Εργαστήριο
Εισαγωγή στον Προγραμματισμό

Εργαστήριο 06

Πίνακες και δείκτες

Βασιλόπουλος Διονύσης

Ε.ΔΙ.Π. Τμήματος Πληροφορικής & Τηλεπικοινωνιών

ΤΜΗΜΑ
ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

6ο Εργαστήριο

define vs constant vs static

#define: Είναι οδηγία στον προεπεξεργαστή. Ότι γράφουμε στο define αντικαθίσταται στον κώδικα πριν το compile.

```
#define size 8
int main(){
int a;
a=size;
}
```

πριν το compile

```
#define size 8
int main(){
int a;
a=8;
}
```

const: Είναι μεταβλητή, δεσμεύει χώρο στη μνήμη και έχει συγκεκριμένη διεύθυνση. Η τιμή ΔΕΝ αλλάζει

static: Είναι μεταβλητή, με συγκεκριμένες ιδιότητες. Σταματόπουλος σελίδες 66-68.

https://stackoverflow.com/questions/6442328/what-is-the-difference-between-define-and-const

6ο Εργαστήριο

Κλήση συνάρτησης – By value

```
#include <stdio.h>
void badf(int x, int y, int sum, int diff) {
sum = x + y;
diff = x - y;
printf("The sum a+b is: %d, the difference a-b is: %d\n", sum, diff);
}
int main() {
int sum=0, diff=0;
badf(2, 3, sum, diff);  //Call by value
printf("The sum a+b is: %d, the difference a-b is: %d\n", sum, diff);
return 0;
}
```

Οι τιμές των μεταβλητών

Οι τιμές των μεταβλητών δεν αλλάζουν μετά την κλήση της συνάρτησης

6ο Εργαστήριο

Κλήση συνάρτησης – By pointer

```
#include <stdio.h>
void goodf(int x, int y, int *sum, int *diff) {
*sum = x + y;
*diff = x - y;
printf("The sum a+b is: %d, the difference a-b is: %d\n", sum, diff);
}
int main() {
int sum=0, diff=0;
badf(2, 3, &sum, &diff);  //Call by pointer
printf("The sum a+b is: %d, the difference a-b is: %d\n", sum, diff);
return 0;
}
```

Η διεύθυνση μνήμης των μεταβλητών

Οι τιμές των μεταβλητών αλλάζουν μετά την κλήση της συνάρτησης

6ο Εργαστήριο

Δήλωση array

```
int arr[5];                 --Άμεση δήλωση μεγέθους
int arr[5] = {1, 2, 3};     --Άμεση δήλωση μεγέθους, μερική αρχικοποίηση

int arr[] = {1, 2, 3, 4, 5};    --Έμμεση δήλωση μεγέθους

int* arr = (int*)malloc(5 * sizeof(int)); --Ως pointer με δήλωση μεγέθους
```

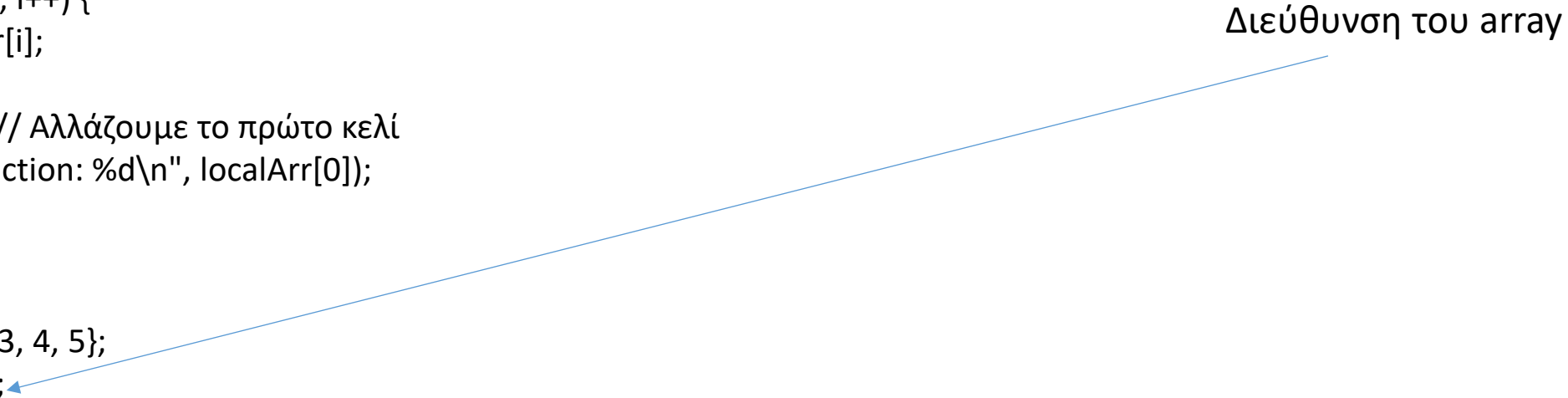
6ο Εργαστήριο

array ως παράμετρος σε συνάρτηση (by value)

```
#include <stdio.h>

void passByValue(int arr[5]) { // Γνωρίζουμε το μέγεθος
    int localArr[5];
    for (int i = 0; i < 5; i++) {
        localArr[i] = arr[i];
    }
    localArr[0] = 42; // Αλλάζουμε το πρώτο κελί
    printf("Inside function: %d\n", localArr[0]);
}

int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    passByValue(arr);
    printf("In main: %d\n", arr[0]); // Το array διατηρεί την αρχική του τιμή
    return 0;
}
```



6ο Εργαστήριο

array ως παράμετρος σε συνάρτηση (by reference) (1/2)

```
#include <stdio.h>

void passByReference(int arr[], int size) {

for (int i = 0; i < size; i++) {
       arr[i]  ++;
   }
}

int main() {
   int arr[5]
   ……
   passByValue(arr, 5);
   ……
   return 0;
}
```

Διεύθυνση του array

Οι τιμές του array αλλάζουν

6ο Εργαστήριο

array ως παράμετρος σε συνάρτηση (by reference) (2/2)

```
#include <stdio.h>

void passByReference(int* arr, int size) {

for (int i = 0; i < size; i++) {
      *(arr+i) ++;
   }
}

int main() {
   int arr[5]
   ……
   passByValue(arr, 5);
   ……
   return 0;
}
```

Διεύθυνση του array

Οι τιμές του array αλλάζουν