

Structures

מבנים

מהו מבנה?

מבנה (או רשומה) הוא סוג של משתנה, שאנו יכולים להגדיר בעצמנו, המכיל, כמו במערך, מספר משתנים. להבדיל מהמערך, המבנה יכול להכיל מספר סוגים של משתנים, והגישה אל המשתנים אינה מספרית אלא שמית.

בשביל מה בעצם צריך מבנים?

בתוכניות אנחנו בדרך כלל משתמשים בנתונים שונים, שיש ביניהם קשר. מבנים יכולים לעזור לנו לאגד ביחד נתונים הקשורים זה לזה, וכך לאפשר לטפל בהם בצורה נוחה ומסודרת.

Structures

מבנים

הגדרת מבנה

כדי להגדיר מבנה מסוג חדש אנו צריכים לעשות את זה בצורה הבאה:

```
struct structName
{
    type1 name1;    /*field 1*/
    type2 name2;    /*field 2*/
    ...
    typeN nameN;
};
```

למשל:

```
struct Person
{
    char name[10];
    int age;
    char address[20];
};
```

חשוב להדגיש שבהצהרה זו לא הגדרנו שום משתנה, אלא רק הכרזנו על טיפוס נתונים חדש. כרגע לא בוצעה שום הקצאת זכרון!!!

Structures

מבנים

לאחר שהגדרנו את הטיפוס החדש, נוכל להשתמש בו כמו בכל טיפוס אחר. ניצור משתנה חדש מסוג המבנה שהגדרנו.

הגדרת משתנה:

להגדרת משתנה בשם x מסוג struct structName יש לכתוב כך:

```
struct structName x;
```

נוכל גם ליצור מצביע לטיפוס זה:

```
struct structName *y;      /* a pointer to a structure */
```

גישה למשתנים :

כדי לגשת למשתנים בתוך מבנה (שנקרא שדה), יש לכתוב את שם המשתנה מסוג המבנה, אחריו נקודה, ואחריה שם המשתנה בתוך מבנה (שדה) :

```
x.age=30;
```

אם המשתנה הוא מצביע למבנה, יש להשתמש בחץ (->) במקום הנקודה, על מנת לגשת למשתנה בתוך מבנה המוצבע:

```
y->age=40;
```

Structures

מבנים

השמה לשדות

בדוגמה הזאת מייצרים משתנה מסוג struct item שמתאר שוקו.

כפי שראינו מקודם בגישה לשדות המבנה, אפשר לגשת לכל אחד משדות המשתנה. נוכל, לכן, להשתמש בהשמה לכל אחד מאיבריו:

```
struct item
{
    int catalog_number;
    char name[20];
    float price;
    unsigned int num;
};

int main()
{
    struct item shoko;           /* variable definition */

    shoko.catalog_number = 23;
    strcpy(shoko.name, "shoko");
    shoko.price = 12.90;
    shoko.num = 100;
    .....
```

Structures

מבנים

שימוש ב-`typedef`:

אם רוצים לחסוך את הצורך לכתוב את המילה השמורה `struct` בכל פעם שמגדירים משתנה מסוג של מבנה מסוים, ניתן להגדיר את המבנה באופן הבא:

```
typedef struct {  
    רשימת המשתנים בתוך המבנה, כפי שהדגמנו /*  
} structName;          /*the new type*/
```

עכשיו אפשר להגדיר משתנים ומצביעים פשוט על ידי:

```
structName x;
```

או

```
structName *y;
```

Structures

מבנים

דוגמא 1:

שימוש במערך מבנים.

```
#include <stdio.h>
```

```
# define NUM 5
```

```
typedef struct student
```

```
{
```

```
    char first_name[8];
```

```
    char last_name[8];
```

```
    int marks[3];    /*array as a structure's field*/
```

```
}student;
```

```
void get_data(student* arr, int size);
```

```
void print_average(student* p, int size);
```

```
int main ( )
```

```
{
```


```
    student array[NUM]; /*array of structures*/
```

Structures

מבנים

```
get_data(array, NUM);  
print_average(array, NUM);  
return 0;  
}
```

הפונקציה מקבלת כתובת של תחילת המערך ואת אורכו

```
void get_data(student *arr, int size)   
{  
    int i, j;  
    for (i=0; i<size; i++)  
    {  
        printf(" \n please enter data for student's name\n");  
        scanf ("%s%s", arr[i].first_name, arr[i].last_name);  
        printf("\n please enter data for 3 marks\n" );  
        for (j=0; j<3 ; j++)  
            scanf ( "%d", &arr[i].marks[j]);  
    }  
}
```

Structures

מבנים

הפונקציה מקבלת כתובת של תחילת המערך
ואת אורכו

```
void print_average(student *p, int size)
{
    int i, j, sum;
    for (i=0; i<size; i++, p++)
    {
        for (j=0, sum=0; j<3 ; j++)
            sum+= p->marks[j];
        printf ("\n The average mark of %s %s is %f",
            p->first_name, p->last_name, (float)sum/3 );
    }
}
```

יש כאן שימוש במצביע לכל תא במערך מבנים. גם כך אפשרי
אם כי שימוש באינדקסים של תאים במערך(כמו בפונקציה
קודמת) בדרך כלל עדיף.

Structures

מבנים

תכונות של מבנים

הצבת תוכן של משתנה אחד מסוג מבנה למשתנה אחר
מסוג מבנה :

ניתן לבצע הצבה פשוטה, בתנאי ששני המשתנים מאותו טיפוס
בדיוק:

```
struct student a, b;
```

```
.....
```

```
a = b;
```

מה שמבוצע במקרה זה הוא העתקת תוכן המשתנה מסוג
מבנה b למשתנה מסוג מבנה a שדה-שדה.

Structures

מבנים

העברת משתמה מסוג מבנה כפרמטר לפונקציה :

```
int main()
{
    struct student a;
    func(a);
    .....
}

void func(struct student b)
{
    .....
}
```

פונקציה יצרה משתנה b שהוא העתק של a (כל השדות

מועתקים אחד-אחד). השינויים שנעשים על משתנה b

בפונקציה, אינם נשמרים ביציאה מהפונקציה.

בגלל העתקת מידע, עדיף להעביר לפונקציה מצביע למבנה

ולא משתנה מסוג מבנה.

Structures

מבנים

דוגמא 2:

שימוש במערך מצביעים למבנים

```
#include <stdio.h>
```

```
#include <string.h>
```

```
#define MAX_CL 10
```

```
#define NAME_LEN 20
```

```
typedef struct cl_struct
```

```
{
```

```
    int id;
```

```
    char name[NAME_LEN];
```

```
} cl_struct;
```

```
int get_cl_data(cl_struct *A[], int num);
```


```
void what_is_it(cl_struct *A[], int a);
```

Structures

מבנים

```
int main()
{
    int i, cl_num;
    cl_struct cl_data_vec[MAX_CL];
    cl_struct *cl_data_ptr_vec[MAX_CL];
    for( i = 0; i<MAX_CL; i++)
        cl_data_ptr_vec[i] = &cl_data_vec[i];    /* Array of
                                                    pointers */
    cl_num = get_cl_data(cl_data_ptr_vec, MAX_CL);
    if (cl_num >=1)    /*If there is at least one client */
        what_is_it(cl_data_ptr_vec, cl_num);
    return 0;
}
```

מערך מצביעים




Structures

מבנים

```
int get_cl_data(cl_struct *A[], int num)
{
    int i;

    for (i = 0; i < num; i++)
    {
        printf("\nPlease enter client %d data\n", i+1);
        printf("Client id and name (up to 19 chars):");
        scanf("%d %s", &A[i]->id, A[i]->name);
        if (A[i]->id == 0)
            break; /*If the customer id is 0 - stop! */
    }
    return i;      /* We need to return the number of
                    clients and not the index of the last one. */
}
```



אפשר לכתוב: **cl_struct **A**

Structures

מבנים

עבודה עצמית לסטודנטים : מה מבצעת הפונקציה?

```
void what_is_it(cl_struct *A[], int act_num )
```

```
{
```

```
    int i;
```

```
    cl_struct tmp_s;
```

A הוא מצביע כפול



```
    strcpy( tmp_s.name, A[0]->name);
```

```
    tmp_s.id = A[0]->id;
```

```
    for (i = 1; i < act_num; i++)    /*Find the "biggest" id
                                     and name */
```

```
{
```

```
    if (tmp_s.id < A[i]->id)
```

```
        tmp_s.id = A[i]->id;
```

```
    if ( strcmp( tmp_s.name, A[i]->name)<0 )
```

```
        strcpy( tmp_s.name, A[i]->name);
```

```
}
```

```
    printf("%d, %s", tmp_s.id, tmp_s.name);
```

```
}
```