A picture containing shape

Description automatically generated

**SEMINARSKA NALOGA**

PROGRAMIRANJE 1

|  |  |
| --- | --- |
|  | **Avtor naloge:** |
|  | URŠA KUMELJ |

Ljubljana, 2022

|  |  |  |  |
| --- | --- | --- | --- |
|  | Ime naloge | Aktivna povezava | Število težavnostnih točk |
| 1 del | Suspension Bridges | <https://open.kattis.com/problems/suspensionbridges> | 3.5T |
|  | Engineering English | <https://open.kattis.com/problems/engineeringenglish> | 2.3T |
|  | Tripple Texting | <https://open.kattis.com/problems/tripletexting> | 1.6T |
|  | Soft Passwords | <https://open.kattis.com/problems/softpasswords> | 2.5T |
|  | Matrix Inverse | <https://open.kattis.com/problems/matrix> | 3T |
|  | Polynomial Multiplication 1 | <https://open.kattis.com/problems/polymul1> | 1.9T |
| 2. del | Anthony and Diablo | <https://open.kattis.com/problems/anthonyanddiablo> | 2.7T |
|  | Basic Remains | <https://open.kattis.com/problems/basicremains> | 3.2T |
|  | Shopping List (Easy) | <https://open.kattis.com/problems/shoppinglisteasy> | 1.9T |
|  | Digits | <https://open.kattis.com/problems/digits> | 3.4T |
|  | A Different Problem | <https://open.kattis.com/problems/different> | 2.7T |
|  | ICPC Team Selection | <https://open.kattis.com/problems/icpcteamselection> | 3.3T |
| 3. del | Combination Lock | <https://open.kattis.com/problems/combinationlock> | 2.7T |
|  | Cantor | <https://open.kattis.com/problems/cantor> | 2.9T |
|  | Cinema | <https://open.kattis.com/problems/cinema> | 2.2T |
|  | Accounting | <https://open.kattis.com/problems/bokforing> | 3.3T |
|  | Alex and Barb | <https://open.kattis.com/problems/alexandbarb> | 3.3T |
|  | Average Speed | <https://open.kattis.com/problems/averagespeed> | 3.4T |
|  | Best Compression Ever | <https://open.kattis.com/problems/bestcompression> | 2.2T |
| **Skupaj kandidatnih točk** | | | **50T** |
| **Doseženih** | |  | **49T** |
|  |  |  |  |
| 4. del | Veci | <https://open.kattis.com/problems/veci> | 1.8T |
|  | Scaling Recipe | <https://open.kattis.com/problems/scalingrecipe> | 1.6T |
|  |  |  |  |
|  | **Skupaj kandidatnih točk** | | **3.4T** |
|  |  | |  |
|  | **Skupaj načtovanih točk:** 15T + 15T + 19T + 3.4T | | **52.4T** |
|  | **Predvidena ocena** | | **9** |

# Izjava

Urša Kumelj izjavljam, da sem seminarsko nalogo opravila samostojno in da sem njena avtorica. To pomeni, da sem v poročilu točno označila vse tiste dele kode, ki sem jih povzela iz virov in vire navedla. Prav tako vem, da izjava o avtorstvu pomeni, da znam razložiti vsako podrobnost kode, ki jo oddajam. Zavedam se, da v primeru, če izjava prvega stavka ni resnična, kršim disciplinska pravila.

# Veci

Nasloga je na spletni strani <https://open.kattis.com/problems/veci> in je vredna 1.8 točk.



BESEDILO NALOGE

Vaš program bo dobil celo število X. Poiščite najmanjše število, večje od X, sestavljeno iz istih števk kot X.

**Vhodni podatki**:

Prva vrstica vnosa vsebuje celo število X (1 ≤ X ≤ 999 999). Prva številka v X ne bo ničla.

**Izhodni podatki**:

Izpiši rezultat v eni vrstici. Če te številke ni, izpišite 0.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

Graphical user interface, application

Description automatically generated

Primer vhodnih podatkov 2: Primer izhodnih podatkov 2:

Graphical user interface, diagram

Description automatically generated with medium confidence

Primer vhodnih podatkov 3: Primer izhodnih podatkov 3:

Graphical user interface, text, application

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo podano število. Najti moramo najmanjše število večje od podanega, sestavljeno iz njenih števk. Najlažje je najti vse kombinacije števil, ki jih dobimo iz števk podanega števila. Jaz sem si pomagala z že vgrajenim modulom itertools, kjer imajo na voljo permutacije. Permutacije nam dajo vse možne kombinacije tega števila, ki jih shranjujemo v tabelo. Potem je v tej tabeli potrebno najti najmanjše število, ki je večje od podanega števila. To potem najlažje naredimo z zanko, v primeru, da to število ne obstaja izpišemo 0, kot to od nas zahteva navodilo.

DOKAZ, DA JE NALOGA REŠENA

Timeline

Description automatically generated

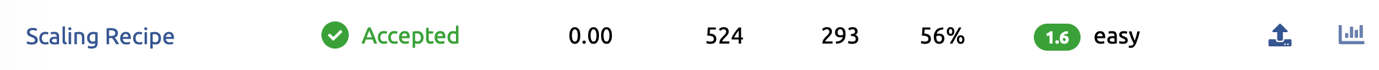
VIRI

Pomagala sem si s spletno stranjo <https://www.geeksforgeeks.org/permutation-and-combination-in-python/>

In <https://github.com/cliodhnaharrison/kattis/blob/main/veci.py>.

# Scaling Recipe

Naloga je na spletni strani <https://open.kattis.com/problems/scalingrecipe> in je vredna 1.6 točke.



BESEDILO NALOGE

Imate recept, ki določa število sestavin, količino vsake sestavine, ki jo boste potrebovali, in število porcij, ki jih dobite. Vendar število porcij, ki jih potrebujete, ni enako številu porcij, ki je navedeno v receptu! Kako jih lahko določite?

**Vhodni podatki**:

Prva vrstica vnosa vsebuje tri cela števila n (1 ≤ n ≤ 40), x in y (1 ≤ x, y ≤ 40.000), kjer je n število sestavin v receptu, x število porcij, za pripravo recepta in y število porcij, ki jih potrebujete.

Vsaka naslednja n vrstica vsebuje eno celo število a (1 ≤ a ≤ 40.000). To so količine vsake sestavine, potrebne za recept.

Vhodni podatki bodo izbrani tako, da bo količina vsake sestavine potrebna za y celo število.

**Izhodni podatki**:

Izpišite n vrstic, kjer je vsaka vrstica eno celo število, ki je količina te sestavine, potrebna za izdelavo y delov recepta. Izpišite te vrednosti po vrstnem redu vnosa.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

Graphical user interface, application

Description automatically generated

Primer vhodnih podatkov 2: Primer izhodnih podatkov 2:

Graphical user interface, text, application

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo prvo podana tri cela števila, n, koliko sestavin potrebujemo za recept, x, število porcij za ta recept in y, število porcij, ki bi jih mi radi imeli. Potem pa še n vrstic, ki so količine potrebne za posamezno sestavino.

Obdelati je potrebno vsako sestavino posebej, kar naredimo z eno izmed zank. Potem pa je ključno, da izračunamo količino sestavine potrebno za 1 enoto oziroma osebo, saj bomo potem samo pomnožili s količino y, ki jo potrebujemo. Vsak obdelan primer potem izpišemo.

DOKAZ, DA JE NALOGA REŠENA

Timeline

Description automatically generated

# Combination Lock

Naloga je na spletni strani <https://open.kattis.com/problems/combinationlock> in je vredna 2.7 točk.

****

BESEDILO NALOGE

Zdaj, ko ste se vrnili v šolo, se morate spomniti, kako uporabljati kombinacijsko ključavnico na vaši omarici. Pogost dizajn je model Master Brand, prikazan na desni. Ključavnica ima številčnico s 40 kalibracijskimi oznakami oštevilčene od 0 do 39 z 0 na vrhu in nato naraščajo v smeri urinega kazalca. Kombinacija je sestavljena iz treh takih številk; na primer: 15 – 25 – 8. Za odpiranje ključavnice se izvedejo naslednji koraki:

• zavrtite številčnico 2 krat v smeri urinega kazalca (2 polna obrata)

• ustavite se pri prvi številki kombinacije

• zavrtite gumb v nasprotni smeri urinega kazalca (1 polni obrat)

• nadaljujte z vrtenjem v nasprotni smeri urinega kazalca, dokler 2. številka ni dosežena

• ponovno obrnite gumb v smeri urinega kazalca, dokler 3. številka ni dosežena

• povlecite držalo in ključavnica se bo odprla

Glede na začetni položaj številčnice in kombinacijo za ključavnico, za koliko stopinj se številčnica skupaj obrne (v smeri urinega kazalca plus v nasprotni smeri urinega kazalca) pri odpiranju ključavnice?

**Vhodni podatki**:

Vhod je sestavljen iz več testnih primerov, največ 2000. Za vsak primer obstaja vrstica vnosa, ki vsebuje 4 številke med 0 in 39. Prva številka je položaj številčnice. Naslednje tri številke so kombinacija. Zaporedne številke v kombinaciji bodo različne. Vrstica, ki vsebuje štiri ničle, sledi zadnjemu primeru.

**Izhodni podatki**:

Za vsak primer izpišite vrstico z enim celim številom: število stopinj, za katere je treba številčnico obrniti, da odprete ključavnico.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

Table

Description automatically generated with low confidence

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več testnih primerov. Vsaka vrstica je sestavljena iz 4 celih števil. Prva nam pove položaj številčnice, ostale pa so kombinacija. S štirimi ničlami na koncu označimo konec testnih primerov.

Ker moramo obravnavati več testnih primerov na enkrat, je najlažje, da ustvarimo neskončno zanko. Potem je potrebno pravilno prebrati vsako število. Nato pa se osredotočimo na alineje, zapisane v navodilih. Iz začetnega položaja moramo priti po najkrajši možni poti do naslednje številke. To lahko naredimo na dva načina in sicer v smeri urinega kazalca ali obratno. Najti pa je potrebno najkrajšo pot, torej moramo najprej obravnavati ta dva primera. Ker pa s tem obravnavamo le prvi dve številki, nas čaka enak postopek še za drugi dve. Ne smemo pa pozabiti, da če je bolj optimalna pot v obratni smeri urinega kazalca, moramo potem odšteti tiste predhodnje stopinje, saj stopinje štejemo v smeri urinega kazalca. Vse premike, ki jih naredimo, prištevamo k vsoti. Na koncu, ko imamo obravnavano vse, je naša naloga poiskati število stopinj. Kako velika pa je ena stopinja, moramo določiti na začetku, ker so nam v navodilu povedali, da je model Master Brand sestavljen iz 40 kalibracijskih oznak, krog pa ima 360 stopinj, torej je ena stopinja velika ravno 9. Naš rezultat pa je potem seštevek naračunanih stopinj in še 360 \* 3, saj v navodilih piše, da jo moramo prvo zavrteti dvakrat, nato pa še enkrat.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application

Description automatically generated

# Cantor

Naloga je na spletni strani <https://open.kattis.com/problems/cantor> in je vredna 2.9 točk.



BESEDILO NALOGE

Število zapisano v trojiškem zapisu je tisto število, ki je zapisano v osnovi 3. Število ima lahko več kot eno razširitev v trojiškem sistemu. Kot primer in .

Cantorjeva množica je definirana kot realna števila med 0 in 1 vključno, ki imajo takšno razširitev, ki ne vsebuje 1. Če ima število več kot eno trojiško razširitev, je dovolj, da ena sama ne vsebuje števila 1.

Na primer in sta v Cantorjevi množici, , ki je edini zapis v trojiškem sistemu, pa ni del Cantorjeve množice.

Vaša naloga je ugotoviti, ali je dano število v Cantorjevi množici.

**Vhodni podatki**:

Vhodni podatki so sestavljeni iz več testnih primerov, do največ 10.

Vsak testni primer je sestavljen iz ene vrstice, ki vsebuje število x zapisano v decimalnem zapisu, 0 ≤ x ≤ 1 in imajo največ 6 števk za decimalno vejico.

Zadnja vrstica vnosa je END, ki ni testni primer.

**Izhodni podatki**:

Za vsak testni primer izpišite MEMBER, če je x v Cantorjevi množici in NON-MEMBER sicer.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

Graphical user interface, application

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več testnih primerov. Vsak testni primer je neko decimalno število med 0 in 1.

Z neskončno zanko bomo najlažje obravnavali več testnih primerov naenkrat. Potem pa se moramo spomniti, kako decimalno število pretvoriti v trojiški sistem. Algoritem si lahko pogledamo na internetu, kot sem to storila tudi sama. Potem ko smo število pretvorili, moramo pregledati njegove števke. Čim naletimo na število 1 v njenem decimalnem zapisu, izpišemo NON-MEMBER, saj nam navodilo pove, da to število potem ni del Cantorjeve množice.

DOKAZ, DA JE NALOGA REŠENA

Timeline

Description automatically generated

VIRI

Pomagala sem si s spletno stranjo <https://madformath.com/calculators/basic-math/base-converters/decimal-to-base-3-converter-with-steps/decimal-to-base-3-converter-with-steps>, ki mi je pomagala, kako spremenimo decimalno število v trojiški sistem.

# Cinema

Naloga je na spletni strani <https://open.kattis.com/problems/cinema> in je vredna 2.2 točk.



BESEDILO NALOGE

Združenje United Cinema Crowd Association iz Stockholma načrtuje predstavo starih računalniških znanstvenikov in njihovih pietingov v lokalnem kinu KTH Royal Institute of Technology.

Ne davno je revizor društva opozoril, da je upravni odbor v gledališče rezerviral preveč skupin obiskovalcev, ki kvečjemu ustreza N obiskovalcev.

Skupaj se je na predstavo prijavilo M skupin obiskovalcev. Odločeno je bilo, da skupine vstopijo v gledališče eno za drugo, v istem vrstnem redu, kot so se prijavile na predstavo. Če je ob prihodu skupine premalo prostih sedežev, se skupina razjezi in odide.

Glede na velikost vseh gostujočih skupin določite, koliko skupin ne bo sprejetih v gledališče.

**Vhodni podatki**:

Prva vrstica vnosa vsebuje cela števila N (1 ≤ N ≤ 100) in M (1 ≤ M ≤ 50), število sedežev v gledališču in število gostujočih skupin.

Druga vrstica vsebuje M celih števil – velikost vsake gostujoče skupine v vrstnem redu, v katerem so se prijavili na predstavo. Skupino sestavljaja med 1 in 10 obiskovalcev.

**Izhodni podatki**:

Izpišite eno število – število skupin, ki ne bodo sprejete v gledališče.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

Diagram

Description automatically generated

Primer vhodnih podatkov 2: Primer izhodnih podatkov 2:

Diagram, chat or text message

Description automatically generated with medium confidence

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo dve vrstici. V prvi zvemo število prostih sedežev v gledališču in pa število skupin, v drugi vrstici pa velikost posamezne skupine.

Ključno je, da gremo skozi posamezno skupino in če je velikost te skupine večja od števila prostih sedežev, izpade. Tako si zapomnimo, da je ta ena izmed skupin, ki bo odšla domov in potem gledamo naprej. Ko pa naletimo na skupino, ki je manjše velikosti od števila prostih sedežev, velikost odštejemo od števila prostih sedežev v gledališču. Na koncu izpišemo vsoto tistih skupin, ki so odšle domov.

OPIS PROBLEMA IN IDEJA REŠEVANJA

Timeline

Description automatically generated

VIRI

Pomagala sem si z nalogo na spletni strani [https://github.com/JonSteinn/Kattis-Solutions/blob/master/src/Cinema Crowds/Python 3/main.py](https://github.com/JonSteinn/Kattis-Solutions/blob/master/src/Cinema%20Crowds/Python%203/main.py)

Accounting

Naloga je na spletni strani <https://open.kattis.com/problems/bokforing> in je vredna 3.3 točke.



BESEDILO NALOGE

Ekonomistka Erika proučuje ekonomsko neenakost. Njen model se začne v situaciji, ko imajo vsi enako količino denarja. Po tem se finančno stanje ljudi spreminja na različne zapletene načine.

Erika mora velikokrat zagnati simulacijo, da preveri, ali njen model deluje. Simulacija je sestavljena iz N ljudi, od katerih vsak začne z 0 kronami. Potem se zgodi Q dogodkov, treh različnih vrst:

1. Dogodek tipa "SET i x" pomeni, da i-ti osebi nastavi finančno stanje na vrednost x.
2. Dogodek tipa "RESTART x" pomeni, da je finančno stanje vseh nastavljeno na vrednost x.
3. Dogodek tipa "PRINT i" pa izpiše finančno stanje i-te osebe.

Žal je trenutna Erikina implementacija zelo počasna, zahteva preveč časa, da bi spremljali, koliko denarja imajo vsi. Odloči se, da bo svoje algoritemsko znanje uporabila za pospešitev simulacije.

**Vhodni podatki**:

Prva vrstica vključuje dve celi števili N in Q, kjer 1 ≤ N ≤ in 1 ≤ Q ≤ . Vsaka naslednja Q vrstica se začne z nizom, ki je bodisi "SET", "RESTART" ali "PRINT". Zagotovljeno je, da obstaja vsaj en dogodek tipa "PRINT".

Če je niz "SET", mu sledita dve celi števili I in x, kjer 1 ≤ I ≤ N in 0 ≤ x ≤. Če je niz "RESTART", mu sledi celo število x, 0 ≤ x ≤ . Če je niz "PRINT", mu sledi celo število I, 1 ≤ i ≤ N.

**Izhodni podatki**:

Za vsak niz tipa "PRINT", izpiši finančno stanje i-te osebe.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

Graphical user interface, application

Description automatically generated

Primer vhodnih podatkov 2: Primer izhodnih podatkov 2:

Graphical user interface

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več vrstic. Prva vrstica je sestavljena iz dveh celih števil N in Q, ki nam povesta število ljudi in število nadaljnih vrstic oz. dogodkov.

Prva opazka je, da ne bomo mogli naloge rešiti samo z neskončno zanko in pa inputom, saj se nam vse skupaj zalomi, ker pri SET potrebujemo tri inpute, pri PRINT in RESTART pa le dva. Jaz sem se odločila za uvoz modula sys, ki bere vsako vrstico kot svoj input. Tako potem začnemo z obravnavo treh dogodkov. Pred tem pa se moramo še odločiti, kako bomo shranjevali podatke. Meni se je zdel slovar najboljša možnost. Pri dogodku tipa SET ustvarimo slovar, kjer so ključi prvo število za besedo SET, vrednosti pa so drugo število za SET. Potem obravnavamo RESTART. V neko spremenljivko shranimo vrednost za besedo RESTART in ne smemo pozabiti, da slovar po tem počistimo, saj nam v navodilu piše, da vsem nastavi enako vrednost. Vrednost pa shranimo v neko spremenljivko zato, ker dogodka SET in RESTART ne izpisujeta ničesar, le shranjujeta podatke. Ostane nam samo še PRINT. Ko naletimo na to besedo izpišemo, kar je vrednost pri tem ključu, saj imamo pri PRINT ključe našega slovarja. V primeru, da tega ključa ni, pomeni, da je bil slovar počiščen, torej je bil poklican dogodek RESTART, kar nam da vedeti, da moramo izpisati vrednosti, ki smo jih shranili v spremenljivko pri RESTART.

DOKAZ, DA JE NALOGA REŠENA

Timeline

Description automatically generated

VIRI

Pomagala sem si z videom na spletni strani <https://www.youtube.com/watch?v=5JddumIbthE>

# Alex and Barb

Naloga je na spletni strani <https://open.kattis.com/problems/alexandbarb> in je vredna 3.3 točk.



BESEDILO NALOGE

Alex in Barb čakata na obisk svojih bratrancev za večerjo. Ker se njuni bratranci pogosto zapletejo v najrazličnejše zvijače, se Alex in Barb odločita, da si bosta čas krajšala s kratko igro s kartami.

Igra je naslednja: obstaja kup k kart na mizi. Alex in Barb si izmenjujeta od m do n kart, začne vedno Alex. Prvi igralec brez veljavnih potez izgubi.

Za dani k, m in n določi, kateri igralec bo zmagal v igri, če oba igrata z optimalno strategijo.

**Vhodni podatki**:

Vhod je sestavljen iz ene vrstice, ki vsebuje tri cela števila, ločena s presledkom 1 ≤ k ≤ in

1 ≤ m ≤ n ≤ .

**Izhodni podatki**:

V eni vrstici izpišite ime zmagovalca.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

A picture containing text

Description automatically generated

Primer vhodnih podatkov 2: Primer izhodnih podatkov 2:

A picture containing text

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo eno vrstico s tremi števili in sicer k (število kart), m ter n, ki povesta od kod do kod lahko jemlemo karte s kupčka.

En krog (cikel) imenujmo del igre, ko oba igralca vzameta vsak po enkrat karte s kupčka. Cilj igre je pripaviti stanje kart tako, da jih bo v zadnjem ciklu manj kot m. S tem dosežemo, da nasprotnik ne more več pobrati vsaj m kart in tako izgubi. Na stanje v zadnjem ciklu vplivamo z dejanji v predzadnjem ciklu. Ker vemo, da bo nasprotnik pobral nekje med m in n kartami, ukrepamo tako, da bo vsota njegovih in naših pobranih kart m+n. In tako v vseh predhodnih ciklih.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, timeline

Description automatically generated

# Average Speed

Naloga je na spletni strani <https://open.kattis.com/problems/averagespeed> in je vredna 3.4 točk.



BESEDILO NALOGE

Kupili ste avto, da bi se od Waterlooja odpeljali v veliko mesto. Števec kilometrov na njihovem avtomobilu je pokvarjen, zato ne morete izmeriti razdalje. Toda merilnik hitrosti in tempomat delujeta, tako da lahko avto vzdržuje konstantno hitrost, ki jo je mogoče občasno prilagoditi glede na omejitve hitrosti, prometne zastoje in čakalne vrste na mejah. Imate štoparico in zabeležite pretečeni čas vsakič, ko se hitrost spremeni. Od časa do časa se vprašate: "Kako daleč ste prišel?". Če želite rešiti to težavo, morate napisati program, ki se izvaja na vašem prenosnem računalniku na sovoznikovem sedežu.

**Vhodni podatki**:

Standardni vnos vsebuje več vrstic vnosa (največ 100): Vsaka sprememba hitrosti je označena z vrstico, ki določa pretečeni čas od začetka potovanja (v formatu “hh:mm:ss”), ki ji sledi nova hitrost v km/h (celo število med 0 in 1000, vključno). Vsaka poizvedba je označena z vrstico, ki vsebuje pretečeni čas. Na začetku potovanja avto miruje. Pretekli časi so podani v nepadajočem vrstnem redu in v danem trenutku obstaja največ ena sprememba hitrosti.

**Izhodni podatki**:

Za vsako poizvedbo v standardnem vnosu bi morali natisniti vrstico s časom in prevoženo razdaljo, zaokroženo na natanko dve števki za decimalno vejico, v spodnji obliki. (Lahko domnevate, da je prevožena razdalja vsaj stran od katerega koli števila, ki bi bilo drugače zaokroženo.)

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

Graphical user interface, text, application, chat or text message

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več vrstic. Nekatere vsebujejo čas, nekatere pa poleg časa še hitrost.

Zaradi vnosa enkrat dveh inputov, enkrat pa enega, je najlažje uvoziti modul sys, ki bere vsako vrstico kot svoj input.

Iz fizike vemo, da je pot enaka produktu časa in hitrosti. Tukaj nastopi prvi problem, saj nimamo posebej zapisano v eni enoti časa, zato je najbolje čas pretvoriti v eno enoto, na primer v sekunde. Ker pa je pot definirana kot razlika hitrosti pomnožena z razliko časov, je potrebno povedati, kaj bo naša prejšnja hitrost in kaj zdajšnja in enako za čas. Kasneje pa to tako ali tako spreminjamo, saj se na vsaki točki nekoliko spremeni. Potem pa obravnavamo dva primera in sicer, ko je vrstica sestavljena iz časa in hitrosti in pa samo časa. Kjer je vrstica samo s časom, je naša naloga izpisati prepotovano pot na dve decimalki natančno. Paziti pa moramo, da rezultat ponovno vrnemo v pravilnih enotah oziroma tako, kot je bil podan z začetnim vnosom.

DOKAZ, DA JE NALOGA REŠENA

Timeline

Description automatically generated with low confidence

# Best Compression Ever

Naloga je na spletni strani <https://open.kattis.com/problems/bestcompression> in je vredna 2.2 točk.



BESEDILO NALOGE

Biti izobražen na področju računalništva in matematike ni vedno lahko. Še posebej ne, če imate prijatelje, ki vedno znova vztrajajo, da vam pokažejo svoje nove dokaze, da je P enako NP, da je Riemannova hipoteza resnična itd.

Eden od vaših prijateljev je pred kratkim trdil, da je našel fantastičen nov kompresijski algoritem. Kot primer njegove neverjetne zmogljivosti vam je vaš prijatelj povedal, da bi bila vsaka datoteka v vaši dragoceni zbirki naključnih bitnih nizov po stiskanju največ b koščkov dolga! Seveda vam je to malo težko verjeti, zato želite ugotoviti, ali je to sploh teoretično možno.

Vaša zbirka naključnih bitnih nizov je sestavljena iz N datotek, od katerih si nista niti dve enaki in vsaka je natančno 1000 bitov dolga.

**Vhodni podatki**:

Vhod je sestavljen iz dveh celih števil N (1 ≤ N ≤ ) in b (0 ≤ b ≤ 50), ki podaja število datotek v vaši zbirki in največje število bitov, ki jih ima stisnjena datoteka.

**Izhodni podatki**:

Izpišite vrstico, ki vsebuje bodisi yes, če je mogoče vseh N datotek stisniti, v vaši zbirki, v datoteke največ velikosti b bitov, drugače pa no.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:

A picture containing text

Description automatically generated

Primer vhodnih podatkov 2: Primer izhodnih podatkov 2:

A picture containing shape

Description automatically generated

Primer vhodnih podatkov 3: Primer izhodnih podatkov 3:

A picture containing text

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo eno vrstico z dvema številoma. Prvo nam pove število datotek, drugo pa največje število bitov, ki jih lahko ima stisnjena datoteka.

Premisliti je potrebno, koliko možnosti imamo vse skupaj. Ker je bit osnovna in najmanjša enota v računalništvu in prestavlja neko informacijo, ki je lahko 1 ali 0, nastopa v binarnem zapisu. Sprva pomislimo, da je vseh možnosti, vendar nam prvi primer namigne, da temu ni tako. Vseh možnosti je , saj drugače 13 datotek ne bi mogli zapisati z največ 3 biti.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application, timeline

Description automatically generated

VIRI

Pomagala sem si z nalogo na spletni strani:

<https://github.com/rvrheenen/OpenKattis/blob/master/Python/bestcompression/bestcompression.py>

# Anthony and Diablo

Naloga je na spletni strani <https://open.kattis.com/problems/anthonyanddiablo> in je vredna 2.7 točk.



BESEDILO NALOGE

Anthony ima hišnega hrčka po imenu Diablo. Diablo uživa, kadar ima veliko prostora za gibanje, zato mu želi Anthony zgraditi kletko, ki pokriva čim več površine.

Vendar pa Diablo tudi zelo rad koplje in se skriva. Ko se Diablo skrije, Anthonyja boli le tega najti, saj se Diablo zelo dobro skriva. Zato želi Anthony zagotoviti, da kletka, ki jo je zgradil zanj, ni prevelika. Zlasti želi zagotoviti, da je površina kletke točno 𝐴 kvadratnih metrov, katero koli območje, večje od 𝐴 kvadratnih metrov bo predstavljalo preveč dela, da bi našli Diabla. Katerakoli površina, manjša od 𝐴 kvadratnih metrov, bo za Diabla preveč neprijetna.

Anthony ima 𝑁 metrov ograje, ki jo je mogoče na katerikoli točki prerezati/ukriviti, in se sprašuje, ali je mogoče zgraditi kletko katerekoli oblike, ki ima natančno površino 𝐴 z materiali, ki jih ima. Napišite program, ki mu bo pomagal!

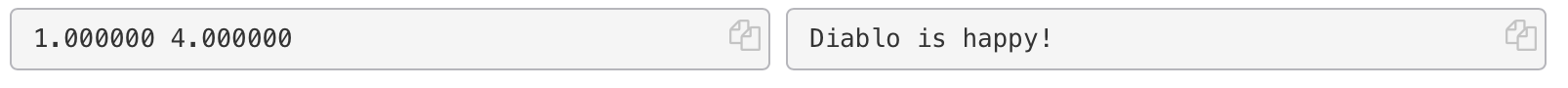
**Vhodni podatki**:

Vhod vsebuje dve realni števili 0 < 𝐴 ≤ 100 in 0 ≤ 𝑁 ≤ 1000.

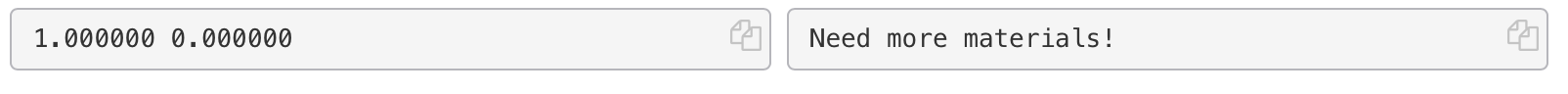
**Izhodni podatki**:

Izpišite eno vrstico z "Diablo is happy!" ("Diablo je srečen! "), če lahko Anthony zgradi svojo kletko za Diabla in "Need more materials!" ("Potrebujem več materiala!") sicer.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 1:



Primer vhodnih podatkov 2: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo eno vrstico: prvi podatek nam pove koliko površine A moramo zgraditi z materiali N, ki jih imamo na voljo, če je to mogoče.

Sprva na ustrezen način preberemo podatke. Ključno je opaziti, da potrebujemo (z nekaj znanja matematike) iz formule za obseg kroga najti maksimalen radij, ki ga lahko dobimo iz materiala N (saj nam N v resnici predstavlja obseg). Ta radij potem uporabimo za izračun površine S. Da lahko kletko naredimo, mora biti (iz prebranih podatkov) podana površina A manjša ali kvečjemu enaka od izračunane površine S.

DOKAZ, DA JE NALOGA REŠENA

Table

Description automatically generated with medium confidence

# Basic Remains

Naloga je na spletni strani <https://open.kattis.com/problems/basicremains> in je vredna 3.2 točk.



BESEDILO NALOGE

Imamo podano osnovo 𝑏 in dve nenegativni celi števili 𝑝 in 𝑚 v osnovi-𝑏. Izračunaj 𝑝 mod 𝑚 in izpiši rezultat kot celo število z osnovo 𝑏. 𝑝 mod 𝑚 je definirano kot najmanjše nenegativno celo število 𝑘 tako, da je 𝑝=𝑎𝑚+𝑘, za neko celo število 𝑎.

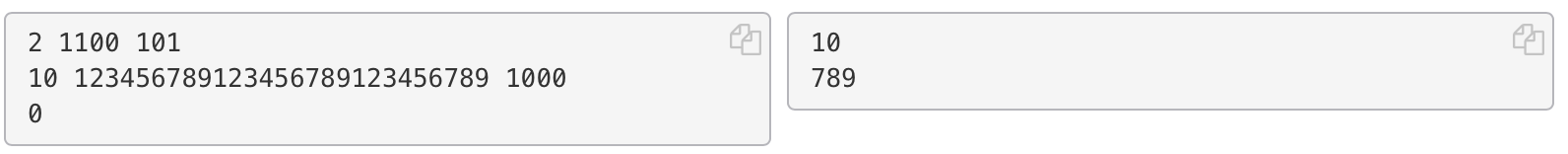
**Vhodni podatki**:

Vnos je sestavljen iz več primerov (največ 1000). Vsak primer je predstavljen z vrstico, ki vsebuje tri nenegativna cela števila. Prvo število 𝑏 je osnova in je število med 2 in 10. Drugo število 𝑝 vsebuje do 1000 števk med 0 in 𝑏−1. Tretje število 𝑚 vsebuje do 9 števk med 0 in 𝑏−1. Zadnjemu primeru sledi vrstica, ki vsebuje 0.

**Izhodni podatki**:

Za vsak testni primer izpiše 𝑝 mod 𝑚 v osnovi-𝑏.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več testnih primerov. Vsaka vrstica je sestavljena iz treh celih števil. Prvo število nam pove, v kateri osnovi b moramo zapisati število, ki ga dobimo z ostankom pri deljenju števila p z m (p mod m), kjer je p drugi prebran podatek in m tretji. S številom 0 pa označimo konec testnih primerov.

Prva ključna stvar je branje podatkov. Naloga od nas zahteva ostanek pri deljenju p z m, v podani osnovi b. Pretvarjanje niza v celo število naredimo s funkcijo int(). Ta funkcija pa sprejema zgolj nize, zato je pomembno, da prebrane podatke pretvorimo v niz. V dokumentaciji za programski jezik Python lahko vidimo, da ima funkcija int() privzeto vrednost 10, torej ostanka pri deljenju p z m ne bo težko dobiti. Ko dobimo ta ostanek, pa je zapisan v osnovi 10. Sedaj je potrebno ta ostanek še pretvoriti v zahtevano osnovo b. Za rešitev tega je najlažje ustvariti funkcijo, ki bo to število pretvorila v osnovo b. Tako funkcijo pa ustvarimo s pomočjo znanja matematike tako, da ustvarimo tabelo, v katero shranjujemo ostanke pri deljenju števila z osnovo b. Ko to delamo ročno na papir, je rezultat število od spodaj navzgor. Enako moramo storiti tudi tu in vrniti tabelo, ki je "narobe obrnjena".

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application

Description automatically generated

VIRI

Pomagala sem si z nalogama:

<https://shareablecode.com/snippets/python-solution-for-uva-online-judge-10551-basic-remains-tsPu-RVYT>

<https://www.codegrepper.com/code-examples/python/python+convert+number+to+base>

# Shopping List (Easy)

Naloga je na spletni strani <https://open.kattis.com/problems/shoppinglisteasy> in je vredna 1.9 točk.



BESEDILO NALOGE

Kelly, ki je odlična programerka, pogosto razmišlja o "naslednji veliki stvari" za aplikacije. Nekega jutra je Kelly pomislila na sijajno idejo, da bi ustvarila družbeno omrežje za nakupovalne sezname živil. Kelly želi, da bi njeno socialno omrežje povezalo ljudi, ki kupujejo iste stvari v trgovini z živili, zato potrebuje vašo pomoč pri pripravi programa, s katerim bi ugotovila, kdaj uporabniki kupujejo iste stvari v trgovini, da bo njeno spletno mesto lahko prikazovalo ustrezne oglase. Kelly želi primerjati uporabnikov najnovejši nakupovalni seznam z vsemi njegovimi prejšnjimi nakupovalnimi seznami in ugotoviti, katere izdelke kupi vsakič, ko gre v trgovino.

**Vhodni podatki**:

Prva vrstica vnosa bo vsebovala dve celi števili, ločeni s presledkom: 𝑛 (1≤𝑛≤20), ki predstavlja število nakupovalnih seznamov, in 𝑚 (1≤𝑚≤20), ki je število elementov na vsakem od teh seznamov. Vsako od naslednjih 𝑛 vrstic je eden od nakupovalnih seznamov, ki vsebuje 𝑚 s presledkom ločenih predmetov. Vsak element je dolžine manj kot 11 znakov in sestavljen samo iz malih angleških črk.

**Izhodni podatki**:

Izpišite celo število 𝑛, ki podaja število elementov, ki se nahajajo na vsakem seznamu v prvi vrstici. V naslednji 𝑛 vrstici, izpišite te elemente po abecednem vrstnem redu tako, da je vsak element v svoji vrstici.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:

Graphical user interface, application

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več vrstic, v prvi imamo dva podatka in sicer število nakupovalnih seznamov in število izdelkov na vsakem nakupovalnem seznamu. Potem pa sledijo izdelki na posameznem nakupovalnem seznamu.

Naloga od nas zahteva, da izpišemo izdelke, ki se pojavijo na vseh seznamih. Tega se je naljažje lotiti z množicami, saj množice ne vsebujejo podvojenih elementov. Od tukaj dobimo idejo, da naredimo najprej množico vseh izdelkov iz vseh nakupovalnih seznamov. Nato se postopoma sprehajamo po posameznem seznamu in ga primerjamo z množico vseh seznamov. Ker vemo, da nas zanimajo samo enaki izdelki, si tako pomagamo z matematično metodo presek. Na koncu torej dobimo množico presekov vseh izdelkov na nakupovalnih seznamih in to je iskana rešitev.

DOKAZ, DA JE NALOGA REŠENAGraphical user interface, text, application, chat or text message

Description automatically generated

VIRI

Pomagala sem si z nalogo [https://github.com/JonSteinn/Kattis-Solutions/blob/master/src/Shopping List (Easy)/Python 3/main.py](https://github.com/JonSteinn/Kattis-Solutions/blob/master/src/Shopping%20List%20(Easy)/Python%203/main.py)

# Digits

Naloga je na spletni strani <https://open.kattis.com/problems/digits> in je vredna 3.4 točke.



BESEDILO NALOGE

Googol, zapisan v decimalki, ima 101 števk. Googolplex ima eno števko več kot googol. To je veliko števk!

Glede na poljubno število , definirajte zaporedje z uporabo naslednjega zaporedja:

= število števk v decimalni predstavitvi

Vaša naloga je določiti najmanjše pozitivno število 𝑖, tako da velja .

**Vhodni podatki**:

Vnos je sestavljen iz več vrstic. Vsaka vrstica vsebuje vrednost . Vsaka vrednost je nenegativna in ne vsebuje več kot milijon števk. Zadnja vrstica vnosa vsebuje besedo END. Imamo največ 10000 vrednosti in velikost vhodne datoteke je največ 2 MB.

**Izhodni podatki**:

Za vsako vrednost , izpišite ustrezno pripadajočo vrednost 𝑖.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več testnih primerov, ki vsebujejo celo število, na koncu pa besedo END.

Ideja je, da dolžino števila spremenimo v niz, nato to dobljeno dolžino ponovno spremenimo v niz njegove dolžine in to ponavljamo, dokler dolžina ni enaka nizu 1, pri tem pa vsako spremembo prištevamo k vsoti. Ko dosežemo niz 1, izstopimo iz zanke in izpišemo število ponovitev. Pomembno pa je, da k končni vsoti prištejemo še 1, saj je pri ustavitvi pri zanki nismo upoštevali. Lahko pa začnemo šteti vsoto že pri 1, ampak se moramo zavedati tega pomena.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface

Description automatically generated with medium confidence

# A Different Problem

Naloga je na spletni strni <https://open.kattis.com/problems/different> in je vredna 2.7 točk.



BESEDILO NALOGE

Napišite program, ki izračuna razliko med nenegativnimi celimi števili.

**Vhodni podatki**:

Vsaka vrstica vnosa je sestavljena iz para celih števil. Vsako celo število je med 0 in (vključno). Vnos se konča s koncem datoteke.

**Izhodni podatki**:

Za vsak par celih števil v vhodu izpišite eno vrstico, ki vsebuje absolutno vrednost njihove razlike.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo več testnih primerov. Vsaka vrstica vsebuje dve nenegativni celi števili.

Naloga je preprosta, saj moramo za vsak primer izpisati le absolutno vrednost razlike teh dveh števil. Z obravnavo več testnih primerov si pomagamo z neskončno zanko while.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, table

Description automatically generated with medium confidence

# ICPC Team Selection

Naloga je na spletni strani <https://open.kattis.com/problems/icpcteamselection> in je vredna 3.3 točke.



BESEDILO NALOGE

Profesor Univerze Nha Trang, g. Van, je pravkar organiziral tekmovanje za oblikovanje svojih ekip ICPC. Na tekmovanju so sodelovali dijaki. študent je dosegel točk na tekmovanju.

Trener želi oblikovati 𝑁 različnih ekip (vsaka ekipa ima 3 dijake), da se na podlagi rezultatov tega tekmovanja udeležijo regijskega tekmovanja. Po njegovih izkušnjah je uspešnost ekipe običajno enaka mediani posameznih rezultatov članov ekipe (tj. rezultatu drugega najboljšega učenca).

Trener želi maksimirati 𝑆– vsota njegovih 𝑁 uspešnosti ekip. Vaša naloga je izračunati 𝑆.

**Vhodni podatki**:

Vhod je sestavljen iz več različnih podatkov. Prva vrstica vnosa vsebuje število naborov podatkov, ki je pozitivno celo število in ni večje od 20. Naslednje vrstice opisujejo nabore podatkov.

Vsak nabor podatkov je opisan z naslednjimi vrsticami:

• Prva vrstica vsebuje pozitivno celo število 𝑁 (𝑁≤100)

• Druga vrstica vsebuje 3𝑁 pozitivna cela števila (≤100)

**Izhodni podatki**:

Za vsak nabor podatkov izpišite vrednost 𝑆.

**Razlaga za vzorčni nabor podatkov:**

Eden od načinov za oblikovanje dveh ekip je:

• Ekipa 1: dijak 1, dijak 2, dijak 3;

• Ekipa 2: dijak 4, dijak 5, dijak 6.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih najprej pridobimo število testnih primerov, število ekip in rezultate posameznih dijakov.

Ker želimo vsoto drugih najboljših dijakov (oz. mediano), je ideja, da vse podatke spravimo v urejeno tabelo. Če pomislimo, moramo torej namišljeno ustvariti nove ekipe s po tremi dijaki, v katerih bo en najslabši, potem en drugi najslabši in pa najboljši ([naslabši, 2.najboljši, najboljši]). Take ekipe sestavimo tako, da najslabšemu dodamo dva najboljša iz tabele in to nadaljujemo do konca. Drugi najboljši bo vedno ravno "ta slabši" izmed teh zadnjih dveh, ki jih vedno dodamo najslabšemu. Ker pa moramo vsakič primerjati med seboj druga dva, je najlažje, da sproti iz tabele brišemo že pregledane in končamo takrat, ko smo jih zbrali ravno toliko kot je bilo ekip na začetku.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application

Description automatically generated

VIRI

Pomagala sem si z nalogo <https://github.com/jed1337/Kattis/blob/master/icpc_team_selection.py>

# Suspension Bridges

Naloga je na spletni strani <https://open.kattis.com/problems/suspensionbridges> in je vredna 3.5 točk.



BESEDILO NALOGE

Gorske vasi rade privabljajo turiste z gradnjo visečih mostov, kot je ta, ki je prikazan tukaj v gorovju Harz v Nemčiji. Ti mostovi omogočajo pustolovsko nagnjenim ljudem, da svoje vznemirjenje poiščejo s prečkanjem globokih sotesk. Da bi zagotovili, da bodo vsi deležni ravno prave meje razburjenja, mora biti nagib na najgloblji točki mostu pomemben glede na razdaljo, ki jo most pokriva.

Glede na razdaljo med sidrnimi točkami, kjer je most pritrjen, in glede na želeno količino nagiba, izračunajte, kako dolg mora biti vsak od kablov, ki držijo viseči most!

V pomoč pri reševanju te naloge je nekaj ozadja: Prosto viseči viseči most bo dobil obliko verižne krivulje (catena je latinsko za verigo), tako kot prosto viseča veriga med dvema drogoma. Glede na vodoravno razdaljo 𝑑 med dvema sidrnima točkama in želeno količino 𝑠 kabel povešen v sredini, obstaja pozitiven parameter 𝑎 tako, da je 𝑎 + 𝑠 = 𝑎 ⋅cosh(𝑑/2𝑎). Dolžina kabla je nato podana z ℓ(𝑎,𝑑)=2𝑎⋅sinh(𝑑/2𝑎).

Funkcije sinh in cosh označujemo hiperbolični sinus oziroma hiperbolični kosinus, ki sta opredeljena na naslednji način:

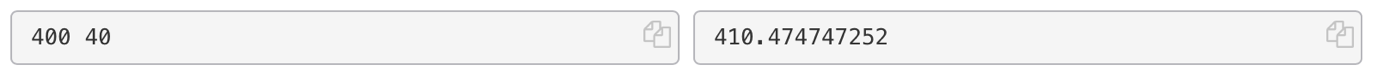
**Vhodni podatki:**

Vhod je sestavljen iz enega testnega primera z dvema celima številoma 𝑑, ločenima s presledkom 𝑠, podano v eni vrstici, tako da je 0 < 𝑑 ≤ 1000 in 0 < 𝑠 ≤ 1000. Število 𝑑 označuje razdaljo med sidrišči in 𝑠 je želeni nagib na sredini mostu. Natančneje, "sag" je navpična razdalja med najnižjo točko kabla na sredini in vodoravno črto, ki jo tvorita 2 sidrišča na obeh straneh soteske.

**Izhodni podatki:**

Izpišite dolžino kabla, ki je potrebna za pokrivanje razdalje med sidrnimi točkami, da dosežete želeno povešanje. Vaš odgovor mora biti pravilen znotraj absolutne napake .

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih imamo eno vrstico, iz katere najprej dobimo dolžino mostu horizontalno (d) in nekako višino mostu. V nalogi imamo podano obnašanje parametra a glede na s in d. To lahko opišemo kar s fukcijo f(a+s) in tako po metodi bisekcije gledamo obnašanje. Določiti moramo interval, ki ima nasprotno predznačeni meji glede na našo funkcijo. Tam, kjer funkcija spremeni predznak, na tistem intervalu se potem nahaja ničla (za ničlo vzamemo ta parameter a) in ta parameter je na začetku aritmetična sredina pozitivne vrednosti in negativne vrednosti. Tako posopek lahko ponavljamo v neskončnost in s tem dobivamo vse bolj in bolj natančen približek. Rezultat pa potem dobimo po drugi podani formuli, ki kot argumenta sprejme parameter a in dolžino d.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, text, application, chat or text message

Description automatically generated

VIRI

Pomagala sem si z nalogo <https://github.com/gkamtzir/Kattis-Problems/blob/master/suspension_bridges.py.>

# Engineering English

Naloga je na spletni strani <https://open.kattis.com/problems/engineeringenglish> in je vredna 2.3 točke.



BESEDILO NALOGE

Inženirji so prevzeli oddelek za angleški jezik univerze! Kot del njihovega načrta, da prevladujejo na univerzi, so vas zaposlili, da napišete njihovo zlobno programsko opremo. Njihov cilj je odstraniti vse podvojene besede iz vseh besedil na angleškem oddelku, saj, kot pravijo, "podvojene besede so podvojeno delo" in "podvojeno delo zapravlja denar." Pravzaprav bi v resnici rekli: "podvojene besede so . delo" in ". . zapravlja denar«. To bi moral storiti vaš program – odstranite vse podvojene besede in jih nadomestite s piko.

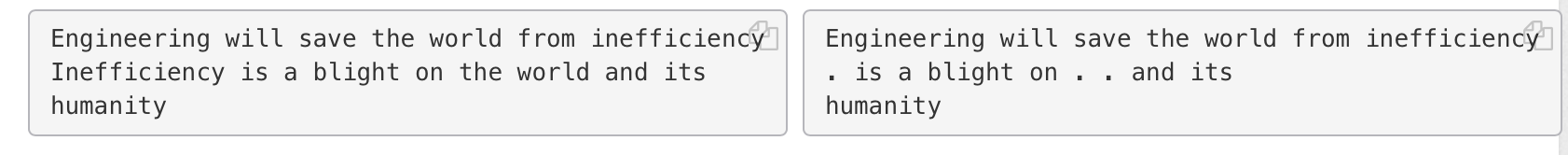
**Vhodni podatki:**

Vhod je učbenik z največ 2000 vrstice. Vsaka vrstica vsebuje do 100 znakov. Ločila se ne prikažejo (odstranil jih je že drug programer); besede so sestavljene iz največ 20 male in velike črke (a–z). Največ jih je 20000 skupno število besed v vhodu. Vsak par sosednjih besed je ločen z enim presledkom. Pri primerjavi besed za dvojnike (t.j. »knjiga« in »knjiga« in »bOoK« je treba šteti za isto besedo) velike in male črke niso pomembne. Vnos se konča na koncu datoteke.

**Izhodni podatki:**

Natisnite vsako besedo, ko se pojavi prvič, potem pa, če se ta beseda znova pojavi, natisnite piko.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

Reševanje te naloge se mi je zdelo najlažje z množicami. Z zanko for sem se sprehodila po vrstici in pregledovala vsako besedo posebej. Če se je beseda že ponovila, sem jo dodala v množico že uporabljenih besed in v prazen niz, ustvarjen že na začetku, dodala piko in presledek, zato ker vidimo v izhodnih podatkih, da je za vsako piko še presledek. Če beseda še ni bila prej vidna, smo jo skupaj s presledkom dodali v ustvarjen niz.

DOKAZ, DA JE NALOGA REŠENAGraphical user interface, application

Description automatically generated

VIR

Pomagala sem si z nalogo na spletni strani <https://github.com/JonSteinn/Kattis-Solutions/blob/master/src/Engineering%20English/Python%203/engineeringenglish.py>.

# Triple Texting

Naloga je na spletni strani <https://open.kattis.com/problems/tripletexting> in je vredna 1.6 točke.



BESEDILO NALOGE

Julia uživa v pogovorih z babico, igranju z lego kockami in izumljanju iger s kartami za dva igralca, kjer ima zmagovalno strategijo. Zadnje čase pa se zaradi nekakšnega "pandemonija" ni mogla osebno pogovarjati z babico. Namesto tega so se zatekli k pošiljanju sporočil, kar je zelo počasen proces, saj babica tipka zelo počasi in pogosto napačno vnaša črke. Da bi bila stvar še hujša, je babica začela vsako besedo pisati trikrat, da bi Julia lahko popravila svoje napačne tipke. Na primer, če želi babica napisati besedo "zdravo", bo namesto tega napisala "zdravo, zdravo". Če napačno vnese eno od teh črk, se lahko namesto tega pošlje kot »zdravo hrllohello«.

Vaša naloga je napisati program, ki po sporočilu, ki ga je poslala babica, kjer je morda ena črka spremenjena v drugo črko, najde izvirno besedo.

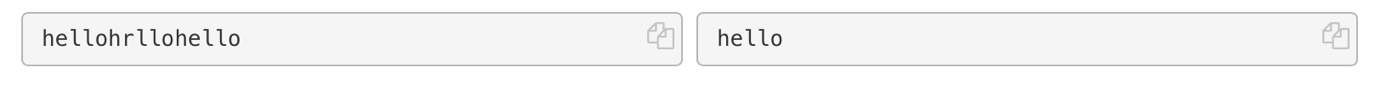
**Vhodni podatki:**

Vhod je sestavljen iz enega niza 𝑠, ki vsebuje male angleške črke (3 ⎢ ⎢ 𝑠 ⎢ 99). To je sporočilo, ki ga je poslala babica. Zagotovljeno je, da je ta niz rezultat trikratnega zapisa besede, pri čemer je bila morda ena črka spremenjena v drugo črko.

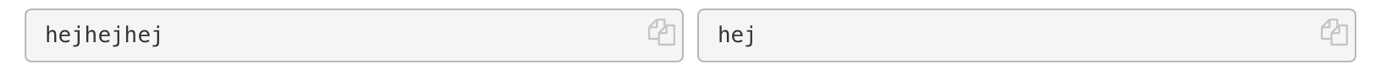
**Izhodni podatki:**

Iznesite en niz 𝑡, izvirna beseda.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

Reševanja problema sem se lotila z rezanjem niza. Vem, da babica vsako besedo napiše natanko trikrat in se lahko vmes kakšno črko zmoti. Tako sem določila dolžino celotnega babičinega besedila in ga razdelila na tri dele. Dobila sem dolžino ene babičine besede in se s tem indeksom sprehajala po celem besedilu. Prava beseda pa je tista, ki se ponovi dvakrat, tretjo tako izločimo iz končnega rezultata.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application

Description automatically generated

# Soft Passwords

Naloga je na spletni strani <https://open.kattis.com/problems/softpasswords> in je vredna 2.5 točk.



BESEDILO NALOGE

Vaše najljubše spletno mesto družbenih medijev spreminja svojo politiko glede preverjanja gesla za prijavo: manjša napaka pri prijavi je zdaj sprejemljiva! Zlasti ob predpostavki, da je geslo, ki ste ga izbrali pri ustvarjanju računa, 𝑆, geslo 𝑃 vneseno med prijavo bo sprejeto, če bo izpolnjen kateri koli od naslednjih pogojev:

• 𝑆 in P sta enaka;

• 𝑆 se lahko oblikuje iz 𝑃 tako, da dodaš eno števko (0-9);

• 𝑆 se lahko oblikuje iz 𝑃 s pripisovanjem ene števke;

• 𝑆 je enako 𝑃 potem, ko obrnete velike črke vseh črk iz 𝑃.

Če želite niz obrniti velike črke, zamenjajte vse velike črke z enakovrednimi malimi črkami in vse male črke z enakovrednimi velikimi črkami, medtem ko pustite vse druge znake enake. Na primer, preobrat velikih in malih črk pa55WORD je PA55word.

Vsako drugo poskusno geslo 𝑃 bo zavrnjeno. Tako na primer, če 𝑆 je c0deninja5, potem bo c0deninja sprejeta, ne pa C0deninja5 ali c0deninja51.

Napišite program, ki glede na alfanumerične nize 𝑆 in 𝑃, določa, ali je 𝑃 je treba sprejeti.

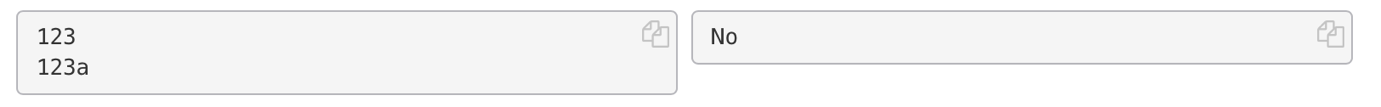
**Vhodni podatki:**

Prva vrstica vnosa je niz 𝑆, shranjeno geslo in druga vrstica vnosa je geslo 𝑃 ki jih je uporabnik vnesel med poskusom prijave. Vsak niz je sestavljen samo iz števk 0–9, malih črk a–z in velikih črk A–Z. Nizovi ne bodo vsebovali presledkov ali kakršnih koli drugih tujih znakov in vsak bo vseboval vsaj enega in največ 101 znakov.

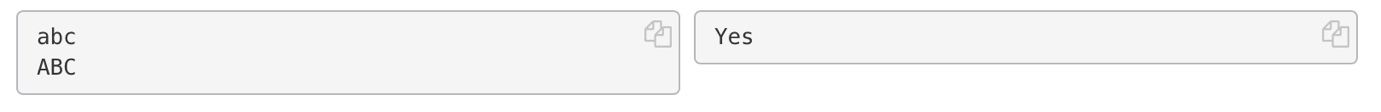
**Izhodni podatki:**

Natisni Yes, če je 𝑃 treba sprejeti v skladu z zgornjimi pravili, drugače pa No.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:



OPIS PROBLEMA IN IDEJA REŠEVANJA

V vhodnih podatkih vnesemo v prvo vrstico geslo S, v drugo pa geslo P, ki ga je uporabnik vnesel med poskusom prijave. Program nam mora pa izpisati ali je geslo P sprejemljivo ali ne. Pri reševanju sem se samo držala alinej (zapisane v navodilih naloge) in sem jih zapisala v kodo. Pomagala sem si z vgrajenimi metodami ter za drugo in tretjo točko uporabila zanko for, da sem zgenerirala nov niz, kateremu sem dodala števko (0-9) na začetku oziroma na koncu.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application

Description automatically generated

# Matrix Inverse

Naloga je na spletni strani <https://open.kattis.com/problems/matrix> in je vredna 3 točke.



BESEDILO NALOGE

Podan kvadrat 𝑛×𝑛 matrika 𝐴, definicija njenega inverza je matrika, ki izpolnjuje enakost:

kjer sta 𝐵 in 𝐼 matriki velikosti 𝑛×𝑛 in 𝐼 je matrika identitete z enicami vzdolž diagonale in ničlami povsod drugod:

A picture containing text, clock

Description automatically generated

Za ta problem bi morali implementirati matrični sistem za računanje inverza za 2×2 matrike.

**Vhodni podatki:**

Vsak testni primer je opisan z dvema vrsticama vnosa. Vsaka vrstica ima dve celi števili z absolutno vrednostjo največ 100, in cela števila, podana po vrstnem redu 𝑎, 𝑏, 𝑐, 𝑑 predstavljajo vrednosti matrike za invertiranje:

A picture containing text, watch, clock, gauge

Description automatically generated

Vsakemu testnemu primeru sledi prazna vrstica, vnos pa se konča s koncem datoteke. Največ je 5 testnih primerov.

**Izhodni podatki:**

Za vsak primer prikažite številko primera, ki ji sledita dve vrstici, ki vsebujeta inverzno vrednost dane matrike. Za vsak testni primer je zajamčeno, da ima inverzno (tj. nobena matrika ni singularna), ta inverzna vrednost pa ima celoštevilsko vrednost. Sledite formatu vzorčnega izhoda.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:

A picture containing graphical user interface

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

Sam problem poiskati inverz 2x2 matrike ni težak, vedeti moramo le nekaj algebre in potem po formuli računamo inverz. Nekoliko težav je bilo z vnašanjem več inputov na nekrat. To rešimo z uporabo zanke while, v katero vnašamo inpute in obdelamo podatke.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application

Description automatically generated

# Polynomial Multiplication 1

Naloga je na spletni strani <https://open.kattis.com/problems/polymul1> in je vredna 1.9 točk.



BESEDILO NALOGE

**Vhodni podatki:**

Standardni vnos se začne s celim število , številom testnih primerov.

Vsak testni primer je sestavljen iz dveh polinomov. Polinom je podan s celim številom , ki označuje stopnjo polinoma, sledi mu zaporedje celih števil , kjer je koeficient polinoma

POZOR: Vhodne in izhodne datoteke za to težavo so precej velike, kar pomeni, da morate biti nekoliko previdni glede učinkovitosti I/0.

**Izhodni podatki:**

Za vsak testni primer izpišite produkt dveh polinomov v enaki obliki kot v vhodu (vključno s stopnjo). Vsi koeficienti v rezultatu se bodo ujemali v predpisano 32-bitno celo število.

Primer vhodnih podatkov 1: Primer izhodnih podatkov 2:

Graphical user interface, application

Description automatically generated

OPIS PROBLEMA IN IDEJA REŠEVANJA

Sprva je potrebno it skozi vse vnešene inpute, ki jih v vhodnih podatkih podamo. Potem pa ustvarimo novo tabelo, ki je velikosti dolžine obeh tabel koeficientov in jo tako polnimo, da pomnožimo vsak element iz prvega polinoma z drugim istoležnim elementov iz drugega polinoma in rezultat shranimo v tabelo na enako mesto, od koder sta bila ta dva elementa. Torej množimo istoležne koeficinete in s tem ustvarjamo novo tabelo. Na koncu moramo tabelo še spremeniti v niz, da bo izpis takšen kot v izhodnih podatkih.

DOKAZ, DA JE NALOGA REŠENA

Graphical user interface, application

Description automatically generated

VIRI

Pomagala sem si s stranjo na spletni strani <https://www.geeksforgeeks.org/multiply-two-polynomials-2/>.