A program célja:

A programban előre rögzített repülőgép járatokra lehet jegyet foglalni, foglalást törölni, menüt választani, valamint lehet járatokat keresni és meg tudja jeleníteni a foglaltsági térképet is. A járatokról pedig egy összesítést is meg tud jeleníteni.

A program felépítése:

A program az indulást követően kiolvassa az eddigi foglalásokat (ha léteznek) a foglalasok.txt fájlból.
Később ugyanebben a fájlban fogja tárolni a program futása során bekövetkezett változásokat (új/törölt foglalás). A kiolvasott adatokat a program eltárolja.

A foglalható járatok a jaratok.txt -ben vannak rögzítve a következőképpen:
járatazonosító mikor(ÉÉÉÉ/HH/NN), honnan, hová, férőhely.
Ezeket is szintén beolvassa a program induláskor és ideiglenesen eltárolja az adatokat, majd a végén frissíti a fájlt. A járatok adatainak ilyen módon történő tárolására a férőhelyek változása miatt van szükség.

A program egy konzolos, menüvezérelt program.
A program a külső adatok beolvasása után kilistázza a menüpontokat, majd a felhasználótól bekéri az általa választott menüpont sorszámát. A sorszám egy egész szám, a program képes felismerni ha a felhasználó hibás adatot vagy nem létező sorszámot ad meg. A menü programrész egy ciklusban van, ami addig ismétlődik, ameddig a kilépő sorszámot nem adja meg a felhasználó. Amennyiben a felhasználó a kilépést választja, a ciklus nem fut le többször és a program folytatódik tovább.

Main függvény:

A main függvényben történik a programkét legfontosabb adattároló elemének, a járatok és a foglalások listájának a megalkotása. Ezen kívül a main függvény meghívja a beolvasó függvényeket, a menüt, a menüből való kilépés után pedig rendezi a foglalásokat, frissíti a txt-ben tárolt adatokat és felszabadítja a felhasznált memóriaterületeket.

Menu.c:

A menu.c egyetlen függvényt tartalmaz: a menu() függvényt, amelynek feladata a menüpontok megjelenítése, a menüpont bekérése a felhasználótól és a megfelelő függvények meghívása a feladat elvégzéséhez. Lényegében ez a függvény biztosítja a kapcsolatot a felhasználó és az egyes programrészek között.

Elérhető menüpontok:

1.: járat keresése: a menüpont választása után a program bekér két várost a felhasználótól(honnan-hová, a városok neveit ékezet nélkül kell megadni), majd egy időintervallumot(ÉÉÉÉ/HH/NN formában) és meghívja a jaratKeres() függvényt.

2.: repülőjegy foglalása: ebben a menüpontban a program egy járatszámot(egy adott járat számát a járat keresése funkciót használva lehet megtudni).
A program megpróbálja megkeresni a járatot a járatszám alapján. Ha a járat szerepel a rögzített adatok között és van férőhely, akkor bekéri a felhasználó nevét, majd megjeleníti a foglaltsági térképet. (Ellenkező esetben a program közli a felhasználóval a hibát: „nem létező járat”, illetve „nincs több férőhely”.)

A foglaltsági térkép alapján a felhasználó el tudja dönteni, hogy melyik szabad helyet szeretné választani. Ezt a program szintén bekéri a felhasználótól. Ha olyan helyre szeretne foglalni, amelyre már létezik foglalás, akkor hibaüzenetet kap, ha a hely még szabad, akkor a foglalás tovább folytatódik.

A következő adat, amit a felhasználónak meg kell adnia az a járaton fogyasztani kívánt étel. Három menü közül lehet választani: normál, vega, laktózmentes. Ezeket a program a megszokott módon kilistázza és a bekért adat egy sorszám.

3.: foglalás törlése: ebben a menüpontban a program bekéri a felhasználó nevét, majd meghívja a jaratTorol() függvényt.

4.: összesítés: meghívja az Osszesit() függvényt.

5.: kilépés: ez a menüpont csak megszakítja a ciklust, a ciklusfeltétel miatt a menü már nem fut le többször.

A program a kilépés választása után még elvégzi a foglalások és a járatok adatainak mentését.
Ehhez megnyitja a már említett „foglalasok.txt” és „jaratok.txt” fájlokat és beleírja az eltárolt adatokat, követve a formai követelményeket.

Fajlkezeles.c:

A fajlkezeles.c 5 függvényt tartalmaz, amelyek feladata a fájlokból való beolvasás és az adatok rögzítése, illetve a menüből való kilépés után az adatok fájlba rögzítése.

Függvények:

beolvas(): 2 bemenet: a stream amelyből olvas(FILE*) és az elválasztó karakter, amelynél megáll az olvasás(char)

A függvény char* típusú, tehát a beolvasott karaktertömbbel tér vissza.

A függvény a megadott streamről egyenként olvas be karaktereket és dinamikusan tárolja azokat, így tetszőleges hosszúságú stringeket is be lehet olvasni akár fájlból akár a szabványos bemenetről.
A függvény ugyan a fájlkezelésnél szerepel, azonban bemenete akár lehet az stdin is, így a programban nem csak fájlokból beolvasásra van használva.

jaratokBeolvas(): 2 bemenet: járatok listája(pointer), járatok listájának mérete(pointer)

A függvény Jarat* típusú, tehát a beolvasott adatokkal teli jaratok tömbbel tér vissza.

A függvény a szövegfájlban a forma szerint tárolt adatokat olvassa be és tárolja el a dinamikusan foglalt jaratok Jarat struktúrák tömbjében.

jaratRogzit(): 2 bemenet: járatok listája(pointer), járatok listájának mérete(pointer)

A függvény void típusú, tehát nincs visszatérési értéke.

A függvény a járatok listájában tárolt adatokat rögzíti a szöveges fájlban a formázási minta szerint.

foglalasokBeolvas(): 4 bemenet: járatok listája(pointer), járatok listájának mérete(int), foglalások listája(pointer), foglalások listájának mérete(pointer)

A függvény Foglalas* típusú, tehát a beolvasott adatokkal teli foglalasok tömbbel tér vissza.

A függvény a szövegfájlban a forma szerint tárolt adatokat olvassa be és tárolja el a dinamikusan foglalt foglalasok Foglalas struktúrák tömbjében a jaratFoglal() függvény segítségével.

foglalasokRogzit(): 2 bemenet: foglalások listája(pointer), foglalások listájának mérete(int)

A függvény a foglalások listájában tárolt adatokat rögzíti a szöveges fájlban a formázási minta szerint.

jaratkezeles.h:

A jaratkezeles. Header fájlban a függvények definícióján kívül két fontos struktúra és egy enum is definiálva van.

Jarat: 6 elem: járatszám(string), Datum struktúra, induló város(string), megérkezés helye(string), férőhely(int), foglalt ülések(számok(int) tömbje)

Foglalas: 4 elem: járatszám(string), név(string), ülőhely(string), menü(int)

enum Menu: normal, vega, laktozmentes

jaratkezeles.c:

A jaratkezeles.c 6 függvényt tartalmaz, melyek feladatai a járatok és foglalások adatainak kezelése, foglaltsági térkép és összegzés készítése.

jaratKeres(): 6 bemenet: járatok listája(pointer), járatlista mérete, indulási hely(string), megérkezés helye(string), kezdő időpont(Datum struktúra), végső időpont(Datum struktúra)
A függvény void típusú, nincs visszatérési értéke.

A függvény kilistázza azokat a járatokat, amelyek megfelelnek az induló városnak, az úticélnak és indulásuk a megadott időintervallumba esik.

ulohelySzam(): 1 bemenet: a foglaláshoz tartozó ülőhely(char*)

A függvény visszatérési értéke int.

A függvény feladata, hogy a választott ülőhelyet számmá alakítsa, hogy a foglaltsági térkép elkészítésekor könnyebb legyen azt felhasználni.

jaratFoglal(): 5 bemenet: járatok listája(pointer), foglalások listája(pointer), a foglalás(Foglalas struktúra), járat indexe a járatlistában(int), foglalások listájának mérete(pointer)
A függvény a foglalások listájával tér vissza, az új foglalással kibővítve.

A függvény kibővíti a foglalások listáját egyel(azért kapja meg a méretet pointerként, hogy ezen egyúttal módosítani is tudjon), majd belehelyezi a foglalást. A függvény ezen kívül még a járathoz tartozó férőhelyeket is csökkenti, illetve rögzíti a foglalt ülést a járathoz tartozó foglaltUlesek tömbben.

jaratTorol(): 3 bemenet: a foglalások listája(pointer), a foglalások listájának mérete(pointer), a felhasználó neve(string)
A függvény a foglalások listájával tér vissza, a kitörölt foglalás nélkül.

A függvény a felhasználó neve alapján megkeresi a foglalását a foglalasok[] tömbben és kitörli azt. Amennyiben nem létezik a megadott néven foglalás, akkor hibaüzenetet ír ki.

foglaltsagiTerkep(): 3 bemenet: járatok listája(pointer), járatok listájának mérete(int) járatszám(char*)
A függvény void típusú, nincs visszatérési értéke.

Ez a függvény a járatszám alapján megjeleníti a foglaltsági térképet és más karakterrel jelöli a foglalt, illetve a szabad üléseket.

Osszesit(): 4 bemenet: járatlista(pointer), foglalások listája(pointer), járatlista mérete(int), foglalások listájának mérete(int)
A függvény void típusú, nincs visszatérési értéke.

Ez a függvény elvégzi az összesítést, kilistázza a járatokat járatszám alapján és megszámolja, majd megjeleníti, hogy melyik menüből hányat kell felvinni a fedélzetre.

datumkezeles.h:

A datumkezeles.h a függvények definícióján kívül egy struktúrát is tartalmaz.

Datum: a megadott időpontok az egész számokra való feldarabolás után ilyen struktúrában tárolódnak el.
3 elem: év(int), hónap(int), nap(int)

datumkezeles.c:

A datumkezeles.c két függvényt tartalmaz, melyek céljai a felhasználó által megadott dátumok összehasonlítása, illetve a dátum beolvasása.

datumBeolvas(): nincs bemenet
A függvény egy Datum struktúrával tér vissza.

A függvény stringként beolvas egy dátumot a standard inputról, majd átalakítja az időpontot olyan formába, hogy az év, a hónap és a nap három külön egész számként legyen felhasználható, majd bele helyezi ezeket egy Datum struktúrába.

datumOsszehasonlit(): 2 bemenet: datum1(Datum struktúra), datum2(Datum struktúra)
A függvény bool típusú, igaz/hamis értékkel tér vissza.

Ez a függvény összehasonlítja a két megadott időpontot és igaz értékkel tér vissza, ha datum1 későbbi időpont, mint datum2. Ellenkező esetben, amikor datum1 korábbi időpont, mint datum2 akkor pedig hamis értékkel tér vissza.

validacio.c:

A validacio.c függvényeinek feladata az inputok validációja, legyen az egy megadott dátum vagy string.

datumValidacio(): 1 bemenet: datum(Datum struktúra)
A függvény bool típusú, igaz/hamis értékkel tér vissza.

Ez a függvény egy előre meghatározott időpont előtti időpontokat hibának érzékel, ezzel feltárhatóak azok az esetek, amikor a felhasználó a múltba szeretne jegyet foglalni. Igaz értékkel tér vissza ha hibát talált, hamissal ha nem.

inputValidacio(): 1 bemenet: input(string)

A függvény visszatérési értéke string.

A függvény képes kiszűrni a helytelen(üres) inputokat, majd addig olvas be adatot a felhasználótól, amíg az nem megfelelő, végül visszatér a helyes inputtal.

ulohelyValidacio(): 3 bemenet: ülőhely száma(int), foglalt ülések tömbje(int[]), tömb mérete(int)

A függvény bool típusú, igaz/hamis értékkel tér vissza.

A függvény képes eldönteni, hogy a megadott ülés foglalt-e már az adott járaton vagy sem. Igaz értékkel tér vissza, ha már foglalt és hamissal ha még nem.

rendezes.c:

rendezes(): 4 bemenet: járatok tömbje(pointer), járatok tömb mérete(int), foglalások tömbje(foglalas), foglalások tömb mérete(int)

A függvény visszatérési értéke a rendezett foglalások tömb.

A rendezes.c egyetlen függvénye, célja a foglalások járatszám szerinti rendezése. Abban a sorrendben fognak szerepelni a foglalások, amilyen sorrendben a jaratok.txt-ben a járatok vannak megadva.

Formai követelmények:

foglalasok.txt: minden foglalás külön sorban, minden adat szóközzel elválasztva ebben a sorrendben: járatszám, név, ülőhely, választott étel.

jaratok.txt: minden járat külön sorban, minden adat szóközzel elválasztva ebben a sorrendben: járatszám, honnan, hová, indulás időpontja(ÉÉ/HH/NN formátumban), max férőhely, szabad férőhely.

Adatszerkezetek:

A program a már említett két dinamikusan foglalt tömbbel dolgozik, a járatok és a foglalások tömbjével, valamint a járatokon belül létezik még egy dinamikusan foglalt foglaltUlesek lista is. Azért választottam a dinamikusan foglalt struktúrák tömbjét, mert nem csak a foglalásuk egyszerű, hanem egy konkrét adat törlése is. A malloc, illetve realloc rengeteg segítséget nyújt. Megfontolandó lenne még esetleg a verem használata is, azonban annak kialakítása ebben az esetben nehézkes lenne, illetve a program nem dolgozik olyan sok adattal, hogy ez indokolttá tegye annak használatát.