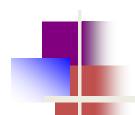


## מבוא לתכנות בשפת C

מבנים



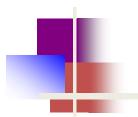
#### מבנים - structures

- קבוצה של ערכים (לא בהכרח מאותו סוג) שמהווים
   יחד יחידה אחת
  - דוגמה:
  - מבנה של פריט למכירה (article) בחנות
    - שם (name) המוצר
    - הכמות (quantity) המאוכסנת
      - המחיר (price)
        - אופן השימוש •
        - מגדירים מבנה
        - יוצרים משתנה
      - מכניסים לתוכו ערכים
        - משתמשים בערכים



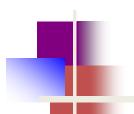
## איך?

```
char name[40];
    int quantity;
    double price;
} a1, *p1, arr1[100];
```



## ?איך?

```
int main() {
      strcpy(a1.name, "Football");
      al.quantity = 50; al.price = 129.99;
      strcpy(arr1[30].name, "Football");
      arr1[30].quantity = 50; arr1[30].price = 129.99;
      p1 = arr1 + 30;
      strcpy((*p1).name, "Football");
       (*p1).quantity = 50; (*p1).price = 129.99;
      strcpy(p1->name, "Football");
      p1->quantity = 50; p1->price = 129.99;
      printf("%s, %d, %lf, %s", al.name,
             arr1[30].quantity, (*p1).price, p1->name);
      return 0:
```



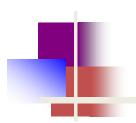
```
struct article {
        char name[40];
        int quantity;
        double price;
};
struct article a2, *p2, arr2[100];
```



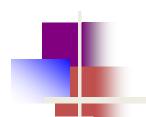
```
typedef int Integer;
Integer n;

typedef struct {
    char name[40];
    int quantity;
    double price;
} Article;

Article a3, *p3, arr3[100];
```



```
Article a4 = { "Football", 50, 129.99 };
Article arr4[] = { { "Football", 50, 129.99 }, { "Football", 50, 129.99 } };
Article a5 = a4;
p3 = (Article *) malloc(sizeof(Article));
*p3 = arr4[1];
*p3 = a5;
free(p3);
```



- בהתאם לשאר הכללים שלמדנו
- למשל ה- name יכול להיות דינאמי
- מצריך כמובן הקצאה, ואחרי השימוש שחרור
- למשל השדות של ה- struct יכולים להיות struct אחר, או מערך של כל דבר
  - ... וכך הלאה –



## דוגמה – הוראות לקומפיילר

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#include <float.h>
#include <stdlib.h>
#define RAND(R) (((double) rand() / RAND MAX) * R)
```



#### נקודה במישור

```
struct point {
                                             by value העברה
       double x, y;
                                                  – כתובת
};
                                                   העתק –
typedef struct point Point;
                                       באותו אופן ניתן להחזיר
void randomPoint(Point *pp) {
       pp->x = RAND(10);
                                                  – כתובת
       pp \rightarrow y = RAND(10);
                                                   העתק –
void printPoint(Point p) {
       printf("(%.21f, %.21f)", p.x, p.y);
int isEqual(Point p1, Point p2) {
        return p1.x == p2.x && p1.y == p2.y;
double dist(Point p1, Point p2) {
        return sqrt(pow(p1.x - p2.x, 2) + pow(p1.y - p2.y, 2));
```



```
Point p1, p2, p3;
srand(time(NULL));
randomPoint(&p1);
printPoint(p1); // (1.25, 0.50)
printf("\n");
randomPoint(&p2);
printPoint(p2); // (6.49, 1.83)
printf("\n");
printf("dist = %.21f\n", dist(p1, p2));
        // dist = 5.41
printf("isEqual = %s\n", isEqual(p1, p2) ? "true" : "false");
        // isEqual = false
p3 = p1;
printf("isEqual = %s\n", isEqual(p1, p3) ? "true" : "false");
         // isEqual = true
```

```
מעגל במישור
  ruct circle {
        Point center:
        double radius;
typedef struct circle Circle;
void randomCircle(Circle *pc) {
        randomPoint(&pc->center);
        pc->radius = RAND(15);
void printCircle (Circle c) {
        printf("(");
        printPoint(c.center);
        printf(", %.21f)", c.radius);
double perimeter(Circle c) {
        return 2 * M PI * c.radius;
double area(Circle c) {
        return M PI * c.radius * c.radius;
int isInside(Point p, Circle c) {
        return dist(p, c.center) < c.radius;</pre>
int isItersect(Circle c1, Circle c2) {
        return dist(c1.center, c2.center) <= c1.radius + c2.radius;
```



```
randomCircle(&c1);
printCircle(c1); // ((3.75, 0.02), 12.08)
printf("\n");
printf("perimeter = %.21f\n", perimeter(c1)); // perimeter = 75.90
printf("area = %.21f\n", area(c1)); // area = 458.43
printf("dist = %.21f\n", dist(p1, c1.center)); // dist = 2.55
printf("isInside = %s\n", isInside(p1, c1) ? "true" : "false");
        // isInside = true
randomCircle(&c2);
printCircle(c2); // ((5.90, 5.72), 13.57)
printf("\n");
printf("isIntersect = %s\n", isItersect(c1, c2) ? "true" : "false");
        // isIntersect = true
```

## קבוצה של נקודות

```
#define CAPACITY 20
struct setOfPoints {
          Point points[CAPACITY];
          int size;
};
typedef struct setOfPoints SetOfPoints;
void emptySetOfPoints (SetOfPoints *ps) {
          ps->size = 0;
void randomSetOfPoints (SetOfPoints *ps) {
          int i:
          int sz = RAND(CAPACITY/2);
          emptySetOfPoints(ps);
          for (i = 0; i < sz; ++i) {</pre>
                     Point p;
                     do {
                                randomPoint(&p);
                     } while (isMember(p, *ps));
                     insert(p, ps);
void printSetOfPoints (SetOfPoints s) {
          int i;
          printf("{\n");
          for (i = 0; i < s.size; ++i) {</pre>
                     printf("\t");
                     printPoint(s.points[i]);
                     printf("\n");
          printf("}");
                                       Tzachi (Isaac) Rosen
```

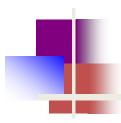


```
int whereIs(Point p, SetOfPoints s) {
         int i;
         for (i = 0; i < s.size; ++i)</pre>
                  if (isEqual(p, s.points[i]))
                           return i:
         return -1;
int isMember(Point p, SetOfPoints s) {
         int i = whereIs(p, s);
         return (i != -1) ? 1 : 0;
void insert(Point p, SetOfPoints *ps) {
         int i = whereIs(p, *ps);
         if (i == -1) {
                  ps->points[ps->size] = p;
                  ++ps->size;
void delete(Point p, SetOfPoints *ps) {
         int i = whereIs(p, *ps);
         if (i != -1) {
                  --ps->size;
                  ps->points[i] = ps->points[ps->size];
                                  Tzachi (Isaac) Rosen
```



## קבוצה של נקודות

```
void uni(SetOfPoints * ps1, SetOfPoints s2) {
       int i;
       for (i = 0; i < s2.size; ++i)
              insert(s2.points[i], ps1);
void intersect(SetOfPoints * ps1, SetOfPoints s2) {
       int i;
       SetOfPoints s:
      emptySetOfPoints(&s);
       for (i = 0; i < ps1->size; ++i)
              if (isMember(ps1->points[i], s2))
                    insert(ps1->points[i], &s);
       *ps1 = s;
```



```
randomSetOfPoints(&s1);
printSetOfPoints(s1); // {
              (6.37, 6.75)
printf("\n");
randomSetOfPoints(&s2);
printSetOfPoints(s2); // {
              (6.20, 9.17)
//
              (7.71, 5.33)
//
//
             (5.30, 8.47)
             (4.86, 8.79)
//
//
             (6.47, 2.94)
//
             (6.02, 5.01)
//
             (4.39, 0.06)
              (7.20, 7.94)
printf("\n");
```



```
printf("isMember = %s\n", isMember(p1, s1) ? "true" : "false");
       // isMember = false
insert(p1, &s1);
printSetOfPoints(s1); // {
// (6.37, 6.75)
             (1.25, 0.50)
//
// }
printf("\n");
printf("isMember = %s\n", isMember(p1, s1) ? "true" : "false");
       // isMember = true
delete(s1.points[(s1.size - 1) / 2], &s1);
printSetOfPoints(s1); // {
//
      (1.25, 0.50)
printf("\n");
```

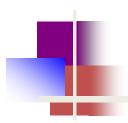
```
uni(&s1, s2);
printSetOfPoints(s1); // {
                  (1.25, 0.50)
//
//
                  (6.20, 9.17)
                  (7.71, 5.33)
//
//
                  (5.30, 8.47)
//
                  (4.86, 8.79)
                  (6.47, 2.94)
//
//
                  (6.02, 5.01)
//
                  (4.39, 0.06)
//
                  (7.20, 7.94)
//
printf("\n");
intersect(&s1, s2);
printSetOfPoints(s1); // {
//
                  (6.20, 9.17)
//
                  (7.71, 5.33)
//
                  (5.30, 8.47)
//
                  (4.86, 8.79)
//
                  (6.47, 2.94)
//
                  (6.02, 5.01)
//
                  (4.39, 0.06)
//
                  (7.20, 7.94)
//
printf("\n");
                         Tzachi (Isaac) Rosen
```

# קבוצה דינאמית של נקודות

```
struct dynSetOfPoints {
          Point *points;
          int size;
typedef struct dynSetOfPoints DynSetOfPoints;
void dynEmptySetOfPoints(DynSetOfPoints *ps) {
          ps->points = (Point *) malloc(0);
          ps->size = 0;
void dynRandomSetOfPoints (DynSetOfPoints *ps) {
          int i:
          int sz = RAND(CAPACITY/2);
          dynEmptySetOfPoints(ps);
          for (i = 0; i < sz; ++i) {
                    Point p;
                     do {
                               randomPoint(&p);
                     } while (dynIsMember(p, *ps));
                     dynInsert(p, ps);
void dynPrintSetOfPoints (DynSetOfPoints s) {
          int i;
          printf("{\n");
          for (i = 0; i < s.size; ++i) {</pre>
                    printf("\t");
                    printPoint(s.points[i]);
                    printf("\n");
          printf("}");
                                       Tzachi (Isaac) Rosen
```

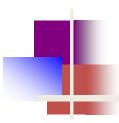


```
int dynWhereIs(Point p, DynSetOfPoints s) {
          int i;
          for (i = 0; i < s.size; ++i)</pre>
                    if (isEqual(p, s.points[i]))
                              return i:
          return -1;
int dynIsMember(Point p, DynSetOfPoints s) {
          int i = dynWhereIs(p, s);
          return (i != -1) ? 1 : 0;
void dynInsert(Point p, DynSetOfPoints *ps) {
          int i = dynWhereIs(p, *ps);
          if (i == -1) {
                    ++ps->size;
                    ps->points = (Point *) realloc(ps->points, ps->size*sizeof(Point));
                    ps->points[ps->size - 1] = p;
void dynDelete(Point p, DynSetOfPoints *ps) {
          int i = dynWhereIs(p, *ps);
          if (i != -1) {
                    --ps->size;
                    ps->points[i] = ps->points[ps->size];
                    ps->points = (Point *) realloc(ps->points, ps->size*sizeof(Point));
```



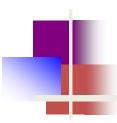
## קבוצה דינאמית של נקודות

```
void dynUni(DynSetOfPoints * ps1, DynSetOfPoints s2) {
       int i;
       for (i = 0; i < s2.size; ++i)
             dynInsert(s2.points[i], ps1);
void dynIntersect(DynSetOfPoints * ps1, DynSetOfPoints s2)
       int i:
      DynSetOfPoints s;
       dynEmptySetOfPoints(&s);
       for (i = 0; i < ps1->size; ++i)
              if (dynIsMember(ps1->points[i], s2))
                    dynInsert(ps1->points[i], &s);
       free (ps1->points);
       *ps1 = s;
```



```
dynRandomSetOfPoints(&ds1);
dynPrintSetOfPoints(ds1); // {
               (1.80, 3.04)
//
//
               (1.31, 9.30)
               (2.90, 7.19)
//
               (7.48, 2.38)
// }
printf("\n");
dynRandomSetOfPoints(&ds2);
dynPrintSetOfPoints(ds2); // {
//
               (4.19, 9.04)
//
               (3.42, 9.57)
               (9.35, 5.54)
//
               (5.69, 3.61)
//
               (6.88, 9.10)
               (2.16, 0.06)
               (2.87, 0.03)
//
               (6.17, 8.48)
printf("\n");
```

Tzachi (Isaac) Rosen



```
printf("isMember = %s\n", dynIsMember(p1, ds1) ? "true" : "false");
        // isMember = false
dynInsert(p1, &ds1);
dynPrintSetOfPoints(ds1); //
     (1.80, 3.04)
//
//
               (1.31, 9.30)
//
               (2.90, 7.19)
//
              (7.48, 2.38)
//
               (1.25, 0.50)
printf("\n");
printf("isMember = %s\n", dynIsMember(p1, ds1) ? "true" : "false");
        // isMember = true
dynDelete(ds1.points[(ds1.size - 1) / 2], &ds1);
dynPrintSetOfPoints(ds1); // {
//
              (1.80, 3.04)
//
               (1.31, 9.30)
               (1.25, 0.50)
//
//
               (7.48, 2.38)
printf("\n");
```

```
dynUni(&ds1, ds2);
dynPrintSetOfPoints(ds1); // {
                     (1.80, 3.04)
//
                     (1.31, 9.30)
//
//
                     (1.25, 0.50)
//
                     (7.48, 2.38)
//
                     (4.19, 9.04)
//
                     (3.42, 9.57)
                     (9.35, 5.54)
//
//
                     (5.69, 3.61)
//
                     (6.88, 9.10)
                     (2.16, 0.06)
//
//
                     (2.87, 0.03)
//
                     (6.17, 8.48)
//
printf("\n");
dynIntersect(&ds1, ds2);
dynPrintSetOfPoints(ds1); // {
                     (9.35, 5.54)
//
//
                     (5.69, 3.61)
//
                     (6.88, 9.10)
//
                     (2.16, 0.06)
//
                     (2.87, 0.03)
//
                     (6.17, 8.48)
//
printf("\n");
free (ds1.points);
free (ds2.points);
                             Tzachi (Isaac) Rosen
```