HÁZI FELADAT

Programozás alapjai 2.

Specifikáció

Nagy Ádám Balázs

CKTO0H

2025. március 30.

Tartalom

1.Feladatválasztás

Naptár osztály

Készítsen naptár tárolására osztályt! Az osztály legyen képes tetszőleges naptári napot lefoglalni, és le lehessen kérdezni, hogy eddig milyen napok foglaltak, valamint egy adott nap foglalt-e vagy szabad (szabad, ha nem foglalt). Számítsa ki, hogy egy adott naptári nap milyen napra esik. Az osztály legyen képes két naptári nap összehasonlítására, eltelt napok számának kiszámítására! Legyen lehetőség táblázatos formában éves, és havi naptár nyomtatására! Valósítsa meg az összes értelmes műveletet operátor átdefiniálással (overload), de nem kell ragaszkodni az összes operátor átdefiniálásához! Legyen az osztálynak iterátora is! Demonstrálja a működést külön modulként fordított tesztprogrammal! A megoldáshoz **ne** használjon STL tárolót!

2.Feladatspecifikáció

2.1Program feladatai

A program célja, hogy egy bővíthető naptár osztály készüljön el. Feladati közé tartoznak a feladatválasztásnál felsoroltak, valamint, hogy különböző típusú események álljanak rendelkezésre a napok lefoglalásához. A naptárnak nincs határa időben, dinamikusan foglal területet az éveknek ha a távoli jövővel, vagy múlttal szeretne valaki műveleteket végezni. A naptárnak részei lesznek az évek, hónapok és a napok. A napok foglalását egy másik osztály végzi el. A naptár osztály, az implicit generálódó függvényeken kívül a következő műveleteket is megvalósítja :

- Létrehozás

- Megszüntetés
- Másolás
- Értékadás
- Más osztálynak üzenés

A felsorolt függvények konstans környezetben is értelmezhető változata is megvalósításra kerül.

2.2Program tesztelése

A felhasználó választhat, hogy melyik funkciót szeretné tesztelni. A feladat utáni számot kell beírnia a felhasználónak. Nap lefoglalás (1), mely napok foglaltak (2), foglalást törölni (3) naptári nap milyen napra esik (4), két nap összehasonlítása (5), két nap között eltelt idő száma (6), naptár kiírása a képernyőre (7). Az azután beírt adatt alapján fog lefutni a megfelelő teszteset. A dátum beírásakor az évek, hónapok és napok elválaszthatóak ponttal, valamint szóközzel. A hónapokat számokkal és betűkkel is ki lehet írni.

Olyan esetben amikor nem megfelelő hónapot vagy napot írnak be, azaz ha 1-nél kisebb, vagy 12-nél vagy az adott hónap napjainak számánál nagyobb, kivétel képződik, aminek elkapása a főprogramban történik. Abban az esetben ha egy foglalt napra szeretnénk új eseményt foglalni, visszakérdez a program, hogy szeretné a felhasználó a korábbi foglalását törölni és újat tenni a helyére.

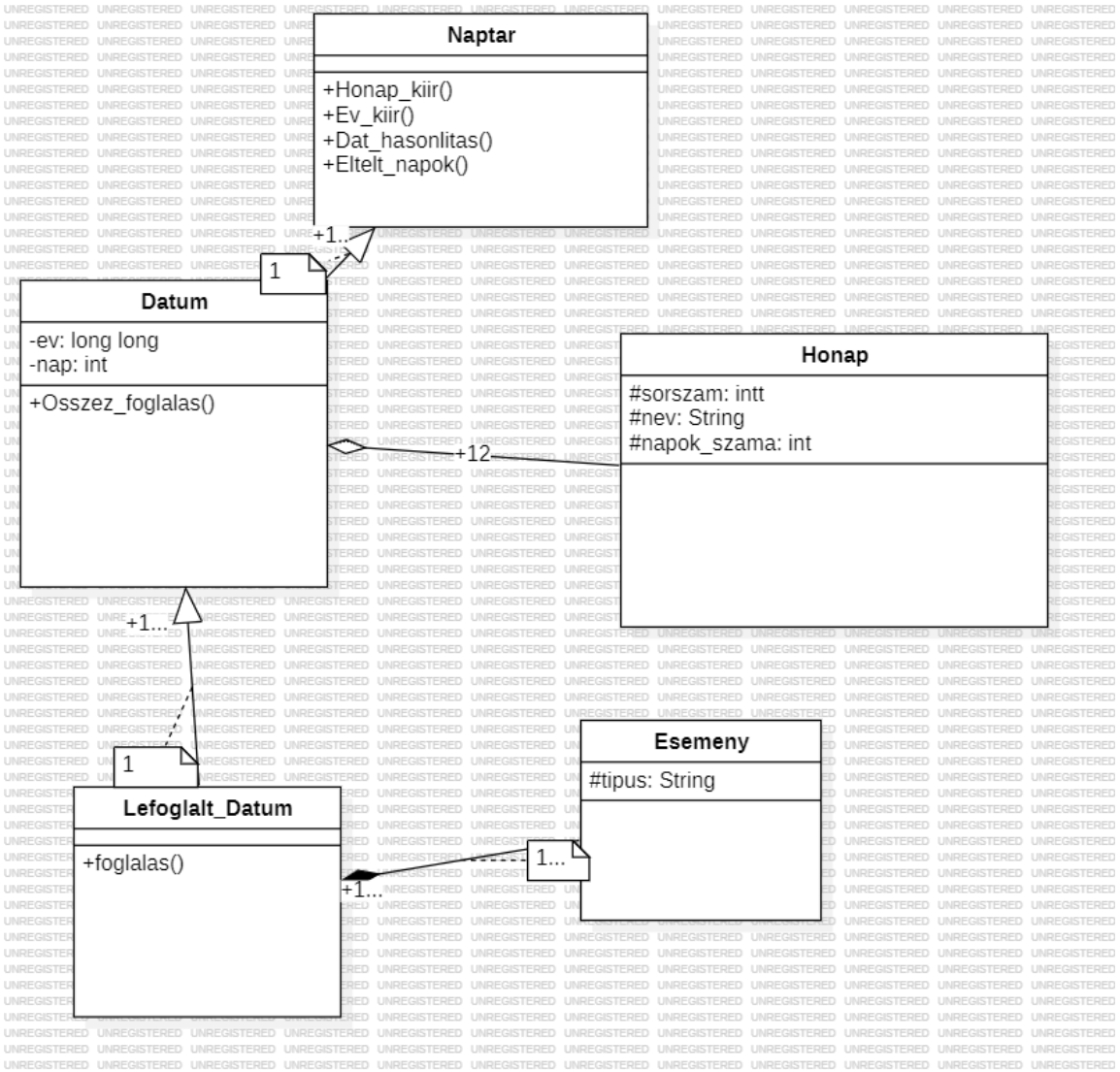
3.Terv

A feladat egy naptár osztály és egy tesztprogram létrehozását követeli. Ez más osztály segítségével lehet megvalósítani.

3.1.Objektum terv

A projekt főosztálya a naptár osztály. Ez dátumokból áll, a Datum osztály örökli a naptár tulajdonságait és viselkedéseit. A dátumok évek és napok mellett az adott hónapot is tartalmazza. A dátum és hónapok között aggregáció van, mivel a hónapok a dátumon kívül is létezhetnek. A Datum

tárolja a foglalt napokat, amiket a Osszez_foglalas függvénnyel lehet kiíratni. A foglalást egy másik osztály végzi, a Lefoglalt_Datum. Közte és a Datum között öröklés áll fent. A lefoglalt dátumok különböző típusúak lehetnek, a típusok neve az Esemeny osztályban van tárolva, amely kompozicióban áll a Lefoglalt_Datum osztállyal.



3.2Algoritmusok

3.2.1Tesztprogram algoritmusai

A tesztprogram a specifikációnak megfelelően a felhasználó inputja alapján teszteli a program következő funkcióit. Azt követően az adatot

beolvassa és annak a teszteset ciklusába lép be amelynek megfelel az adat.

3.2.2 Hónap lekérdezés

Ha hónap <0 akkor

Kivétel alulindexelés

Ha hónap>=12 akkor

Kivétel túlindexelés

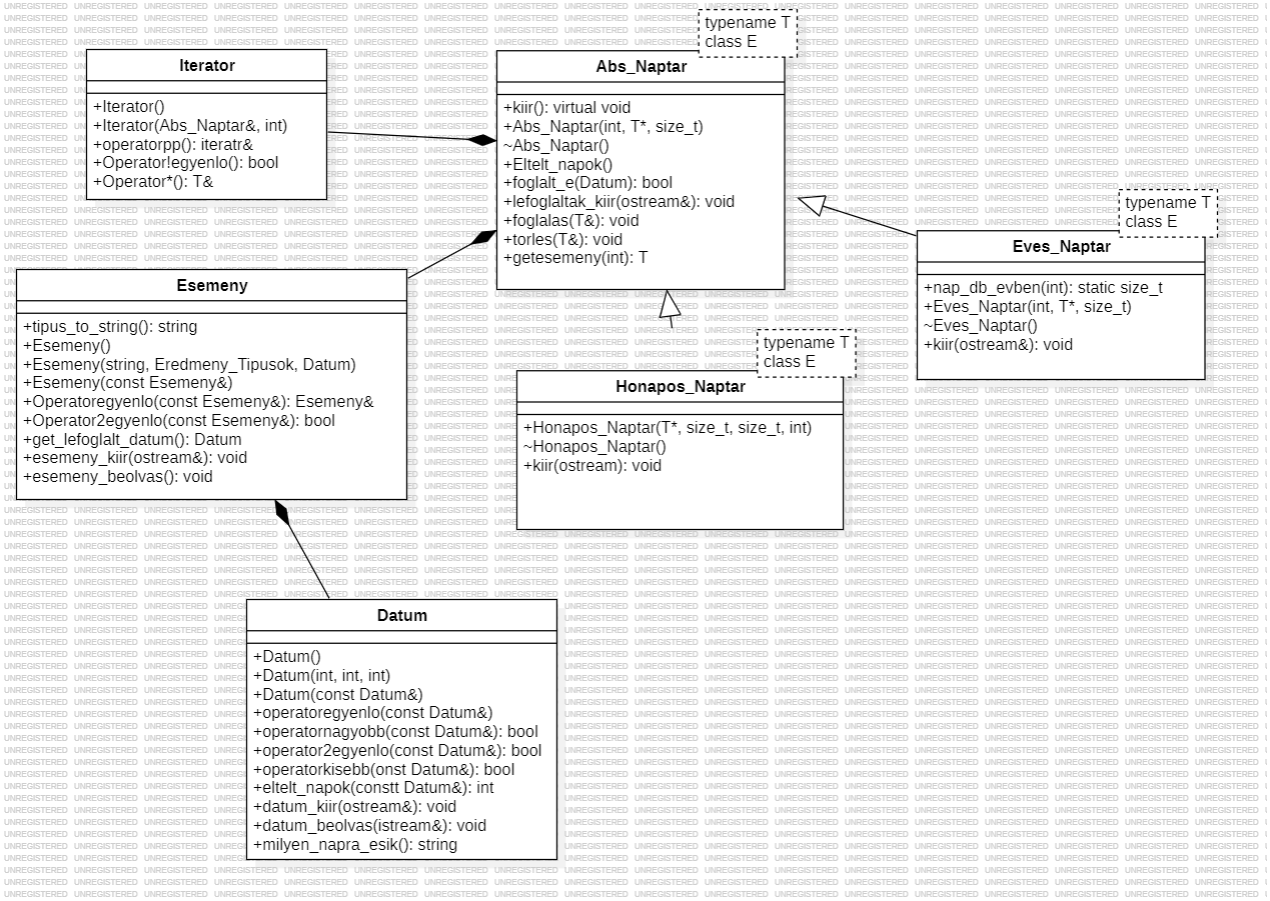
Különben return honapok[honap]

A kivételeket a főprogram kezeli.

3.b Javított terv

3.b.Objektum terv

A projekt alapját az absztrakt `Abs_Naptar` osztály képezi. Ennek tagváltozóit és tagfüggvényeit öröklik a `Honapos_Naptar` és `Eves_Naptar` osztályok. A kiíratás függvénye az alaposztályban virtuális, mivel az örökölt osztályokban eltérő módon kerül megvalósításra.
A T egy típusként (template paraméterként) szerepel, amely osztály típusú (például `Esemeny` vagy `Datum`), és az elemek egy dinamikusan lefoglalt tömbben tárolódnak.
Az E típus a kivételkezeléshez használt osztályt jelöli.
Mivel minden eseménynek van egy dátuma, az `Esemeny` és `Datum` osztályok között kompozíció (összetétel) kapcsolat áll fenn.



4.Megvalósítás

4.1Programzói dokumentáció

A feladat megvalósítása nagy részben megegyezik a tervben bemutatottal. Az osztályok tagfüggvényeinek működése az üzleti modellben tesztelésre kerül. Ennek megvalósítása érdekében a tárgyban megismert "gtest_lite.h" állomány függvényeit használom fel.

Az IO modell megvalósítja a fájlból olvasás valamint fájlba írás műveletek, így az írásra, valamint olvasásra szánt tagfüggvények is bemutatásra kerülnek. A standard bemeneten a felhasználó különböző funkciók közül választhat: eseményt foglalhat, az eddig foglalt eseményeket megtekintheti, eseményt törölhet és naptárakat is megtekinthet, amelyben a lefoglalt napok jelölve vannak, ezen napokon lefoglalt eseményeket is lehetősége van látni.

4.1.1 Az elkészített osztályok bemutatása

A tervtől eltérően a naptár osztályok nem sablonnal működnek, dátumokat magukban nem lehet lefoglalni, mert azzal olyan napok is le lennének foglalva amikor semmi fontos nincs kikötve, hogy történne. A foglalás az Abs_Naptar osztályban történik, és a lefoglalt események növekvő sorrendben vannak tárolva dátum szerint egy dinamikus tömbben. Ehhez a Datum osztáy egyenlőtlenségi operátorai vannak felhasználva. A lefoglalás mellett ugyanebben az osztályban valósul meg egy lefoglalt eseménynek törlése valamint az osztály destruktora által felszabadítva van a lefoglalt memória. A dinamikus tömb bejárásához készült a const_iterator osztály amelyet tartalmaz az Abs_Naptar osztály. Ez fel van használva az események kiiratásánál valamint az ellenőrzésben, hogy egy dátum foglalt-e vagy sem. A void kiir és size_t capacity függvények virtuálisak, más történik a két örökölt osztály megvalósításakor. Az éves naptár kiírásához fel vannak használva az év hónapjainak hónapos naptárként való kirásai.

4.1.2 Elkészített függvények, algoritmusok

A dátum osztályban megvalósított int eltelt_napok függvény két dátum közötti különbséget számolja ki napokban. Ez többféleképpen történhet, ha a két dátum csak a nap terén különbözik egyszerűen a nagyobb dátum napjából ki kell vonni a kisebbet. Ha hónapokban külömböznek, a függvény kivonja a kisebb hónap összes napjaiból a dátum napjait, így megkapva a hónap elkövetkezendő napjait. Ehhez még hozzáadódik a köztes hónapok összes napjainak száma valamint a nagyobb dátumban a hónapban eltelt napok száma. Az évek különbözése esetén is hasonlóan számolható ki a különbség.

Az Abs_Naptar foglalás függvényében a foglalás azon elven valósul meg, hogy az események dátum szerint növekvő sorrendben legyenek eltárolva. Ennek érdekében a függvény addig másolja az új tömbbe az eredeti tömb elemeit amíg nem talál olyan elemet amelyik nagyobb a foglalandó esemény dátumánál. Ebben az esetben megjegyzi ezt a poziciót, a ciklusból kilép és az új esemény bekerül erre a pozícióra. Ezután az új tömbbe mindig az eredeti eggyel kisebb indexű elemek kerülnek be.

A hónapos naptár kiiratásában nagy szerepet játszik a het_kezdo nap függvény ami alapján lehet tudni, hogy milyen a hónap kezdése milyen napra esik, és ez figyelembe van véve a naptár elrendezésében.

A függvények nagy részének működéséről szó esik a függvények felett lévő kommentekben.

4.1.3Felhasznált header fileok, stl tárolók,állományok

A projekt megvalósításához fel voltak használva különböző header fileok. Az <iomanip> a dátumok valamint a naptárak rendezett kiiratásához voltak szükségesek. Ez könnyebben olvashatóvá tette a naptárakat. Stl tárolókból a <string> osztály használata volt megengedett, mást nem használtam. Ebben tároltam a különböző szöveges változókat. Az <fstream> a fájlok kezeléséhez volt szükséges, amelyből van olvasásra, írásra és mindkettőre alkalmas fájl. A <locale> a magyar betűket megfelelően kezelő nyelv beállításához volt szükséges. Az <stdexcept> általak kerülnek a kivételek az outputra. A memóriaszivárgás ellenőrzéséhez a “memtrace.h” állomány van felhasználva, a teszteléshez pedig a “gtest_lite.h”.

4.1.4 Tesztprogram bemutatása

A program a tesztprogramot futtaja ha definiálva van a CPORTA makró. A tesztprogram a dátum osztály tagfüggvényeit teszteli először, majd ezután az őt tartalmazó Esemény osztály funkcióit, ezután az Abs_Naptarban megvalósított függvényeket, végül pedig az ebből az osztályból öröklődő hónapos és éves naptár függvényeit. Különböző eseteket is vizsgál, azt is ellenőrzi, hogy hibás adat esetén érkezett-e kivétel. A dátumok és az események egyenlőségének vizsgálata egyben magába vonja azt is, hogy ezeknek az osztályoknak == operátorai helyesen meg lettek valósítva. Ez igaz a EXPECT_GT, EXPECT_LE... ellenőrzésekre is amiket a dátum osztály egyenlőtlenségi operátorai biztosítanak.

A tesztprogram továbbá ellenőrzi, hogy egy esemény valóban bekerült-e a dinamikus tömbbe foglalás után, valamint, hogy az esemény már nem található meg a tömbben törlés esetén.

4.1.5Fájlkezelés, IO modell

A Beolvaso_Teszt.Txt–ben szerepel egy dátum, egy hónap , egy év, egy n ami azt mutatja hány eseményt szeretnénk lefoglalni és ezután n esemény. Ezek az adatok be vannak olvasva a fájlból majd a Kiiro_Teszt.txt-be beírjuk a kapott dátumot és a lefoglalt eseményeket. Az inputban szereplő hónapnak és évnek megfelelő Naptárakat kiiratjuk, feltüntetve rajtuk a lefoglalt események helyét. Ezzel egyben megmutatjuk a fájlkezelés, valamint az osztályok beolvasó és kiíró tagfüggvényeinek működését.

A foglalt_esemenyek.txt biztosítja, hogy a program lefoglalt eseményei ne vesszenek el. Ezt a fájl olvasásra és írásra is használandó, ezért fstream-ként van deklarálva. Ha a lefoglalt események kiírására szükség van a naptár innen olvassa be az eseményeket, majd ide írja az újonnan foglalt. Törlés esetén töröljük az eseményt a dinamikus tömbből, majd annak tartalmát beirjuk a fájlba, így a fájlban se lesz már jelen a törölt esemény.

A program exceptionbol származó saját Gond_Van kivételosztállyal kezeli a nem szabványos hibákat, ezekhez üzeneteket is társít amit hiba elkapás esetén a kimentre kiír. Az , hogy a Gond_Van kivételeket előbb keresi a program mint más exceptionoket biztosítja, hogy a saját kivételeket ne az általános exception kapja el. Vannak a saját kivételosztályon kívül is kivételek, ezeket elkapja az általános exception, ha pedig egyikre se igaz az azt jelenti, hogy ismertelen a hiba ezért “Nagy baj van.” íródik ki a kimenetre.

4.2 Felhasználói dokumentáció

A program felhasználójának futtatás esetén tud választani, hogy milyen funkcióját a programnak szeretné látni. Ezen funkciók eléréséhez a neki megfelelő kifejezés zárójelében szereplő nagy betűjét kell beírja (pl. törlés-T, esemény foglalás- F, kilépés-K). Hibás betű beírása esetén újrapróbálkozhat a felhasználó. Ha foglalást vagy törlést választ a felhasználó be kell írnia az illető eseményt. Ez először az esemény nevének beírásával történik, majd egy tipust kell beírni. Ez lehet a következők közül: iskolai, globális, magánéleti, munkahelyi vagy egyéb. Ha nem ezek közöl választ a felhasználó a típus szintén egyéb lesz. Utolsónak az esemény dátumát szükséges begépelni. Ez is három lépésben történik. Először az év , majd a hónap és végül az év kerül beolvasásra. Az újévet nem lehet törölni

az események közül, mivel ez minden naptárban le van foglalva első hívás után.

Naptár kiiratás esetén azt kell eldöntenie a felhasználónak, hogy hónapos vagy éves naptárt szeretne kiiratni. A hónapos naptár egy hónapot tartalmaz, az éves pedig értelemszerűen egy évet. Éves naptár esetén a felhasználónak be kell írnia a kívánt évet, hónapos esetén pedig először a hónaphoz tartozó évet, majd magát a hónapot. A lefoglalt események *nap* formában jelennek meg a naptárban. Lehetőség van ezután egy ilyen dátumhoz tartozó eseményt megtekinteni. Ha olyan dátumot ír be a felhasználó ami nincs lefoglalva kivétel keletkezik.

Ha a felhasználó az összes lefoglalt esemény kiírását választja, megjelenik a lefoglalt események száma, ezután pedig a lefoglalt dátumok időrendi sorrendben. Ha a felhasználó ki szeretne lépni, egy funkció beteljesülése után R betűvel végezheti ezt el.