Διάλεξη 14η: Δυναμική Διαχείρηση Μνήμης, μέρος 2ο

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

Εισαγωγή στην Επιστήμη Υπολογιστών

Βασίζεται σε διαφάνειες του Κ. Παναγιωτάκη

Δείκτες σε δείκτες

- Ένας δείκτης μπορεί να δείχνει σε οποιοδήποτε τύπο δεδομένων
  - Ακόμη και δείκτη
  - κ.ο.κ.

```
char ch;
char *pch;
char **ppch;
char ***pppch;
```

- Η ch περιέχει ένα χαρακτήρα
- Η pch περιέχει δείκτη σε χαρακτήρα
  - Ισοδύναμο με πίνακα χαρακτήρων με άγνωστο μέγεθος
- Η ppch περιέχει δείκτη σε πίνακες χαρακτήρων
  - Ισοδύναμο με πίνακα με άγνωστο μέγεθος από πίνακες χαρακτήρων άγνωστου μεγέθους, ή πίνακα με strings
- Η pppch περιέχει δείκτη σε πίνακα από πίνακες με strings
- κ.λ.π.

Συνοπτικά

- Με τις μεταβλητές:

```
char ch;
char *pch = &ch;
char **ppch = &pch;
char ***pppch = &ppch;
```

- Η εντολή `*pch = 'a'` γράφει `'a'` στο `ch`
- Η εντολή `**ppch = 'a'` γράφει `'a'` στο `ch`
- Η εντολή `**pppch = 'a'` είναι λάθος

Παράδειγμα

ptptr.c

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

int main(void) {
  int lines = 5;
  int cols = 10;
  int i, **arr;

  arr = (int **) malloc(lines * sizeof(int *));
  if(arr == NULL) {
    printf("Cannot allocate row pointers.\n");
    exit(0);
  }

  for(i = 0; i < lines; i++) {
    arr[i] = (int *) malloc(cols * sizeof(int));
    if(arr[i] == NULL) {
      printf("Cannot allocate row[%d]\n", i);
      exit(0);
    }
    printf("%d %p %ld", i,
           (void *)arr[i],
           (long)arr[i]);
    if(i > 0) {
      printf(" %d\n",
             (int)(arr[i] - arr[i - 1]));
    } else {
      printf("\n");
    }
  }
  /* free the memory */
  for (i = 0; i < lines; i++) {
    free (arr[i]);
  }
  free(arr);
  return 0;
}
```

Δείκτες και πίνακες

- Ομοιότητες και διαφορές μεταξύ των δεικτών σε δείκτες και τους δισδιάστατους πίνακες
- Δισδιάστατοι πίνακες
  - `int a[10][20];`
  - Η έκφραση `a[i][j]` είναι συντακτικά σωστή
  - Ο πίνακας δεσμεύει 10×20 ακέραιους στη μνήμη
  - Η διεύθυνση του `a[i][j]` είναι `a + i*20 + j`, ή
  - η διεύθυνση του `a[i][j]` είναι `(char*)a + (i * 20 + j) * sizeof(int)`

Δείκτες και πίνακες (2)

- Δείκτες σε δείκτες
  - `int **b;`
  - Η έκφραση `b[i][j]` είναι συντακτικά σωστή
  - Για δέσμευση 10 δεικτών χρειάζεται

```
b = (int**)malloc(10 * sizeof(int*))
```

  - Για δέσμευση μνήμης για 20 ακέραιους σε κάθε “γραμμή” χρειάζεται

```
for(i = 0; i < 10; i++) {
  b[i] = (int*)malloc(20 * sizeof(int))
}
```

  - Η διεύθυνση του `b[i][j]` είναι `b[i] + j`, ή
  - η διεύθυνση του `b[i][j]` είναι `(char*)b[i] + j * sizeof(int)`, ή
  - Πριν τη χρήση του `b` ως πίνακα πρέπει να έχει γίνει *σωστή αρχικοποίηση* για τους δείκτες `b[i]` και *δέσμευση μνήμης*