ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ & ΥΛΙΚΟΥ
ΕΡΓΑΣΤΗΡΙΑΚΕΣ ΑΣΚΗΣΕΙΣ ΓΙΑ ΤΟ ΜΑΘΗΜΑ:
ΗΡΥ 201 - ΨΗΦΙΑΚΟΙ ΥΠΟΛΟΓΙΣΤΕΣ

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2018-2019

Άσκηση 1: Η γλώσσα C και η μνήμη.

Σκοπός Άσκησης
Η συμπεριφορά της γλώσσας C σε συγκεκριμένο υλικό. Εξοικείωση με δεκαεξαδική αναπαράσταση στη C.

Προετοιμασία
Μελέτη της C, διαβάστε τα εγχειρίδια.

A) Δείτε πως διατάσσονται τα στοιχεία ενός πίνακα στη μνήμη. Εκτελέστε το ακόλουθο πρόγραμμα:

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

int var1 = 31;
int var2 = -2;

int main(void)
{
          int A[10], i;       /* A = array of 10 ints, i = scalar int variable */
          int * p;            /* p is a scalar variable that points to an int */

          for(i = 0; i < 10; i++) {
                    A[i] = i;
                    printf("Element A[%d] = %d is stored in address : %#010x\n", i, A[i], &A[i]);
          }

          p = &var1;
          printf("Var addresses (hex): %#010x, %#010x\n&p=%#010x, p=%#010x, *p=%d\n",
                    &var1, &var2, &p, p, *p);
          printf("Var values 1: var1=%5d (%#010x), var2=%5d (%#010x)\n", var1, var1, var2, var2);
          *p = 0xffff;
          printf("Var values 2: var1=%5d (%#010x), var2=%5d (%#010x)\n", var1, var1, var2, var2);
          *(p+1) = 1500;
          printf("Var values 3: var1=%5d (%#010x), var2=%5d (%#010x)\n", var1, var1, var2, var2);
          return 0;
}
```

Τι παρατηρείτε;

Το όνομα A είναι μια σταθερά για την C. Έτσι δεν μπορούμε να την αλλάξουμε γράφοντας A++. Μπορούμε όμως να δούμε την τιμή της. Αλλάξτε το παραπάνω πρόγραμμα και τυπώστε σε δεκαεξαδικό (%x στην printf) και δεκαδικό τις ακόλουθες εκφράσεις: A, A+1, (int)A + 1, &A[1]. Πώς συγκρίνονται τα A+1 και (int)A + 1 και γιατί; Τυπώστε επίσης το μέγεθος του πίνακα καθώς και το μέγεθος ενός στοιχείου του πίνακα χρησιμοποιώντας τη `sizeof()`.

Για το καθένα από τα παρακάτω, φτιάξτε ένα μικρό πρόγραμμα και εκτελέστε το για να πάρετε την απάντηση.

B) Πώς τοποθετούνται τα δεδομένα στη μνήμη: δηλώστε τρεις μεταβλητές ενός τύπου της αρεσκείας σας. Τυπώστε σε δεκαδικό τις διευθύνσεις τους χρησιμοποιώντας το & (π.χ. &i). Τι παρατηρείτε;

C) Δηλώστε δύο struct. Το πρώτο έχει έναν χαρακτήρα X, ένα ακέραιο C και άλλον ένα χαρακτήρα Y. Το δεύτερο έχει δύο χαρακτήρες X και Y, και ένα ακέραιο C. Δηλώστε τα πεδία με τη σειρά που σας δίνεται.

Το χρήσιμο μέγεθος στις δύο περιπτώσεις είναι 6 bytes, 4 για τον ακέραιο, και από ένα για κάθε χαρακτήρα. Τυπώστε το μέγεθος του κάθε struct με την sizeof (παρατήρηση: δε χρειάζεται να δηλώσετε μεταβλητή για να χρησιμοποιήσετε τη `sizeof`). Τι παρατηρείτε και γιατί πιστεύετε ότι συμβαίνει αυτό;

D) Η δυναμική δέσμευση μνήμης είναι απαραίτητη σε σύνθετες εφαρμογές και δομές δεδομένων. Κάνετε τέσσερα malloc με χώρο: 1, 10, 16, και 32 bytes. Τυπώστε σε δεκαδικό τις διευθύνσεις που επιστράφηκαν. Τι παρατηρείτε; (υπόδειξη: δείτε τη σχετική θέση και απόσταση των διαδοχικών malloc σε σχέση με τον ζητούμενο χώρο).

E) Χάρτης χρήσης μνήμης: Τυπώστε σε δεκαεξαδικό (α) την main (όπως και ο πίνακας A, η main είναι μια σταθερή για την C, με τιμή τη διεύθυνση του κώδικα της main στην μνήμη), (β) τη διεύθυνση μιας καθολικής μεταβλητής, (γ) τη διεύθυνση μιας τοπικής μεταβλητής, και (δ) τη διεύθυνση που επιστρέφει ένα malloc. Ποιές από αυτές είναι «κοντά» και ποιές είναι «μακρυά»; Σχεδιάστε ένα «χάρτη» της μνήμης, δηλαδή ένα πίνακα πλάτους 1 λέξης και δείξτε καθ' ύψος την τοποθέτηση των μεταβλητών σας.

Παραδοτέα

(α) Κώδικες για τα ανωτέρω

(β) Μικρό αλλά περιεκτικό κείμενο με την ανάλυσή και τα συμπεράσματά σας (δείτε το πρότυπο αναφορών)