**A**

Egy büfében 10 féle üdítőt lehet vásárolni. Egy szekvenciális inputfájlban tároljuk, hogy melyik napon, melyik üdítőből, mennyi fogyott el (dl-ben megadva). Az állomány minden sorában egy-egy nap üdítőital forgalmának adatait találjuk: az adott nap dátumát (string) és utána az egyes üdítőfajtákból fogyasztott mennyiségeket (10 természetes szám) sorközökkel és/vagy tabulátorjelekkel elválasztva. Feltehetjük, hogy az állomány helyesen van kitöltve.

Listázza ki a konzolra azon napok dátumát, amikor nem volt olyan üdítő, amelyből legalább 10 liter ne fogyott volna, valamint határozza meg ezen napok közül az egyik olyat (annak dátumát), amikor a legkisebb volt a forgalom (egységár\*mennyiség értékek összege)! Az egyes üdítők egységárát (billentyűzetről) olvassa be a program.

**Például az input fájl adatai:**

2010.01.01 0 10 20 4 30 7 25 14 24 12

2010.01.02 40 10 2 14 32 8 2 24 4 55

2010.01.03 11 0 0 44 31 27 2 4 21 0

2010.01.04 0 5 34 6 11 0 2 24 18 40

Megfelelt szintű (közepes) a megoldás akkor, ha

* az érvényes tesztesetekre jól működik a program,
* a program ciklusai tanult programozási tételekből származnak,
* a program a szöveges állománynak legfeljebb egy sornyi adatát tárolja egyszerre,
* a szöveges állományt csak egyszer olvashatja végig.

Kiválóan megfelelt szintű a megoldás akkor, ha megfelelt, és

* az érvénytelen teszteseteket is kezeli: nem létező állománynév,
* egy nap adatainak az input fájlból történő beolvasására egy külön függvény szolgál,
* nem ágyaz közvetlenül ciklust ciklusba, hanem ezt elkerülendő megfelelően paraméterezett függvényeket használ.

Értékelés után a programját EHAkód.zip formában az alábbi helyre töltse fel:

[\\inf.elte.hu\dfs\zh\PROG\C](file:///\\inf.elte.hu\dfs\zh\PROG\C)

**B**

Egy büfében 10 féle üdítőt lehet vásárolni. Egy szekvenciális inputfájlban tároljuk, hogy melyik napon, melyik üdítőből, mennyi fogyott el (dl-ben megadva). Az állomány minden sorában egy-egy nap üdítőital forgalmának adatait találjuk: az adott nap dátumát (string) és utána az egyes üdítőfajtákból fogyasztott mennyiségeket (10 természetes szám) sorközökkel és/vagy tabulátorjelekkel elválasztva. Feltehetjük, hogy az állomány helyesen van kitöltve.

Listázza ki a konzolra azoknak a napoknak a dátumát, amikor az egyes üdítőkből elfogyasztott mennyiség egymástól való eltérése nem volt nagyobb, mint 5 liter, és számolja meg, hogy ezen napok között hány volt olyan, amikor a forgalom (egységár\*mennyiség értékek összege) 5000 Ft-nál nagyobb volt. Az egyes üdítők egységárát (billentyűzetről) olvassa be a program.

**Például az input fájl adatai:**

2010.01.01 0 10 20 4 30 7 25 14 24 12

2010.01.02 40 10 2 14 32 8 2 24 4 55

2010.01.03 11 0 0 44 31 27 2 4 21 0

2010.01.04 0 5 34 6 11 0 2 24 18 40

Megfelelt szintű (közepes) a megoldás akkor, ha

* az érvényes tesztesetekre jól működik a program,
* a program ciklusai tanult programozási tételekből származnak,
* a program a szöveges állománynak legfeljebb egy sornyi adatát tárolja egyszerre,
* a szöveges állományt csak egyszer olvashatja végig.

Kiválóan megfelelt szintű a megoldás akkor, ha megfelelt, és

* az érvénytelen teszteseteket is kezeli: nem létező állománynév,
* egy nap adatainak az input fájlból történő beolvasására egy külön függvény szolgál,
* nem ágyaz közvetlenül ciklust ciklusba, hanem ezt elkerülendő megfelelően paraméterezett függvényeket használ.

Értékelés után a programját EHAkód.zip formában az alábbi helyre töltse fel:

[\\inf.elte.hu\dfs\zh\PROG\C](file:///\\inf.elte.hu\dfs\zh\PROG)

**C**

Egy szöveges állományban tároljuk egy évfolyam ZH eredményét. Az állomány minden sorában egy-egy hallgató adatait találjuk: a hallgató EHA kódját és mellette a 6 db feladatra kapott pontszámát. Az adatok között szóköz(ök) és/vagy tabulátor jelek találhatók. Az egyes feladatokra 0-10-ig lehetett pontot kapni, ha valaki nem oldott meg egy feladatot, akkor X-et írtak a pont helyére. (Ez csak jelzés a feladat hiányára.) Az osztályzat az összpontszámtól függ: 20 ponttól kettes, 30-tól hármas, 40-től négyes, ötventől ötös. Feltehetjük, hogy az állomány helyesen van kitöltve.

Listázza ki a hallgatók kódját és összpontszámát a konzolra, valamint mondja meg, hány ötös, négyes, hármas, kettes és egyes jegy született!

**Például az input fájl adatai:**

EEADAAI.ELTE X 9 6 0 10 5

MDADAAI.ELTE 10 9 7 8 10 9

NVADAAT.ELTE 9 10 10 8 9 X

Megfelelt szintű (közepes) a megoldás akkor, ha

* az érvényes tesztesetekre jól működik a program,
* a program ciklusai tanult programozási tételekből származnak,
* a program a szöveges állománynak legfeljebb egy sornyi adatát tárolja egyszerre,
* a szöveges állományt csak egyszer olvashatja végig.

Kiválóan megfelelt szintű a megoldás akkor, ha megfelelt, és

* az érvénytelen teszteseteket is kezeli: nem létező állománynév,
* egy nap adatainak az input fájlból történő beolvasására egy külön függvény szolgál,
* nem ágyaz közvetlenül ciklust ciklusba, hanem ezt elkerülendő megfelelően paraméterezett függvényeket használ.

Értékelés után a programját EHAkód.zip formában az alábbi helyre töltse fel:

[\\inf.elte.hu\dfs\zh\PROG\C](file:///\\inf.elte.hu\dfs\zh\PROG)

**D**

Egy szöveges állományban tároljuk egy magasugró verseny eredményét. Az állomány minden sorában egy-egy versenyző adatait találjuk: a versenyző nevét (szóközök nélküli szöveg) és mellette azt, hogy az egyes próbáknál mekkorát ugrott. A versenyen egy versenyző 6-szor próbálkozhatott. Ha leverte a lécet, X a próba eredménye, ha kihagyta az adott magasságot, akkor O. Feltehetjük, hogy az állomány helyesen van kitöltve.

Listázza ki azoknak a nevét a konzolra, akiknek csak egy sikeres próbálkozása (150 cm-nél magasabb ugrása) volt, valamint adja meg a legjobb ugrást teljesítő sportoló nevét és összes eredményét!

**Például az input fájl adatai:**

Nagy\_Bela 120 X O 125 130 X

Kiss\_Agoston 140 145 X 150 155 X

Toth\_Pal 135 O 140 145 X 155

Megfelelt szintű (közepes) a megoldás akkor, ha

* az érvényes tesztesetekre jól működik a program,
* a program ciklusai tanult programozási tételekből származnak,
* a program a szöveges állománynak legfeljebb egy sornyi adatát tárolja egyszerre,
* a szöveges állományt csak egyszer olvashatja végig.

Kiválóan megfelelt szintű a megoldás akkor, ha megfelelt, és

* az érvénytelen teszteseteket is kezeli: nem létező állománynév,
* egy nap adatainak az input fájlból történő beolvasására egy külön függvény szolgál,
* nem ágyaz közvetlenül ciklust ciklusba, hanem ezt elkerülendő megfelelően paraméterezett függvényeket használ.

Értékelés után a programját EHAkód.zip formában az alábbi helyre töltse fel:

[\\inf.elte.hu\dfs\zh\PROG\C](file:///\\inf.elte.hu\dfs\zh\PROG)

**E**

Egy vállalat portaszolgálatán feljegyzik a belépő és kilépő személyek adatait. Belépéskor rögzítik az aktuális időpontot, a belépő nevét és azonosítására szolgáló igazolványszámot, távozáskor pedig a fenti három adat mellé kerül a kilépés időpontja. A napi adatokat azonosító szerint, azon belül belépési idő szerint növekvően rendezve egy szöveges állományban tárolják.

Például az input fájl adatai:

10:30 Gipsz\_Jakab 1234567A 10:15

10:45 Gipsz\_Jakab 1234567A 11:30

08:35 Gyanús\_Móric 1x1x1x1x1x

08:35 Gyanús\_Móric 1x1x1x1x1x 08:10

08:05 Gyanús\_Móric 1x1x1x1x1x

08:45 Szer\_Elek 33332222 09:15

Feltehetjük, hogy az állomány helyesen van kitöltve, annak egy sorában három vagy négy sztringet találunk, amelyeket tetszőleges számú elválasztó jel határol, és ezeknek a sztringeknek a belsejében viszont nem találunk elválasztó jelet.

Mondja meg, hány hibás sora van az állománynak (a hibás sor definíciója lejjebb olvasható) és listázza ki azokat a személyeket, akik jelenleg benn vannak a vállalatnál (beléptek, de még nem távoztak)!

Megfelelt szintű (közepes) a megoldás akkor, ha

* az érvényes tesztesetekre jól működik a program: hibás sor az, ahol a belépési időpont későbbi, mint a távozás időpontja,
* a program ciklusai tanult programozási tételekből származnak,
* a programban nem használhat olyan tömböt vagy sztringet, amelynek hossza a szöveges állomány sorainak számától függ,
* Csak egyszer olvassa végig az állományt.

Kiválóan megfelelt szintű a megoldás akkor, ha megfelelt, de

* az is hibás sornak számít (például a 4. és 5. sorok), amelyet olyan sor előz meg, ahol ugyanaz a személy szerepel és vagy nincs távozási időpont (lásd Gyanús\_Móricot a 3. sorban), vagy a távozási időpont későbbi, mint a vizsgált sor belépési időpontja (lásd Gyanús\_Móricot a 4. sorban),
* az érvénytelen teszteseteket is kezeli: nem létező állománynév észlelése, üres sorok az állományban nem okoznak problémát,
* nem ágyaz közvetlenül ciklust ciklusba, hanem ezt elkerülendő megfelelően paraméterezett függvényeket használ,
* a szekvenciális inputfájl egy sorának beolvasására olyan függvényt használ, amely csak a feldolgozáshoz szükséges adatokat (kilépett-e illetve mennyi ideig volt benn) adja vissza.

Értékelés után a programját EHAkód.zip formában az alábbi helyre töltse fel:

[\\nas1.inf.elte.hu\zh\PROG\C](file:///\\nas1.inf.elte.hu\zh\PROG\C)

**F**

Egy vállalat portaszolgálatán feljegyzik a belépő és kilépő személyek adatait. Belépéskor rögzítik az aktuális időpontot, a belépő nevét és azonosítására szolgáló igazolványszámot, távozáskor pedig a fenti három adat mellé kerül a kilépés időpontja. A napi adatokat azonosító szerint, azon belül belépési idő szerint növekvően rendezve egy szöveges állományban tárolják.

Például az input fájl adatai:

10:30 Gipsz\_Jakab 1234567A 10:15

10:45 Gipsz\_Jakab 1234567A 11:30

08:35 Gyanús\_Móric 1x1x1x1x1x

08:35 Gyanús\_Móric 1x1x1x1x1x 08:10

08:05 Gyanús\_Móric 1x1x1x1x1x

08:45 Szer\_Elek 33332222 09:15

Feltehetjük, hogy az állomány helyesen van kitöltve, annak egy sorában három vagy négy sztringet találunk, amelyeket tetszőleges számú elválasztó jel határol, és ezeknek a sztringeknek a belsejében viszont nem találunk elválasztó jelet.

Listázza ki a hibás sorait az állománynak (a hibás sor definíciója lejjebb olvasható) és számolja meg azokat a személyeket, akik jelenleg benn vannak a vállalatnál (beléptek, de még nem távoztak)!

Megfelelt szintű (közepes) a megoldás akkor, ha

* az érvényes tesztesetekre jól működik a program: hibás sor az, ahol a belépési időpont későbbi, mint a távozás időpontja,
* a program ciklusai tanult programozási tételekből származnak,
* a programban nem használhat olyan tömböt vagy sztringet, amelynek hossza a szöveges állomány sorainak számától függ,
* Csak egyszer olvassa végig az állományt.

Kiválóan megfelelt szintű a megoldás akkor, ha megfelelt, de

* az is hibás sornak számít (például a 4. és 5. sorok), amelyet olyan sor előz meg, ahol ugyanaz a személy szerepel és vagy nincs távozási időpont (lásd Gyanús\_Móricot a 3. sorban), vagy a távozási időpont későbbi, mint a vizsgált sor belépési időpontja (lásd Gyanús\_Móricot a 4. sorban),
* az érvénytelen teszteseteket is kezeli: nem létező állománynév észlelése, üres sorok az állományban nem okoznak problémát,
* nem ágyaz közvetlenül ciklust ciklusba, hanem ezt elkerülendő megfelelően paraméterezett függvényeket használ,
* a szekvenciális inputfájl egy sorának beolvasására olyan függvényt használ, amely csak a feldolgozáshoz szükséges adatokat (kilépett-e illetve mennyi ideig volt benn) adja vissza.

Értékelés után a programját EHAkód.zip formában az alábbi helyre töltse fel:

[\\nas1.inf.elte.hu\zh\PROG\C](file:///\\nas1.inf.elte.hu\zh\PROG\C)