NÉV:	Eredmény:
NEPTUN KÓD:	nem jó megfelelt kiváló
GÉPSORSZÁM:	Elfogadó tanár:

A

Az alábbi feladatok megoldásához az előadáson bevezetett osztálykönyvtárat kell használnia. Az osztálysablonok kódja megtalálható a http://people.inf.elte.hu/gt/oaf/lib.zip állományban. A megoldásokat az előadáson látott módon tevékenység objektumokkal kell megvalósítani, amelyeknek osztálya vagy az öt programozási tétel (Summation, Counting, Selection, LinSearch, MaxSearch) osztálysablonjának valamelyikéből származik, vagy az általános felsoroló (Enumerator) osztálysablonból. Nem definiálhatja felül a run(), do(), loopCond() metódusokat, az init()-et is csak akkor, ha a Summation osztályból származtat! A saját kódban egy ifstream típusú objektum >> operátora csak az Enumerator osztályból származtatott osztály first() illetve next() metódusaiban használható. A saját kódban egyáltalán ne szerepeljen ciklus, illetve rekurzív függvényhívás! A bemeneti adatokat tartalmazó szöveges állományokról feltételezzük, hogy helyesen vannak kitöltve, csak a létezésüket kell ellenőrizni.

Egy szekvenciális inputfájlban egy mozi egy napi vetítései és forgalma található. A fájl egy eleme egy eladott jegy adatait tartalmazza szóközökkel tagolva: a vetített film azonosítója (6 karakter), a terem nevét (ez szóközt nem tartalmaz), a vetítés kezdetének idejét (HH:MM formában), és a vásárolt hely számából és az eladott jegy típusából álló érték-párt. Három különböző típusú jegyet lehet vásárolni: gyermek (G) – 500 forint, diák (D) – 800 forint, felnőtt (F) – 1000 forint. A fájl sorai kezdési időpontok szerint rendezettek. Egy adott időpontban legfeljebb egy film vetítése kezdődik. A szöveges állományt csak egyszer olvashatja végig és nem használhat a szöveges állomány sorainak számától függő méretű belső tárolót.

cím	terem	idő	hely	típus
AAAAAA	Jancsó	10:15	9	G
AAAAAA	Jancsó	10:15	3	G
AAAAAA	Jancsó	10:15	5	F
BBBBBB	Coppola	19:45	15	F
BBBBBB	Coppola	19:45	7	D

2 pontért: Melyik vetítés hozta a legnagyobb bevételt és mennyit?

3 pontért: Oldja meg az előző feladatot úgy, hogy a feladat megoldását legfelső szinten a Summation-ból származtatja (a MaxSearch helyett).

Segítség: Természetes számok maximumának meghatározásánál a maximumkeresést visszavezethetjük összegzésre úgy, hogy kezdetben *maxérték*:=*0*, amit minden lépésben a *maxérték*:=**max**(*maxérték*, *aktuálisérték*) módosíthat (ahol a **max** ilyenkor egy asszociatív egységelemes művelet).

4 pontért: Válaszolja meg mindkét alábbi kérdést: Melyik vetítés hozta a legnagyobb bevételt és mennyit? Hány vetítésnek volt az összbevétele 2000 Ft-nál kevesebb?

Segítség: A számlálás is egy összegzés, és két ugyanazon felsoroláson értelmezett összegzés összevonható egyetlen olyan összegzésbe, amely eredménypárt állít elő.

5 pontért: Válaszolja meg mindkét alábbi kérdést: Melyik vetítés hozta a legnagyobb bevételt és mennyit? Hány olyan vetítés volt, amelyikre csak diákok váltottak jegyet?

Miután a programját bemutatta és azt elfogadták, töltse fel azt Neptun-kód.zip formában Windows alól a \\nas2.inf.elte.hu\zh\OAF, Linux alól az **smb://nas2.inf.elte.hu** a **zh\OAF** könyvtárba.

NÉV:	Eredmény:
NEPTUN KÓD:	nem jó megfelelt kiváló
GÉPSORSZÁM:	Elfogadó tanár:

B

Az alábbi feladatok megoldásához az előadáson bevezetett osztálykönyvtárat kell használnia. Az osztálysablonok kódja megtalálható a http://people.inf.elte.hu/gt/oaf/lib.zip állományban. A megoldásokat az előadáson látott módon tevékenység objektumokkal kell megvalósítani, amelyeknek osztálya vagy az öt programozási tétel (Summation, Counting, Selection, LinSearch, MaxSearch) osztálysablonjának valamelyikéből származik, vagy az általános felsoroló (Enumerator) osztálysablonból. Nem definiálhatja felül a run(), do(), loopCond() metódusokat, az init()-et is csak akkor, ha a Summation osztályból származtat! A saját kódban egy ifstream típusú objektum >> operátora csak az Enumerator osztályból származtatott osztály first() illetve next() metódusaiban használható. A saját kódban egyáltalán ne szerepeljen ciklus, illetve rekurzív függvényhívás! A bemeneti adatokat tartalmazó szöveges állományokról feltételezzük, hogy helyesen vannak kitöltve, csak a létezésüket kell ellenőrizni.

Egy szekvenciális inputfájlban egy mozi egy napi vetítései és forgalma található. A fájl egy eleme egy eladott jegy adatait tartalmazza szóközökkel tagolva: a vetített film azonosítója (6 karakter), a terem nevét (ez szóközt nem tartalmaz), a vetítés kezdetének idejét (HH:MM formában), és a vásárolt hely számából és az eladott jegy típusából álló érték-párt. Három különböző típusú jegyet lehet vásárolni: gyermek (G) – 500 forint, diák (D) – 800 forint, felnőtt (F) – 1000 forint. A fájl sorai kezdési időpontok szerint rendezettek. Egy adott időpontban legfeljebb egy film vetítése kezdődik. A szöveges állományt csak egyszer olvashatja végig és nem használhat a szöveges állomány sorainak számától függő méretű belső tárolót.

cím	terem	idő	hely	típus
AAAAAA	Jancsó	10:15	9	G
AAAAAA	Jancsó	10:15	3	G
AAAAAA	Jancsó	10:15	5	F
BBBBBB	Coppola	19:45	15	F
BBBBBB	Coppola	19:45	7	D

2 pontért: Melyik vetítésen vett részt a legtöbb gyerek és hány?

3 pontért: Oldja Oldja meg az előző feladatot úgy, hogy a feladat megoldását legfelső szinten a Summation-ból származtatja (a MaxSearch helyett).

Segítség: Természetes számok maximumának meghatározásánál a maximumkeresést visszavezethetjük összegzésre úgy, hogy kezdetben *maxérték*:=*0*, amit minden lépésben a *maxérték*:=**max**(*maxérték*, *aktuálisérték*) módosíthat (ahol a **max** ilyenkor egy asszociatív egységelemes művelet).

4 pontért: Válaszolja meg mindkét alábbi kérdést: Melyik vetítésen vett részt a legtöbb gyerek és hány? Hány vetítésen nem voltak egyáltalán gyerekek?

Segítség: A számlálás is egy összegzés, és két ugyanazon felsoroláson értelmezett összegzés összevonható egyetlen olyan összegzésbe, amely eredménypárt állít elő.

5 pontért: Válaszolja meg mindkét alábbi kérdést: Melyik vetítésen vett részt a legtöbb gyerek és hány? Hány vetítésen vettek részt kizárólag diákok?

Miután a programját bemutatta és azt elfogadták, töltse fel azt Neptun-kód.zip formában Windows alól a \\nas2.inf.elte.hu\zh\OAF, Linux alól az **smb://nas2.inf.elte.hu** a **zh\OAF** könyvtárba.

NÉV:	Eredmény:
NEPTUN KÓD:	nem jó megfelelt kiváló
GÉPSORSZÁM:	Elfogadó tanár:

C

Az alábbi feladatok megoldásához az előadáson bevezetett osztálykönyvtárat kell használnia. Az osztálysablonok kódja megtalálható a http://people.inf.elte.hu/gt/oaf/lib.zip állományban. A megoldásokat az előadáson látott módon tevékenység objektumokkal kell megvalósítani, amelyeknek osztálya vagy az öt programozási tétel (Summation, Counting, Selection, LinSearch, MaxSearch) osztálysablonjának valamelyikéből származik, vagy az általános felsoroló (Enumerator) osztálysablonból. Nem definiálhatja felül a run(), do(), loopCond() metódusokat, az init()-et is csak akkor, ha a Summation osztályból származtat! A saját kódban egy ifstream típusú objektum >> operátora csak az Enumerator osztályból származtatott osztály first() illetve next() metódusaiban használható. A saját kódban egyáltalán ne szerepeljen ciklus, illetve rekurzív függvényhívás! A bemeneti adatokat tartalmazó szöveges állományokról feltételezzük, hogy helyesen vannak kitöltve, csak a létezésüket kell ellenőrizni.

Egy tömegközlekedéssel foglalkozó vállalat bevezette az elektronikus bérleteket tesztelési céllal egy napra, és egy szöveges fájl tartalmazza az eltárolt utazásokat: egy sorban az utas azonosítóját (8 jegyű szám), és egy időkód (HH:MM) – járat (sztring) párt. A fájl azonosítók, azon belül időpontok szerint rendezett. A szöveges állományt csak egyszer olvashatja végig és nem használhat a szöveges állomány sorainak számától függő méretű belső tárolót.

azonosító	idő	járat
11111111	06:30	9
11111111	07:45	17A
22222222	07:36	7E

2 pontért: Vajon utazott-e minden utas a 7E járattal?

3 pontért: Oldja meg az előző feladatot úgy, hogy a feladat megoldását legfelső szinten a Summation-ból származtatja (a LinSearch helyett).

Segítség: Az optimista lineáris keresést visszavezethetjük összegzésre, ha „összeéseljük” a keresés során vizsgált feltételeket.

4 pontért: Válaszolja meg mindkét alábbi kérdést: Vajon utazott-e minden utas a 7E járattal? Hány utas utazott a 7E járaton!

Segítség: A számlálás is egy összegzés, és két ugyanazon felsoroláson értelmezett összegzés összevonható egyetlen olyan összegzésbe, amely eredménypárt állít elő.

5 pontért: Válaszolja meg mindkét alábbi kérdést: Vajon utazott-e minden utas a 7E járattal? Hány utas utazott legalább háromszor!

Miután a programját bemutatta és azt elfogadták, töltse fel azt Neptun-kód.zip formában Windows alól a \\nas2.inf.elte.hu\zh\OAF, Linux alól az **smb://nas2.inf.elte.hu** a **zh\OAF** könyvtárba.

NÉV:	Eredmény:
NEPTUN KÓD:	nem jó megfelelt kiváló
GÉPSORSZÁM:	Elfogadó tanár:

D

Az alábbi feladatok megoldásához az előadáson bevezetett osztálykönyvtárat kell használnia. Az osztálysablonok kódja megtalálható a http://people.inf.elte.hu/gt/oaf/lib.zip állományban. A megoldásokat az előadáson látott módon tevékenység objektumokkal kell megvalósítani, amelyeknek osztálya vagy az öt programozási tétel (Summation, Counting, Selection, LinSearch, MaxSearch) osztálysablonjának valamelyikéből származik, vagy az általános felsoroló (Enumerator) osztálysablonból. Nem definiálhatja felül a run(), do(), loopCond() metódusokat, az init()-et is csak akkor, ha a Summation osztályból származtat! A saját kódban egy ifstream típusú objektum >> operátora csak az Enumerator osztályból származtatott osztály first() illetve next() metódusaiban használható. A saját kódban egyáltalán ne szerepeljen ciklus, illetve rekurzív függvényhívás! A bemeneti adatokat tartalmazó szöveges állományokról feltételezzük, hogy helyesen vannak kitöltve, csak a létezésüket kell ellenőrizni.

Egy tömegközlekedéssel foglalkozó vállalat bevezette az elektronikus bérleteket tesztelési céllal egy napra, és egy szöveges fájl tartalmazza az eltárolt utazásokat: egy sorban az utas azonosítóját (8 jegyű szám), és egy időkód (HH:MM) – járat (sztring) párt. A fájl azonosítók, azon belül időpontok szerint rendezett. A szöveges állományt csak egyszer olvashatja végig és nem használhat a szöveges állomány sorainak számától függő méretű belső tárolót.

azonosító	idő	járat
11111111	06:30	9
11111111	07:45	17A
22222222	07:36	7E

2 pontért: Volt-e olyan utas, aki 06:00 óra előtt utazott?

3 pontért: Oldja meg az előző feladatot úgy, hogy a feladat megoldását legfelső szinten a Summation-ból származtatja (a LinSearch helyett).

Segítség: A lineáris keresést visszavezethetjük összegzésre úgy, ha „összevagyoljuk” a keresés során vizsgált feltételeket.

4 pontért: Válaszolja meg mindkét alábbi kérdést: Volt-e olyan utas, aki 06:00 óra előtt utazott? Hány utas utazott 06:00 óra előtt?

Segítség: A számlálás is egy összegzés, és két ugyanazon felsoroláson értelmezett összegzés összevonható egyetlen olyan összegzésbe, amely eredménypárt állít elő.

5 pontért: Válaszolja meg mindkét alábbi kérdést: Volt-e olyan utas, aki 06:00 óra előtt utazott? Hány utas nem utazott egyáltalán a 7E járattal?

Miután a programját bemutatta és azt elfogadták, töltse fel azt Neptun-kód.zip formában Windows alól a \\nas2.inf.elte.hu\zh\OAF, Linux alól az **smb://nas2.inf.elte.hu** a **zh\OAF** könyvtárba.