Korepondenčný seminár z programovania
XIX. ročník, 2001/2002
Katedra vyučovania informatiky FMFI UK,
Mlynská Dolina, 842 48 Bratislava

KSP finančne podporuje nadácia Open Society Fund Bratislava a IUVENTA – zariadenie pre vožný čas detí, mláde« e a dospelých MICROSTEP spol s.r.o.

Kategória Z

Vzorové riešenia 2. kola letnej časti

Milí riešitelia!

Je tu leto a preto[1] si rýchlo prečítajte vzoráky, mrknite na výsledkovú listinu a choďte sa radšej kúpať. Prefíkanejšie jedince si môžu zobrať vzoráky k vode.

S neštvrtákmi, ktorí sa umiestnili na začiatku výsledkovej listiny sa stretneme na jesennom sústredení.

Krátke vlasy, dlhý rozum. Navyše aj skôr uschnú.

KSPáci

1. Zúrivý Ubu

opravoval YoYoδ
(max. 15 bodov)

Už pohľad do výsledkovej listiny naznačuje, že skoro všetky riešenia boli temer vzorové.[2] Za chýbajúci popis sa strhávali zvyčajné 2 body. Jeden bod sa strhával, ak ste nemali dostatočne zdôvodnené, prečo sa počet okresov rovná počtu miest v najväčšom komponente. No a riešenia s horšou časovou zložitosťou ako vzorové t.j. $O(M + N)$, mohli získať najviac 12 bodov.

Skôr než prejdeme k vzorovému riešeniu, uveďme si najskor niekoľko označení z teórie grafov, aby sme boli jednotní. Cestnú sieť v Burundi zloženú z miest a ciest budeme nazývať *graf*. Mestá budú *vrcholy* grafu a cestu budú *hrany* grafu (jedna hrana spája vždy dva vrcholy). Množina vrcholov, v ktorej z každého z nich sa dá dostať do každého z nich sa nazýva *komponent* grafu (niektorí z vás preň vymysleli exotické mená typu „cesta“, „kružnica“, čo síce sú pojmy z teórie grafov, ale označujúce niečo úplne iné).

Všetci, čo tento príklad odovzdali, zistili, že počet okresov, ktoré je treba zriadiť, je rovný počtu vrcholov v najväčšom komponente grafu (označme ho K). A prečo? No najskôr si ukážme, prečo ich nemôže byť menej ako K. Ak by tak bolo, zoberme si vrcholy najväčšieho komponentu. Keďže by ich bolo viac ako okresov, museli by nejake dva byť v tom istom okrese. To ale znamená, že by v jednom okrese existovali dve mestá, medzi ktorými existuje cesta, čo nemôže byť[3], a preto okresov nemôže byť menej.

Vieme už teda, že ich musí byť aspoň K. Ešte je treba ukázať, že K okresov naozaj stačí, a že sa tak vrcholy dajú rozdeliť. Zoraďme si teda nejako vrcholy v každom komponente. V prvom okrese budú prvé vrcholy z každého komponentu. V druhom budú druhé, atď. Každému vrcholu sa nejaký okres ujde, pretože okresov je toľko čo vrcholov v najväčšom komponente. No a v rámci každého okresu sa nevieme dostať z nijakého vrcholu do žiadneho iného, pretože každý z nich je v inom komponente grafu.

Na napísanie vzorové riešenia už teda stačí vedieť len ako odhaliť jednotlivé komponenty grafu. Zamyslime sa teda, čo spraví také prehľadávanie grafu (či už do hĺbky alebo do

[1] toto je vnútorný rým

[2] zdôrazňujem slová **skoro všetky, temer vzorovo**, to aby si náhodou neupadol/a do omylu, že práve Ty si ho mal vzorovo

[3] alebo ináč povedané dostávame spor

šírky). No „prejde“ všetky vrcholy, ktoré sú pospájané – teda patria do jedného komponentu. Program je teda jednoduchý: Postupne prechádzame vrcholy grafu a keď nájdeme nejaký nenavštívený, pustíme z neho prehľadávanie. Ním zrátame počet vrcholov v tomto komponente a zároveň si o nich poznačíme, že už sú navštívené. No a samozrejme nezabudneme vybrať maximum z počtu vrcholov.

No a aby to bolo zaujímavejšie, na rozdiel od minulej série použijeme prehľadávanie do šírky[4], čo značí, že namiesto zásobníka (respektíve rekurzie) použijeme frontu. Na začiatku do fronty vložíme vrchol, z ktorého chceme prehľadávať. Potom, kým fronta nie je prázdna, vždy vyberieme vrchol zo začiatku a všetky vrcholy, do ktorých sa z neho vieme dostať priamou hranou a ešte sme ich do fronty nevložili, zaradíme do fronty. Keď sa fronta vyprázdni, prešli sme všetky vrcholy v komponente, do ktorého patrí vrchol, z ktorého sme začínali.

Frontu realizujeme poľom F. Pamätáme si index na prvý prvok – z a index na miesto v poli za posledným prvkom – k. Prvok vložíme do fronty tak, že vložíme do poľa na pozíciu k ($F[k]$) a k následne zvýšime. Prvok vyberáme tak, že sa pozrieme na $F[z]$ a z následne zvýšime. No a fronta je prázdna vtedy, ak $z = k$.

Časová zložitosť prehľadávania grafu o N vrcholoch a M hranách má zložitosť $O(N+M)$, čo je teda aj zložitosť celého nášho algoritmu. Pamäťová je tak isto $O(N + M)$.

Listing programu:

```
program Ubu_zuri;
const MaxN=100;
var N,M:integer;
    H:array[0..MaxN,0..MaxN] of integer;
    F:array[0..MaxN] of integer;
    bol:array[0..MaxN] of boolean;
    z,k:integer;
    i,j,pocet,maxpocet,a,b:integer;

procedure prehladaj(v:integer);
var i,j:integer;
begin
  z:=0; k:=1; F[0]:=v;
  bol[v]:=true;                                {oznacime, ze v bol navstiveny}
  while k>z do begin
    i:=F[z]; inc(z);                             {vyberieme vrchol z fronty}
    inc(pocet);                  {zvysime pocet vrcholov v tomto komponente}
    for j:=1 to H[i,0] do                       {prejdeme hrany z vrcholu v}
      if not bol[H[i,j]] then            {ak vedu do nenavstiveneho vrcholu}
        begin F[k]:=H[i,j]; inc(k); bol[H[i,j]]:=true; end;
                                                 {tak ich pridame do fronty}
  end;
end;

begin
  write('Pocet miest: '); readln(N);
  write('Pocet ciest: '); readln(M);
  for i:=0 to N do begin
    H[i,0]:=0;                                 {zatial nemame ziadne hrany}
    bol[i]:=false;                       {ziadne mesto este nebolo navstivene}
```

[4] v našom prípade je úplne jedno, ktoré použijeme

```
  end;
  for i:=1 to M do begin
    write(i,'. hrana: ');
    readln(a,b);
    inc(H[a,0]); H[a,H[a,0]]:=b;                    {pridame hranu z a do b}
    inc(H[b,0]); H[b,H[b,0]]:=a;                    {pridame hranu z b do a}
  end;

  maxpocet:=0;
  for i:=1 to N do if not bol[i] then begin      {vrchol patriaci do zatial}
    pocet:=0;                                     {neprehladaneho komponentu}
    prehladaj(i);
    if pocet>maxpocet then maxpocet:=pocet;   {ak sme nasli vacsi komponent}
  end;                                                      {zapamatame si ho}

  writeln('Treba zriadit ',maxpocet,' okresov.');
end.
```

2. Zdrtený Jurko

opravoval Poko
(max. 15 bodov)

Tento príklad bol zaujímavý tým, že krabičiek bolo vždy práve 5, a teda nebolo treba načítať ich počet a riešiť úlohu všeobecne. Tak sa stalo, že mnohé riešenia (najmä také, ktoré nevyužívali pomocné procedúry) boli na dve strany a človek sa šiel v tej spleti všakovakých *if*-ov stratiť. Ale ako si môžete všimnúť môj listing, nemám právo sa za to na vás hnevať :-)

Poďme však na riešenie. Celkovo máme $5! = 120$ možností, ako môžu byť nugety v krabičkách usporiadané podľa veľkosti. Prvé riešenie je založené na binárnom vyhľadávaní. Predpokladáme, že krabičky 1 až k máme už usporiadané a snažíme sa vložiť $k+1$-vú krabičku. Rozdelíme si krabičky (tak ako sú usporiadané) na dve polovice, novú krabičku porovnáme so strednou krabičkou. V prípade, že nuget v $k+1$-vej krabičke je väčší, vkladáme krabičku do druhej polovice, v opačnom prípade do prvej polovice. Týmto spôsobom pokračujeme, až kým sa dĺžka intervalu, v ktorom môže byť krabička, nezmenší na 1. Teda $k+1$-vú krabičku vieme vložiť na správne miesto pomocou $\lceil \log_2 k \rceil$ otázok. Prvé dve krabičky usporiadame jednou otázkou, tretiu zaradíme na správne miesto pomocou dvoch otázok, štvrtú takisto a na zaradenie piatej krabičky potrebujeme 3 otázky. V najhoršom prípade teda položíme 8 otázok.

Vzorové riešenie však položí najviac 7 otázok. Základom úspechu je dávať otázky tak, aby nám každá možná odpoveď vylúčila polovicu všetkých zostávajúcich možností (resp. pri nepárnom počte možností nám jedna odpoveď vylúči o jednu možnosť viac ako druhá). Takto nám po prvej vhodne položenej otázke ostane 60 možností, po druhej 30 možností, po tretej 15, atď. Celkovo teda položíme najviac $\lceil \log_2 120 \rceil = 7$ otázok. Pri prvej otázke sa opýtame na krabičky 1 a 2. Pre každé možné poradie, kde v prvej krabičke je väčší nuget, máme práve jedno poradie, kde sú tieto dve krabičky v opačnom poradí a ostatné na tom istom mieste. Napr. nech sú krabičky v poradí $1, 4, 2, 5, 3$. Potom tomuto poradiu zodpovedá poradie $2, 4, 1, 5, 3$ pri opačnej polohe krabičiek 1 a 2. Podobnú úvahu prevedieme aj pri otázke na poradie tretej a štvrtej krabičky. Označme v_1 číslo krabičky, v ktorej je väčší nuget z krabičiek 1 a 2, m_1 nech je číslo tej druhej. Podobne nech v_2 je číslo tej krabičky z 3. a 4., v ktorej je väčší nuget a m_2 je číslo tej druhej. V tretej otázke sa spýtame na poradie krabičiek v_1 a v_2. Zrejme odpoveď na túto otázku nám dá číslo tej krabičky, v ktorej je spomedzi týchto štyroch najväčší nuget. Pri obidvoch možných odpovediach získame nasledujúci stav: vieme o troch krabičkách, v akom sú poradí a vieme o jednej krabičke, že v nej je menší nuget

ako v krabičke s dosiaľ najväčším zisteným nugetom. Tak napr., keď $v_1 > v_2$, tak vieme, že $v_1 > v_2 > m_2$ a tiež že $v_1 > m_1$.

Čísla krabičiek si budeme udržiavať v poli a postupne budeme meniť ich poradie. Našou snahou bude usporiadať toto pole podľa toho, ako sú usporiadané krabičky. V tomto kroku výpočtu pre čísla v poli (aj pre nugety v krabičkách s príslušným číslom) platí: $p_1 < p_2$ a $p_1 < p_3 < p_4$, kde p_i je i-ty prvok poľa p. Ostáva nám teda správne zaradiť krabičku s číslom p_2 a $p_5 = 5$. Máme už iba 15 možností, ako ich umiestniť vzhľadom na krabičky p_1, p_3 a p_4. Ako ste si mohli všimnúť, doteraz nikdy nezávisela každá ďalšia otázka od výsledku predošlej (až na prípadné zmeny polohy v poli p). Odteraz to tak už nebude `:-(` Podľa mňa nemá význam prepisovať ešte raz presne to, čo je v listingu programu, myslím si, že je dosť čitateľný. Stačí si niekde na zbytočný kúsok papiera napísať, ktoré možnosti nám po tretej otázke ostávajú, pričom nevypisujeme si čísla krabičiek, ale ich terajšiu polohu v poli p (teda napr. možnosťou $5, 1, 3, 2, 4$ myslíme poradie p_5, p_1, p_3, p_2, p_4). Postupne nám stačí sledovať, ako treba klásť ďalšie otázky a podľa odpovedí poprípade vymieňať prvky poľa. Po tomto procese budú v poli p čísla krabičiek v poradí, ako sú usporiadané v nich nachádzajúce sa nugety. Kto neverí, nech si odskúša všetkých 120 možností. Však ich je len 120 `:-)`

A prečo nemôžeme poradie krabičiek zistiť na menej otázok? Keby sme položili otázku tak, že jedna z odpovedí vylúči aspoň o 2 možnosti menej ako druhá odpoveď, môže sa nám stať, že po odpovedi nám ostane práve tá väčšia časť možností usporiadania krabičiek. Teda v najhoršom prípade po každej ďalšej otázke ostane viac možností usporiadania, ako keby sme sa pýtali podľa vzorového riešenia. Z toho vyplýva, že na menej ako 7 otázok nevieme nugety usporiadať.

Nakoniec ešte bodovanie:

- Riešenia, ktoré položili najviac 7 otázok získali maximálne 15 bodov.
- Riešenia, ktoré položili najviac 8 otázok získali maximálne 11 bodov.
- Za každú ďalšiu položenú otázku bolo o 2 body menej.

Ďalej sa dali body stratiť za chyby v programe, chýbajúci alebo zlý odhad maximálneho počtu položených otázok (1 bod dolu), slabý popis (1 až 2 body dolu) a za nie veľmi ideálne konštrukcie vo vašom programe (1 až 2 body). Keďže krabičiek bolo 5, tentoraz som nebral ohľad na časovú a pamäťovú zložitosť, nakoľko tá sa dá pokladať za konštantnú. Takisto som nevyžadoval dôkaz toho, že krabičky nie je možné zoradiť v najhoršom prípade na menej ako 7 otázok.

Listing programu:

```
Program ZdrtenyJurko; {by Poko}

var p: array[1..5] of integer;
    i: integer;

procedure Vymen(a,b: integer);
  var i: integer;
begin
  i:=p[a]; p[a]:=p[b]; p[b]:=i;
end;

function Ask(a,b: integer): boolean;
  var ans: integer;
begin
  writeln('V ktorej krabicke je vacsi nuget, v ',p[a],'. alebo ',p[b],'.?');
  write('> '); readln(ans);
```

```
  if ans = p[a] then Ask:=true else Ask:=False;
end;

begin
  for i:=1 to 5 do p[i]:=i;
  if Ask(2,1) then Vymen(1,2);
  if Ask(4,3) then Vymen(3,4);
  if Ask(3,1) then begin Vymen(1,3); Vymen(2,4); end;
  {a ideme sa vetvit :) }
  if Ask(3,5)
    then begin
           if Ask(5,4) then Vymen(4,5);
           if Ask(2,4) then begin
                              if Ask(3,2) then Vymen(2,3);
                            end
                       else begin
                              Vymen(2,3); Vymen(3,4);
                              if Ask(5,4) then Vymen(4,5);
                            end;
         end
    else begin
           Vymen(4,5); Vymen(3,4);
           if Ask(1,3) then begin
                              if Ask(2,4) then begin
                                                 if Ask(3,2) then Vymen(2,3);
                                               end
                                          else begin
                                                 Vymen(2,3); Vymen(3,4);
                                                 if Ask(5,4) then Vymen(4,5);
                                               end
                            end
                       else begin
                            Vymen(2,3); Vymen(1,2);
                            if Ask(4,3) then begin
                                               Vymen(3,4);
                                               if Ask(5,4) then Vymen(4,5);
                                             end;
                            end;
         end;
   write('Nugety su od najvacsieho po najmensi v krabickach ',p[1]);
   for i:=2 to 5 do write(', ',p[i]);
   writeln('.');
end.
```

3. Chudáci zajačikovia

opravoval Martin
(max. 15 bodov)

Chudáci zajačikovia, zasa ostanú hladní, pretože takmer všetky vaše programy boli správne, a tak drotár mohol urobiť plôtiky pre všetky mrkvičky. Ak sa vášmu programu podarilo všetky mrkvičky ochrániť, mohli ste získať najviac 15 bodov. Ak ste však mali v programe drobné chyby, alebo bol váš program príliš pomalý, prípadne ste zabudli k nemu napísať výstižný

popis alebo ste zabudli na odhad časovej alebo pamäťovej zložitosti, mohli ste zopár bodov stratiť.

Vieme, že každý plôtik má tvar obdĺžnika, ktorého strana je celočíselným násobkom strany jedného štvorčeka. Preto dĺžka každého takého plôtika je vždy párne číslo. Teraz sa pokúsme zistiť maximálnu a minimálnu dĺžku plôtika. Plôtik bude mať zrejme minimálnu dĺžku vtedy, keď sa bude čo najviac podobať štvorcu. Teda ak $N = A^2$, kde A je nejaké prirodzené číslo, potom plôtik bude mať tvar štvorca $A \times A$ s obvodom $2(A + A)$. Ak $A^2 < N \leq A(A+1)$, potom sa nám mrkvičky do štvorca $A \times A$ nezmestia, preto plôtik bude musieť chrániť obdĺžnik tvaru $A \times (A + 1)$ s obvodom $2(A + A + 1)$. Nakoniec ak $A(A + 1) < N < (A + 1)(A + 1)$, potom sa nám mrkvičky nezmestia už ani do obdĺžnika tvaru $A \times (A + 1)$, preto plôtik bude musieť chrániť trochu väčší štvorec $(A+1) \times (A+1)$ s obvodom $2(A+A+2)$. Naopak plôtik bude najdlhší vtedy, keď bude ním obklopený obdĺžnik čo najužší a čo najdlhší. Keďže najmenšia šírka obdĺžnika je 1, dĺžka plôtika bude $2(N + 1)$.

Poznáme maximálnu a minimálnu dĺžku plôtika, ukážeme, že vieme dosiahnuť každú párnu dĺžku plôtika väčšiu ako minimálnu a menšiu ako maximálnu. Nech plôtik s minimálnou dĺžkou chráni záhradku s N mrkvičkami s rozmermi $A_0 \times B_0$. Keďže mrkvičky sú na záhradke poukladané do súvislého útvaru, vieme ich popresúvať tak, aby v každom štvorčeku prvého riadku a v každom štvorčeku prvého stĺpca boli nejaké mrkvičky. Potom počet mrkvičiek nachádzajúcich sa v prvom riadku alebo v prvom stĺpci je rovný $X_0 = A_0 + B_0 - 1$ a počet mrkvičiek nenachádzajúcich sa ani v prvom riadku ani v prvom stĺpci rovný $Y_0 = N - X_0 = N - A_0 - B_0 + 1$. Ak $Y_0 > 0$, zoberieme nejakú mrkvičku ktorá sa nenachádza ani v prvom riadku, ani v prvom stĺpci (taká určite existuje) a presunieme ju na koniec prvého riadku, čím chránený obdĺžnik predĺžime o jeden štvorček na obdĺžnik tvaru $A_1 \times B_1$, kde $A_1 = A_0 + 1$ a $B_1 = B_0$, v tomto novom obdĺžniku bude platiť, že $X_1 = A_1 + B_1 - 1 = A_0 + B_0 = X_0 + 1$ a $Y_1 = N - A_1 - B_1 + 1 = N - A_0 - B_0 = Y_0 - 1$. Tento postup opakujeme kým $Y > 0$. Každým zopakovaním postupu sa Y zmenší o 1, X a A sa zväčšia o 1 a B ostane nezmenené, takže po Y_0 opakovaniach bude konečné $Y = Y_0 - Y_0 = 0$, $X = X_0 + Y_0 = (A_0 + B_0 - 1) + (N - A_0 - B_0 + 1) = N$, $A = A_0 + Y_0 = A_0 + (N - A_0 - B_0 + 1) = N - B_0 + 1$ a $B = B_0$. Inými slovami, dostaneme obdĺžnik tvaru $A \times B$, v ktorom všetky mrkvičky budú v prvom riadku alebo v prvom stĺpci. Obvod takého obdĺžnika je $2(A + B) = 2((N - B_0 + 1) + B_0) = 2(N + 1)$. Vyšli sme z obdĺžnika s minimálnym obvodom, v každom kroku sme zmenili práve jeden rozmer obdĺžnika o 1, takže obvod obdĺžnika sa v každom kroku zmenil práve o 2 a dostali sme sa k obdĺžniku s maximálnym obvodom. Keďže minimálny aj maximálny obvod sú párne čísla, ukázali sme, že vieme nájsť obdĺžnik pre každý plôtik párnej dĺžky kratší ako najdlhší a dlhší ako najkratší plôtik.

Napriek tomu, že výpočet dĺžok najkratšieho a najdlhšieho plôtika má konštantnú časovú zložitosť, celková časová zložitosť algoritmu je $O(N)$, pretože musíme vypísať $O(N - \sqrt{N}) = O(N)$ riešení. Pamäťová zložitosť je konštantná.

Listing programu:

```
var
  N,a,b,i: Integer;
begin
  Writeln('Zadaj pocet mrkviciek:');
  Readln(N);
  a := Trunc(Sqrt(N)); b := a;

  if (n > a*b) then a := a+1;
  if (n > a*b) then b := b+1;
```

```
  Writeln('Drotar ma vyrobit plotiky dlzky:');
  for i := a+b to n+1 do Writeln(2*i);
end.
```

4. Z vesmíru

opravovala Meri
(max. 15 bodov)

Buúúm treéésk, dve lodičky sa zrazili. Zdá sa, že vedenie nepoužilo ani jeden z vašich programov. Tie boli totižto skoro všetky dobre. Nuž, komu niet rady, tomu niet pomoci. A aby vám nebolo ľúto, tak sa (okrem škodoradosti) môžete vytešovať aj z nejakých tých bodov.

Za programy pracujúce s časovou zložitosťou $O(MN)$ ste mohli získať 7 bodov a za programy s časovou zložitosťou $O(M \log N)$ ste mohli získať 15 bodov. Bod sa dal stratiť, ak ste sa pomýlili o ± 1 pri výpise najkratšej vzdialenosti, alebo nedodali popis programu.

Nuž, a ako pracuje vzorák (niektorí tomu hovoria aj vzorové riešenie)? Vezme si jeden riadok fotografie, nájde poslednú jednotku a nulu. Tieto dve číselká odpočíta a dostane vzdialenosť lodí v riadku. Hľadanie poslednej jednotky prebieha takto: na začiatku vie, že posledná jednotka je niekde medzi prvým a posledným kúskom fotografie. Tak sa pozrie do stredu a ak tam je jednotka, vie, že posledná jednotka je až niekde za stredom. Ak je tam 0 alebo 2 tak vie, že jednotka bude naľavo od stredu. Teraz vie, na ktorej polovici jednotka je. Tak si ju rozdelí na dve časti. Toto robí dovtedy, kým je čo deliť. Poslednú nulu hľadá podobne. Takéto vyhľadávanie pracuje v čase $\log_2 N$, a keďže ho musíme vykonať pre každý riadok, celková zložitosť programu je $O(M \log N)$.

Tááááák a to je koniec rozprávky deti, už by ste mali ísť spať, pretože čím skôr pôjdete spať, tým skôr tu budú prázdniny.

Listing programu:

```
program Z_vesmiru;
const
 MaxX=10;
 MaxY=10;
var
 M,N:integer;
 A:array[0..MaxX,0..MaxY] of integer;
 l,r,minimum:integer;
 Prva,Druha:integer;

procedure Vstup;
var
  i,j:integer;
  f:text;
begin
  assign(f,'Z4.txt');
  reset(f);
  readln(f,M,N);
  for i:=0 to M-1 do for j:=0 to N-1 do
    read(f,A[i][j]);
  close(f);
end;


begin
  Vstup;
```

```
  minimum:=N;
  while (M>0) do begin
    M:=M-1;
    l:=0; r:=N-1;
    while (l+1<>r) do begin
      if (A[M][(l+r) div 2]=1) then l:=(l+r) div 2
         else r:=(l+r) div 2;
    end;
    Prva:=l;
    l:=0; r:=N-1;
    while (l+1<>r) do begin
      if (A[M][(l+r) div 2]=2) then r:=(l+r) div 2
       else l:=(l+r) div 2;
    end;
    Druha:=l;
    if(minimum>Druha-Prva) then minimum:=Druha-Prva;
  end;
  writeln('Lodicky su od seba vzdialene ',minimum,' km');
  readln;
end.
```

5. Zkzymdaa

opravovala Ňaňka
(max. 15 bodov)

Ako obvykle v poslednej sérií, došlo veľmi málo riešení. Neviem, či je to tou poslednou sériou, alebo obtiažnosťou tohto príkladu. Možno obidve...

Príklad mal, asi to nebolo vidieť priamo zo zadania, dve časti. Z nich prvá bola výrazne jednoduchšia a bola hintom k druhej časti. Poďme sa venovať najprv tej.

- vyriešenie – 3 body
- popis alebo program – 2 body

A ako sa dala táto časť rozlúštiť? Dôležité bolo si všimnúť, že sú tam nejaké veľké písmená. Tie sú začiatočnými písmenami jednotlivých slov. Slová už potom ľahko zistíme, keď poprehadzujeme písmená vo vzniknutých zhlukoch písmen. A budeme ich prehadzovať vždy rovnako, v poradí prvé, posledné, druhé, predposledné... Výsledok, teda nápoveda k ďalšej časti je:

```
Zakazdym Ked Sa Fanusik Sifier A Tajnych Odkazov Dostane K Sprave
Ktoru Treba Rozlustit Aspon Na Kratky Okamih Pokazi Jeho Radost
Skutocnost Ze Spravu Rozlustit Nevie Potom Je Stastny Ako Blcha Ked
Sa Mu Dostane Do Ruk Aj Ta Najnepotrebnejsia Indicia Ktora By Mu
V Desifrovani Mohla Pomoc Ved Komu By Bolo Na Co Dobre Vediet Ze Ma
Pred Sebou N Schodov A Na Jeden Krok Moze Prejst Jeden Alebo Dva
```

A teraz hor'sa na druhú, výrazne zložitejšiu časť. Dalo sa za ňu získať až 10 bodov a letmým nahliadnutím do výsledkovej listiny zistíte, že sa tam nenachádza veľa väčších čísel!

A čo nám hovorí nápoveda? Hovorí o nejakých N schodoch, z ktorých môžeme na jeden krok prejsť jeden alebo dva. Môžeme vyskúšať zobrať z textu každé druhé alebo každé písmeno. Alebo raz tak a raz tak. Vždy však vyjde úplný nezmysel. Radšej nejdem popisovať, čo sa všetko dá ešte vymyslieť, bola to predsa len posledná séria a teda najťažší príklad :((

Takže, čo s našimi N schodmi? Vyrátame si, koľkými spôsobmi ich môžeme prejsť.

1. nula schodov môžem prejsť jedným spôsobom – $F_0 = 1$
2. jeden schod môžem prejsť jediným spôsobom – $F_1 = 1$

3. N schodov môžeme prejsť tak, spravíme krok o jeden schod a potom nejako prejdeme $N-1$ schodov, alebo spravíme krok o dva schody a potom nejako prejdeme $N-2$ schodov – $F_N = F_{N-1} + F_{N-2}$

Nepripomína vám to niečo? Že vraj Fibbonaciho čísla! Už sme skoro v cieli. Ešte nám ostáva zistiť, že čo s nimi. To už myslím nie je veľa možností. To správne riešenie vyznačí prvé písmeno a ďalej postupne písmená také, že od posledného vyznačeného vzdialené F_i písmen. Takto nám vynikne výsledná správa: `KSPseminarSNOV`

Výsledková listina po 2. sérii kategórie KSP-Z

	Meno a priezvisko	kola	Trieda		21	22	23	24	25	Σ
1.	Jakub Závodný	Gym. Grösslingová BA	2	70	15	15	15	15	15	145
2.	Michal Ďuriš	Gym. Grösslingová BA	1	67	15	12	15	15	15	139
3.	Martin Rejda	Gym. Grösslingová BA	2	69	15	15	15	7	5	126
4.	Michal Kevický	Gym. Grösslingová BA	2	68	15	14	15	7	5	124
5.	Matúš Petruľák	Gym. Grösslingová BA	1	69	15	10	15	7	5	121
6.	Miso Hrobár	Gym. Jura Hronca BA	2	63	15	10	14	15	3	120
7.	Miroslav Štolc	Gym. Párovská Nitra	2	58	14	11	15	15	5	118
8.	Michal Nánási	Gym. Jura Hronca BA	1	61	15	7	11	15	5	114
9.	Peter Perešíni	ZS Radvanská B. Bystrica	9	50	14	11	15	15	5	110
	Michal Repovský	Gym. Štefánika Trebišov	2	57	15	11	15	7	5	110
11.	Andrej Mikulík	Gym. Grösslingová BA	2	48	14	11	15	15	3	106
12.	Juraj Kollar	Gym. Jura Hronca BA	1	55	14	7	15	6	5	102
13.	František Šmitala	Gym. Farská Nitra	2	52	15	10	14	7	3	101
14.	Juraj Porubský	Gym. Farská Nitra	2	49	12	8	15	7	3	94
15.	Pavol Szórád	Gym. Farská Nitra	2	47	2	8	15	7	3	82
16.	Peter Ružička	Gym. Jura Hronca BA	1	34	10	7	15	7	5	78
17.	Stano Bustor	Gym. Jura Hronca BA	1	45	7		10	15		77
18.	Ivan Trejbal	Gym. Jura Hronca BA	2	45			14	15		74
19.	Marek Zeman	Gym. Jura Hronca BA	1	37	10	..	14	7		68
20.	Milan Šatka	Gym. Liptovský Hrádok	3	65						65
21.	Imrich Živčák	Gym. Komenského Trebišov	4	58						58
22.	Anna Kovacicova	Gym. Jura Hronca BA	1	19	12	12		7	3	53
23.	Igor Rjabinin	Gym. Farská Nitra	2	36	0	1	5	7	3	52
24.	Maroš Bajtoš	Gym. Jura Hronca BA	1	49						49
	Lukáš Poláček	Gym. K. Štúra Modra	2	49						49
26.	Rastislav Lenhardt	GMMH Liptovský Mikuláš	2	48						48
	Pavel Struhár	Gym. Jura Hronca BA	1	30			11	7		48
28.	Jozef Legeny	Gymnázium	1	46						46
29.	Tibor Blénessy	Gym. Poštová Košice	2	45						45
	Ján Valaska	Gym. Jura Hronca BA	2	45						45
31.	Martin Bies	Gym. Jura Hronca BA	2	44						44
32.	Magdaléna Jurkovičová	Gym. Golianova Nitra	3	43						43
33.	Michal Dobiš	Gym. Hollého Trnava	3	42						42
34.	Ľuboš Duda	Gym. Nová Baňa	3	41						41
35.	Marek Jančuška	Gym. Párovská Nitra	2	40						40
36.	Katarína Holešová	Gym. Haanova BA	4	39						39
37.	Michal Kottman	Gym. Jura Hronca BA	1	37						37
	Michal Kramarič	Gym. Jura Hronca BA	2	28			3	6		37
	Radoslav Zapotocky	Gym. Alejová Košice	2	37						37
	Andrea Štefánková	Gym. Jura Hronca BA	1	22				15		37
41.	Peter Ambroz	Gym. Jura Hronca BA	2	36						36
	Miroslav Jagoš	Gym. Varšavská Žilina - Vlčince	1	36						36
43.	Miloš Čiernik	Gym. Lettricha Martin	1	35						35
44.	Michal Lukáč	Gym. Alejová Košice	4	34						34
	Veronika Petráková	Gym. Jura Hronca BA	1	34						34
46.	Miroslav Priesol	Gym. Dolný Kubín	4	33						33
	Marek Tomacha	Gym. sv. Uršule BA	4	33						33
48.	Tomáš Balogh	Gym. Jura Hronca BA	2	25				7		32
	Michal Dzetkulič	Gym. P. Horova Michalovce	1	32						32
	Róbert Šimon	Gym. Hlohovec	2	32						32

	Meno a priezvisko	kola	Trieda		21	22	23	24	25	Σ
51.	Mariana Kuchynárová	Gym. Jura Hronca BA	2	31						31
52.	Katarína Gergelyová	Gym. Jura Hronca BA	1	30						30
	Matúš Svrček	Gym. Stará Ľubovňa	2	30						30
54.	Peter Goga	Gym. 17. novembra Topoľčany	2	15			2	7	5	29
	Ľubomír Kundrák	Ev. lýceum BA	1	29						29
	Martin Salaj	Gym. Čadca	3	22				7		29
	Martin Valacsai	Gym. Jura Hronca BA	1	29						29
58.	Denis Donauer	Gym. Jura Hronca BA	1	28						28
	Roman Petrinec	Gym. Jura Hronca BA	2	28						28
60.	Jana Adamková	Gym. Grösslingová BA	2	27						27
	Martin Paulech	Gym. Hlohovec	2	27						27
	Martin Slavik	Gym. Jura Hronca BA	2	27						27
63.	Peter Čunderlík	Gym. Jura Hronca BA	2	25						25
	Ján Lalinský	Gym. Varšavská Žilina - Vlčince	1	25						25
	Robert Sasak	SPŠE Piešťany	4	25						25
	Ondrej Székely	GLN Tomášikova BA	3	25						25
67.	Štefan Krištofík	SPŠ Levice	3	24						24
	Jakub Šimko	Gym. Jura Hronca BA	1	24						24
69.	Igor Andruška	SPŠ Levice	1	16				7		23
70.	Michal Beňo	Gym. Kremnica	2	21						21
71.	Vojtech Villaris	Gym. Jura Hronca BA	1	19						19
72.	Peter Ivanov	Gym. Jura Hronca BA	1	17						17
	Lucia Kuklicová	Gym. Jura Hronca BA	2	17						17
74.	Michal Lipták	Gym. Jura Hronca BA	4	16						16
	Daniela Mikušová	Gym. Jura Hronca BA	1	16						16
76.	Rastislav Halamíček	Gym. Jura Hronca BA	2	0	4			7		11
77.	Peter Balko	Gym. Jura Hronca BA	1	0	1			7		8
78.	Martin Hudec	Gym. Jura Hronca BA	1	6						6
	Michal Mraz	Gym. Jura Hronca BA		6						6
	Marian Ridilla	Gym. Jura Hronca BA	1	6						6

1