Varjú Ákos
SSNTUX
2021.11.28.

Autószerviz

Varjú Ákos

Felhasználói dokumentáció

A program célja egy autószerviz adatainak az átlátható tárolása, módosítása, mentése.

Működés:
A program megnyitásakor egy szöveges fájlból ”adatok.txt” beolvassa az adatokat és a programon belül lehet őket módosítani, hozzáadni, törölni. Ha ez a txt fájl nem létezik, létre kell hozni és bele kell írni egy 0-t. Bezáráskor (6) a módosított adatokat visszatölti a fájlba.

Főmenü:

(1)Modositas
(2)Kereses
(3)Javítások
(4)Szerviztörténet
(5)Listázás
(6)Mentés és kilépés

Almenü:**(1)**Modositas

(1)Auto hozzáadás:
-Tulajdonost:
-Rendszamot:
-Típus:
-Forgalmi:EEEE:HH:NN formátumban kell megadni pl. 2023:01:23
-Javitasok:

Ezen adatok megadásakor a program vizsgálja a tulajdonost és a rendszámot, nem engedi meg az egyezést. Ha ilyen történik hibajelzést ad a program.A másik 2 adat megadása kötetlenül be lehet gépelni.

(2) Ügyfél hozzad:
-Név:
-Telefonszam
-Email

Ezen adatok megadásakor a program vizsgálja az ügyfél nevét, nem engedi az egyezést. Ha ilyen történik hibajelzést ad a program. A másik 2 adat megadása kötetlenül be lehet gépelni.

(3) Autó törlése:

Varjú Ákos
SSNTUX
2021.11.28.

-Rendszám: megadása után a program rákérdez hogy biztos törölje-e az autót.

(4)Vissza a főmenübe: visszavisz a főmenübe.

Almenü:**(2)**Keresés

(1)Autó keres

-Rendszám megadása után az autó adatai megjelennek

(2)Ügyfél keres

-Ügyfél neve megadása után megjelennek az adatai az autóinak a száma és a hozzá tartozó autok rendszámai

(3) Vissza a főmenübe

Almenü:**(3)**Javítás

(1)Olajcsere

-Rendszám megadása után, rögzíti az elvégzett javítást a mai dátummal és annak árát:25000Ft

(2) Müszaki vizsgaztatas

-Rendszám megadása után, rögzíti az elvégzett javítást a mai dátummal és annak árát:30000Ft és módosítja az adott autó forgalmijának az érvényességet a mai naphoz képest pontosan két évvel. pl 2019:03:12-->2023:11:28 mai dátum 2 év.

(3)Gumicsere

--Rendszám megadása után, rögzíti az elvégzett javítást a mai dátummal és annak árát:10000Ft

(4) Egyéb javítás

-Rendszám megadása után, rögzíti az elvégzett javítást amit a felhasználó ad meg. Ezt pontosan dátumozza a program és beolvassa a javítás árát amit szintén a felhasználó ad meg.

(5) Vissza a főmenübe

Almenü:**(4)**Szervíztörténet

(1) Szerviztörténet:

-Rendszám megadása után, megjeleníti az adott autó szerviztörténetét és a befizetendő összeget.

(2)Befizetés

-Rendszám megadása után, megjelenik a befizetendő összeg, majd a felhasználó által megadott összeget levonja a tartozásból.

(3) Vissza a főmenübe

Almenü:**(5)**Listázás

(1) Listázás: kiírja a rendszerben lévő összes autót a forgalmi vizsga érvényessége alapján

(2) Lejártak megjelenítése: kiírja a rendszerben lévő összes lejárt forgalmival rendelkező autót

Varjú Ákos
SSNTUX
2021.11.28.

(6) Mentés és kilépés

A program minden adatot elment az adatok.txt nevű fájlba.

Varjú Ákos
SSNTUX
2021.11.28.

Autószerviz

Programozói dokumentáció

A program célja egy autószerviz adatainak az átlátható tárolása, módosítása, mentése. Ezeket a program struktúrákban és fésűs listákban tartalmazza.

Struktúrák:

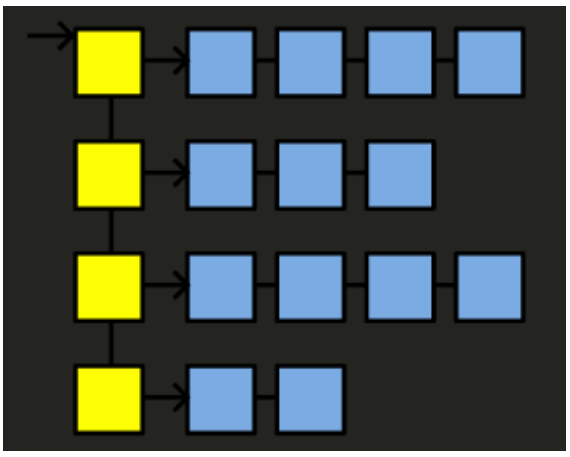
```
typedef struct Autok{
    char* tulaj;
    char *rendszam;
    char *tipus;
    char *vizsga;
    char *javitasok;
    int datum;
    int ar;
    struct Autok *kov;
}Autok;
typedef struct Ugyfelek{
    char *nev;
    char *email;
    char *telefonszam;
    int dbauto;
    struct Ugyfelek *kov;
    struct Autok *elsoauto;
```

AUTÓK struct:
-A tulaj sztringre mutató pointer
-A rendszám sztringre mutató pointer
-A típus sztringre mutató pointer
-A vizsga sztringre mutató pointer
-A javítások sztringre mutató pointer
-Dátum intiger formában
-Ár intiger formában
-Kov: auto structra mutató ptr

Ügyfelek struct:
-A ügyfél neve sztringre mutató ptr
-A email sztringre mutató ptr
-A telefonszam sztringre mutató ptr
-int: hány autója van az ügyfélnek
-Kov: ugyfel structra mutató ptr
-Elsoauto: elso Auto structra mutato pointer

Adatszerkezet: Listák listája

 Az ügyfelek  az autók struktúrája.



A fésűs lista egy olyan láncolt lista, amelynek elemei láncolt listák. Az adatok kétszintű hierarchiában helyezkednek el.

Varjú Ákos
SSNTUX
2021.11.28.

A program indulásakor: az létrehoz egy strázsa nevű üres Ügyfél Struktúrát aminek annyi lesz a feladatot hogy azok után fűz a program minden adatot.

Fájlkezelés fv-ei:

Ha az adatok. txt fájl nem létezik, létre kell hozni és bele kell írni egy 0-t különben a program nem működi rendesen..

Az adatokat “adatok.txt” nevű fájlból olvassa a köv fv:

```
int fesufeltolt(Ugyfelek *strazsa)
```

Feladata a programot feltölteni a fájlból érkező adatokkal a paraméterként megadott strazsa struktúrához láncolva. Az adatokat a fajlbololvas fv dolgozza fel.

```
char *fajlbololvas(FILE *fp)
```

Feladata: fájlból ‘\n’ -ig olvasás

Visszatérési értéke a sztring címe, ezen cím van hozzáadva a struktúrák adataihoz pl. rendszám

```
int ugyfelszamol(Ugyfelek* strazsa)
```

Paraméter a strazsa, visszatérési érték: hány db ügyfél van

```
int fesulement(Ugyfelek *strazsa, int dbugyfel)
```

Feladata: beleírni az összes adatot a fájlba.

Paramétere a strazsa és a dbügyfel, ami az ügyfelek számát jelenti.

```
void felszabadit(Ugyfelek *strazsa)
```

Feladata: körbejárni a fésűs listat és felszabadítani az összes struktúrát.

```
void autofelszabadit (Autok *ptr)
```

Feladata: felszabadítani a paraméterként kapott autó struktúra adatait.

```
void ugyfelfelszabadit(Ugyfelek *ptr)
```

Feladata: felszabadítani a paraméterként kapott ügyfél struktúra adatait.

modosit.c fv-ei

```
void modosit_menu_kiir()
```

Feladata: kiírja az almenüt

```
char *olvas()
```

Feladata a konzolba írt sort dinamikusan lefoglalni és eltárolni. Visszatérési értéke a sztring címe.

```
Ugyfelek * tulaj_keres(char *tulajneve,Ugyfelek *strazsa)
```

Paraméter: strázsa és egy sztring.

A sztring alapján vizsgálja az Ügyfelek listáját, ha talál egyezést visszatérési értéke a talált struktúra címe, ha nincs egyezés NULL ptr-rt ad vissza.

```
Autok * auto_keres(char *rendszam,Ugyfelek *strazsa)
```

Paraméter: strázsa és egy sztring.

A sztring alapján vizsgálja az Autók listáját, ha talál egyezést visszatérési értéke a talált struktúra címe, ha nincs egyezés NULL ptr-rt ad vissza.

```
Autok * auto_beolvas(char *rendszam,char *tulaj)
```

Varjú Ákos
SSNTUX
2021.11.28.
Paraméter: sztring ami a rendszam, és még egy sztring ami a tulaj.
Feladata: létrehozni egy Autó struktúrát és annak értéket adni. A két stringre mutató pointert azért kéri be h egyszerűbb legyen a felhasználónak.
Visszatérési értéke: az fv- által létrehozott struktúra címe.
`void auto_befuz(Autok *ezt,Ugyfelek *tulaj)`
Paraméter: Ügyfél struktúra mutato pointer és egy Autó struktúrára mutató pointer.
Feladata: Hozzáfűzni az Autót a Ügyfél listájához.
`Ugyfelek *ugyfel_beolvas(char *ugyfelneve)`
Paraméter: sztring , ami az ügyfél nevét tartalmazza
Feladata: létrehozni az egy Ügyfél struktúrát és annak értéket adni.
Visszatérési értéke: az fv- által létrehozott struktúra címe.
`void ugyfel_beszur(Ugyfelek *befuzendo,Ugyfelek *strazsa)`
Paraméter: Ügyfél struktúra mutato pointer és a strázsára mutató pointer.
Feladata: hozzáfűzni az Ügyfelet az Ügyfelek listájához
`Autok * torlendo_elotti(Ugyfelek *strazsa, Autok *torlendo)`
Parameter: strázsa és egy Autóra struktúrára mutató pointer.
Feladata: megkeresni azt a struktúrát aminek a ->kov vagy ->elsoauto megegyezik a paraméterként kapott címmel. Ha van találat annak a címét adja vissza.
`void auto_hozzaad(Ugyfelek *strazsa)`
Feladata, autót hozzáadni, beolvasni az adatokat, vizsgálni hogy létezik e már az adott autó.
Használja a `tulaj_keres,auto_keres, auto_befuz ,auto_beolvas-t,`
`void ugyfel_hozzaad(Ugyfelek *strazsa)`
Feladata, ügyfelet hozzáadni, beolvasni az adatokat, vizsgálni hogy létezik e már az adott ügyfél.
Használja a `tulaj_keres,ugyfel_beszur,ugyfel_beolvas.`
`void auto_torol(Ugyfelek *strazsa)`
Feladata: autót törölni, felszabadítani az összes adatát.
Használja a `auto_keres,tulaj_keres,torlendo_elotti,autofelszabadit-t.`
`void modosit_menu(Ugyfelek *strazsa)`

Feladata: a utóbbi 3 fv-nt switch case szerkezetben menü segítségével működtetni

kereses.c fv.ei

`void autokiir(Autok *ezt)`
Feladata: paraméterként kapott struktúra adatainak kiírása
`void ugyfel_kiir(Ugyfelek *ezt)`
Feladata: paraméterként kapott struktúra adatainak kiírása
`void keres_auto_kiir(Ugyfelek *strazsa)`
Feladata: Bekérni az autó rendszámát az alapján keres, és kiírja az adatai.
`void keres_ugyfel_kiir(Ugyfelek *strazsa)`
Feladata: Bekérni az Ügyfél nevét az alapján keres, és kiírja az adatai.
`void keresesmenukiir()`

Varjú Ákos
SSNTUX
2021.11.28.
Feladata kiírja az menüt.
```
void keres(Ugyfelek *strazsa)
```
Feladata: a fenti fv-ekkel switch case szerkezetű menüt működtet

javitasok_szerviz.c fv.ei

```
char *hozzafuz(char *ehhez,char *ezt)
```
Paraméter: két sztring.
Feladata egymashoz fűzni őket és visszaadni az *ehhez módosított címet
```
void menukiir()
```
Feladata: Kiírja a menüt.
```
char *pontosido()
```
Feladata: lekérdezi a rendszer pillanatnyi idejét, majd azt a helyes formára modosítja és visszaadja annak a címet.

A következő fv-ek lényegében ugyan azok.
```
void olaj_csere(Ugyfelek *strazsa)
void muszaki_vizsgaztatas(Ugyfelek *strazsa)
void gumicsere(Ugyfelek *strazsa)
void egyebjavitas(Ugyfelek *strazsa)
```
Mind módosításokat hajtanak végre az adott autón amit elment a pontos idővel és rögzíti a módosítás árát.

```
void menu3(Ugyfelek *strazsa)
```
Az előző fv-ekből csinál switch case szerkezetű menüt.
```
void szerviztortenet(Ugyfelek *strazsa)
```
Feladata: Bekér egy rendszámot a felhasználótól, majd az alapján keresi az autó struktúra címet, majd minden múltbeli javítást kiír és annak költséget.
```
void befizet(Ugyfelek *null)
```
Feladata:Bekér egy rendszámot a felhasználótól, majd az alapján keresi az autó struktúra címet. Megjeleníti a tartozást majd felhasználótól beolvassa a fizetendő összeget,mellyel módosítja a tartozást.

listazas.c fv.ei

```
char* muszakivalt (char*datum)
```
Feladata: módosítani az autó forgalmijának az érvényességet. A pontos dátumhoz +2 év.
```
int dbauto(Ugyfelek *null)
```
Paraméter:Ügyfél strázsa
Feladata: Megszámolni a rendszerben lévő autók számát.
```
void listaz(Ugyfelek* null,int dbauto)
```
Paraméter:Ügyfél strázsa
Feladata: Létrehozni egy dbauto méretű tömböt, majd beleteszi az .datum adatait. Majd ezt rendezi, így rendezett sorrendbe kiírja azokat.
```
void lejart(Ugyfelek *null)
```
Paraméter: Ügyfél strázsa

Varjú Ákos
SSNTUX
2021.11.28.
Feladata: Kiírni az összes lejárt forgalmival rendelkező autót.

```
char *atvalt(char *datum)
```

Feladata: a paraméterként kapott sztringet alakítja át kellő formátumra.

```
int datumszam(char *datum)
```

Feladata: a paraméterként kapott dátumot alakítja integerré.

```
void listazas (Ugyfelek *null)
```

Feladata: switch case szerkezetbe rakni a *lejart*, és a *listaz* fv-t.

A helyes működés érdekében, mindig adjon meg olyan értéket, amit a konzol mondd. A forgalmit ,pedig a helyes formátumban ÉÉÉÉ:HH:NN