A. 773. Legyen $n \geqslant 3$ egy pozitív egész szám, σ pedig a $\{0, 1, \ldots, n-1\}$ halmaz identitástól különböző olyan permutációja, melyre $\sigma(0) = 0$. A C_σ *titkosítás* minden m pozitív egészt elkódol olyan módon, hogy az m szám n-es számrendszerben felírt alakjában minden egyes a számjegyet $\sigma(a)$-ra cserél. Legyen d egy n-nel nem osztható pozitív egész. Azt mondjuk, hogy a C_σ titkosítás *kompatibilis* d-vel, ha C_σ d minden többszörösét d többszörösévé kódolja el. A d számot *titokzatosnak* nevezzük, ha van hozzá olyan C_σ titkosítás, mely kompatibilis d-vel.

Legyen k egy pozitív egész szám, és legyen $p = 2^k + 1$.

a) Keressük meg a 2 legnagyobb hatványát, amely titokzatos a $2p$-s számrendszerben, és bizonyítsuk be, hogy csak egy titkosítás kompatibilis vele.

b) Keressük meg a p legnagyobb hatványát, amely titokzatos a $2p$-s számrendszerben, és bizonyítsuk be, hogy csak egy titkosítás kompatibilis vele.

c) Tegyük fel, továbbá hogy a fenti p szám prímszám. Keressük meg a legnagyobb titokzatos számot a $2p$-s számrendszerben, és bizonyítsuk be, hogy csak egy titkosítás kompatibilis vele.

Javasolta: *Nikolai Beluhov* (Bulgária)

A. 774. Az ABC háromszög körülírt körének középpontja O, és D egy tetszőleges pont a körülírt körön. Legyen X, Y és Z a D pont merőleges vetülete rendre az OA, OB és OC egyenesen. Bizonyítandó, hogy az XYZ háromszög beírt körének középpontja rajta van az ABC háromszög D ponthoz tartozó Simson-egyenesén.

Javasolta: *Fonyó Lajos* (Keszthely)

Beküldési határidő: 2020. április 10.
Elektronikus munkafüzet: `https://www.komal.hu/munkafuzet`
Cím: KöMaL feladatok, Budapest 112, Pf. 32. 1518

Informatikából kitűzött feladatok

I. 505. Egy tájfutó versenyen nem lehet mindenki egyszerre a terepen, ezért különböző időben indítják a versenyzőket. A célba befutó sportoló csak az addig beérkezett versenyzők eredményeit tudhatja meg.

Egy lezajlott verseny után ismerjük az N számú ($3 < N \leqslant 200$) versenyző `START` indulási és `CEL` beérkezési idejét ($0 < \texttt{START}, \texttt{CEL} \leqslant 10\,000$). Készítsünk programot, amely meghatározza, hogy hány versenyző gondolhatta magáról a beérkezés pillanatában, hogy a legjobb háromban végezhet, tehát van esélye a dobogón állni.

A *standard bemenet* első sora a versenyzők N számát és az ezt követő N sor soronként két számot tartalmaz: a versenyzők indulási és érkezési idejét másodpercben, a beérkezés ideje szerint növekvő sorrendben.

A *standard kimenetre* írjuk ki azon versenyzők számát, akik beérkezéskor dobogós helyre számíthattak.

Bemenet (a / jel sortörést helyettesíti)	**Kimenet**
`6 / 6 1798 / 8 1805 / 13 1809` `14 1815 / 9 1824 / 35 1830`	`4`

Beküldendő egy `i505.zip` tömörített állományban a program forráskódja és egy rövid leírás, ami megadja, hogy a forrásállomány melyik fejlesztői környezetben fordítható.

I. 506. Az idei Nemes Tihamér Nemzetközi Informatikai Tanulmányi Verseny 2. fordulójában a 9–10. évfolyamosoknak a következő, Fasor nevű programozási feladatot kellett megoldaniuk:

„A Százholdas Pagonyban van egy N fából álló fasor, a szomszédos fák távolsága 1 pagométer. Bagoly akkor boldog, ha olyan fa tetején ül, ahonnan nem lát magasabb fát. Mivel Bagoly öregszik, ezért csak a legfeljebb K pagométer távolságra lévő fákat látja. Egy sajátjánál magasabb fát tehát akkor láthat, ha a fasorban a sorszámuk különbsége nem nagyobb, mint K.”

Adjuk meg *táblázatkezelő* segítségével az első olyan fát, amelynek tetején Bagoly boldogan ücsöröghet. A munkafüzet B1-es cellájába lévő N érték alapján készítsünk egy véletlen számsorozatot egymás melletti cellákba, amely a fasor fáinak magasságát adja meg pagométerre kerekített értékben. A munkafüzet B2-es cellájában lévő K érték segítségével jelöljük feltételes formázással az első megfelelő fa magasságát mutató cellát. Ha nincs ilyen fa, akkor ne jelöljünk meg egy cellát sem.

A megoldáshoz segédszámításokat lehet végezni a munkafüzetben, de csak a táblázatkezelő beépített függvényei használhatók, vagyis a megoldás makrót vagy programot ne tartalmazzon.

Beküldendő egy `i506.zip` tömörített mappában a táblázatkezelő munkafüzet, valamint egy rövid leírás, ami megadja az alkalmazott táblázatkezelő nevét és verzióját.

I. 507 (É). A honlapok látogatottságáról a webszerverek legtöbbször naplót vezetnek. Az általunk vizsgált weboldal naplójából részletek találhatók a webstat.txt szöveges állományban. A napló időrend szerint rendezett, egy-egy sorában egy látogatás adatai szerepelnek:

- a használt böngésző neve, vagy egy kötőjel, ha a böngésző típusa nem volt megállapítható;
- a böngészés dátuma (minden dátum 2020. februári);
- a weboldalt felkereső kliensszámítógép IP-címe;
- amennyiben a látogató az oldal címét beírva kereste föl a weboldalt, akkor a „honlap” szó, egyébként annak a weboldalnak vagy alkalmazásnak a címe, ahonnan hivatkozással a honlapra került a látogató.

A szöveges állományban a fenti adatokat szóköz választja el a mintának megfelelően:

```
Chrome 2020.02.11 130.43.220.233 www.google.com
Firefox 2020.02.11 134.255.106.38 www.google.com
Safari 2020.02.11 134.255.91.250 honlap
Safari 2020.02.11 146.255.156.230 www.google.hu
```

Készítsünk programot, amellyel megoldjuk a következő feladatokat. Minden feladatrész elkészítésekor írjuk ki a feladat sorszámát (pl. 1. feladat:), valamint a beolvasás és a kiírás formátumát a minták alapján oldjuk meg. Az ékezetmentes kiírás is elfogadott.

1. Olvassuk be és tároljuk el a webstat.txt állományt, majd adjuk meg, hogy hány adatsor szerepel a naplóban. Például: A beolvasott sorok száma: 300.
2. Adjuk meg táblázatos elrendezéssel, hogy az egyes napokon hány látogató adatai szerepelnek a naplóban. Például: 2020.02.11 59 látogató.
3. Soroljuk fel azokat a böngészőket, amelyek szerepelnek a naplóban. A listában minden név egyszer szerepeljen és a neveket vesszővel válasszuk el. Például: A böngészők: Chrome, Firefox, Safari, Edge, Opera.
4. Adjunk statisztikát arról, hogy a honlapot *Chrome* böngészővel felkeresők hogyan érték el a weboldalt. Számítsuk ki, hogy hány százalékuk adta meg a honlap címét, illetve hány százalékuk jött máshonnan a honlapra. Az eredményt egy tizedesjegyre kerekítve írjuk ki, például:

```
www.google.com:      50.5%
honlap:              44.7%
android-app:          2.1%
kereso.startlap.hu:   0.5%
www.google.hu:        1.1%
hu.m.wikipedia.org:   1.1%
```

5. Vizsgáljuk meg az adatokat, és adjuk meg azokat az IP címeket, amelyekről egy adott napon többször is fölkeresték a weboldalt. A listában minden IP-cím csak egyszer szerepeljen. Az eredményt az alábbi formában adjuk meg:
Amely címekről többször is jártak az oldalon egy adott napon: 176.63.29.84, 176.63.7.203, 188.156.108.17 ...
6. Kérjünk be egy IP-címet, és adjuk meg, hogy mely napokon keresték föl a weboldalt a bekért címnek legalább az első két bájtjával azonos címekről. Készítsünk egy szöveges állományt, amelybe soronként megadjuk a talált napokat és IP-címeket a napló szerinti sorrendben. Az állomány neve a bekért IP-címből épüljön fel úgy, hogy a címben szereplő pontok helyére az aláhúzásjel kerüljön, és a kiterjesztése txt legyen.

Beküldendő egy `i507.zip` tömörített állományban a program forráskódja és egy rövid leírás, ami megadja, hogy a forrásállomány melyik fejlesztői környezetben fordítható.

I/S. 43. Jelölje $f(n)$ az n-edik Fibonacci-számot, ahol $f(0)=1$, $f(1)=1$, valamint $f(n+2)=f(n)+f(n+1)$. Készítsünk programot, amely adott N-re meghatározza az $f\big(f(N)\big)$ értékének utolsó két számjegyét.

Bemenet: az első sor tartalmazza az N nemnegatív egész számot.

Kimenet: az egyetlen sorban $f\big(f(N)\big)$ utolsó két számjegye.

Példa:

Bemenet	Kimenet
6	77

Korlátok: $1 \leqslant N \leqslant 10^{15}$. Időkorlát: 0,4 mp.

Értékelés: a pontok 50%-a kapható, ha $N \leqslant 10$.

Beküldendő egy `is43.zip` tömörített állományban a megfelelően dokumentált és kommentezett forrásprogram, amely tartalmazza a megoldás lépéseit, valamint megadja, hogy a program melyik fejlesztői környezetben futtatható.

S. 142. Egy élelmiszerfeldolgozással foglalkozó cég raktárában az almák egy futószalagon érkeznek sorban. Mindegyik almáról tudjuk, hogy mennyire finom. Az almákat szeretnék zsákokba rendezni úgy, hogy az első néhány az első zsákba kerül, a következő néhány a második zsákba, a következő néhány a harmadikba, és így tovább. Egy zsákba legfeljebb K darab alma fér. Egy zsák alma annyira finom, mint a benne lévő legfinomabb alma. Írjunk programot, ami úgy osztja be az almákat a zsákokba, hogy a legfinomabb és legkevésbé finom zsák almák finomsága közötti különbség a lehető legkisebb legyen.

Bemenet: az első sor tartalmazza az almák N számát és a zsákok K méretét. A második sor N darab számot tartalmaz: az i-edik szám azt jelenti, hogy az i-edik alma finomsága F_i.

Kimenet: egyetlen szám, amely megadja a legkisebb finomságbeli különbséget a legfinomabb és legkevésbé finom zsák alma között optimális beosztás esetén.

Példa:

Bemenet	Kimenet
5 2 31 88 41 72 9	47

Korlátok: $1 \leqslant N, K \leqslant 100\,000$, $1 \leqslant F_i \leqslant 10^9$. Időkorlát: 0,3 mp.

Értékelése: a pontok 30%-a kapható, ha $N \leqslant 1000$.

Beküldendő egy `s142.zip` tömörített állományban a megfelelően dokumentált és kommentezett forrásprogram, amely tartalmazza a megoldás lépéseit, valamint megadja, hogy a program melyik fejlesztői környezetben futtatható.

❄

A feladatok megoldásai regisztráció után a következő címen tölthetők fel:
`https://www.komal.hu/munkafuzet`
Beküldési határidő: 2020. április 10.

❄