ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ 2

ΔΕΙΚΤΕΣ

ΤΜΗΜΑ

ΠΛΗΡΟΦΟΡΙΚΗΣ &

POINTERS

- ΔΗΛΩΣΗ ΔΕΙΚΤΗ→τύπος *όνομα_δείκτη→Π.χ int *x;
- sizeof(*)→8 byte
- Ανάθεση διεύθυνσης μεταβλητής σε δείκτη→

  int *x;

  int y=2;

  x=&y;
- Δυναμική δέσμευση μνήμης για 1 μεταβλητή(dynamic allocation of a pointer)

  int *x=(int *)malloc(1*sizeof(int));
- Δυναμική δέσμευση μνήμης για πίνακα(μονοδιάστατο)

  double *x=(double *)malloc(n*sizeof(double));

  n→ακέραια μεταβλητή που ισούται με το μέγεθος του πίνακα

  x→αναφορά σε μία θέση μνήμης.

  *χ→αποαναφοροποίηση (αναφορά στο περιεχόμενο μίας θέσης μνήμης)
- Δυναμική δέσμευση μνήμης για πίνακα(δυσδιάστατο)

  int **a;

  int a=(int **)malloc(rows * sizeof(int *));

```
for(int i=0;i<rows;i++)
{
    a[i]=(int *)malloc(columns * sizeof(int))
}
```

- Άρα δείκτης είναι μία μεταβλητή που περιέχει την διεύθυνση μίας άλλης μεταβλητής.
- Αποδέσμευση μνήμης.

  →Για δείκτη σε 1 μεταβλητή και μονοδιάστατο πίνακα.

```
free(x);
```

  →Για δυσδιάστατο πίνακα.

```
for(int i=0;i<rows;i++)
{
    free(a[i]);
}
free(a);
```

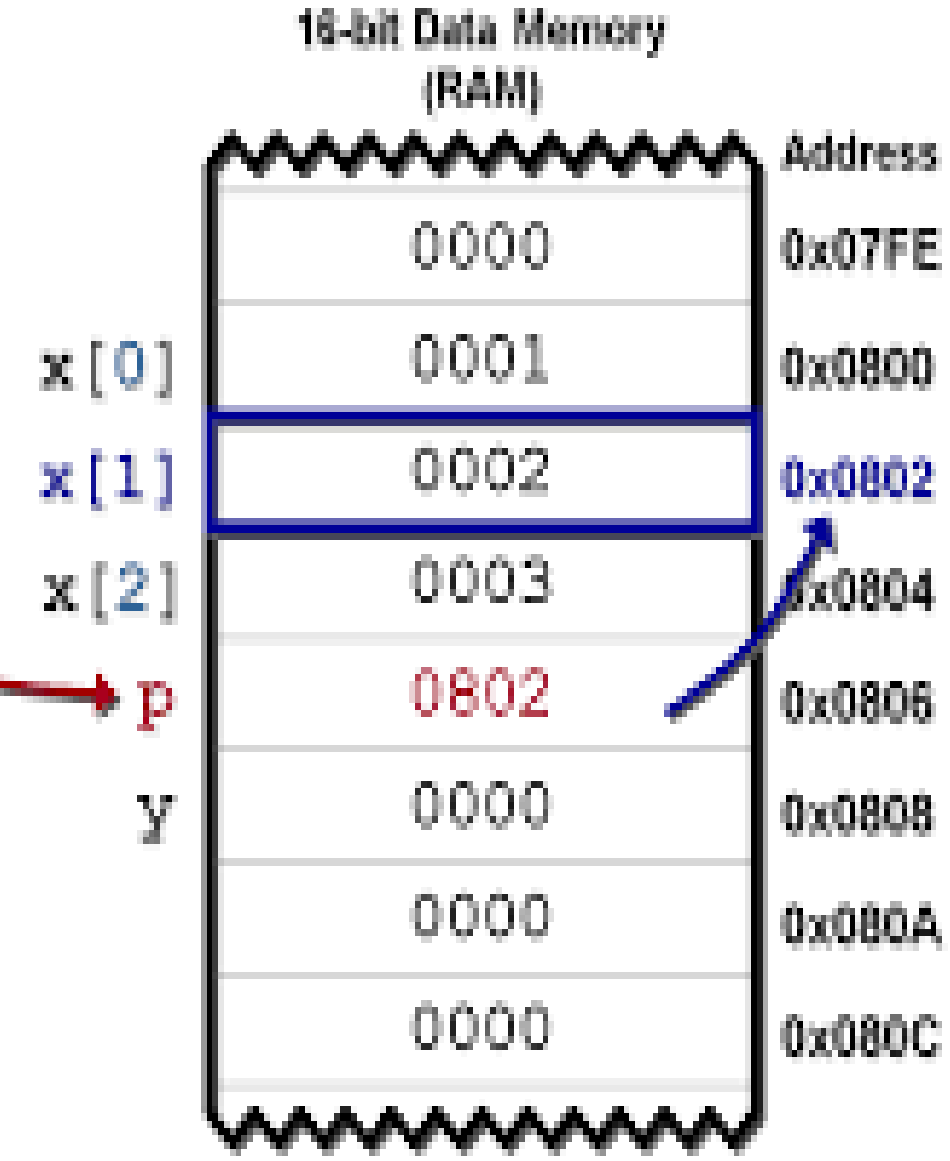
- sizeof()→μέγεθος μίας μεταβλητής=byte τύπου μεταβλητής *Αριθμός στοιχείων μεταβλητής.

  →Π.Χ int x[5];=>sizeof(x)=5*4(sizeof(int))=20 bytes

```
{
  int x[3] = {1,2,3};
  int y;
  int *p = &x;

  y = 5 + *(++p);

  y = 5 + ++(*p);
}
```



ΠΑΡΑΔΕΙΓΜΑ_1

```c
#include <stdio.h>
//Ανάθεση μίας μεταβλητής σε ένα δείκτη.
int main()
{
    int *p;//Δήλωση δείκτη
    int x=2;//Δήλωση μεταβλητής
    p=&x;//Ο δείκτης θα δείχνει στην ίδια θέση μνήμης
    //με την μεταβλητή
    //*p->πρόσβαση στο περιέχομενο της θέσης
    //μνήμης που αντιστοιχεί το p
    printf("%d | %d\n",x,*p);//Ο δείκτης *p και η μεταβλητή χ θα έχουν
    //την ίδια τιμή.
    printf("%d\n",p);//θα εκτυπώσει την θέση μνήμης που βρίσκεται ο p
    //Εξαρτάται από ποια θέση μνήμης έχει δεσμεύσει για την μεταβλητή χ
    (*p)++;//Αύξηση του περιεχόμενου του δείκτη p
    //κατά 1.
    printf("%d\n",*p);//εκτύπωση του περιεχόμενου του δείκτη p.
    printf("%d\n",++*p);//Αύξηση του περιεχόμενου του δείκτη p
    //κατά 1 και στην συνέχεια εκτύπωση του.
}
```

ΠΑΡΑΔΕΙΓΜΑ_2

```c
#include <stdio.h>
void find_max_min(int *x,int size,int *max,int *min)
{
    *max=x[0];
    *min=x[0];
    for(int i=1;i<size;i++)
    {
       if(x[i]>*max)   *max=x[i];
       if(x[i]<*min)   *min=x[i];
    }
}
int main()
{
    int a[]={5,8,9,2,4,1,7,10,6};
    int s=sizeof(a)/sizeof(a[0]);//μέγεθος πίνακα
    //sizeof(a)=9*4(bytes for int)
    //sizeof(a[0]=>1 integer)=4
    int min,max;//Δήλωση 2 μεταβλητών
    //a->Όνομα πίνακα που είναι δείκτης
    //στο 1ο στοιχείο του πίνακα και έφοσον ο πίνακας
    //είναι δείκτης που δεσμεύει συνέχομενες θέσεις μνήμης
    //θα προσπελάσει και τις υπόλοιπες.
    find_max_min(a,s,&max,&min);
    //Περνάω σαν όρισμα την διεύθυνση μνήμης που δείχνει η μεταβλητή
    //max ώστε και ο δείκτης στην συνάρτηση να δείξει στην ίδια θέση
    //μνήμης,αντίστοιχα πράττω για την min
    printf("max=%d\n",max);//10
    printf("min=%d\n",min);//1
}
```

ΠΑΡΑΔΕΙΓΜΑ_3

```c
#include <stdio.h>
#include <stdlib.h>//malloc()!!!
#define s 5
void read_array(int *a)
{
    //διάβασμα τιμών
    for(int i=0;i<s;i++)
    {
        printf("give a[%d]:",i);
        scanf("%d",&a[i]);
    }
}
void print_array(int *a)
{
    //εκτύπωση τιμών
    for(int i=0;i<s;i++)
    {
        printf("a[%d]:%d\n",i,a[i]);
    }
}
int main()
{
    int *a=(int *)malloc(s*sizeof(int));
    //Δυναμική δέσμευση συνεχόμενων θέσεων μνήμης
    //για έναν δείκτη
    read_array(a);
    print_array(a);
}
```

ΠΑΡΑΔΕΙΓΜΑ_4

```c
#include <stdio.h>
#include <stdlib.h>
#define r 3
#define c 3
void read_array(int **a)
{
    for(int i=0;i<r;i++)
    {
        for(int j=0;j<c;j++)
        {
            printf("Give a[%d][%d]:",i,j);
            scanf("%d",&a[i][j]);
        }
    }
}
void print_array(int **a)
{
    for(int i=0;i<r;i++)
    {
        for(int j=0;j<c;j++)
        {
            printf("a[%d][%d]:%d\n",i,j,a[i][j]);
        }
    }
}
```

```c
int main()
{
    int **pin;//Δήλωση διπλού δείκτη
    pin=(int **)malloc(r * sizeof(int *));
    //Δέσμευση μνήμης για r μεταβλητές που
    //το περιεχόμενο του θα είναι διεύθυνση μνήμης
    //Πρακτικά δέσμευση r δεικτών
    for(int i=0;i<r;i++)
    {
        pin[i]=(int *)malloc(c * sizeof(int));
        //Δέσμευση μνήμης για καθέναν από τους δείκτες
    }
    read_array(pin);
    print_array(pin);
    for(int i=0;i<r;i++)
    {
        free(pin[i]);
        //Απελευθέρωση μνήμης για τους 5 δείκτες
        //που έφτιαξα για κάθε γραμμή
    }
    free(pin);
    //απελευθέρωση  όλου του δείκτη
}
```

ΑΣΚΗΣΕΙΣ

1.Να κατασκευαστεί μία συνάρτηση που να δέχεται σαν όρισμα 2 μεταβλητές και να αντιμεταθέτει(swap) τις τιμές τους.Στην κύρια συνάρτηση ο χρήστης θα διαβάζει 2 ακέραιες και 2 δεκαδικές τιμές,θα τις αντιμεταθέτει και θα εμφανίζει τα αποτελέσμα τους

2.Να κατασκευαστεί συνάρτηση που θα δέχεται σαν όρισμα 1 δείκτη σε πίνακα και θα υπολογίζει το μέγιστο,το ελάχιστο και τον μέσο όρο του πίνακα.Στην κύρια συνάρτηση να κατασκευαστεί ένας πίνακας 5 θέσεων,στον οποίο τιμές θα εισάγει ο χρήστης,να κληθεί η συνάρτηση και να εμφανιστούν τα αποτελέσματα της.

3. Να κατασκευαστεί συνάρτηση που θα δέχεται σαν όρισμα 1 δείκτη σε πίνακα και θα υπολογίζει την θέση του μέγιστου στοιχείου,και το άθροισμα του πίνακα.Στην κύρια συνάρτηση ο χρήστης να δεσμεύει δυναμικά 5 θέσεις μνήμης για τον δείκτη και να εισάγει τιμές για το περιεχόμενο τους.Έπειτα να καλεί την συνάρτηση και να εμφανίζει τα αποτελέσματα της.

4. Να κατασκευαστεί συνάρτηση που θα δέχεται σαν όρισμα 1 δείκτη σε πίνακα και θα υπολογίζει το μέγιστο γραμμής,το ελάχιστο γραμμης και τον μέσο όρο του πίνακα.Στην κύρια συνάρτηση να δεσμευθεί μνήμη δυναμικα για ένα δυσδιάστατο πίνακα δεκαδικών,να εισαχθούν οι τιμές από τον χρήστη,να κλήθει η συνάρτηση και να εμφανιστούν τα αποτελέσματα.Τέλος να αποδεσμευτεί η μνήμη που δεσμεύτηκε για τον πίνακα.