Παραδείγματα Assembly Λύσεις 2ου Διαγωνίσματος και 2ης Άσκησης (Εύρεσης Πρώτου Αριθμού)

(Εργαστήριο 4)

Άσκηση 1 (α)

Γράψτε ένα πρόγραμμα (4 εντολών) με το οποίο μπορείτε να προσθέσετε το περιεχόμενο των θέσεων μνήμης 0X30000000 και 0X30000001. Το αποτέλεσμα να αποθηκευτεί ως λέξη των 32 μπιτς στις θέσεις μνήμης 0X30000004 – 0X30000007

A.

1. **lb $t1, 0x30000000**
2. **lb $t2, 0x30000001**
3. **add $t3, $t2, $t1**
4. **sw $t3, 0x30000004**

B.

1. **li $t0 0x30000000**
2. **lb $t1, ($t0)**
3. **lb $t2, 1($t0)**
4. **add $t3, $t1, $t2**
5. **sw $t3, 4($t0)**

Άσκηση 1 (β)

Εξηγήστε την εντολή div $t5, $t4 εάν τα περιεχόμενα των καταχωρητών πριν την εκτέλεση της είναι ως ακολούθως: $t4 = 00000005 και $t5 = 0000002a. Ποιο θα είναι το αποτέλεσμα και που θα αποθηκευτεί; (Δώστε συγκεκριμένες τιμές.)

Εκτελεί την διαίρεση **$t5/$t4 = 42/5**

Αποτέλεσμα = 8 αποθηκεύεται στον καταχωρητή Lo

Υπόλοιπο = 2 αποθηκεύεται στον καταχωρητή Hi

Άσκηση 1 (γ)

Γράψτε ένα σύντομο πρόγραμμα το οποίο θα ελέγχει εάν ο χρήστης έχει πληκτρολογήσει τον σωστό αριθμό ορισμάτων που είναι 3. Αν δεν είναι, να συνεχίζει στο label "arg_error"

1. **li $to, 3**
2. **bne $a0, $t0, arg_error**

Άσκηση 2 (α)

Γράψτε ένα πρόγραμμα (3 εντολών) το οποίο να διαιρεί το περιεχόμενο του $t5 με το περιεχόμενο του $t6 και να τοποθετεί το αποτέλεσμα στον $t8 και το υπόλοιπο στον $t9.

A.

1. **div $t5, $t6**
2. **mflo $t8**
3. **mfhi $t9**

B.

1. **div $t8, $t5, $t6**
2. **mfhi $t9**

Άσκηση 2 (β)

Δώστε το περιεχόμενο των θέσεων στη μνήμη όπου θα αποθηκευτεί το δεύτερο όρισμα της εντολής «spim –file sqrt.s 225» εάν η αρχή του 2ου ορίσματος είναι αποθηκευμένη στη διεύθυνση μνήμης 0X60000009.

0X60000009 : 0x32 # ascii of key 2 =(00110010)
0X6000000a : 0x32 # ascii of key 2 =(00110010)
0X6000000b : 0x35 # ascii of key 5 =(00110101)

Άσκηση 2 (γ)

Εξηγήστε ποιοι καταχωρητές θα αλλάξουν τιμή άμεσα μετά την εντολή «spim –file sqrt.s 225». Δώστε τη νέα τους τιμή εάν την γνωρίζετε.

$a0 = 2

$a1 = διεύθυνση μνήμης όπου αποθηκεύτηκε το όνομα του αρχείου (η πρώτη από τις 4).

Άσκηση 3 (α)

Δίνονται οι ακόλουθες εντολές:

```
.data
input:   .byte 20, 25, 30
.text
main:  la $t6, input
```

Περιγράψτε τι θα αποθηκευτεί στον $t6 μετά την εκτέλεση των πιο πάνω εντολών.

Ο $t6 θα περιέχει την διεύθυνση μνήμης του αριθμού 20 (ο οποίος θα είναι αποθηκευμένος ως 00010100)

Άσκηση 3 (β)

Δίνονται οι ακόλουθες εντολές:

```
.data
input:  .byte 20, 25, 30
.text
main:  la $t6, input
```

Συμπληρώστε την κενή γραμμή ώστε ο $t5 να πάρει το 3ο byte του input (δηλ. το 30).

lb $t5, 2($t6) ή lb $t5, input+2

Άσκηση 3 (γ)

Δίνονται οι ακόλουθες εντολές:

```
.data
input:   .byte 32, 10, 15, 6, 20, 22
.text
main:  la $t6, input
lw $t4, 1($t6)
```

Δώστε τα περιεχόμενα του $t4 (σε δεκαεξαδική ή δυαδική μορφή) μετά την εκτέλεση των πιο πάνω εντολών.

$t4 = 14 06 0f 0a

(10_{10}=0a$_{hex}$, 15_{10}=0f$_{hex}$, 6_{10} = 06$_{hex}$, 20_{10}=14$_{hex}$)

Άσκηση 4 (α)

Ο καταχωρητής $t1 περιέχει 4 κώδικες ascii που αντιστοιχούν στα πλήκτρα 2, 7, 9 και 0. Γράψτε ένα πρόγραμμα (με loop) το οποίο μετατρέπει τους κώδικες ascii σε ξεχωριστούς (μονοψήφιους) αριθμούς και τους αποθηκεύει σε διαδοχικές θέσεις στη μνήμη χρησιμοποιώντας ως δείκτη μνήμης τον καταχωρητή $t2 ο οποίος πρέπει να δείχνει αρχικά στη θέση μνήμης 0X50000000.

Άσκηση 4 (α)

```
1.                              # $t1 = 0x32373930
2.          li $t3, 0x30        # ascii–0x30 = integer
3.          li $t4, 4           # counter
4.          li $t2, 0x50000000  # t2 = pointer
5.   loop1:
6.          andi $t0, $t1, 0xff # use 8 bits only
7.          sub $t0, $t0, $t3   # -0x30
8.          sb $t0, 0($t2)      # store
9.          addi $t2, $t2, 1    # pointer + 1
10.         addi $t4, $t4, -1   # counter -1
11.         srl $t1, $t1, 8     # shift >> by 8
12.         bnez $t4, loop1     # branch until t4=0
```

Άσκηση 4 (β)

Γράψτε ένα πρόγραμμα (με loop) το οποίο να διαιρεί τον αριθμό που βρίσκεται στη θέση μνήμης 0X55000000 διαδοχικά με το 2, 3, 4, 5 και 6 και κάθε φορά να ελέγχει το υπόλοιπο της διαίρεσης. Εάν το υπόλοιπο μιας διαίρεσης ισούται με 1, τότε το πρόγραμμα να φτάνει στο τέλος του. Αν δεν βρεθεί υπόλοιπο = 1, τότε να τυπώνει τον αρχικό αριθμό.

Άσκηση 4 (β)

```
1.          lb $t0, 0x55000000# get the number from mem
2.          li $t3, 2         # $t3 initial divider
3.          li $t4, 7         # stop divisions at 7
4.          li $t1, 1         # check remainder with $t1
5.  loop1:
6.          div $t0, $t3      # divide
7.          mfhi $t5          # get remainder
8.          beq $t5, $t1 exit # check rem, if =1 exit
9.          addi $t3, $t3, 1  # divider + 1
10.         bne $t3, $t4, loop1# until divider = 7, loop1
11.         li $v0, 1         # syscall 1 to print integer
12.         move $a0, $t0     # move int. for printing
13.         syscall
14.  exit:                    # exit
```

Άσκηση 4 (γ)

Στις θέσεις μνήμης 0X40000000 - 0X40000003 υπάρχουν αποθηκευμένοι οι κώδικες ascii που αντιστοιχούν στα πλήκτρα 3, 4, 5 και 6. Γράψτε ένα πρόγραμμα το οποίο μετατρέπει τους κώδικες ascii σε ξεχωριστούς (μονοψήφιους) αριθμούς και τους αποθηκεύει στις θέσεις μνήμης 0X40000006 - 0X40000009 αντίστοιχα. Ο καταχωρητής $t8 να χρησιμοποιηθεί ως δείκτης στη μνήμη και το πρόγραμμα να γίνει με loop.

Άσκηση 4 (γ)

```
1.          li $t8, 0x40000000      # t8 = pointer to memory
2.          li $t0, 0x30            # ascii – 30 = integer
3.          li $t1, 4               # bytes to do
4. loop2:   lb $t2, ($t8)           # load byte from memory
5.          sub $t2, $t2, $t0       # ascii – 30 = integer
6.          sb $t2, 6($t8)          # store to memory (t8+6)
7.          addi $t8, $t8, 1        # memory pointer +1
8.          addi $t1, $t1, -1       # couter -1
9.          bnez $t1, loop2   # branch to loop2 until t1 = 0
```

Άσκηση 2
(Εύρεση Πρώτου Αριθμού)

1. Get number
2. Jal atoi
3. Jal riza
4. Jal check (rem. of: num/2, num/3…. n/riza)
5. If prime (always rem > 0), print and exit
6. If not a prime (rem=0), num = num -1
7. Branch to step 3 or 4 (either way will work)