Grob_Laszlo_NHF

1.0

Készítette Doxygen 1.12.0

1. fejezet

Adatszerkezet-mutató

1.1. Adatszerkezetek

Az összes adatszerkezet listája rövid leírásokkal:

2. fejezet

Fájlmutató

2.1. Fájllista

Az összes dokumentált fájl listája rövid leírásokkal:

3. fejezet

Adatszerkezetek dokumentációja

3.1. Egyedi_osszetevok struktúrareferencia

Egyedi összetevőket tartalmazó lista minden összetevő egyszer kell hogy szerepeljen benne, tartalmazza az összetevők számát és az összetevők tömbjére mutató pointert. A benne levő összetevők rendszerint nem tartalmaznak mennyiségeket csak a nevet és típust.

Adatmezők

- Osszetevo ∗ **egyedi_osszetevok**
- int **egyedi_osszetevok_szama**

3.1.1. Részletes leírás

Egyedi összetevőket tartalmazó lista minden összetevő egyszer kell hogy szerepeljen benne, tartalmazza az összetevők számát és az összetevők tömbjére mutató pointert. A benne levő összetevők rendszerint nem tartalmaznak mennyiségeket csak a nevet és típust.

Ez a dokumentáció a struktúráról a következő fájlok alapján készült:

- file_utils.c
- file_utils.h

3.2. Etel struktúrareferencia

Étel struktúra tartalmazza az étel nevét(max 50 karakter) az összetevőinek számát, az összetevők tömbjének pointerét, és az elkészítési útmutatót(max 1000 karakter)

Adatmezők

- char **nev** [51]
- int **osszetevok_szama**
- Osszetevo ∗ **osszetevok**
- char **elkeszites** [1001]

3.2.1. Részletes leírás

Étel struktúra tartalmazza az étel nevét(max 50 karakter) az összetevőinek számát, az összetevők tömbjének pointerét, és az elkészítési útmutatót(max 1000 karakter)

Ez a dokumentáció a struktúráról a következő fájlok alapján készült:

- file_utils.c
- file_utils.h

3.3. Osszetevo struktúrareferencia

Összetevők struckt tartalmazza az összetevő nevét (max 50 karakter), típusát(max 50 karakter), és mennyiségét(double)

Adatmezők

- char **nev** [51]
- char **tipus** [51]
- double **mennyiseg**

3.3.1. Részletes leírás

Összetevők struckt tartalmazza az összetevő nevét (max 50 karakter), típusát(max 50 karakter), és mennyiségét(double)

Ez a dokumentáció a struktúráról a következő fájlok alapján készült:

- file_utils.c
- file_utils.h

3.4. Receptkonyv struktúrareferencia

Receptkönyv struktúra tartalmazza a benne levő ételek számát és az ételek tömbjére egy mutatót.

Adatmezők

- int **etelek_szama**
- Etel ∗ **etelek**

3.4.1. Részletes leírás

Receptkönyv struktúra tartalmazza a benne levő ételek számát és az ételek tömbjére egy mutatót.

Ez a dokumentáció a struktúráról a következő fájlok alapján készült:

- file_utils.c
- file_utils.h

4. fejezet

Fájlok dokumentációja

4.1. file_utils.c fájlreferencia

Ebben a modulban kaptak helyet a filokból beolvasó illetve azokba kiíró fv-k, illetve az általánosan használt stdin/stdout olvasó író fv-ek.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "debugmalloc.h"
```

Adatszerkezetek

- struct Osszetevo

  Összetevők struckt tartalmazza az összetevő nevét (max 50 karakter), típusát(max 50 karakter), és mennyiségét(double)
- struct Etel

  Étel struktúra tartalmazza az étel nevét(max 50 karakter) az összetevőinek számát, az összetevők tömbjének pointerét, és az elkészítési útmutatót(max 1000 karakter)
- struct Receptkonyv

  Receptkönyv struktúra tartalmazza a benne levő ételek számát és az ételek tömbjére egy mutatót.
- struct Egyedi_osszetevok

  Egyedi összetevőket tartalmazó lista minden összetevő egyszer kell hogy szerepeljen benne, tartalmazza az összetevők számát és az összetevők tömbjére mutató pointert. A benne levő összetevők rendszerint nem tartalmaznak mennyiségeket csak a nevet és típust.

Típusdefiníciók

- typedef struct Osszetevo **Osszetevo**
- typedef struct Etel **Etel**
- typedef struct Receptkonyv **Receptkonyv**
- typedef struct Egyedi_osszetevok **Egyedi_osszetevok**

Függvények

- Receptkonyv ∗ receptek_beolvas (void)

  beolvassa a recepteket az előre megadott receptek.txt fileból az r receptkönyv struktúrába (lehetett volna argumentum a filenév de felesleges), a működése egyszerű csak felbloatolja egy rakás hibakezelés.
- void receptet_fileba_ment (Receptkonyv ∗r)

  Elmenti a kapott r struktúrát az előre megadott formátumban a receptek.txt fileba, ha nem létezik létrehozza a program gyökérkönyvtárába.
- void receptkonyv_felszabadit (Receptkonyv ∗r)

  felszabadítja a kapott r struktúrát és minden alstruktúráját
- int recept_letezik (Receptkonyv ∗r, const char ∗etel_neve)

  Kap egy receptek structot és egy stringet és megnézi hogy a string egyezik e valamely r.etelek.nev stringgel elvileg case független, de mivel utf karakterek ezért néha bugos, az esetek 99ában működik.
- void recept_kiir (Etel ∗m)

  kiirja az adott etel struktura adatait(név összetevők elkészítés)
- Egyedi_osszetevok ∗ osszetevo_beolvas (void)

  Beolvassa az összetevőket az előre megadott osszetevok.txt fileból az e egyedi összetevők tömbbe.
- void osszetevo_fileba_ment (Egyedi_osszetevok ∗e, Receptkonyv ∗r)

  Összefésüli az e struktúrában található összetevőket a receptkönyv struktúra ételeinek az összetevőivel, hogy az egyedi összetevő struktúra pontosan egyszer tartalmazzon minden összetevőt, majd elmenti azt az előre megadott formátumban az osszetevok.txt fileba, ha nem létezik létrehozza a program gyökérkönyvtárába.
- void egyedi_osszetevo_felszabadit (Egyedi_osszetevok ∗e)

  Felszabadítja a kapott e struktúrát és minden alstruktúráját.
- int osszetevo_letezik (Egyedi_osszetevok ∗e, const char ∗osszetevo_neve)

  Kap egy egyedi osszetevok structot és egy stringet és megnézi hogy a string egyezik-e valamely r.osszetevok.nev stringgel elvileg case független, de mivel utf karakterek ezért néha bugos, az esetek 99ában működik.
- Osszetevo **o_beolvas1** (void)
- Osszetevo **o_beolvas2** (void)
- Osszetevo **o_beolvas3** (void)
- Etel **i_beolvas** (void)

4.1.1. Részletes leírás

Ebben a modulban kaptak helyet a filokból beolvasó illetve azokba kiíró fv-k, illetve az általánosan használt stdin/stdout olvasó író fv-ek.

Dátum

2024-11-08

4.1.2. Függvények dokumentációja

4.1.2.1. egyedi_osszetevo_felszabadit()

```
void egyedi_osszetevo_felszabadit (
            Egyedi_osszetevok * e)
```

Felszabadítja a kapott e struktúrát és minden alstruktúráját.

Paraméterek

e	

4.1.2.2. osszetevo_beolvas()

```
Egyedi_osszetevok * osszetevo_beolvas (
            void )
```

Beolvassa az összetevőket az előre megadott osszetevok.txt fileból az e egyedi összetevők tömbbe.

Visszatérési érték

Egyedi_osszetevok*

4.1.2.3. osszetevo_fileba_ment()

```
void osszetevo_fileba_ment (
            Egyedi_osszetevok * e,
            Receptkonyv * r)
```

Összefésüli az e struktúrában található összetevőket a receptkönyv struktúra ételeinek az összetevőivel, hogy az egyedi összetevő struktúra pontosan egyszer tartalmazzon minden összetevőt, majd elmenti azt az előre megadott formátumban az osszetevok.txt fileba, ha nem létezik létrehozza a program gyökérkönyvtárába.

Paraméterek

e	
r	

4.1.2.4. osszetevo_letezik()

```
int osszetevo_letezik (
            Egyedi_osszetevok * e,
            const char * osszetevo_neve)
```

Kap egy egyedi osszetevok structot és egy stringet és megnézi hogy a string egyezik-e valamely r.osszetevok.nev stringgel elvileg case független, de mivel utf karakterek ezért néha bugos, az esetek 99ában működik.

Paraméterek

e	Az összetevőket tartalmazó struktúra.
osszetevo_neve	Keresett string.

Visszatérési érték

int Ezután visszatér a keresett elem tömbbeli poziciójával, vagy nullával ha nem található.

4.1.2.5. recept_kiir()

```
void recept_kiir (
            Etel * m)
```

kiirja az adott etel struktura adatait(név összetevők elkészítés)

Paraméterek

m	Az étel struktúra amit ki akarunk írni.

4.1.2.6. recept_letezik()

```
int recept_letezik (
            Receptkonyv * r,
            const char * etel_neve)
```

Kap egy receptek structot és egy stringet és megnézi hogy a string egyezik e valamely r.etelek.nev stringgel elvileg case független, de mivel utf karakterek ezért néha bugos, az esetek 99ában működik.

Paraméterek

r	receptek struct
etel_neve	keresett string

Visszatérési érték

int Visszatér a keresett elem tömbbeli indexével plusz 1(i+1-el), vagy nullával ha nem található (azért kell az i+1 hogy a 0. elemet is helyesen kezelje).

4.1.2.7. receptek_beolvas()

```
Receptkonyv * receptek_beolvas (
            void )
```

beolvassa a recepteket az előre megadott receptek.txt fileból az r receptkönyv struktúrába (lehetett volna argumentum a filenév de felesleges), a működése egyszerű csak felbloatolja egy rakás hibakezelés.

Visszatérési érték

Receptkonyv∗

4.1.2.8. receptet_fileba_ment()

```
void receptet_fileba_ment (
            Receptkonyv * r)
```

Elmenti a kapott r struktúrát az előre megadott formátumban a receptek.txt fileba, ha nem létezik létrehozza a program gyökérkönyvtárába.

Paraméterek

r	Receptkönyv amit menteni szeretnénk

4.1.2.9. receptkonyv_felszabadit()

```
void receptkonyv_felszabadit (
            Receptkonyv * r)
```

felszabadítja a kapott r struktúrát és minden alstruktúráját

Paraméterek

r	

4.2. file_utils.h

```
00001 #ifndef FILE_UTILS_H
00002 #define FILE_UTILS_H
00003 #include <stdio.h>
00004 #include <stdlib.h>
00005 #include <string.h>
00006 #ifdef _WIN32
00007 #include <windows.h>
00008 #include <locale.h>
00009 #include <wchar.h>
00010 #include <windows.h>
00011 #include <locale.h>
00012 #include <fcntl.h>
00013 #include <io.h>
00014 #endif
00015 #include "debugmalloc.h"
00016
00017 typedef struct Osszetevo
00018 {
00019     char nev[51];
00020     char tipus[51];
00021     double mennyiseg;
00022 } Osszetevo;
00023 typedef struct Etel
00024 {
00025     char nev[51];
00026     int osszetevok_szama;
00027     Osszetevo* osszetevok;
00028     char elkeszites[1001];
00029 } Etel;
00030 typedef struct Receptkonyv
00031 {
00032     int etelek_szama;
00033     Etel* etelek;
00034 } Receptkonyv;
00035
00036 typedef struct Egyedi_osszetevok
00037 {
00038     Osszetevo* egyedi_osszetevok;
00039     int egyedi_osszetevok_szama;
00040 } Egyedi_osszetevok;
00041
00042 Receptkonyv* receptek_beolvas(void);
00043 void receptet_fileba_ment(Receptkonyv* r);
00044 void receptkonyv_felszabadit(Receptkonyv* r);
00045 int recept_letezik(Receptkonyv* r, const char* etel_neve);
00046 void recept_kiir(Etel* m);
00047
00048
00049 Egyedi_osszetevok* osszetevo_beolvas(void);
00050 void osszetevo_fileba_ment(Egyedi_osszetevok* e, Receptkonyv* r);
00051 void egyedi_osszetevo_felszabadit(Egyedi_osszetevok* e);
00052 int osszetevo_letezik(Egyedi_osszetevok* e, const char* osszetevo_neve);
00053
00054 Osszetevo o_beolvas1(void);
00055 Osszetevo o_beolvas2(void);
00056 Osszetevo o_beolvas3(void);
00057 Etel i_beolvas(void);
00058
00059 #endif
```

4.3. main.c fájlreferencia

Egynelőre elég redundáns, de megtartottam ha később bővíteni akarom a projectet akkor innen lehet. A példaként mellékelt osszetevok.txt és receptek.txt chatgpt által lettek generálva, nem mindig értelmes a tartlmuk.

```
#include <stdio.h>
#include "menu.h"
#include "file_utils.h"
```

Függvények

- int **main** (void)

4.3.1. Részletes leírás

Egynelőre elég redundáns, de megtartottam ha később bővíteni akarom a projectet akkor innen lehet. A példaként mellékelt osszetevok.txt és receptek.txt chatgpt által lettek generálva, nem mindig értelmes a tartlmuk.

4.4. menu.c fájlreferencia

A főmenüt kezelő modul, csak a főmenü, és a kilépés funkcionalitását tartalmazza.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "file_utils.h"
#include "o_menu.h"
#include "r_menu.h"
#include "debugmalloc.h"
```

Függvények

- int main_menu (void)

  A főmenü logikájával foglalkozó fv. Beolvassa a fileokból az összetevők listáját(továbbiakban e), és receptek listáját(továbbiakban r). A véletlenszerűnek tűnően elhelyezett "while (getchar() != '\n') {}" az utf-8 as beolvasás miatt kellenek, mivel inkonzisztensen olvassa az stdio streamet ezért ezzel törlöm az esetlegesen bennmaradt adatot a következő beolvasás előtt, mivel az fflush nem definiált c-ben és vicces dolgokat művel néha. A "system("@cls||clear");" csak kozmetikai célok miatt van ott, törli a konzolablakot az átláthatóság kedvéért. Tartalmaz egy összetevők, receptek, qol, bev. lista, és kedvencek menüpontot.
- int kilepes (Egyedi_osszetevok *e, Receptkonyv *r)

  Meghívja a kilépés előtti fv-eket: elmenti a recepteket és összetevőket és meghívja a megfelelő felszabadító fv-eket.
- void **menu_kiir** (void)

  Kilistázza a menüopciókat, stdio-n.

4.4.1. Részletes leírás

A főmenüt kezelő modul, csak a főmenü, és a kilépés funkcionalitását tartalmazza.

Dátum

2024-11-08

4.4.2. Függvények dokumentációja

4.4.2.1. kilepes()

```
int kilepes (
            Egyedi_osszetevok * e,
            Receptkonyv * r)
```

Meghívja a kilépés előtti fv-eket: elmenti a recepteket és összetevőket és meghívja a megfelelő felszabadító fv-eket.

Paraméterek

e	Egyedi_osszetevo struktúra, amiben az összes összetevő van.
r	Receptkony struktúra, amiben az összes receptet tárolom.

Visszatérési érték

int visszatér nullával ha sikerült minden, majd később hibakezeléshez kell.

4.4.2.2. main_menu()

```
int main_menu (
             void )
```

A főmenü logikájával foglalkozó fv. Beolvassa a fileokból az összetevők listáját(továbbiakban e), és receptek listáját(továbbiakban r). A véletlenszerűnek tűnően elhelyezett "while (getchar() != '\n') {}" az utf-8 as beolvasás miatt kellenek, mivel inkonzisztensen olvassa az stdio streamet ezért ezzel törlöm az esetlegesen bennmaradt adatot a következő beolvasás előtt, mivel az fflush nem definiált c-ben és vicces dolgokat művel néha. A "system("@cls||clear");" csak kozmetikai célok miatt van ott, törli a konzolablakot az átláthatóság kedvéért. Tartalmaz egy összetevők, receptek, qol, bev. lista, és kedvencek menüpontot.

Visszatérési érték

int ha minden rendben visszatér nullával, majd később hibakezeléshez.

4.5. menu.h

```
00001 #ifndef MENU_H
00002 #define MENU_H
00003 #include <stdio.h>
00004 #include <stdlib.h>
00005 #include <string.h>
00006 #ifdef _WIN32
00007 #include <windows.h>
00008 #include <locale.h>
00009 #endif
00010 #include "file_utils.h"
00011 #include "debugmalloc.h"
00012
00013 void osszetevok_almenu(Egyedi_osszetevok* e);
00014 int main_menu(void);
00015 int kilepes(Egyedi_osszetevok* e, Receptkonyv* r);
00016 void menu_kiir(void);
00017 void osszetevo_felvesz(Egyedi_osszetevok* e);
00018 void osszetevo_kiir(Egyedi_osszetevok* e);
00019
00020 #endif
```

4.6. o_menu.c fájlreferencia

Az összetevők almenühöz tartozó fv-ek, egy külön modulba szétszedve.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "file_utils.h"
#include "debugmalloc.h"
```

Függvények

- void osszetevok_almenu (Egyedi_osszetevok **e)

  *Az összetevők almenü vezérlőegysége, azért kell **e-t kapnia hogy hozzáférjen az eredeti pointerhez ha azon akar módosítani Tartalmaz egy összetevő felvétel, törlés, listázás, és keresés menüpontot.*
- void osszetevo_felvesz (Egyedi_osszetevok **e)

  Megnézi hogy van e összetevő a listában, ha nincs akkor inicializálja a listát. Ezután beolvas egy nevet, ha tartalmazza már a tömb akkor nem engedi újra felvenni egyébként hozzáadja a tömbhöz átméretezés után.
- void osszetevo_kiir (Egyedi_osszetevok *e)

  Kiírja az argumentumban kapot struct összetevő elemeinek a listáját.
- void osszetevok_keres (Egyedi_osszetevok *e)

  Keres az adott struktúrában a név alapján amit stdin-ről olvas be.
- void osszetevo_torol (Egyedi_osszetevok *e)

  Törli az adott összetevőt amit a nevével kell megadni, amennyiben létezik az e structban. Készít egy ideiglenes struckt pointert, lefoglalja neki az új hoszzát, majd átmásolja az összes elemet kivéve a törlendőt, végül felszabadítja az eredetit.

4.6.1. Részletes leírás

Az összetevők almenühöz tartozó fv-ek, egy külön modulba szétszedve.

Dátum

2024-11-08

4.6.2. Függvények dokumentációja

4.6.2.1. osszetevo_felvesz()

```
void osszetevo_felvesz (
            Egyedi_osszetevok ** e)
```

Megnézi hogy van e összetevő a listában, ha nincs akkor inicializálja a listát. Ezután beolvas egy nevet, ha tartalmazza már a tömb akkor nem engedi újra felvenni egyébként hozzáadja a tömbhöz átméretezés után.

Paraméterek

e	Receptek listája

4.6.2.2. osszetevo_kiir()

```
void osszetevo_kiir (
            Egyedi_osszetevok * e)
```

Kiírja az argumentumban kapot struct összetevő elemeinek a listáját.

Paraméterek

e	

4.6.2.3. osszetevo_torol()

```
void osszetevo_torol (
            Egyedi_osszetevok * e)
```

Törli az adott összetevőt amit a nevével kell megadni, amennyiben létezik az e structban. Készít egy ideiglenes struckt pointert, lefoglalja neki az új hoszzát, majd átmásolja az összes elemet kivéve a törlendőt, végül felszabadítja az eredetit.

Paraméterek

e	

4.6.2.4. osszetevok_almenu()

```
void osszetevok_almenu (
            Egyedi_osszetevok ** e)
```

Az összetevők almenü vezérlőegysége, azért kell ∗∗e-t kapnia hogy hozzáférjen az eredeti pointerhez ha azon akar módosítani Tartalmaz egy összetevő felvétel, törlés, listázás, és keresés menüpontot.

Paraméterek

e	A korábban beolvasott összetevők listáját tartalmazó egyedi összetevők struct.

4.6.2.5. osszetevok_keres()

```
void osszetevok_keres (
            Egyedi_osszetevok * e)
```

Keres az adott struktúrában a név alapján amit stdin-ről olvas be.

Paraméterek

e	

4.7. o_menu.h

```
00001 #ifndef O_MENU_H
00002 #define O_MENU_H
00003 #include <stdio.h>
00004 #include <stdlib.h>
00005 #include <string.h>
00006 #ifdef _WIN32
00007 #include <windows.h>
00008 #endif
00009 #include "file_utils.h"
00010 #include "debugmalloc.h"
00011
00012 void osszetevok_almenu(Egyedi_osszetevok** e);
00013 void osszetevo_felvesz(Egyedi_osszetevok** e);
00014 void osszetevo_kiir(Egyedi_osszetevok* e);
00015 void osszetevok_keres(Egyedi_osszetevok* e);
00016 void osszetevo_torol(Egyedi_osszetevok* e);
00017
00018
00019 #endif
```

4.8. r_menu.c fájlreferencia

A receptek almenühöz tartozó fv-ek, egy külön modulba szétszedve.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "file_utils.h"
#include "debugmalloc.h"
```

Függvények

- void receptek_almenu (Receptkonyv **r)

  *A receptek almenü vezérlőegysége, azért kell **r-t kapnia hogy hozzáférjen az eredeti pointerhez ha azon akar módosítani Tartalmaz egy receptek felvétel, törlés, listázás, és keresés menüpontot.*
- void recept_felvesz (Receptkonyv **r)

  Megnézi hogy van-e recept a listában, ha nincs akkor inicializálja a listát. Ezután beolvas egy nevet, ha tartalmazza már a tömb akkor nem engedi újra felvenni, egyébként hozzáadja a tömbhöz átméretezés után. A beolvasás kicsit körülményes de működik.
- void recept_keres (Receptkonyv *r)

  Kiírja az argumentumban kapot struct receptekben lévő ételek neveit.
- void recept_torol (Receptkonyv *r)

  Törli az adott receptet amit az étel nevével kell megadni, amennyiben létezik az r structban. Készít egy ideiglenes struckt pointert, lefoglalja neki az új hoszzát, majd átmásolja az összes elemet kivéve a törlendőt, végül felszabadítja az eredetit.
- void recept_listaz (Receptkonyv *r)

  Keres az adott struktúrában a név alapján amit stdin-ről olvas be.

4.8.1. Részletes leírás

A receptek almenühöz tartozó fv-ek, egy külön modulba szétszedve.

Dátum

2024-11-08

4.8.2. Függvények dokumentációja

4.8.2.1. recept_felvesz()

```
void recept_felvesz (
            Receptkonyv ** r)
```

Megnézi hogy van-e recept a listában, ha nincs akkor inicializálja a listát. Ezután beolvas egy nevet, ha tartalmazza már a tömb akkor nem engedi újra felvenni, egyébként hozzáadja a tömbhöz átméretezés után. A beolvasás kicsit körülményes de működik.

Paraméterek

e	Receptek listája

4.8.2.2. recept_keres()

```
void recept_keres (
            Receptkonyv * r)
```

Kiírja az argumentumban kapot struct receptekben lévő ételek neveit.

Paraméterek

e	

4.8.2.3. recept_listaz()

```
void recept_listaz (
            Receptkonyv * r)
```

Keres az adott struktúrában a név alapján amit stdin-ről olvas be.

Paraméterek

e	

4.8.2.4. recept_torol()

```
void recept_torol (
            Receptkonyv * r)
```

Törli az adott receptet amit az étel nevével kell megadni, amennyiben létezik az r structban. Készít egy ideiglenes struckt pointert, lefoglalja neki az új hoszzát, majd átmásolja az összes elemet kivéve a törlendőt, végül felszabadítja az eredetit.

Paraméterek

e	

4.8.2.5. receptek_almenu()

```
void receptek_almenu (
            Receptkonyv ** r)
```

A receptek almenü vezérlőegysége, azért kell ∗∗r-t kapnia hogy hozzáférjen az eredeti pointerhez ha azon akar módosítani Tartalmaz egy receptek felvétel, törlés, listázás, és keresés menüpontot.

Paraméterek

e	A korábban beolvasott receptek listáját tartalmazó Receptköny struct.

4.9. r_menu.h

```
00001 #ifndef R_MENU_H
00002 #define R_MENU_H
00003 #include <stdio.h>
00004 #include <stdlib.h>
00005 #include <string.h>
00006 #ifdef _WIN32
00007 #include <windows.h>
00008 #endif
00009 #include "file_utils.h"
00010 #include "debugmalloc.h"
00011
00012 void receptek_almenu(Receptkonyv** r);
00013 void recept_felvesz(Receptkonyv** r);
00014 void recept_keres(Receptkonyv* r);
00015 void recept_torol(Receptkonyv* r);
00016
00017 #endif
```

Tárgymutató