NHF

Verzió 1.0

Adatszerkezet-mutató

Adatszerkezetek

Az összes adatszerkezet listája rövid leírásokkal:

Adatszerkezetek dokumentációja

address struktúrareferencia

Cím tárolására használt struktúra.
`#include <vcard.h>`

Adatmezõk

char **zip** [16]
char **street_no** [128]
char **city** [96]
char **country** [96]
char **county** [96]

Részletes leírás

Cím tárolására használt struktúra.

Adatmezõk dokumentációja

char city[96]

char country[96]

char county[96]

char street_no[128]

char zip[16]

Ez a dokumentáció a struktúráról a következõ fájl alapján készült:

vcard.h

contact struktúrareferencia

Egy kontakt összes adatának tárolására használt struktúra.
`#include <vcard.h>`

Adatmezõk

char **phone** [20]
char **fn** [324]
fullname name
char **email** [256]
address address
char **bday** [9]
char **note** [256]
char **org** [96]
char **title** [96]

Részletes leírás

Egy kontakt összes adatának tárolására használt struktúra.

Adatmezõk dokumentációja

address address

char bday[9]

char email[256]

char fn[324]

fullname name

char note[256]

char org[96]

char phone[20]

char title[96]

Ez a dokumentáció a struktúráról a következõ fájl alapján készült:

vcard.h

fullname struktúrareferencia

Teljes név tárolására használt struktúra.
`#include <vcard.h>`

Adatmezõk

char **prefix** [16]
char **first** [96]
char **middle** [96]
char **last** [96]
char **suffix** [16]

Részletes leírás

Teljes név tárolására használt struktúra.

Adatmezõk dokumentációja

char first[96]

char last[96]

char middle[96]

char prefix[16]

char suffix[16]

Ez a dokumentáció a struktúráról a következõ fájl alapján készült:

vcard.h

ListaElem struktúrareferencia

A láncolt lista egy eleme.
`#include <lista.h>`

Adatmezõk

contact adat
struct **ListaElem** * **next**

Részletes leírás

A láncolt lista egy eleme.

Paraméterek

adat	Az elem által tárolt adat.
next	A következõ listaelem címe. Ha nincs, NULL.

Adatmezõk dokumentációja

contact adat

struct ListaElem* next

Ez a dokumentáció a struktúráról a következõ fájl alapján készült:

lista.h

lista.c fájlreferencia

```
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
#include <sys/stat.h>
#include "vcard.h"
#include "debugmalloc.h"
#include "lista.h"
#include "menu.h"
```

Függvények

int **listahossz** (**ListaElem** *elso)

Láncolt lista hosszát számolja meg, laborhoz használt függvény átalakított verziója.

void **lista_free** (**ListaElem** *eleje)

Láncolt listát szabadít fel, laborhoz használt függvény átalakított verziója.

ListaElem * **elore_beszur** (**ListaElem** *eleje, **contact** adat)

Láncolt lista elejére szúr be adatot, laborhoz használt függvény átalakított verziója.

ListaElem * **GetLastItem** (**ListaElem** *eleje)

Láncolt lista utolsó elemét keresi meg, laborhoz használt függvény átalakított verziója.

ListaElem * **vegere_beszur** (**ListaElem** *eleje, **contact** adat)

Láncolt lista végére szúr be adatot, laborhoz használt függvény átalakított verziója.

ListaElem * **keres** (**ListaElem** *eleje, char *needle)

Megkeres egy kifejezést az adatbázisban, és visszaad egy listát az összes olyan kontakttal, ami tartalmazza a keresett kifejezést valamilyen mezõben.

char * **sor_olvas** ()

Dinamikus memórával egy hosszú sort beolvasó függvény.

void **lista_kiir_short** (**ListaElem** *eleje)

Kiírja egy kontaktokból álló láncolt lista elemeinek nevét és telefonszámukat.

ListaElem * **import_all** (**ListaElem** *eleje)

Az összes kártyát importálja a /cards mappából.

ListaElem * **nth** (**ListaElem** *eleje, size_t n)

Egy láncolt lista n-edik elemét keresi meg és adja vissza.

Függvények dokumentációja

ListaElem * elore_beszur (ListaElem * *eleje*, contact *adat*)

Láncolt lista elejére szúr be adatot, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.
adat	A beszúrandó adat.

ListaElem * GetLastItem (ListaElem * *eleje*)

Láncolt lista utolsó elemét keresi meg, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.

ListaElem * import_all (ListaElem * *eleje*)

Az összes kártyát importálja a /cards mappából.

Paraméterek

eleje	A lista eleje, amibe beilleszti az elemeket.

ListaElem * keres (ListaElem * *eleje*, char * *needle*)

Megkeres egy kifejezést az adatbázisban, és visszaad egy listát az összes olyan kontakttal, ami tartalmazza a keresett kifejezést valamilyen mezõben.

Paraméterek

eleje	A láncolt lista (elsõ elemének címe), amelyben megkeresi a kifejezést.
keresett	A keresett kifejezés (string).

Visszatérési érték

Láncolt lista a keresési eredményekkel.

void lista_free (ListaElem * *eleje*)

Láncolt listát szabadít fel, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.

void lista_kiir_short (ListaElem * *eleje*)

Kiírja egy kontaktokból álló láncolt lista elemeinek nevét és telefonszámukat.

Paraméterek

eleje	A lista eleje.

int listahossz (ListaElem * *elso*)

Láncolt lista hosszát számolja meg, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.

ListaElem * nth (ListaElem * *eleje*, size_t *n*)

Egy láncolt lista n-edik elemét keresi meg és adja vissza.

Túlindexeléskor figyelmeztet és NULL pointert ad vissza.

Paraméterek

eleje	A lista eleje
n	Hányadik elemet adja vissza

Visszatérési érték

A lista n-edik eleme

char * sor_olvas ()

Dinamikus memórával egy hosszú sort beolvasó függvény.

A program végül nem használja, a sok realloc helyett hatékonyabb a tipikusnál nem sokkal nagyobb (vagy szabványos adatok, pl. irányítószám esetén a szabvány által meghatározott maximális méretû) terület foglalása.

Visszatérési érték

Dinamikusan foglalt string

ListaElem * vegere_beszur (ListaElem * *eleje*, contact *adat*)

Láncolt lista végére szúr be adatot, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.
adat	A beszúrandó adat.

lista.h fájlreferencia

```
#include <stdlib.h>
#include "vcard.h"
```

Adatszerkezetek

struct **ListaElem***A láncolt lista egy eleme.*

Típusdefiníciók

typedef struct **ListaElem ListaElem**

A láncolt lista egy eleme.

Függvények

int **listahossz** (**ListaElem** *eleje)

Láncolt lista hosszát számolja meg, laborhoz használt függvény átalakított verziója.

void **lista_free** (**ListaElem** *eleje)

Láncolt listát szabadít fel, laborhoz használt függvény átalakított verziója.

ListaElem * **elore_beszur** (**ListaElem** *eleje, **contact** adat)

Láncolt lista elejére szúr be adatot, laborhoz használt függvény átalakított verziója.

ListaElem * **GetLastItem** (**ListaElem** *eleje)

Láncolt lista utolsó elemét keresi meg, laborhoz használt függvény átalakított verziója.

ListaElem * **vegere_beszur** (**ListaElem** *elso, **contact** uj)

Láncolt lista végére szúr be adatot, laborhoz használt függvény átalakított verziója.

ListaElem * **keres** (**ListaElem** *eleje, char *keresett)

Megkeres egy kifejezést az adatbázisban, és visszaad egy listát az összes olyan kontakttal, ami tartalmazza a keresett kifejezést valamilyen mezõben.

char * **sor_olvas** ()

Dinamikus memórával egy hosszú sort beolvasó függvény.

void **lista_kiir_short** (**ListaElem** *eleje)

Kiírja egy kontaktokból álló láncolt lista elemeinek nevét és telefonszámukat.

ListaElem * **import_all** (**ListaElem** *eleje)

Az összes kártyát importálja a /cards mappából.

ListaElem * **nth** (**ListaElem** *eleje, size_t n)

Egy láncolt lista n-edik elemét keresi meg és adja vissza.

Típusdefiníciók dokumentációja

typedef struct ListaElem ListaElem

A láncolt lista egy eleme.

Paraméterek

adat	Az elem által tárolt adat.
next	A következõ listaelem címe. Ha nincs, NULL.

Függvények dokumentációja

ListaElem * elore_beszur (ListaElem * *eleje*, contact *adat*)

Láncolt lista elejére szúr be adatot, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.
adat	A beszúrandó adat.

ListaElem * GetLastItem (ListaElem * *eleje*)

Láncolt lista utolsó elemét keresi meg, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.

ListaElem * import_all (ListaElem * *eleje*)

Az összes kártyát importálja a /cards mappából.

Paraméterek

eleje	A lista eleje, amibe beilleszti az elemeket.

ListaElem * keres (ListaElem * *eleje*, char * *needle*)

Megkeres egy kifejezést az adatbázisban, és visszaad egy listát az összes olyan kontakttal, ami tartalmazza a keresett kifejezést valamilyen mezõben.

Paraméterek

eleje	A láncolt lista (elsõ elemének címe), amelyben megkeresi a kifejezést.
keresett	A keresett kifejezés (string).

Visszatérési érték

Láncolt lista a keresési eredményekkel.

void lista_free (ListaElem * *eleje*)

Láncolt listát szabadít fel, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.

void lista_kiir_short (ListaElem * *eleje*)

Kiírja egy kontaktokból álló láncolt lista elemeinek nevét és telefonszámukat.

Paraméterek

eleje	A lista eleje.

int listahossz (ListaElem * *elso*)

Láncolt lista hosszát számolja meg, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.

ListaElem * nth (ListaElem * *eleje*, size_t *n*)

Egy láncolt lista n-edik elemét keresi meg és adja vissza.

Túlindexeléskor figyelmeztet és NULL pointert ad vissza.

Paraméterek

eleje	A lista eleje
n	Hányadik elemet adja vissza

Visszatérési érték

A lista n-edik eleme

char * sor_olvas ()

Dinamikus memórával egy hosszú sort beolvasó függvény.

A program végül nem használja, a sok realloc helyett hatékonyabb a tipikusnál nem sokkal nagyobb (vagy szabványos adatok, pl. irányítószám esetén a szabvány által meghatározott maximális méretû) terület foglalása.

Visszatérési érték

Dinamikusan foglalt string

ListaElem * vegere_beszur (ListaElem * *eleje*, contact *adat*)

Láncolt lista végére szúr be adatot, laborhoz használt függvény átalakított verziója.

Paraméterek

eleje	A lista eleje.
adat	A beszúrandó adat.

lista.h

Ugrás a fájl dokumentációjához.

```
1 #include <stdlib.h>
2 #include "vcard.h"
3
4 #ifndef NHF_LISTA_H
5 #define NHF_LISTA_H
6
11 typedef struct ListaElem{
12       contact adat;
13       struct ListaElem *next;
14 }ListaElem;
15
16 int listahossz(ListaElem *eleje);
17
18 void lista_free(ListaElem *eleje);
19
20 ListaElem *elore_beszur(ListaElem *eleje, contact adat);
21
22 ListaElem *GetLastItem(ListaElem *eleje);
23
24 ListaElem *vegere_beszur(ListaElem *elso, contact uj);
25
26 ListaElem *keres(ListaElem *eleje, char *keresett);
27
28 char *sor_olvas();
29
30 void lista_kiir_short(ListaElem *eleje);
31
32 ListaElem *import_all(ListaElem *eleje);
33
34 ListaElem *nth(ListaElem *eleje, size_t n);
35
36 #endif //NHF_LISTA_H
```

menu.c fájlreferencia

```
#include <stdio.h>
#include "menu.h"
#include "vcard.h"
#include <string.h>
```

Függvények

void **olvas** (char *s, size_t len)

Egy egyszerûbb, buffer overflow mentes scanf, csakis stringekhez.

char **menukiir** (const char **lista)

Kiírja az argumentumként megadott lista elemeit számozva, külön-külön sorba.

char * **beker** (const char *prompt, char *dest, size_t size)

A paraméterben megadott szöveg kiírása után bekér a felhasználótól egy karaktert a szabványos bemeneten.

char * **beker_keres** (const char *prompt, char *dest, size_t size)

char **contactmenu** (**contact** *c)

Kiírja a kontakt adatait tartalmazó menüt.

char **namemenu** (**contact** *c)

Kiírja a kontakt nevét tartalmazó menüt.

char **addressmenu** (**contact** *c)

Kiírja a kontakt címének adatait tartalmazó menüt.

void **clear** ()

Ez a függvény egy regular expressiont használ a képernyõ törlésére.

contact * **edit_contact** (**contact** *c, char *next)

Ez a függvény egy kontakt szerkesztését teszi lehetõvé.

char **contextmenu** (**contact** *c)

Változók

const char * **main_options** []

A fõmenü opcióit tartalmazó lista.

const char * **newc_options** []

Az új kontakt létrehozásánál megjelenített opciókat tartalmazó lista.

const char * **name_options** []

A név opcióit tartalmazó lista.

const char * **org_options** []

A cég opcióit tartalmazó lista.

const char * **view_options** []
A megtekintés opcióit tartalmazó lista.

const char * **address_options** []
A cím opcióit tartalmazza.

Függvények dokumentációja

char addressmenu (contact * *c*)

Kiírja a kontakt címének adatait tartalmazó menüt.

Paraméterek

c	A kontakt adatait tartalmazó struktúrára mutató pointer.

Visszatérési érték

A cím azon részének sorszáma, amit megnyitunk szerkesztésre/megtekintésre.

char * beker (const char * *prompt*, char * *dest*, size_t *size*)

A paraméterben megadott szöveg kiírása után bekér a felhasználótól egy karaktert a szabványos bemeneten.

Paraméterek

prompt	

Visszatérési érték

Visszatér a felhasználótól bekért karakterrel.

char * beker_keres (const char * *prompt*, char * *dest*, size_t *size*)

void clear ()

Ez a függvény egy regular expressiont használ a képernyõ törlésére.

Nem platform- vagy compiler specifikus.

Hiba:

CLion-ban az integrált terminálban nem töröl semmit.

char contactmenu (contact * *c*)

Kiírja a kontakt adatait tartalmazó menüt.

Paraméterek

c	A kontakt adatait tartalmazó struktúrára mutató pointer.

Visszatérési érték

A kontakt azon adatának sorszáma, amit megnyitunk szerkesztésre/megtekintésre.

char contextmenu (contact * *c*)

contact * edit_contact (contact * *c*, char * *next*)

Ez a függvény egy kontakt szerkesztését teszi lehetõvé.

Paraméterek

c	A kontakt amit szerkeszt.
next	Miután visszatér a függvény, kilépjen-e a szerkesztésbõl vagy sem.

Visszatérési érték

A módosított kontakt struktúra.

char menukiir (const char ** *lista*)

Kiírja az argumentumként megadott lista elemeit számozva, külön-külön sorba.

A felhasználó megadhatja a szabványos bemeneten az általa választott opció sorszámát.

Paraméterek

lista	Lista, amely tartalmazza a kiírandó opciókat.

Visszatérési érték

Visszatér a felhasználótól bekért karakterrel.

char namemenu (contact * *c*)

Kiírja a kontakt nevét tartalmazó menüt.

Paraméterek

c	A kontakt adatait tartalmazó struktúrára mutató pointer.

Visszatérési érték

A név azon részének sorszáma, amit megnyitunk szerkesztésre/megtekintésre.

void olvas (char * *s*, size_t *len*)

Egy egyszerûbb, buffer overflow mentes scanf, csakis stringekhez.

Paraméterek

s	A string amibe olvasunk.
len	A string maximális hossza.

Változók dokumentációja

const char* address_options[]

Kezdõ érték:= {

```
    "utca, házszám",
```

```
        "város",
        "megye",
        "irányítószám",
        "ország"
}
```

A cím opcióit tartalmazza.

const char* main_options[]

Kezdõ érték:
```
= {
        "új kontakt",
        "kontaktok megtekintése",
        "keresés",
        "összes exportálása",
        "összes importálása",
        NULL
}
```

A fõmenü opcióit tartalmazó lista.

const char* name_options[]

Kezdõ érték:
```
= {
        "név elõtag",
        "keresztnév",
        "második név",
        "vezetéknév",
        "név utótag",
        NULL
}
```

A név opcióit tartalmazó lista.

const char* newc_options[]

Kezdõ érték:
```
= {
        "név",
        "telefonszám",
        "email",
        "cím",
        "születésnap",
        "cég",
        "foglalkozás",
        "megjegyzés",
        NULL
}
```

Az új kontakt létrehozásánál megjelenített opciókat tartalmazó lista.

const char* org_options[]

Kezdõ érték:
```
= {
        "szervezet",
        "beosztás",
        NULL
}
```

A cég opcióit tartalmazó lista.

const char* view_options[]

Kezdõ érték:
```
= {
        "szerkesztés",
        "exportálás",
        "törlés",
        NULL
}
```

A megtekintés opcióit tartalmazó lista.

menu.h fájlreferencia

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

Függvények

void **olvas** (char *s, size_t len)

Egy egyszerûbb, buffer overflow mentes scanf, csakis stringekhez.

char **menukiir** (const char **lista)

Kiírja az argumentumként megadott lista elemeit számozva, külön-külön sorba.

char * **beker** (const char *prompt, char *dest, size_t size)

A paraméterben megadott szöveg kiírása után bekér a felhasználótól egy karaktert a szabványos bemeneten.

char * **beker_keres** (const char *prompt, char *dest, size_t size)

char **contactmenu** (**contact** *c)

Kiírja a kontakt adatait tartalmazó menüt.

char **namemenu** (**contact** *c)

Kiírja a kontakt nevét tartalmazó menüt.

void **clear** ()

Ez a függvény egy regular expressiont használ a képernyõ törlésére.

contact * **edit_contact** (**contact** *c, char *next)

Ez a függvény egy kontakt szerkesztését teszi lehetõvé.

Változók

const char * **main_options** []

A fõmenü opcióit tartalmazó lista.

const char * **newc_options** []

Az új kontakt létrehozásánál megjelenített opciókat tartalmazó lista.

const char * **name_options** []

A név opcióit tartalmazó lista.

const char * **org_options** []

A cég opcióit tartalmazó lista.

const char * **view_options** []

A megtekintés opcióit tartalmazó lista.

const char * **address_options** []
A cím opcióit tartalmazza.

Függvények dokumentációja

char * beker (const char * *prompt*, char * *dest*, size_t *size*)

A paraméterben megadott szöveg kiírása után bekér a felhasználótól egy karaktert a szabványos bemeneten.

Paraméterek

prompt	

Visszatérési érték

Visszatér a felhasználótól bekért karakterrel.

char * beker_keres (const char * *prompt*, char * *dest*, size_t *size*)

void clear ()

Ez a függvény egy regular expressiont használ a képernyõ törlésére.

Nem platform- vagy compiler specifikus.

Hiba:

CLion-ban az integrált terminálban nem töröl semmit.

char contactmenu (contact * *c*)

Kiírja a kontakt adatait tartalmazó menüt.

Paraméterek

c	A kontakt adatait tartalmazó struktúrára mutató pointer.

Visszatérési érték

A kontakt azon adatának sorszáma, amit megnyitunk szerkesztésre/megtekintésre.

contact * edit_contact (contact * *c*, char * *next*)

Ez a függvény egy kontakt szerkesztését teszi lehetõvé.

Paraméterek

c	A kontakt amit szerkeszt.
next	Miután visszatér a függvény, kilépjen-e a szerkesztésbõl vagy sem.

Visszatérési érték

A módosított kontakt struktúra.

char menukiir (const char ** *lista*)

Kiírja az argumentumként megadott lista elemeit számozva, külön-külön sorba.

A felhasználó megadhatja a szabványos bemeneten az általa választott opció sorszámát.

Paraméterek

lista	Lista, amely tartalmazza a kiírandó opciókat.

Visszatérési érték

Visszatér a felhasználótól bekért karakterrel.

char namemenu (contact * *c*)

Kiírja a kontakt nevét tartalmazó menüt.

Paraméterek

c	A kontakt adatait tartalmazó struktúrára mutató pointer.

Visszatérési érték

A név azon részének sorszáma, amit megnyitunk szerkesztésre/megtekintésre.

void olvas (char * *s*, size_t *len*)

Egy egyszerûbb, buffer overflow mentes scanf, csakis stringekhez.

Paraméterek

s	A string amibe olvasunk.
len	A string maximális hossza.

Változók dokumentációja

const char* address_options[][extern]

A cím opcióit tartalmazza.

const char* main_options[][extern]

A fõmenü opcióit tartalmazó lista.

const char* name_options[][extern]

A név opcióit tartalmazó lista.

const char* newc_options[][extern]

Az új kontakt létrehozásánál megjelenített opciókat tartalmazó lista.

const char* org_options[][extern]

A cég opcióit tartalmazó lista.

const char* view_options[][extern]

A megtekintés opcióit tartalmazó lista.

menu.h

Ugrás a fájl dokumentációjához.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include "vcard.h"
4 #include "debugmalloc.h"
5 #ifndef NHF_MENU_H
6 #define NHF_MENU_H
7
8 extern const char *main_options[];
9
10 extern const char *newc_options[];
11
12 extern const char *name_options[];
13
14 extern const char *org_options[];
15
16 extern const char *view_options[];
17
18 extern const char *address_options[];
19
20 void olvas(char *s, size_t len);
21
22 char menukiir(const char **lista);
23
24 char *beker(const char *prompt, char *dest, size_t size);
25
26 char *beker_keres(const char *prompt, char *dest, size_t size);
27
28 char contactmenu(contact *c);
29
30 char namemenu(contact *c);
31
32 void clear();
33
34 contact *edit_contact(contact *c, char *next);
35
36 #endif //NHF_MENU_H
```

vcard.c fájlreferencia

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
#include "vcard.h"
```

Függvények

fullname InitName (**fullname** empty)

Egy nevet inicializál, üres értékekkel tölti fel.

address InitAddress (**address** empty)

Egy címet inicializál, üres értékekkel tölti fel.

contact InitContact (**contact** empty)

Egy kontaktot inicializál, üres értékekkel tölti fel.

char * **straddr** (**address** *a)

A struktúra address változóját egyben, stringként adja vissza.

char * **strfn** (**fullname** *n)

A struktúra fullname változóját egyben, stringként adja vissza.

int **writecard** (char *filename, **contact** *out)

A paraméterként megadott contact típusú változó adatait vCard kiterjesztésû fájlba írja.

contact * **readcard** (char *filename, **contact** *c)

A paraméterként megadott contact típusú változóba másolja be a vCard kiterjesztésû fájl adatait.

Függvények dokumentációja

address InitAddress (address *empty*)

Egy címet inicializál, üres értékekkel tölti fel.

Visszatérési érték

Üres értékekkel feltöltött, address típusú változó.

contact InitContact (contact *empty*)

Egy kontaktot inicializál, üres értékekkel tölti fel.

Visszatérési érték

Üres értékekkel feltöltött, contact típusú változó.

fullname InitName (fullname *empty*)

Egy nevet inicializál, üres értékekkel tölti fel.

Visszatérési érték

Üres értékekkel feltöltött, fullname típusú változó.

contact * readcard (char * *filename*, contact * *c*)

A paraméterként megadott contact típusú változóba másolja be a vCard kiterjesztésû fájl adatait.

Paraméterek

filename	Olvasandó fájl neve.
out	A contact típusú változó, amibe írunk.

char * straddr (address * *a*)

A struktúra address változóját egyben, stringként adja vissza.

Visszatérési érték

A teljes cím egyetlen stringben

Paraméterek

n	A kontakt neve

char * strfn (fullname * *n*)

A struktúra fullname változóját egyben, stringként adja vissza.

Visszatérési érték

A teljes név egyetlen stringben

Paraméterek

n	A kontakt neve

int writecard (char * *filename*, contact * *out*)

A paraméterként megadott contact típusú változó adatait vCard kiterjesztésû fájlba írja.

Paraméterek

filename	Kimenõ fájl neve
out	A kiírandó contact típusú változó

vcard.h fájlreferencia

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
```

Adatszerkezetek

struct **fullname***Teljes név tárolására használt struktúra.*

struct **address***Cím tárolására használt struktúra.*

struct **contact***Egy kontakt összes adatának tárolására használt struktúra.*

Típusdefiníciók

typedef struct **fullname fullname**

Teljes név tárolására használt struktúra.

typedef struct **address address**

Cím tárolására használt struktúra.

typedef struct **contact contact**

Egy kontakt összes adatának tárolására használt struktúra.

Függvények

fullname InitName (**fullname** empty)

Egy nevet inicializál, üres értékekkel tölti fel.

contact InitContact (**contact** empty)

Egy kontaktot inicializál, üres értékekkel tölti fel.

address InitAddress (**address** empty)

Egy címet inicializál, üres értékekkel tölti fel.

char * **straddr** (**address** *a)

A struktúra address változóját egyben, stringként adja vissza.

char * **strfn** (**fullname** *n)

A struktúra fullname változóját egyben, stringként adja vissza.

int **writecard** (char *filename, **contact** *out)

A paraméterként megadott contact típusú változó adatait vCard kiterjesztésû fájlba írja.

contact * **readcard** (char *filename, **contact** *c)

A paraméterként megadott contact típusú változóba másolja be a vCard kiterjesztésû fájl adatait.

Típusdefiníciók dokumentációja

typedef struct address address

Cím tárolására használt struktúra.

typedef struct contact contact

Egy kontakt összes adatának tárolására használt struktúra.

typedef struct fullname fullname

Teljes név tárolására használt struktúra.

Függvények dokumentációja

address InitAddress (address *empty*)

Egy címet inicializál, üres értékekkel tölti fel.

Visszatérési érték

Üres értékekkel feltöltött, address típusú változó.

contact InitContact (contact *empty*)

Egy kontaktot inicializál, üres értékekkel tölti fel.

Visszatérési érték

Üres értékekkel feltöltött, contact típusú változó.

fullname InitName (fullname *empty*)

Egy nevet inicializál, üres értékekkel tölti fel.

Visszatérési érték

Üres értékekkel feltöltött, fullname típusú változó.

contact * readcard (char * *filename*, contact * *c*)

A paraméterként megadott contact típusú változóba másolja be a vCard kiterjesztésû fájl adatait.

Paraméterek

filename	Olvasandó fájl neve.
out	A contact típusú változó, amibe írunk.

char * straddr (address * *a*)

A struktúra address változóját egyben, stringként adja vissza.

Visszatérési érték

A teljes cím egyetlen stringben

Paraméterek

n	A kontakt neve

char * strfn (fullname * *n*)

A struktúra fullname változóját egyben, stringként adja vissza.

Visszatérési érték

A teljes név egyetlen stringben

Paraméterek

n	A kontakt neve

int writecard (char * *filename*, contact * *out*)

A paraméterként megadott contact típusú változó adatait vCard kiterjesztésû fájlba írja.

Paraméterek

filename	Kimenõ fájl neve
out	A kiírandó contact típusú változó

vcard.h

Ugrás a fájl dokumentációjához.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <dirent.h>
5
6 #ifndef NHF_VCARD_H
7 #define NHF_VCARD_H
8
10 typedef struct fullname{
11      char prefix[16];
12      char first[96];
13      char middle[96];
14      char last[96];
15      char suffix[16];
16 }fullname;
17
19 typedef struct address{
20      //char type[64];
21      char zip[16];
22      char street_no[128];
23      char city[96];
24      char country[96];
25      char county[96];
26 }address;
27
29 typedef struct contact{
30      char phone[20];
31      char fn[324];
32      fullname name;
33      char email[256];
34      address address;
35      char bday[9];
36      char note[256];
37      char org[96];
38      char title[96];
39 }contact;
40
41 fullname InitName(fullname empty);
42
43 contact InitContact(contact empty);
44
45 address InitAddress(address empty);
46
47 char *straddr(address *a);
48
49 char *strfn(fullname *n);
50
51 int writecard(char *filename, contact *out);
52
53 contact *readcard(char *filename, contact *c);
54
55 #endif //NHF_VCARD_H
```