PRASK

Leták letnej časti V. ročníka

Ahojte milí riešitelia.

Sme veľmi radi, že ste sa dozvedeli o PRASKu a asi by vás zaujímalo, čo to vlastne je, ako to celé funguje a prečo by ste to mali riešiť. Na všetko z toho sa vám teraz pokúsime odpovedať.

Čo to je a pre koho je to určené?

PRASK je korešpodenčný seminár určený pre všetkých základoškolákov, ktorých zaujíma matematika, informatika alebo by sa chceli naučiť programovať. Je to súťaž zameraná hlavne pre siedmakov a starších, môžete ju však riešiť aj keď ste v nižšom ročníku.

Seminár je organizovaný študentami informatiky na Fakulte matematiky, fyziky a informatiky na Univerzite Komenského.

Priebeh súťaže

Počas roka prebiehajú dve nezávislé časti – letná a zimná. Priebeh častí je už potom úplne rovnaký. Každá časť pozostáva z dvoch sérií piatich príkladov – dvoch teoretických, jedného praktického a dvoch programátorských. Ak aj neviete programovať nezúfajte. Namiesto programátorských úloh si môžete prejsť programátorským tutoriálom, ktorý vás to naučí a naviac v ňom získate body, ktoré sa vám rátajú do PRASKu.

Na riešenie série je vyhradených niekoľko týždňov. Až do dňa odovzdania môžete doma riešiť zadané príklady. Môžete riešiť ľubovoľné príklady z danej série, nemusíte vyriešiť všetko, nemusíte vyriešiť ani celú úlohu[1]. Najneskôr do dňa odovzdania (ktorý je napísaný na zadaniach aktuálnej série) je potrebné poslať vaše riešenia pomocou webového rozhrania.

Po konci série si pozrieme vaše odovzdané riešenia a opravíme ich. Pre každý príklad je v zadaní napísané, koľko bodov sa zaň dá dostať. Samozrejme, je možné získať čiastkové body, aj keby ste nevyriešili celú úlohu, alebo by vaše riešenie nebolo úplne správne. Dokonca, ak nás prekvapíte originálnym riešením, môžete získať bonusové body. Opravené riešenie vám potom pošleme späť aj s poznámkami ohľadom vašeho riešenia.

Prečo to chcem riešiť?

Riešenie korešpodenčného seminára prináša mnoho výhod. Riešením úloh a čítaním našich vzorových riešení **objavíte a naučíte sa** mnoho nových vecí, ktoré by ste sa v škole skoro určite nenaučili. Napríklad sa môžete naučiť **programovať**. To vám potom vie **pomôcť pri prijímačkách**, či už na stredné alebo vysoké školy. Takisto vám to pomôže pri **riešení Olympiády z informatiky alebo Korešpodenčného Seminára z Programovania**. No a v neposlednom rade, pri **pohovoroch** do veľkých firiem ako Google, Facebook alebo Eset častokrát zaváži znalosť algoritmického programovania, ktoré si môžete pomocou nášho semináru trénovať.

Je tu však ešte jedna výhoda určená pre najlepších riešiteľov. Dvakrát ročne sa bude organizovať týždenné **sústredenie**. Naň pozývame niekoľko[2] najlepších riešiteľov. Na sústredení si užiješ kopec zábavy, športu, nových ľudí a možno sa aj niečo naučíš.

A samozrejme, víťazov čakajú pekné **vecné ceny** vo forme knižky, hry alebo menšej elektroniky.

Ako má vyzerať správne riešenie

To závisí od typu úlohy, ktorú riešite. Pri teoretických úlohách musí správne riešenie okrem výsledku obsahovať aj popis postupu, akým ste sa k danému výsledku dopracovali. Dôraz sa pri opravovaní dáva hlavne na tento slovný popis, ktorý by mal byť napísaný čo najzrozumiteľnejšie, aby sme si pri opravovaní nemuseli lámať hlavu. Mal by obsahovať všetky podstatné kroky, ktoré vás viedli k riešeniu.

V prípade praktických úloh sa to líši. Občas od vás chceme slovný popis, občas sa stačí dostať k nejakému tajnému heslu alebo kliknúť na správnu linku. Presný spôsob nájdete v zadaní.

[1] Aj keď budeme radi, ak sa vám to podarí.

[2] zhruba 15, ale aj nižšie umiestení riešitelia sa môžu dostať ako náhradníci

No a pri programátorských úlohách a programátorskej liahni odovzdávate iba váš program, ktorý sa vám okamžite automaticky otestuje a do pár sekúnd sa dozviete, či ste úlohu vyriešili správne. A ak nie, môžete skúsiť odovzdať opravený program znova.

A nebojte sa, ak ste ešte nikdy nespisovali postupy svojich riešení. Keď vám riešenia opravíme, napíšeme vám k nim aj komentáre, ktoré vám môžu pomôcť v riešení ďalšej série. To je najlepší spôsob, ako sa zlepšovať.

Spôsob odovzdávania

Ako prvú vec, ktorú musíte urobiť pred tým, ako budete môcť odovzdávať svoje riešenia, je **zaregistrovanie** sa na našej webovej stránke `prask.ksp.sk`. V časti `Zadania a vzoráky` nájdete okrem zadaní aj odkaz, na ktorom môžete odovzdať vaše riešenie.

Riešenie každej teoretickej úlohy má byť jeden súbor formátu **.pdf**. Ten nahráte na našu stránku a stlačíte zelené tlačítko `Submit`. Opravovať sa bude **posledné odovzdané** riešenie, takže si dajte pozor, aby ste si niečo neprepísali.

Myslím, že vytvoriť pdf súbor by pre vás nemal byť problém, ak by ste s tým predsa len problém mali, pokúste sa použiť nejaký online converter ako napríklad `www.freepdfconvert.com`.

V prípade programátorských úloh sa dá rovnakým spôsobom odovzdať zdrojový kód vášho programu, teda súbor s príponou **.cpp**, **.py** alebo **.pas**.

PRASK
V. ročník, 2018/19
Katedra základov a vyučovania informatiky FMFI UK,
Mlynská Dolina, 842 48 Bratislava

Úlohy 1. kola letnej časti

Termín odoslania riešení tejto série je pondelok **6. mája 2019**.

Teoretické úlohy

V tejto časti ťa čaká niekoľko matematickejších úloh, ktoré úzko súvisia s informatikou. Ako riešenie týchto úloh treba poslať podrobne spísaný postup toho, ako si riešil danú úlohu.

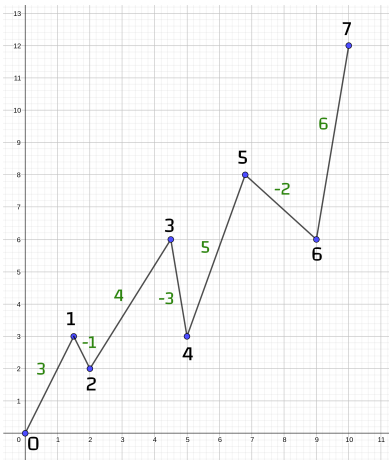
A ak by ťa to zaujímalo, podobné úlohy môžeš nájsť aj v Olympiáde v informatike, kategória B (http://oi.sk/archiv/2018/sl-2018-1-zad-B.pdf). Vrelo ti ju odporúčame riešiť tiež, naučíš sa veľa nových vecí a môžeš sa dostať aj na krajské kolo Olympiády.

1. Prefixová expedícia

15 bodov za riešenie

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Romanovi na roman.sobkuliak@trojsten.sk

Macín a Ondro sú ostrieľaní lezci a na jar sa rozhodli vyraziť za adrenalínom do Tatier. Pred expedíciou si kúpili turistickú mapu, ktorá vyzerala nasledovne:



Medzi každými dvoma vrcholmi na mape je jedno číslo, ktoré reprezentuje rozdiel nadmorskej výšky pravého a ľavého vrcholu. Macín s Ondrom sa po mape budú pohybovať zľava doprava. Nevedia však, v ktorom vrchole svoju expedíciu začnú a preto potrebujú pomoc pri výbere trasy.

Úloha

Vrcholy z mapy si očíslujeme zľava doprava 0 až N. Údaje o nadmorských výškach budeme reprezentovať pomocou poľa A, ktoré obsahuje jednotlivé rozdiely. Prvok $A[i]$ obsahuje rozdiel medzi výškou vrchola $i+1$ a i. Pole A pre našu mapu vyzerá nasledovne:

A	3	-1	4	-3	5	-2	6

Terka chalanom poradila, aby si predpočítali pre pole A *prefixové súčty*. Ide o list, na ktorého i-tej pozícii je súčet všetkých čísel, ktoré sú v pôvodnej postupnosti na pozíciách od 0 po $i-1$. Pre pole A bude preto pole P prefixových súčtov vyzerať nasledovne:

P	0	3	2	6	3	8	6	12

Všimnite si, že tento list je o jedna dlhší ako A a začína 0. Na 0-tej pozícii totiž musí byť súčet čísel od 0 po -1, čo je 0. Napríklad na 3-tej pozícii je 6, pretože súčet prvých 3 čísel poľa A je $3 - 1 + 4 = 6$.

a) (2 body) Ondru pred cestou zaujíma, kde má mapa svoje *stredové vrcholy*. Stredový vrchol je taký vrchol, pre ktorý je súčet rozdielov z najľavejšieho vrcholu na mape rovnaký ako súčet rozdielov od najpravejšieho vrcholu na mape. Presnejšie teda hľadáme všetky vrcholy k také, že

$$A[0] + \cdots + A[k-1] = A[k] + \cdots + A[N-1]$$

Napríklad na mape zo začiatku zadania sú stredovými vrcholmi vrcholy 3 a 6. Vymyslite ako *efektívne* nájsť stredové vrcholy pre pole A. Efektívnym riešením v tomto prípade myslíme také, ktoré pristupuje k jednotlivým prvkov poľa A a P čo najmenejkrát. Pošlite nám slovný popis vášho postupu. Váš popis by mal byť zrozumiteľný aj bez znalosti programovania a mal by ho vedieť odsimulovať ktokoľvek s perom a papierom.

b) (2 body) Macín a Ondro plánujú špeciálnu túru, pri ktorej skončia v rovnakej nadmorskej výške, z akej vyrážali. To znamená, že celkový súčet rozdielov nadmorských výšok na trase je rovný 0. Viete im poradiť, ako pomocou prefixových súčtov nájsť súvislý úsek v poli A, ktorého súčet je 0? Pri tejto podúlohe neposielajte algoritmus, iba popíšte, ako si v poli prefixových súčtov všimnúť úsek so súčtom 0.

c) (3 body) Chalanov pri plánovaní často zaujíma, aký je celkový rozdiel nadmorskej výšky medzi konkrétnymi dvoma vrcholmi. Napríklad pre vrcholy 2 a 5 je celkový rozdiel $4 - 3 + 5 = 6$. Všimnite si, že pre vrcholy a a b také, že $a < b$, vieme vypočítať celkový rozdiel ako súčet čiastkových rozdielov, tj. $A[a] + A[a+1] + ... + A[b-1]$. Macín ale stále prichádza s novými nápadmi na trasy a Ondrovi sa už nechce počítať rozdiely prvok po prvku. Musí to ísť predsa lepšie! Vašou úlohou je preto prísť na to, ako šikovne spočítať celkový rozdiel medzi výškami vrcholov a a b pomocou poľa prefixových súčtov P.

d) (3 body) Ako pre pole A vytvoriť pole prefixových súčtov P? Podobne ako v podúlohe a) vymyslite *efektívne* riešenie, teda také, ktoré k prvkom poľa A a P prístupi čo najmenejkrát, a pošlite nám jeho slovný popis.

e) (5 bodov) Telo sa musí vyššej nadmorskej výške vždy prispôsobiť. Výstupiť preto za jeden deň oveľa vyššie a prenocovať tam by mohlo byť nebezpečné. Chalani si určili výšku k, o ktorú chcú za jeden deň vystúpiť vyššie. Vašou úlohou bude nájsť všetky dvojice vrcholov, medzi ktorými je celkový rozdiel nadmorských výšok práve k. Napríklad na našej mape by pre $k = 4$ splňovali túto podmienku dvojice vrcholov $(2, 3)$ a $(2, 6)$. Opäť nám pošlite slovný popis čo *najefektívnejšieho* riešenia.

Ak viete programovať, niektoré z podúloh si môžete vyskúšať naprogramovať a poslať nám vaše kódy. Nedostanete síce body naviac, ale vaše programy okomentujeme a poradíme vám, čo by sa dalo spraviť lepšie alebo krajšie. Pri programátorských úlohách vám nedávame spätnú väzbu k vašim programom, preto máte teraz jedinečnú šancu.

P.S.: Ondro s Macínom a kamarátmi Tatry skutočne navštívili. Ako sa Čechom v Tatrách darilo a či ich zachraňovala horská služba, sa dozviete v článku od Ondru.

2. Rafinované vyzvedanie

15 bodov za riešenie

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Timke na szollosova.t@gmail.com

Lucka a Evka su veľmi dobré kamarátky. Nedávno mala Lucka narodeniny a Evka jej pripravila úžasnú párty. Lucka by to chcela Evke oplatiť, zabudla však, kedy má Evka narodeniny. Jediné, čo si pamätá je, že ich má v marci.

Úloha

Lucka sa nechce priznať Evke, že zabudla kedy má narodeniny. Radšej na to príde nejakým nenápadným spôsobom, aby si to Evka nevšimla. Rozhodla sa, že jej bude dávať nenápadné otázky, na ktoré jej Evka bude odpovedať áno alebo nie. Týchto otázok však nemôže byť priveľa, pretože potom by sa to Evke určite zdalo podozrivé.

a) (5 bodov) Lucka vie, že sa Evky môže spýtať postupne na všetkých 31 dní a zistiť tak, v ktorý má Evka narodeniny, ale to by bolo príliš nápadné. Pomôžte jej vymyslieť spôsob, akým sa dozvie kedy má Evka narodeniny tak, že jej položí čo najmenej áno/nie otázok.

Otázky sa Lucka pýta postupne, môže ich teda vymýšľať na základe Evkiných odpovedí. Evka je pravdovravná a Lucke nikdy neklame. Dajte si však pozor, že váš spôsob musí počítať so všetkými možnými odpoveďami a musí fungovať bez ohľadu na to, či sa Evka narodila 1., 12. alebo 31.

Lucka sa už chystala Evky opýtať niektorú z otázok, keď jej napadol oveľa rafinovanejší spôsob ako zistiť dátum Evkiných narodenín. Napíše **všetky** svoje otázky na papier a podstrčí jej ho v škole do lavice. Tak Evka nebude vedieť, kto jej otázky poslal, takže nepríde na to, že Lucka zabudla kedy má narodeniny. Stále jej však nemôže položiť priveľa otázok pretože potom by ju Evka mohla spoznať podľa písma a kvôli zdĺhavosti posúvania papierikov to musí zvládnuť jednou sadou otázok.

b) (6 bodov) Zmenia sa nejako otázky, ktoré musí Lucka klásť, ak ich musí všetky napísať dopredu? Na koľko najmenej otázok sa vie Lucka dozvedieť kedy má Evka narodeniny v tomto prípade?

Všimnite si, že v tomto prípade Lucka nevie otázky upravovať na základe Evkiných odpovedí. Napriek tomu má z jej odpovedí vedieť jednoznačne určiť, kedy sa narodila.

c) (4 bodov) Lucku by zaujímalo či použila najmenej otázok ako mohla (iba tak bude totiž najmenej nápadná). Ukážte jej, že váš spôsob zadávania otázok je najlepší možný, teda že pri ňom použijete najmenší možný počet otázok.

Praktická úloha

Pri práci s počítačom je potrebné vedieť pracovať aj s rôznymi nástrojmi, ktoré slúžia na úpravu obrázkov, prácu so zvukom či vyhľadávaním na internete. V tejto časti ťa preto zakaždým čaká nejaká netradičná úloha.

3. Akútny upgrade

15 bodov za riešenie

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Andrejovi na ajo@ksp.sk

Timke sa pokazil počítač. Je to pre ňu ako programátorku beztak ťažké, a akoby nestačilo, je nedeľa ráno a ona musí na pondelok do školy vypracovať úlohu z programovania. To by ale bola hanba keby ju ona, ktorá rieši všemožné súťaže nemala. Rozhodla sa preto zísť do pivnice a pokúsiť sa nájsť svoj starý počítač. Namiesto neho však našla iba pár zbytočností a jeden naozaj starý počítač. Oprášila ho a uvidela na ňom názov : "PRAStarý Kalkulátor". Potešila sa. Keď ho však zapojila, ostala nemilo prekvapená. Počítač nebol úplne klasický, nemal ikonky ani žiadne pokročilé príkazy a hlavne nemal klasickú pamäť. Našla však k nemu naozaj starý manuál, ktorý popisuje jeho fungovanie a tajomnú *rúru*, ktorá sa pri programovaní v ňom používa.

Úloha

Pomôžte Timke a naprogramujte jej úlohu na tomto starom počítači.

a) (1 bod) V rúre sa nachádza číslo x ($0 \leq x \leq 100$). Vypíšte na výstup číslo $3 \cdot x + 5$.

vstup	výstup
10	35

b) (2 body) Vypíšte na výstup čísla od 1 po x, kde x ($1 \leq x \leq 100$) je jediné číslo, nachádzajúce sa na začiatku v rúre.

vstup	výstup
4	1 2 3 4

c) (2 body) Poznáte fibonnacciho postupnosť? Je to postupnosť tvaru $0, 1, 1, 2, 3 \ldots$ Prvé dve čísla postupnosti sú 0 a 1, pre ďalšie čísla platí, že každé je súčtom dvoch predchádzajúcich. V rúre máte na začiatku číslo n ($1 \leq n \leq 100$). Vypíšte na výstup jediné číslo a to n-té fibbonacciho číslo.

vstup	výstup
5	3

d) (3 body) V rúre sa nachádza jediné číslo x ($1 \leq x \leq 1000$). Zistite počet jeho deliteľov a vypíšte tento počet na výstup.

vstup	výstup
8	4

e) (3 body) Na vstupe sa nachádza n ($0 \leq n \leq 300$) kladných čísel. Vypíšte na výstup jediné čislo, ich počet.

vstup	výstup
10 15 7 8 12 3	6

f) (4 bodov) V rúre sa nachádza n ($1 \leq n \leq 2000$) kladných čísel. Vypíšte na výstup tie z nich, ktoré sú deliteľné 3. Navyše musí platiť, že ich vypíšete v rovnakom poradí v akom sa nachádzali na začiatku v rúre.

vstup	výstup
10 20 30 4 9 7	30 9

Odovzdávanie

Odvzdávanie tejto úlohy je trochu špecifické. Vaše riešenia si môžete akokoľvek dlho testovať pomocou online simulátora, ku ktorému sme pre vás taktiež pripravili krátky manuál. Na testovanie môžete odoslať zozipovaný archív, ktorý má obsahovať pre každú podúlohu práve jeden textový súbor s názvom `x.txt` kde x je názov podúlohy. Tento textový súbor má obsahovať kód vášho programu. Pokiaľ v podúlohe uvádzame, že sa v rúre niečo nachádza, váš program má predpokladať, že to tam naozaj je. Neodosielajte teda na testovanie verziu, v ktorej si nejaký testovací vstup vyrobíte tak, ako to zrejme treba robiť v simulátore. Dajte si tiež pozor aby zip neobsahoval priečinok ale priamo textové súbory. Testovač by mal váš kód otestovať, pričom v detailoch jednotlivých stupov sa viete dozvedieť, čo bolo zle. Pokiaľ sa testovač tvári, že váš kód testuje a trvá to nejak dlho, je veľmi pravdepodobné, že je niečo zle, napr. neodoslali ste súbory v požadovanom formáte a podobne. Ak ste si napriek tomu istý, že robíte všetko správne a testovač štrajkuje, neváhajte nás kontaktovať na mail vyššie. Odovzdávať môžete aj iba časť programov, teda napr. môžete odoslať zip s obsahom `a.txt`, `c.txt` a `e.txt`. Toto výrazne odporúčame, nakoľko si viete otestovať, či všetko funguje na menej ťažkých podúlohách.

Programátorské úlohy

Tieto úlohy sú zamerané na praktickú tvorbu programov v niektorom vyššom programovacom jazyku ako je napríklad Python, C++ alebo Pascal. Na stránke odovdzávaš **iba zdrojový kód** svojho programu riešiaceho zadanú úlohu, ktorý bude okamžite automaticky otestovaný a do pár sekúnd sa dozvieš, koľko bodov tvoj program získal. Tieto body ti už nikto nemôže zobrať, ale ak si nezískal plný počet bodov, môžeš opakovane odovzdávať opravený program, až kým nebudeš spokojný s výsledkom.

Ak už vieš programovať, ale ešte si nepracoval s naším testovacím systémom, odporúčam ti zájsť na Programátorskú Liaheň (`http://liahen.ksp.sk`), kde si o tom môžeš prečítať úvodný text a vyriešiť si niekoľko jednoduchých úloh.

Ak však **nevieš programovať, tak nezúfaj!** Pripravili sme pre teba **Programátorskú Liaheň**, ktorá ťa **naučí základy programovania** v jazyku `C++`. Navyše, za riešenie týchto tutoriálových úloh na Liahni môžeš získať body priamo do PRASKu a tým si vynahradiť neriešenie niektorej z programátorských úloh.

Presnejšie to funguje takto. Na Liahni sa nachádzajú dve sady úloh, prvá zameraná na premenné a druhá na podmienky v jazyku `C++`. V týchto sadách sa nachádzajú bodované aj nebodované úlohy, ktoré môžeš postupne riešiť a ktoré ti postupne vysvetlia danú problematiku. Dokopy sa v jednej sade dá získať až 15 bodov.

Týmito bodmi si potom môžeš nahradiť úlohy 4 a 5. Samozrejme, toto môžeš urobiť **s každou sadou najviac raz**.

No a v budúcej sérii budeš môcť za body riešiť ďalšie dve sady z Liahne.

Samozrejme, nič ti nebráni riešiť aj úlohy z Liahne aj klasické programátorské úlohy v PRASKu.

Programátorskú Liaheň nájdeš na tejto stránke: http://liahen.ksp.sk

4. Slizká pochúťka

15 bodov za riešenie

Ak nevieš programovať, nezúfaj! Môžeš sa to naučiť a ešte za to získať body, ktoré sa ti budú počítať namiesto tejto úlohy. Stačí, že pôjdeš na stránku Programátorskej Liahne (liahen.ksp.sk). Keď budeš riešiť sadu **variables_cpp**, bodmi, ktoré získaš si môžeš nahradiť riešenie tejto úlohy. Stačí ak na spodku tejto stránky odovzdáš pdf-ko s prezývkou, ktorú používaš na Liahni.

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Danovi na danoravec@gmail.com

Krtko má rád dážďovky, najlepšie natreté omáčkou. Nemyslite si ale, že to je len tak. Na niektoré časti dážovky sa omáčka proste nehodí. To môže byť spôsobené napríklad nadmerným množstvom povrchového slizu, ktorý je sám o sebe lahodný, s omáčkou je však k ničomu. Krtko je veľmi skúsený dážďovkár, vďaka čomu vie hneď na prvý pohľad povedať, ktoré úseky dážďovky sa oplatí natrieť.

Krtko dážďovky natiera tak, že sa postaví do dostatočnej vzdialenosti, aby dážďovku videl celú. Následne si zapíše pod seba úseky, ktoré treba natrieť. Ako všetci vieme, každá dážďovka sa skladá z n políčok očíslovaných v poradí od 1 po n. Krtko si teda bude zapisovať úseky $[a, b]$, kde a je prvé políčko úseku, ktorý chce natrieť a b je posledné políčko tohoto úseku. Často sa mu však stáva, že počas zapisovania zabudne, že niektoré políčko si už zapísal a tak ho zapíše aj do ďalšieho úseku. Jeho papier teda môže vyzerať tak, že sú na ňom napríklad úseky $[1, 4]$ a $[3, 7]$. Môžeme si všimnúť, že políčka 3 a 4 sú tu zahrnuté v oboch úsekoch napísaných na papieri a budú teda natreté dvakrát. Toľké plytvanie omáčkou! Uvedomil si to aj sám Krtko a tak vás požiadal, či by ste mu nenapísali program, ktorému ukáže svoj papier a on mu podľa toho nakreslí dážďovku tak, ako má byť natretá.

Úloha

Vašou úlohou je napísať program, ktorý na vstupe dostane Krtkov papier a na výstup nakreslí natretú dážďovku. To znamená, že pre každé políčko dážďovky určí, či má alebo nemá byť natreté. Nezaujíma nás, či niektoré políčko má byť podľa papiera natreté viackrát. Ak má byť natreté aspoň raz, prehlásime ho za natreté, v opačnom prípade za nenatreté.

Formát vstupu

Na prvom riadku vstupu sú dve celé čísla n ($1 \leq n \leq 10^6$) a q ($0 \leq q \leq 10^5$). Číslo n je počet políčok dážovky, číslo q je počet úsekov napísaných na Krtkovom papieri.

Nasleduje q riadkov popisujúcich Krtkov papier. Na každom sú dve celé čísla a, b ($1 \leq a \leq b \leq n$). Tie hovoria, že má byť natretý úsek dážďovky od a-teho políčka po b-te vrátane. Úseky na papieri sa môžu ľubovoľne prekrývať. Je teda možné aj to, že niektoré políčka dážďovky budú zahrnuté vo všetkých úsekoch na papieri.

Formát výstupu

Na výstup vypíšte n medzerou oddelených čísiel. Ako i-te číslo vypíšte 0, ak je i-te políčko dážďovky nenatreté, v opačnom prípade vypíšte ako i-te číslo 1.

Hodnotenie

Číslo sady	1	2	3	4	5
maximálne n	200	20 000	100 000	500 000	1 000 000
maximálne q	100	100	100 000	100 000	100 000

Príklady

vstup

```
8 4
4 7
1 1
3 5
6 7
```

výstup

```
1 0 1 1 1 1 1 0
```

5. Kandidát

15 bodov za riešenie

Ak nevieš programovať, nezúfaj! Môžeš sa to naučiť a ešte za to získať body, ktoré sa ti budú počítať namiesto tejto úlohy.

Stačí, že pôjdeš na stránku Programátorskej Liahne (liahen.ksp.sk). Keď budeš riešiť sadu **conditions_cpp**, bodmi, ktoré získaš si môžeš nahradiť riešenie tejto úlohy. Stačí ak na spodku tejto stránky odovzdáš pdf-ko s prezývkou, ktorú používaš na Liahni.

Ak máte akékoľvek otázky ohľadom tejto úlohy, napíšte Jitke na jitka.mur@gmail.com

Za siedmimi horami a siedmimi dolinami, v ďalekej krajine menom Trojsten sa chystajú prezidentské voľby. Kampane kandidátov sú v plnom prúde, no Emovi sa nepáči volebný program ani jedného z nich. Rozhodol sa teda sám vsúpiť do tohto kolotoču ako vhodný adept. No ak sa chce stať prezidentom, má pred sebou ešte dlhú cestu. Musí pozbierať dostatočný počet podpisov, aby mohol vôbec kandidovať a následne voľby vyhrať.

V Trojstene majú prezidentské voľby jednoduché pravidlá. Kandidát sa stane prezidentom ak sa pozná s každým občanom. Emo by rád vedel, či pozná každého občana, no má plné ruky práce so zháňaním podpisov. Pomôžte mu a zistite to preňho.

Úloha

Vašou úlohou je napísať program, ktorý na vstupe dostane zozmam známostí v Trojstene a zistí, či sa Emo pozná s každým občanom. Občan (aj Emo je občanom) sa s niekým pozná ak sa poznajú priamo alebo ob jedného človeka. Teda ak Emo pozná Andreja a Andrej pozná Timku, tak môžeme povedať, že aj Emo pozná Timku. Ak však Timka pozná Gabiku, ktorá inak nikoho iného nepozná, tak Emo Gabiku nepozná, pretože je to príliš vzdialená známosť.

Formát vstupu

Na prvom riadku vstupu sú tri celé čísla n, m a e. Číslo n ($1 \leq n \leq 80\,000$) je počet občanov Trojstenu, ktorí sú pre jednoduchosť očíslovaní od 1 po n, číslo m($0 \leq m \leq 100\,000$) je počet známostí a číslo e ($1 \leq e \leq n$) hovorí, ktorý občan je Emo.

Nasleduje m riadkov popisujúcich jednotlivé známosti. Na každom sú dve celé čísla a, b ($1 \leq a < b \leq n$), ktoré udávajú, že občan s číslom a a občan s číslom b sa navzájom poznajú.

Formát výstupu

Na výstup vypíšte do jedného riadku slovo **Ano** ak sa Emo stane prezidentom alebo slovo **Nie** ak sa ním nestane. Do druhého riadku vypíšte vzostupne čísla občanov, ktorých Emo nepozná, oddelených medzerou. Ak pozná každého, do druhého riadku nevypisujte nič.

Hodnotenie

Číslo sady	1	2	3	4	5
maximálne n	20	70	80 000	80 000	80 000
maximálne m	100	1 000	100 000	100 000	100 000

Príklady

vstup

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

výstup

```
Ano
```

Ema označuje číslo 1. *Priamo pozná* $2, 3, 4$. *Občana* 5 *pozná skrz občana* 2.

vstup

```
6 5 2
1 2
3 6
3 4
2 3
2 5
```

výstup

```
Ano
```

Ema označuje číslo 2. *Priamo pozná občanov* $1, 3, 5$. *Občanov* $4, 6$ *pozná skrz občana* 3.

vstup

```
7 6 1
1 2
1 3
2 3
2 5
4 6
5 7
```

výstup

```
Nie
4 6 7
```

Ema označuje číslo 1. *Priamo pozná občanov* $2, 3$. *Občana* 5 *pozná skrz občana* 2. *Občania* 4 *a* 6 *sa poznajú len navzájom a Emo ich nepozná. Občan číslo* 7 *sa s Emom tiež nepozná, neexistuje totiž priamy známy Ema, ktorý by ho poznal.*