ACM Programming Contest
kódování čeątiny

Katedra počítačů ČVUT FEL & Czech ACM Chapter

Soutěľ v programování 2000

Váľení soutěľící,

Jak jistě víte, minulý měsíc se v Praze konalo zasedání Mezinárodního měnového fondu a Světové banky. Do Prahy se chystalo mnoho takových, kterým ąlo pouze o vlastní zviditelnění a touhu projevit svou agresivitu (a tím teď opravdu nemáme na mysli ctihodné pány finančníky). Naątěstí naąe Policie svou úlohu tentokráte zvládla, alespoň se to vąeobecně soudí. Moľná vás nepřekvapí, ľe za organizací této obrovské akce, která nemá v naąich zemích obdoby, je mimo jiné skryt také rozsáhlý softwarový systém, který umoľňuje policistům efektivně pracovat a rychle se rozhodovat a reagovat na aktivity demonstrantů.

Máte nyní jedinečnou příleľitost vyzkouąet si, co obnáąí vytváření programů pro takové příleľitosti. Aby reakce Policie byla opravdu dostatečně rychlá, je důleľitá především časová efektivita pouľitých programů. Pamě»ová spotřeba je aľ na druhém místě vzhledem k mnoľství peněz vyčleněných na tuto akci. Vąechny programy, které máte vytvořit, budou číst vstupní data ze standardního vstupu a výsledky vypisovat na standardní výstup. V obou případech je nutné přesně dodrľet předepsaný formát, aby byla moľná bezchybná komunikace mezi různými policejními jednotkami. Je-li vstupem více hodnot na jednom řádku a není-li uvedeno jinak, jsou čísla oddělena právě jednou mezerou. Vstup ani výstup neobsahuje ľádné přebytečné mezery, není-li v zadání přísluąné úlohy uvedeno jinak.

Protoľe některé případy vyľadují hromadné zpracování více vypjatých situací, musí být vámi vytvořené programy schopny řeąit více instancí úlohy najednou. Obvykle je na prvním řádku uveden počet úloh, pak následují postupně vąechna zadání. Přesný formát je uveden u kaľdého příkladu. Pokud se v zadání vyskytují celá čísla a pokud není u konkrétní úlohy uvedeno něco jiného, pak je velikost čísel volena vľdy tak, aby se vąechna tato čísla i případné výsledky veąly do standardního typu *integer*.

Nyní nám jiľ zbývá pouze vám popřát mnoho úspěchů při řeąení těchto úloh. Doufáme, ľe Vaąe dojmy z této soutěľe budou alespoň tak dobré, jako dojem, který udělala Policie minulý měsíc na větąinu obyvatel.

Vaąi organizátoři

Tato stránka, včetně vzorových řeąení a testovacích vstupů, je k dispozici ke staľení jako archiv `ctu2000.tar.gz` (9.3MB).

d Zpětné inľenýrství
g Gnome
h Herci
l Linie
p Počítáme ponorky
r Rozejděte se!
s Praľská spojka
t Tokeny

Zpětné inľenýrství

debug.c, debug.p, debug.C
debug.in, debug.out

Policistům se nedávno podařilo zachytit fragment zdrojového kódu, který si mezi sebou předávaly anarchistické skupiny *MCA* a *MBI*. Tento fragment můľete vidět níľe, pro přehlednost je kromě původního jazyka (anarchisté pouľívají zásadně *C*) uveden také přepis do jazyka *Pascal*.

```
int main()
{ int u,v,k,t ;
  scanf("%d %d",&u,&v) ;
  for (k=0;!(u%2)&&!(v%2);k++)
    { u/=2 ; v/=2 ; }
  if (u%2) t=-v ;
  else t=u/2 ;
  while(t) {
    while(!(t%2)) t/=2 ;
    if (t>0) u=t ; else v=-t ;
    t=u-v ;
  }
  while(k-->0) u*=2 ;
  printf("%d\n",u) ;
  return 0 ;
}
```

```
program zahada;
var u,v,k,t:integer;
begin
  readln(u,v); k:=0;
  while(u mod 2=0)and(v mod 2=0) do
    begin u:=u div 2; v:=v div 2; k:=k+1; end;
  if(u mod 2<>0) then t:=-v
  else t:=u div 2;
  while(t<>0) do begin
    while(t mod 2=0) do t:=t div 2;
    if(t>0) then u:=t else v:=-t;
    t:=u-v;
  end;
  while(k>0) do begin u:=u*2; k:=k-1; end;
  writeln(u:1);
end.
```

Na první pohled je zřejmé, ľe program čte ze vstupu dvě čísla, pro která spočítá nějaký výsledek. Bohuľel se vąak dosud nepodařilo proniknout do tajů výpočetního procesu, a tak nevíme přesně, jaká je souvislost mezi vstupem a výstupem. Pro přehlednost si vąak realizovanou funkci označíme jako *f*. Definičním oborem této funkce bude mnoľina celých kladných čísel. Nech» $f(a,b)$ je hodnota, kterou program vypíąe, pokud na jeho vstup zadáme celá čísla a a b. Jestliľe program neskončí nebo skončí s chybou, není hodnota $f(a,b)$ pro přísluąný vstup definována. Abychom mohli chování funkce lépe sledovat, je třeba vytvořit funkci inverzní, tedy nalézt taková čísla a a b, pro která je $f(a,b)$ rovno předem dané hodnotě.

Specificace vstupu:

První řádek vstupního souboru obsahuje celé kladné číslo Z, za kterým následuje postupně Z zadání. Kaľdé zadání je tvořeno jediným řádkem obsahujícím dvě celá čísla N a M, $0 < N,M <= 1000000$. Čísla jsou oddělena mezerou.

Specifikace výstupu:

Pro kaľdé zadání vypíąe program na výstup jediný řádek se dvěma čísly U a V oddělenými mezerou. Tato čísla musí splňovat podmínku $1 <= V < U < N$. Jestliľe existuje více dvojic vyhovujících zadání, vypiąte ten případ, kdy je U maximální. Pokud existuje více řeąení se stejným maximálním U, vypiąte to, kdy je maximální V. Jestliľe neexistuje ľádná dvojice čísel, která by zadání vyhověla, vypiąte na řádek větu "`Reseni neexistuje.`".

Příklad vstupu:

```
2
11 2
2 10
```

Příklad výstupu:

```
10 8
Reseni neexistuje.
```



Gnome

gnome.c, gnome.p, gnome.C
gnome.in, gnome.out

Jednou z nejméně příjemných věcí v ľivotě policisty během občanských nepokojů je čekání. Představte si několik hodin strávených v deąti čekáním na to, kterým směrem se pohne rozlícený dav. Svým způsobem je to horąí neľ v čekárně u zubaře, kde obvykle alespoň neprąí. Proto vedení Policie rozhodlo, ľe vąechny policisty vybaví vodovzdorným herním automatem, který bude simulovat jednu z variant legendární hry Gnome, tzv. *Same-Gnome*.

Vaąím úkolem je napsat simulátor stejnojmenné hry, který by byl dostatečně jednoduchý, aby mohl být implementován v přenosné verzi počítače PMD-85 (takzvaný *PMD-Pilot*). Pravidla jsou poměrně jednoduchá. Hrací pole se skládá z *N x M* políček, na kaľdém políčku je umístěn právě jeden kámen určité barvy. Cílem hry je odstranit co nejvíce kamenů a především získat přitom co nejvíce bodů. Pravidla pro odstraňování kamenů jsou následující:

- Lze odstranit pouze takový kámen, který sousedí s alespoň jedním kamenem stejné barvy (sousední kameny jsou nahoře, dole, vlevo a vpravo, ale ne ąikmo).
- Při odstranění kamene je současně odstraněna celá souvislá oblast sousedních kamenů stejné barvy.
- Hrací pole se nachází v gravitačním poli, tzn. po odstranění kamenů propadnou výąe umístěné kameny dolů, takľe zaplní vzniklý prázdný prostor.
- Pokud po odstranění kamenů a případném propadnutí výąe poloľených kamenů vzniknou prázdné sloupce, jsou sloupce napravo posunuty doleva tak, aby prazdné sloupce zanikly.
- Jestliľe je v rámci jednoho tahu odstraněno *K* kamenů, získá hráč $(K\text{-}2)^2$ bodů.
- Pokud jsou během hry odstraněny popsaným způsobem vąechny kameny, získá hráč dodatečnou prémii ve výąi 1000 bodů.

Specificace vstupu:

První řádek vstupního souboru obsahuje celé kladné číslo *Z*, za kterým následuje postupně *Z* zadání. Kaľdé zadání začíná řádkem obsahujícím dvě celá čísla *C* a *R* (1 <= *R*<150, 1 <= *C*<300) oddělená mezerou, která udávají počet sloupců a řádků hracího pole. Poté následuje *R* řádků popisujících postupně řádky hracího pole shora dolů. Na kaľdém z těchto řádků je právě *C* písmen velké abecedy, tedy znaků od `A` do `Z`, které udávají různé barvy kamenů na daném řádku postupně zleva doprava.

Za popisem hracího pole následuje seznam tahů, které hráč provedl, tedy souřadnice kamenů, které mají být odstraněny. Nejprve je na samostatném řádku uveden počet tahů *M*. Poté následuje *M* řádků, z nichľ kaľdý obsahuje dvě celá čísla *I* a *J* oddělená mezerou. *I* (1 <= *I* <= *C*) určuje číslo sloupce vybraného kamene zleva počínaje od jedné, *J* (1 <= *J* <= *R*) číslo řádku zespoda počínaje také od jedné.

Specifikace výstupu:

Vaąím úkolem je simulovat celou hru a odstraňovat souvislé oblasti kamenů stejné barvy podle sekvence zadané na vstupu. Pokud na dané pozici není ľádný kámen nebo pokud nelze

vybraný kámen odstranit, tah ignorujte a pokračujte daląím v řadě. Pro kaľdé zadání musí program vypsat tento text:

```
"Game over!
Score dosazene v teto hre je S bodu.
Byli bychom radi, kdybyste si zahrali jeste jednou.
Prejete si hrat znovu?
Prijemnou zabavu Vam preje firma ACMTENDO.",
```

kde *S* je počet bodů získaný podle uvedených pravidel. Kaľdá věta je na samostatném řádku.

Příklad vstupu:

```
2
3 2
GZZ
ZZG
2
2 1
2 1
6 3
ABABAB
BABABA
ABABAA
11
6 1
5 1
4 1
3 1
2 1
1 1
2 1
3 1
4 1
5 1
6 1
```

Příklad výstupu:

```
Game over!
Score dosazene v teto hre je 1004 bodu.
Byli bychom radi, kdybyste si zahrali jeste jednou.
Prejete si hrat znovu?
Prijemnou zabavu Vam preje firma ACMTENDO.
Game over!
Score dosazene v teto hre je 5 bodu.
Byli bychom radi, kdybyste si zahrali jeste jednou.
Prejete si hrat znovu?
Prijemnou zabavu Vam preje firma ACMTENDO.
```



Herci

herci.c, herci.p, herci.C
herci.in, herci.out

Policisté, stejně jako kaľdý normální člověk, nemohou pouze pracovat, ale musí mít také nějaké zábavné koníčky. Proto zaloľila skupina psovodů amaterský divadelní soubor nazvaný *Kynologický soubor italského národního divadla a literatury* (*KSINDL*). Tito ochotníci hrají veąkerá představení s ohromnou vervou a osobním nasazením. Bohuľel vąak doąlo k nepříjemné situaci, kdy byla značná část divadelního souboru povolána na zásah proti neukázněným demonstrantům, a to právě v den, kdy měla být veřejná premiéra jejich nové hry. Co se dá dělat, práce je práce, proto muselo jít divadlo stranou a někteří členové se tak premiéry nemohou zúčastnit. Naątěstí, jako v kaľdém správném ochotnickém divadle, mnoho herců umí nejen svou roli, ale také několik daląích rolí, které mají zaľité natolik, ľe by byli schopni je sehrát. Bohuľel vąak není vľdy jednoduché najít přijatelné řeąení, proto by

policisté uvítali program, který by zhodnotil nastalou situaci a rozhodl, zda je moľné představení v dané sestavě sehrát.

Specificace vstupu:

První řádek vstupního souboru obsahuje celé kladné číslo Z, za kterým následuje postupně Z zadání. Kaľdé zadání začíná řádkem obsahujícím tři čísla oddělená mezerou, jsou to po řadě počet herců v souboru (H, $1 <= H <= 100$), počet rolí ve hře, která se má večer sehrát (R, $1 <= R <= 60$), a počet herců, kteří se nemohou premiéry účastnit ze sluľebních důvodů (A, $0 <= A <= H$). Poté následuje právě H řádků, kaľdý z nich obsahuje jedno jméno herce. Jméno herce je tvořeno pouze malými a velkými písmeny, kterých je vľdy minimálně 1 a maximálně 30. Velká a malá písmena se rozliąují. Dále následuje R řádků reprezentujících role, kaľdý z nich obsahuje jméno jedné role. Pro toto jméno platí stejná pravidla jako pro jména herců.

Po výpisu rolí následuje seznam, kterou roli je který herec schopen zahrát. Pro kaľdého herce je uvedeno jeho jméno, dále na stejném řádku jedna mezera, číslo U_i, které představuje počet rolí, mezera a dále následuje $U_i$názvů rolí oddělených mezerami. Celý seznam pro jednoho herce je na jediném řádku, daląí herec následuje na daląím řádku. Seznam má právě H prvků, tedy kaľdý herec je zde uveden právě jednou, ale nemusí to nutně být ve stejném pořadí, v jakém byla zapsána jména herců.

Za vąemi seznamy následuje ve vstupním souboru seznam herců, kteří nemohou hrát. Ten je tvořen právě A řádky, na kaľdém je jedno jméno herce.

Specifikace výstupu:

Program vypíąe právě jeden řádek pro kaľdé zadání. Jestliľe je za daných okolností moľno hru sehrát, bude na řádku věta `"Premiera bude!"`, jinak vypiąte větu `"Zatraceni demonstranti!"`.

Příklad vstupu:

```
1
5 3 2
Pepan
Venda
Cvalda
Bozka
Vendula
Kecal
Jenik
Marysa
Bozka 1 Marysa
Vendula 1 Marysa
Pepan 2 Kecal Jenik
Venda 3 Jenik Marysa Kecal
Cvalda 1 Kecal
Bozka
Venda
```

Příklad výstupu:

```
Premiera bude!
```



Linie

linie.c, linie.p, linie.C
linie.in, linie.out,

Tato úloha (včetně řeąení a testovacích dat) byla převzata z archivu. My jsme pouze dopsali českou legendu.

Policisté se při své práci často mohou dostat i do velmi nebezpečných situací, zvláątě při podobných akcích, jako bylo zasedání MMF a SB. A proto, aby se při jejich práci pokud moľno předeąlo jakýmkoli zraněním, jsou Policisté vybaveni bezpečnostími *Směrnicemi pro Mimořádně Rychlé Tlačenice*, ve zkratce *SMRT*.

Často se stávalo, ľe kdyľ na policisty letěl nějaký předmět vrľený demonstranty, první policista mu uhnul, ale druhý, který přes prvního neviděl, byl zasaľen. A proto jedna ze směrnic určených pro policisty, kteří mají za úkol blokovat demonstrantům cestu, říká, ľe policisté nesmí stát v řadách, aby se nemohli stát snadným cílem pro létající předměty velkých ráľí typu dlaľební kostky. Jak známo, vąechny směrnice se musí dodrľovat. Je tedy třeba také toto dodrľování kontrolovat a případné poruąení kázně okamľitě hlásit. A protoľe velící důstojníci nemohou být vąude, potřebuje Policie nástroj, který by tato poruąení odhalil sám. A to bude váą úkol.

U kaľdého zásahu bude přítomna kamera, která bude snímat policejní kordon shora tak, ľe bude moľno kaľdému policistovi přiřadit souřadnici v myąlené souřadné soustavě. Kaľdý policista bude tak jednoznačně identifikován právě jedním párem celých čísel X a Y, kde $0 <= X, Y <= 9999$. Protoľe rohové pozice jsou ze strategického hlediska nevýhodné, ľádný policista nikdy nestojí na souřadnici 0, 0.

Vaąím úkolem je napsat program, který pro kaľdé zadání načte skupinu souřadnic pro kordon policistů a zjistí, zda někde nestojí na jedné přímce více neľ dva policisté. Vzhledem k tomu, ľe tento předpis je poměrně nový a jeątě ne úplně zaľitý, můľe se stát, ľe bude existovat více přímek se třemi a více policisty, ale vzhledem k jejich vysoké morálce a dobrému výcviku se dá předpokládat, ľe jich v jedné řadě nebude nikdy více neľ deset.

Specificace vstupu:

Vstup obsahuje několik zadání. Kaľdé zadání začíná na nové řádce, je tvořeno souřadnicemi 3 aľ 300 bodů (včetně) a ukončeno párem nul (`0 0`). Kaľdý bod je zadán párem celých čísel v rozsahu 0 aľ 9999 (včetně). Jedno zadání můľe být rozděleno i na více řádků, v tomto případě vąak nebude nikdy rozdělen pár souřadnic, dělení bude vľdy provedeno mezi celými páry souřadnic. Celý vstup bude opět ukončen jednou daląí dvojicí nul.

Specifikace výstupu:

Na výstupu bude věta "`Smernice byla dodrzena.`", pokud neexistuje ľádná trojice policistů stojících na jedné přímce. Jinak program vypíąe na výstup větu "`Tito policiste porusuji smernici:`" a na kaľdém z dalších řádků bude vľdy výpis vąech policistů stojících na jedné přímce. Takto musí být postupně uvedeny vąechny přímky, na kterých jsou tři a více policistů. Souřadnice musí být v rámci jednoho řádku seřazeny nejprve podle x-ové souřadnice, a v případě shody pak podle souřadnice y. Vąechny body jsou uzavřeny v závorkách a jejich sloľky jsou oddělené čárkou, bez jakýchkoli mezer. Samotné body nejsou kromě závorek navzájem nijak odděleny. Řádky jsou řazeny podobně jako souřadnice na řádcích, tedy podle prvního bodu, a pokud ten je stejný, pak postupně podle dalších bodů. Vąimněte si, ľe jeden policista se můľe vyskytovat i na více řádcích. Za kaľdým zadáním (včetně posledního) vytiskněte jeden prázdný řádek.

Příklad vstupu:

```
5 5 8 7 14 11 4 8 20 15
12 6 18 21 0 0
5 5 8 8 14 13 0 0
5 5 25 17 20 23 10 11 20 14 15 11 0 0
0 0
```

Příklad výstupu:

```
Tito policiste porusuji smernici:
(4,8)(8,7)(12,6)
(5,5)(8,7)(14,11)(20,15)
(12,6)(14,11)(18,21)

Smernice byla dodrzena.

Tito policiste porusuji smernici:
(5,5)(10,11)(20,23)
(5,5)(15,11)(20,14)(25,17)
```

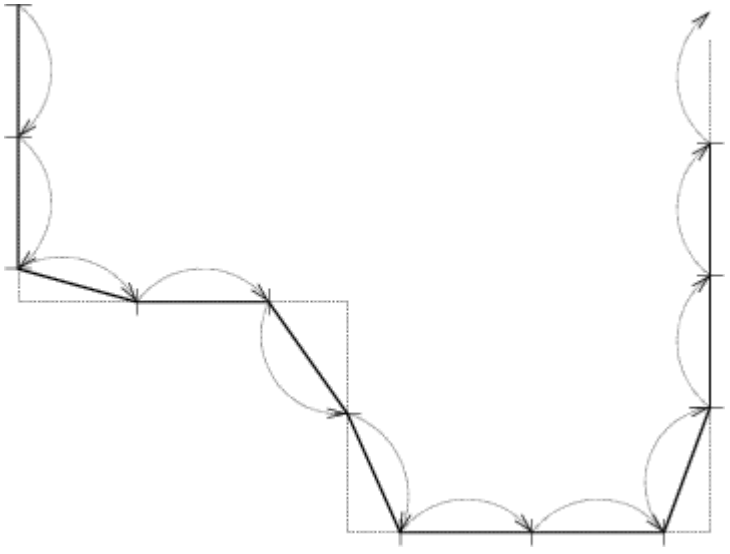


Počítáme ponorky

ponork.c, ponork.p, ponork.C
ponork.in, ponork.out,

Protoľe hlavní část zasedání se konala v Kongresovém Finančním Centru (KFC), bylo samozřejmě nutné zajistit tomuto místu kvalitní ochranu. Jak jste jistě sami viděli, přes den byla tato stavba hlídána více neľ dostatečně. Nás ale bude zajímat poněkud vąednějąí noc, kdy i větąina demonstrantů jiľ ulehla ke spánku.

Je jasné, ľe i v noci je třeba budovu ohlídat. Bývá k tomu určeno několik speciálně ąkolených stráľných, kteří obchází KFC celou noc po různých trasách. Nejdůleľitějąí z nich vede po úzké zídce kolem celé budovy. Tato trasa byla vytyčena ze strategických důvodů, nebo» z ní je velmi dobrý výhled do celého okolí. A aby bylo celé okolí pokryto, musí být na tuto trasu nasazeno stráľných několik. Ale protoľe lidí na práci je vľdy méně, neľ by bylo třeba, vedení Policie potřebuje určit minimální počet stráľných tak, aby stále měli celé okolí budovy v dohledu. K řeąení tohoto problému je vąak třeba znát, jak je trasa dlouhá a za jak dlouho ji stráľný projde. A to právě bude vaąím úkolem.



Pro určení délky trasy se u Policie pouľívá měření tzv. *policejních normokroků* (zkratka *ponork*). Zídka má podobu rovných úseků, které na sebe navazují v pravém úhlu. Kaľdý úsek má délku nejméně jeden ponork. Pro účely měření je v policejních předpisech stanoveno, ľe stráľný jdoucí po zídce nezkracuje při změně směru krok, ale překračuje rohy tak, ľe délka jeho kroku je stále přesně jeden ponork. ©ířku zídky povaľujeme za nulovou. Pokud se celá trasa nedá projít tak, ľe posledním krokem stráľný skončí přesně v koncovém bodě zídky, počítá se zbytek jako celý krok pouze v případě, ľe je dlouhý nejméně půl ponorku. Vaąím úkolem je určit délku trasy, která vede po zídce, v ponorcích. (Ilustrační obrázek odpovídá prvnímu z příkladů vstupu.)

Specificace vstupu:

První řádek vstupního souboru obsahuje celé kladné číslo Z, za kterým následuje postupně Z zadání. Kaľdé zadání začíná řádkem obsahujícím dvě hodnoty: K a U, ($1 <= K<1000$, $1 <=$

U<5000) kde K je délka kroku stráľného (velikost ponorku) a U je počet úseků zídky. Po nich následuje U čísel určujících délky jednotlivých úseků, kaľdé číslo je na samostatném řádku.

Specifikace výstupu:

Pro kaľdé zadání vypíąe program právě jeden řádek s textem "`Strazny ujde X ponorku.`" kde X je počet ponorků potřebných k projití celé zídky.

Příklad vstupu:

```
3
4 5
9
10
7
11
15
1 3
5
3
7
4 2
6
7
```

Příklad výstupu:

```
Strazny ujde 12 ponorku.
Strazny ujde 15 ponorku.
Strazny ujde 3 ponorku.
```



Rozejděte se!

rozchod.c, rozchod.p, rozchod.C
rozchod.in, rozchod.out,

Nejoblíbenějąí větou kaľdého správného policisty je „Občané, rozejděte se! Není tady nic k vidění!". Ovąem samotná věta vľdy nepomůľe. Existují obsáhlé příručky, které popisují, jak se má policista zachovat, pokud občané neuposlechnou dobře míněné výzvy a nerozejdou se. Nejdůleľitějąí rada pro takovou chvíli říká, ľe je dobré hlouček lidí rozdělit tak, aby jich bylo pohromadě co nejméně. A o takové dělení nám půjde i v této úloze. Úkolem bude hledat různá dělení hloučku, aby si z nich policista mohl vybrat to nejsnáze realizovatelné.

Pro zjednoduąení uvaľujeme pouze takzvané „ortogonální normohloučky stejnoměrně rozlehlých občanů", coľ je termín zavedený v dnes jiľ klasické příručce na toto téma *Rozcházení se snadno a rychle*. Takový normohlouček je moľné modelovat jako čtvercový útvar sloľený z N x N čtvercových občanů stejné půdorysové velikosti. Úkolem policisty je tento hlouček rozdělit na dvě poloviny. A ideální poloviny jsou samozřejmě takové, které jsou obě stejné, a to nejen co do velikosti, ale i do tvaru.

Kromě toho je vąak také nutné uvaľovat způsob oddělení obou polovin od sebe. Existuje speciální oddělovací policejní páska určená speciálně k tomuto účelu, v policejním ľargonu se jí říká „oddělovačka". Tuto páskou je moľné rychle ovinout kolem skupiny občanů, a tak je oddělit od zbytku davu. Vzhledem k tomu, ľe oddělovačka je velmi drahá, nesmí se nikdy stříhat ani krátit. Je proto nutné najít takové rozdělení normohloučku, kdy kaľdá ze vzniklých polovin má přesně daný obvod.

Vaąím úkolem je určit počet různých způsobů, kterými je moľné rozdělit čtvercový normohlouček o velikosti strany N takovým způsobem, aby vznikly dvě části o přesně daném obvodu M. Tyto části musí být stejné, to znamená, ľe jedna musí být převeditelná na druhou

pouze pomocí operací posunu, otáčení a zrcadlení. Dělící linie může samozřejmě vést pouze po hranách „jednotkových čtverečků", aby žádný občan neutrpěl zranění.

Specificace vstupu:

První řádek vstupního souboru obsahuje celé kladné číslo *Z*, za kterým následuje postupně *Z* zadání. Každé zadání sestává ze dvou čísel *N* a *M*, 2 <= *N* <= 20, 2 <= *M* <= 60, uvedených na stejném řádku a oddělených mezerou. První číslo (*N*) udává velikost strany čtverce, druhé (*M*) je délka pásky, tedy obvod každé z polovin, které mají rozdělením vzniknout.

Specifikace výstupu:

Pro každé zadání musí program vypsat právě jeden řádek. Pokud není možné rozdělit normohlouček na dvě stejné části se zadaným obvodem, bude na řádku věta "`Rozdeleni neni mozne.`". V ostatních případech vytisknětě větu "`Existuje` *R* `ruznych moznosti.`", kde *R* je počet různých způsobů, jak hlouček rozdělit předepsaným způsobem. Dva způsoby jsou považovány za různé, pokud vzniklé poloviny jsou rozdílné, tedy pokud nelze polovinu z jednoho dělení převést na polovinu podle druhého dělení pomocí operací posunutí, otáčení a zrcadlení.

Příklad vstupu:

```
3
2 5
4 12
6 24
```

Příklad výstupu:

```
Rozdeleni neni mozne.
Existuje 1 ruznych moznosti.
Existuje 10 ruznych moznosti.
```



Praľská spojka

spojka.c, spojka.p, spojka.C
spojka.in, spojka.out,

Dobrá informovanost o aktuální situaci je pro Policii velmi důležitá. V případě poruchy telekomunikačního vedení nemají mezi sebou jednotlivá stanoviště spojení, které tak musí být nahrazeno pomocí poslů. Každý posel může vyřídit vzkaz na daném místě a v daný okamžik. Přestože poslové používají moderních dopravních prostředků (bicyklů), je přeci jen při jejich práci velice důležité znát vzdálenost, na kterou je třeba zprávu dopravit. Proto musí posel vždy volit nejkratší cestu, aby dorazil do cíle své cesty co nejdříve. Kromě toho se někdy může stát, že některé veřejné komunikace není možné kvůli občanským nepokojům použít. V krajním případě se dokonce všechna stanoviště rozpadnou na dvě nebo i více skupin, mezi kterými není žádné spojení.

Policejní složky mají velmi propracovaný projekt infrastruktury, je vždy přesně dáno rozmístění stanovišť v terénu a vzdálenosti mezi nimi. Proto je možné zmapovat vzdálenosti mezi jednotlivými stanovišti a rozhodnout, která cesta je pro případného posla nejhorší. Podle ní se potom zjišťuje, jaká aktuálnost krizového zpravodajství může být zaručena.

Specificace vstupu:

První řádek vstupního souboru obsahuje celé kladné číslo *Z*, za kterým následuje postupně *Z* zadání. Každé zadání začíná řádkem obsahujícím dvě čísla oddělená mezerou. Jsou to po řadě počet jednotek (stanovišť) *J*, 2 <= *J* <= 300 a počet existujících spojnic mezi nimi *S*, 0

<= *S* <= *J*.(*J*-1) / 2. Dále následuje *S* řádků, kaľdý z nich obsahuje právě tři čísla oddělená mezerou. První dvě čísla udávají čísla spojených stanovią» (jednotky jsou číslovány od jedné do *J*) a třetí číslo je délka spojnice mezi nimi. Je přitom moľné, aby mezi nimi existovala také kratąí cesta, která vede přes některé jiné stanoviątě. I takovou samozřejmě můľe posel vyuľít. ®ádná dvojice stanovią» se v rámci jednoho zadání nevyskytne dvakrát.

Specifikace výstupu:

Úkolem je najít takovou dvojici stanovią», mezi kterými je maximální vzdálenost v případě pouľití nejkratąí moľné cesty. Cesta můľe vést pouze po zadaných spojnicích. Pro kaľdé zadání vypíąe program větu "`Nejvetsi vzdalenost je V.`", kde *V* je délka nejdeląí cesty mezi dvěma stanoviąti, mezi kterými nelze nalézt cestu kratąí. Pokud by zadaná situace znamenala, ľe existuje alespoň jedna dvojice stanovią» taková, ľe mezi nimi neexistuje ľádné spojení, vypíąe program místo toho větu "`Bez spojeni neni veleni!`".

Příklad vstupu:

```
2
5 5
1 2 1
2 3 2
3 4 3
4 5 4
5 1 5
5 2
1 2 1
2 3 1
```

Příklad výstupu:

```
Nejvetsi vzdalenost je 7.
Bez spojeni neni veleni!
```



Tokeny

tokeny.c, tokeny.p, tokeny.C
tokeny.in, tokeny.out,

Důleľitou sloľkou přípravy na očekáváné demonstrace je pozorné sledování toho, co aktivisté chystají. V dneąní době jiľ i odpůrci globalizace běľně pouľívají globální Internet k tomu, aby proti globalizaci protestovali. Nepřekvapí, ľe i ostatní skupiny obyvatel, kteří svolávají veřejné shromáľdění, často vytvářejí webovské stránky informující o chystaných aktivitách. Tento způsob je velmi výhodný k oslovení velkého počtu lidí, ale naątěstí zároveň také pomáhá Policii, aby si udělala předběľný obrázek o tom, co se chystá.

Bohuľel je vąak na současném Internetu stránek velmi mnoho, a tak není snadné nalézt takové, na kterých jsou podobné akce svolávány. Proto bylo rozhodnuto, ľe se pro Policii vytvoří speciální program vyuľívající nejnovějąí poznatky z oblasti umělé inteligence. Úkolem tohoto programu bude procházet internetové stránky a automaticky detekovat „podezřelé" texty. Ty jsou potom nahláąeny policistům k přezkoumání. Základní činností programu je rozbor anglického textu a pokus o určení jeho přibliľného obsahu. K tomu je nutná podrobná analýza textu, nejprve lexikální (na úrovni slov a symbolů), poté syntaktická (na úrovni vět). V tomto příkladě budeme uvaľovat pouze první, lexikální část.

Jak jistě víte, lexikální analyzátor čte posloupnost znaků textu a vytváří posloupnost lexikálních elementů (tokenů). Kdyľ lexikální analyzátor přečte např. text `abc+123`, vytvoří (za předpokladu, ľe text je analyzován podle jazyka pro aritmetický výraz) tokeny [`abc`] [+] [`123`] .

Podívejme se nyní podrobněji na proces čtení znaků. Poté, co jsou přečteny znaky `a`, `b` a `c`, je rozhodnuto o tom, ľe se vytvoří token [`abc`] , na základě toho, ľe za nimi následuje znak `+`, protoľe znak + nemůľe být součástí identifikátoru. O vytvoření tokenu tedy v tomto případě rozhoduje první znak dosud nepřečtené části textu. Naproti tomu po přečtení znaku + víme s jistotou, ľe token můľe být ihned vytvořen, protoľe znaky za ním uľ na jeho vytvoření nemají vliv. Tokeny se tedy liąí podle toho, kolik znaků dosud nepřečtené části textu rozhoduje o tom, jak budou vytvořeny.

Problém je v tom, ľe internetové stránky se často mění a bylo by neefektivní po kaľdé změně provádět analýzu celého textu znovu. Proto se vľdy analyzuje pouze změněné místo a jeho okolí. Bohuľel, kvůli výąe uvedeným závislostem, pokud u nějakého tokenu dojde ke změně textu, můľe tato změna ovlivnit i tvorbu několika tokenů před ním. U kaľdého tokenu *t* proto definujeme číslo *lookback*(*t*), které udává vzdálenost nejzaząího tokenu (ve směru od tokenu *t* k začátku textu), který je při svém vytváření ovlivněn textem tokenu *t*. Pokud takový token není, bude *lookback*(*t*)=0. Vzdálenost dvou tokenů chápeme intuitivně jako počet mezilehlých tokenů plus jedna.

V naąem příkladu jsou hodnoty *lookback*jednotlivých tokenů rovny po řadě 0, 1 a 0. Vaąím úkolem je napsat program, který pro zadanou posloupnost tokenů vypočte *lookback* kaľdého tokenu.

Specificace vstupu:

Vstup se skládá z *N* zadání. Na prvním řádku vstupu je přirozené číslo *N*, *N*>0. Dále následují jednotlivá zadání. Kaľdé zadání je tvořeno posloupností tokenů. Kaľdý token je popsán na samostatném řádku dvojicí čísel *L*, *A*, *L* >= 1, *A* >= 0, kde *L* je počet znaků tokenu, *A* je počet znaků dosud nepřečtené části vstupního textu, které rozhodují o okamľiku vytvoření tokenu. Posloupnost tokenů je zakončena dvojicí 0 0, která není popisem tokenu. Délka posloupnosti a celkový počet znaků vstupního textu se vejdou do pascalského typu `integer` nebo céčkového typu `int`, jinak vąak nejsou omezeny. Velikost hodnoty *lookback* nepřesahuje u ľádného tokenu 250000.

Specifikace výstupu:

Pro kaľdé zadání vypiąte nejprve na samostatný řádek text "`Zadani X:`", kde *X* je pořadové číslo zadání počítáno od jedničky. Dále vypiąte pro kaľdý token (v pořadí podle vstupu) na samostatný řádek jediné číslo udávající *lookback* přísluąného tokenu. Na konci kaľdého zadání (včetně posledního) vytiskněte jeden prázdný řádek.

Příklad vstupu:

```
2
3 1
1 0
3 1
0 0
4 2
1 5
2 1
2 1
0 0
```

Příklad výstupu:

```
Zadani 1:
0
1
0

Zadani 2:
0
1
2
2
```

Na těchto stránkách jste návątěvníkem číslo  od 25. září 2000.