# Εργαστήριο 1

## Χειμερινό Εξάμηνο 2015-2016

## Στόχοι του εργαστηρίου

- Εντολές εισόδου-εξόδου
- Χρήση συνθηκών σε δομές επιλογής
- Χρήση συνθηκών σε δομές επανάληψης

## Εντολές εισόδου-εξόδου

### Εντολές για είσοδο ακεραίων αριθμών από την κονσόλα:

```
li $v0, 5                 # κωδικός 5 από τα syscall στον $v0
syscall
move $t0, $v0             # καταχώριση της τιμής που εισάγαμε, στον
                          # register $t0
```

### Εντολές για εμφάνιση ακεραίου στην κονσόλα:

```
li $v0, 1                 # pseudo-instruction για addi $v0, $0, 1
                          # κωδικός 1 από τα syscall στον $v0
```

```
move $a0, $t0                    # ο $t0 έχει την τιμή που θέλουμε να     # εμφανίσουμε και
                                 το φορτώνουμε στον $a0
                                 # σαν παράμετρο
syscall
```

## Εντολές για εμφάνιση μηνυμάτων στην κονσόλα:

Για να τυπώσετε ένα string στην οθόνη θα πρέπει να το δηλώσετε πρώτα στον τμήμα .data του προγράμματός σας ως εξής:

Αρχικά πρέπει να ορίσετε το string στην περιοχή της μνήμης με ένα label (στην περίπτωση του παραδείγματος String_name1) για να μπορείτε να έχετε πρόσβαση με χρήση της διεύθυνσης του.

```
.data
String_name1:  .asciiz   "msg"
```

Σε περίπτωση που θέλετε να αλλάξετε γραμμή (για να είναι πιο ευανάγνωστη η εκτέλεση) πρέπει να ορίσετε ένα string αλλαγής γραμμής.

```
String_name2:  .asciiz   "\n"
```

Για να τυπώσετε το string στην κονσόλα πρέπει να γράψετε τις εξής εντολές.

```
li  $v0, 4                       # κωδικός 4 από τα syscall στον $v0
la  $a0, String_name1    # φορτώνουμε στον $a0 την διεύθυνση
                                 # του String που θα εκτυπώσουμε
syscall
```

Οι εντολές syscall είναι κλήσεις λειτουργιών του συστήματος. Με αυτές μπορείτε εκτός από I/O λειτουργίες να τερματίσετε ένα πρόγραμμα (syscall με παράμετρο $v0=10) ή να ζητήσετε δυναμικά μνήμη (αντίστοιχη εντολή malloc της C). Μπορείτε να βρείτε όλες τις λειτουργίες στο Help menu του MARS.

# Υλοποίηση δομών ελέγχου ροής προγράμματος

Εντολές ελέγχου ροής είναι το σύνολο των εντολών το οποίο αλλάζουν την ροή εκτέλεσης προγράμματος. Κανονικά η σειρά εκτέλεσης των εντολών είναι μετά από κάθε εντολή να εκτελείται η αμέσως από κάτω. Υπάρχουν όμως εντολές οι οποίες αλλάζουν είτε στατικά (χωρίς συνθήκη) είτε δυναμικά (με συνθήκη) την ροή εκτέλεσης των εντολών.

Παράδειγμα ελέγχου ροής, μετατροπή από C σε assembly:

## Παράδειγμα 1 - if

```
if( x = = 0){
```

```
        …       // Case true
}
else
{
        …       // Case false.
}
…               // Next instruction
```

Αντιστοιχίζουμε την μεταβλητή x στον καταχωρητή $t3

```
bnez $t3,else
…                       # case true
j endif

else:
…                       # case false

endif:
…                       # next instruction
```

## Παράδειγμα 2 - for

```
for( i = 0 ; i < 10 ; i++ )
{
        …               // code
}
…                       // next instruction
```

Αντιστοιχίζουμε την μεταβλητή i στον καταχωρητή $t3 και αποθηκεύουμε το όριο του loop (εδώ 10) στον καταχωρητή $t4.

```
li $t3,0        # add $t3, $0, $0
li $t4, 10       # addi $t4,$0,10

for:
bge $t3,$t4,endfor

…               # code

addi $t3,$t3,1
j for

endfor:
…               # next instruction
```

## Παράδειγμα 3 - multiple if

```
if( x > 0 && x < 10)
{
        …       // code
```

```
}
…                   // next instruction
```

Αντιστοιχίζουμε την μεταβλητή x στον καταχωρητή $t3 και αποθηκεύουμε την σταθερά με την οποία θα συγκρίνουμε (εδώ 10) στον καταχωρητή $t4.

```
li  $t4,10
blez $t3, endif
bge $t3,$t4, endif

…       # code

endif :
…       # next instruction
```

## β’ τρόπος:

```
addi $t4,$0,10
bgtz $t3,cond2          # if(x > 0)
j endif
cond2:
blt $t3,$t4,is_true     # if(x < 10)
j endif
is_true:

…       #code

endif:
…       # next instruction
```

Το πλήθος των εντολών είναι σχετικά μεγάλο αλλά η λειτουργία τους είναι πολύ απλή. Συνιστάται κατά τον προγραμματισμό σε assembly να κάνετε χρήση του instruction set. Μπορείτε επίσης να χρησιμοποιήσετε και ψευδο-εντολές σαν να ήταν κανονικές εντολές (για παράδειγμα *li, la, bgez, blt* κοκ). Η αποστήθιση όλων των εντολών δεν είναι ζητούμενο σε αυτό το εργαστήριο ωστόσο πρέπει να είσθε σε θέση να μπορείτε να χρησιμοποιήσετε τις εντολές που αναγράφονται σε αυτό το αρχείο.

Για να πάρετε άριστα στην εξέταση, η λύση σας θα πρέπει να είναι σωστή αλλά και αποδοτική. Θα πρέπει να χρησιμοποιείτε τον ελάχιστο δυνατό αριθμό εντολών στην λύση σας, αλλά και η λύση σας να είναι ευανάγνωστη και καλά σχολιασμένη.

***Για το επόμενο εργαστήριο έχετε να υλοποιήσετε τις παρακάτω εργαστηριακές ασκήσεις. Την ώρα του εργαστηρίου θα εξετασθείτε προφορικά πάνω στους κώδικες που θα παραδώσετε.***

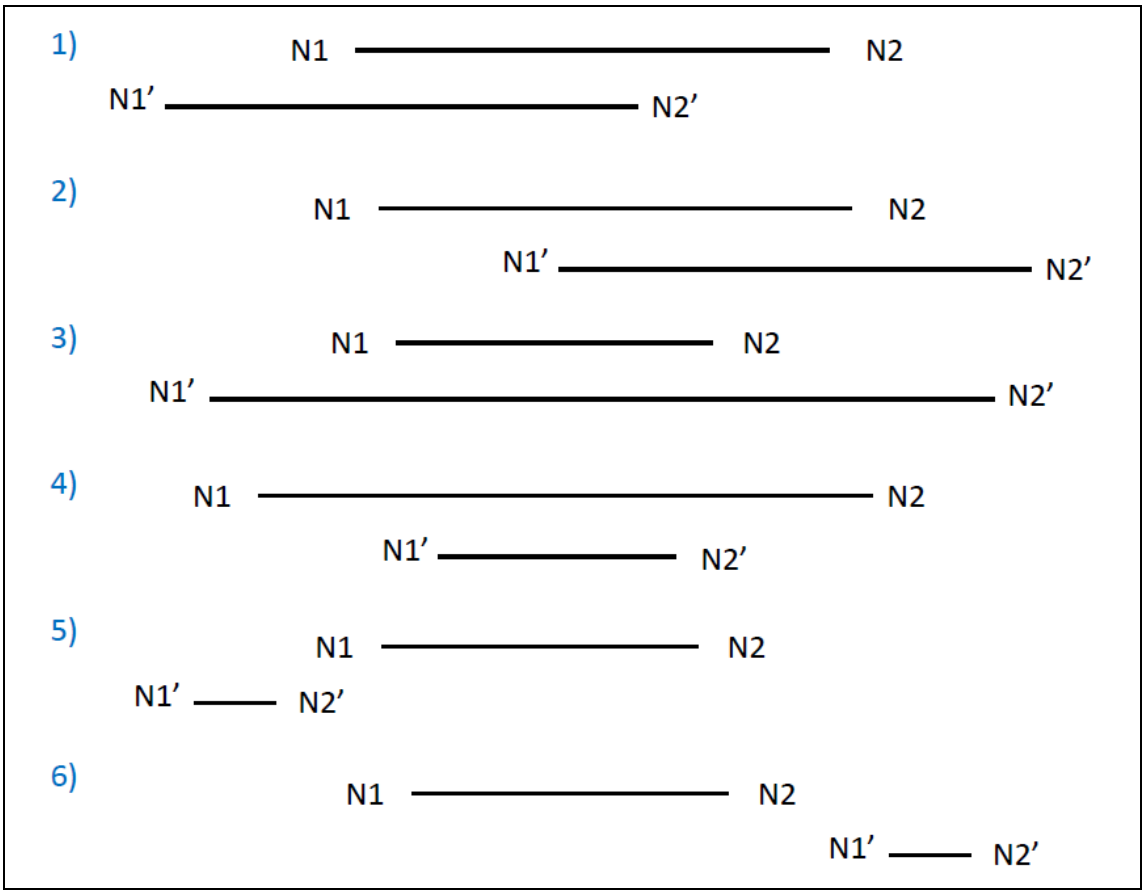
## Άσκηση 1 – Count Leading Zeros (4 μονάδες)

Να υλοποιήσετε την εντολή *clz* (count leading zeros) σε MIPS assembly. Η εντολή *clz* μετράει τον αριθμό των 0 στα most significanct bits ενός καταχωρητή 32-bits. Το πρόγραμμά σας θα διαβάζει έναν ακέραιο αριθμό *num* από την κονσόλα, θα τον τοποθετεί σε κάποιον καταχωρητή, και θα μετράει τον αριθμό των leading zeros. Για παράδειγμα, το πρόγραμμά

σας θα πρέπει να γράφει στην κονσόλα *"Number 45 has 26 leading zeros"* εάν ο αριθμός που δώσατε είναι ο *num* = 45 = $(000\ldots00101101)_2$.

## Άσκηση 2-Εύρεση επικαλυπτόμενων διαστημάτων (6 μονάδες)

Να υλοποιήσετε σε MIPS assembly ένα πρόγραμμα το οποίο θα ζητά από τον χρήστη να εισάγει επαναληπτικά δυο ακέραιους αριθμούς N1 και N2, οι οποίοι αντιπροσωπεύουν ένα διάστημα [N1, N2]. Αν ο χρήστης εισάγει αρνητικό N1 < 0 ή αρνητικό N2< 0 το πρόγραμμα τερματίζει.
Το πρόγραμμα θα πρέπει να βρίσκει το μέγιστο διάστημα που προκύπτει είτε από αλληλοεπικάλυψη διαστημάτων είτε όχι. Για την εύρεση του μέγιστου διαστήματος κάθε φορά, θα πρέπει να ελέγχονται έξη περιπτώσεις που απεικονίζονται στην Εικόνα. Το πρόγραμμα θα πρέπει να κρατάει σε κάθε επανάληψη το αριστερό άκρο και το δεξί άκρο του τρέχοντος μέγιστου διαστήματος (min και max). Στην τελευταία επανάληψη, τα min και max περιέχουν το τελικό μέγιστο διάστημα.
Πιο συγκεκριμένα, ο χρήστης, στην επανάληψη 0, εισάγει ένα διάστημα [N1, N2], οπότε στην αρχή min = N1 και max = N2. Στην επανάληψη 1, ο χρήστης εισάγει το επόμενο διάστημα [N1, N2] και τα min, max ανανεώνονται ανάλογα με τον έλεγχο των περιπτώσεων της Εικόνας μεταξύ του αποτελέσματος της προηγούμενης επανάληψης και του εισαχθέντος διαστήματος κοκ. Αν δυο διαστήματα δεν αλληλεπικαλύπτονται (περιπτώσεις 5,6 της Εικόνας), τότε το αποτέλεσμα θα πρέπει να είναι το μέγιστο διάστημα από αυτά τα δύο διαστήματα. Στο τέλος του προγράμματος εκτυπώνεται το τελικό μέγιστο διάστημα.
Για να βρεθεί το μέγιστο των διάστημα, υπάρχουν έξη περιπτώσεις αλληλοεπικάλυψης διαστημάτων:



**Περίπτωση 1,2**: Μερική επικάλυψη με το αποτέλεσμα της τρέχουσας επανάληψης.
**Περίπτωση 3,4:** Ολική επικάλυψη μεταξύ της τρέχουσας και των προηγούμενων επαναλήψεων.
**Περίπτωση 5,6:** Καμία επικάλυψη μεταξύ της τρέχουσας και των προηγούμενων επαναλήψεων.

## Παράδειγμα Εκτέλεσης:

Iteration 0
Please give N1:
**** user input : 1
Please give N2:
**** user input : 3

Iteration 1
Please give N1:
**** user input :0
Please give N2:
**** user input :2

Iteration 2
Please give N1:
**** user input :1
Please give N2:
**** user input :3

Iteration 3
Please give N1:
**** user input :1
Please give N2:
**** user input :4

Iteration 4
Please give N1:
**** user input :5
Please give N2:
**** user input :6

Iteration 6
Please give N1:
**** user input :5
Please give N2:
**** user input :10

Iteration 5
Please give N1:
**** user input :-1

The max final union of ranges is [5,10].
-- program is finished running --

Θα πρέπει να στέλνετε με email τις λύσεις των εργαστηριακών ασκήσεων σας στους διδάσκοντες στο uth_ece232@gmail.com.

Το email σας θα πρέπει να περιέχει ως attachment **ένα zip file** με τον κώδικα σας.

Κάθε διαφορετική άσκηση στην εκφώνηση θα βρίσκεται και σε διαφορετικό asm file. **Το όνομα των asm files θα ΠΡΕΠΕΙ να αρχίζει με το ΑΕΜ σας.**
*Για παράδειγμα*, το lab1.zip θα περιέχει 2 asm files, ένα για κάθε μία από τις ασκήσεις του lab1, με ονόματα 9999_lab1a.asm, 9999_lab1b.asm για τον φοιτητή με ΑΕΜ 9999.

Το email σας θα έχει Subject: CE232, lab N (N ο αριθμός του lab, N=1, 2, …). Το email σας θα έχει body: το όνομα σας και το ΑΕΜ σας.

Θα πρέπει να στέλνετε το email σας πριν βγείτε από την εξέταση του εργαστηρίου**.**