“: испробају се *сви начини* како игре може да се одвија. С обзиром на то да овакво решење не доноси свих 100 поена и због чињенице да је поприлично компликованије од оног званичног решења Комисије за 100 поена, нећемо му посветити превише пажње. Ипак, следи кратак опис (дакле, без важних детаља о имплементацији) једног од могућих начина да се ово обави.

Низ из игре може да буде подвргнут различитим променама, али се сваким потезом смањује дужина низа бар за 1. Уведимо помоћни *низ низова* (матрицу): сваки низ представља једно од узастопних стања игре. За задато тренутно стање можемо да одредимо једно од два следећа: прво се добија применом операције 1, а друго применом операције 2. За одређивање сваког могућег даљег тока игре може да се користи нека рекурзивна процедура, с тим што у случају „празног низа“ знамо ко је победник. За свако добитно стање се памти који играч је победник. Пажљивом анализом може да се одреди који играч има победничку стратегију за дато стање уколико знамо који играчи имају победничке стратегије у каснијим стањима. Горепоменути низ низова служи да бисмо имали представу о претходним стањима.

За овакво решење је Комисија предвидела 50 поена. Ипак, овакве идеје о „грубој сили“ су изузетно важне и постоје проблеми који не могу да се реше на било који други начин.

100 поена

Сваким потезом се избацују сва појављивања неке вредности из низа. Укупан број потеза у овој игри је зато једнак броју вредности које се појављују у овом низу, ма како А и Б играли. Међутим, број

потеза одређује победника: уколико је број потеза у игри непаран, побеђује А; у супротном, Б је победник. Оправдајмо ово запажање:

- ако један потез завршава игру, тада побеђује А (однос играч чији је потез);
- ако два потеза завршавају игру, тада – ма како А играо – долазимо до стања у ком један потез играча завршава игру, те је Б победник; другим речима, губи играч чији је потез;
- понављати претходни резон.

(Приметимо, дакле, да нема смисла говорити о победничкој „стратегији“: победник је одређен почетним стањем и не постоји никакав низ потеза који може да промени овај исход.)

Дакле, свели смо задатак на одређивање броја различитих елемената у неком низу. Ово може да се уради на више начина, а овде ће бити представљена два (за оба је Комисија предвидела 100 поена):

Прво решење

Нека је полазни низ X. Тада желимо да добијемо низ Y у ком се свака вредност присутна у X налази тачно једном и у њему се не налази ниједна друга вредност. На пример, за $X = (1,2,3,2,5,3)$ имамо $Y = (1,2,3,5)$. Постепено (редом) додајемо неке елементе из X на крај Y. Нека тренутно разматрамо да ли ћемо додати $X[k]$ на крај низа Y (чија је „тренутна дужина“ једнака l):

- уколико ниједан од $Y[0]$, $Y[1]$, …, $Y[l-1]$ није једнак $X[k]$, тада додајемо $X[k]$ у Y ($Y[l] \coloneqq X[k]$) и одржавамо тражено својство;
- у супротном, у Y се већ налази ова вредност и не врши се никакво додавање.

На пример, нека у горњем примеру разматрамо $X[3]$ $(= 2)$, при чему је тада $Y = (1,2,3)$. Како се 2 налази у Y, не долази ни до какве промене. Међутим, при разматрању $X[4]$ $(= 5)$, видимо да се 5 не налази у Y, те морамо да ажурирамо: $Y \coloneqq (1,2,3,5)$.

Решење, дакле, зависи од дужине низа Y на крају.

Следи псеудокод овог решења:

```
broj_razlicitih_1(X, n):
|  Y[0] := X[0]
|  l := 1
|  za sve k od 1 do n – 1:
|  |  jedinstven := TRUE
|  |  za t od 0 to k – 1:
|  |  |  ako (X[t] = X[k]) onda jedinstven := FALSE
|  |  if (jedinstven) onda
|  |  |  Y[l] := X[k]
|  |  |  l := l + 1
|  vrati l
```

Временска сложеност оваквог приступа је $O(N^2)$, што је довољно за 100 поена. Могуће је изменити имплементацију тако да се не уводи Y.

Друго решење

Нека је X дато решење. Приметимо да редослед елемената низа X није битан, те сортирање низа (у растућем/неопадајућем поретку) неће променити одговор. Приметимо да се сада X састоји од „блокова“ једнаких вредности. У поменутом примеру се добије $X = (1,2,2,3,3,5)$. При постепеном разматрању свих елемената низа X, можемо да установимо да ли је $X[k]$ елемент који се до сада

није појављивао само упоређивањем са претходним елементом тј. испитивањем да ли важи $X[k] = X[k-1]$. Ово следи из чињенице да је низ сортиран.

Следи псеудокод овог решења:

```
broj_razlicitih_2(X, n):
|  sortiraj X
|  l := 1
|  za sve k od 1 do n - 1:
|  |  ako (X[k] =/= X[k - 1]) onda l := l + 1
|  vrati l
```

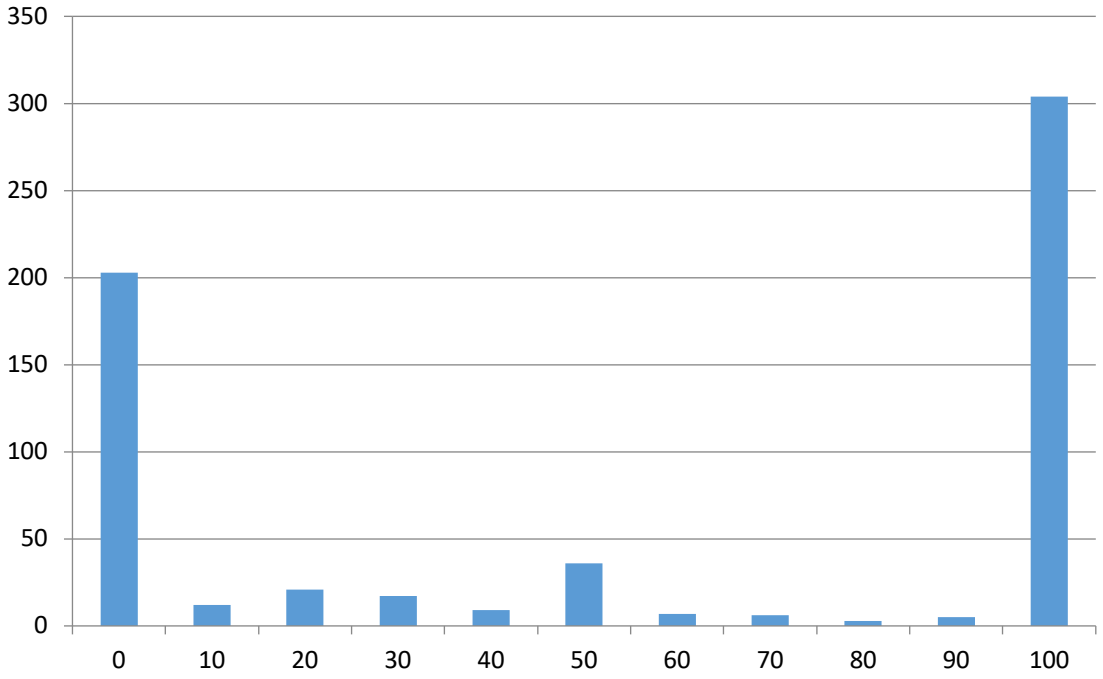
„Уско грло“ овог алгоритма, однос „најскупљи део“, јесте сортирање. Због природе ограничења је прихватљив било који од стандардних алгоритама сортирања за 100 поена, па и они у класи $O(N^2)$ нпр. *Selection sort*.

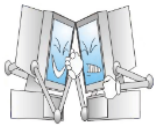
Такмичари који знају ефикасније алгоритме сортирања су исто добили 100 поена (наравно, под условом да је имплементација била добра), али ово није неопходно при оваквим ограничењима. Ипак, важно је знати да се упит „број различитих елемената“ појављује често као део неких много компликованијих задатака, а један од основних начина да се ово уради јесте управо сортирање.

Статистика

Просечан број поена на овом задатку је био поприлично висок – око 56. Међутим, стандардна девијација је много велика (око 46) због чињенице да је поприличан број такмичара имао доста поена, али је у исто време било и много такмичара са мало поена: од укупно 623 такмичара, 304 је имало 100 поена, а 203 је имало 0 поена.

Следи хистограм урађености задатака (однос броја поена и броја такмичара који су остварили толико поена).





Тестирање.

Решења такмичара су тестирана на десет тест примера. Следи њихов опис у наредној табели:

Тест пример	$\max\{N_1, N_2, N_3\}$
1	2
2	2
3	19
4	20
5	14
6	402
7	561
8	913
9	858
10	1000

Уобичајене грешке.

После задатка „Мистерија“, овај задатак је урађен поприлично добро од стране такмичара. Ипак, поткрале су се неке грешке. Најчешће су се тицале непажње у погледу граница низова (низ је индексиран од јединице, а није уведена исправка – ови такмичари су углавном имали 90 поена) или неисправног тумачења наведених ограничења. Једна C++ имплементација је имала наредну грешку (копирано дословно):

```
for(i=0;i<(n2+1);i++)
if(b[i]<b[i+1])
m2++;
```

Иако се квалитет кода не оцењује на такмичењу, добра је пракса направити код колико-толико модуларним. Велики број такмичара је сва три случаја размотрио у главном делу програма, па су ручно мењани неопходни параметри на сваком месту. Овакво копирање врло лако доводи до грешке уколико нешто треба да се измени, будући да тада треба да се иста измена обави три пута.

Неки такмичари су имали идеју да броје појављивања елемената низа и на основу тога одреде број различитих вредности (дакле, нешто слично *Counting sort* алгоритму). Међутим, због природе ограничења (бројеви нису довољно мали да могу да буду индекси низа), ово није донело 100 поена.

Неки (искуснији) такмичари су претпоставили да је реч о задатку који се решава динамичким програмирањем. Међутим, у овом задатку је рекурентна веза било каквог решења које ради у квадратној сложености компликована. Понеки такмичари су користили различите структуре података попут хеш табела и бинарних стабала претраге, али су тада често било лоше имплементиране.