ΛΥΣΕΙΣ

3η Σειρά Ασκήσεων στο μάθημα «Οργάνωση Υπολογιστών»

(Οι λύσεις είναι με παρόμοιες ασκήσεις – διαφορετικά νούμερα)

Άσκηση 1:

Να γραφτεί μια υπορουτίνα που θα μετατρέπει τα περιεχόμενα των θέσεων μνήμης **$400400 - $40040F** σε χαρακτήρες **ASCII**. Σε κάθε θέση είναι αποθηκευμένο ένα δεκαεξαδικό ψηφίο (το περισσότερο σημαντικό nibble είναι μηδέν).

Οι χαρακτήρες **ASCII** να αποθηκευτούν στις θέσεις μνήμης **$400410 - $40041F** αντίστοιχα.

Θεωρήστε ότι τα περιεχόμενα θέσεων μνήμης **$400400 - $40040F** είναι οι αριθμοί:

$00, $01, $02, $03, $04, $05, $06, $07, $08, $09, $0A, $0B, $0C, $0D, $0E, $0F

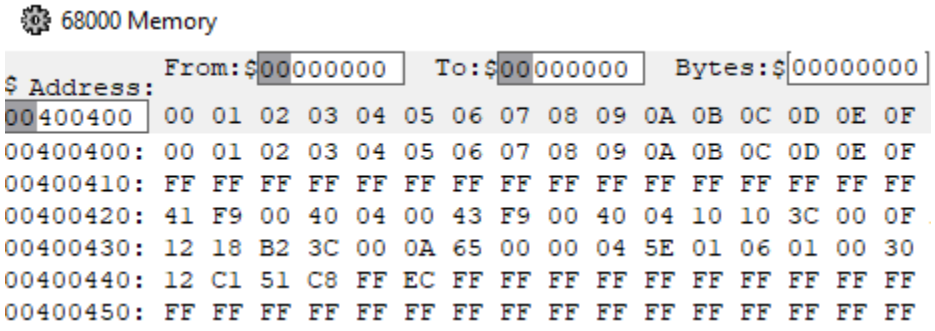
Να γράψετε το πρόγραμμα και να επαληθεύσετε τα αποτελέσματα με το Easy68K.

Να δοθεί η αρχική και τελική κατάσταση των καταχωρητών και της μνήμης στο Easy68K.

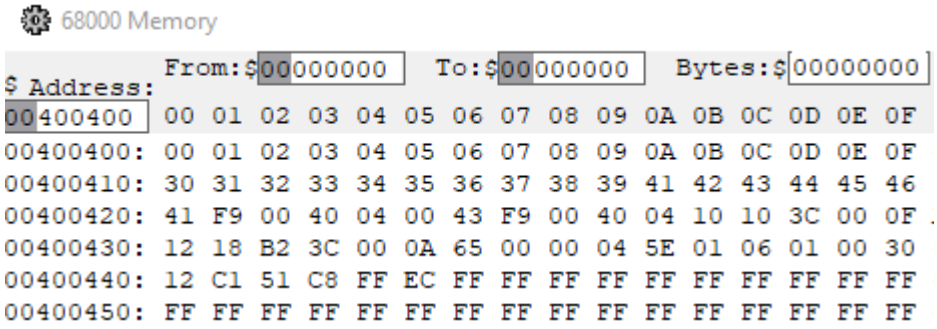
Απάντηση:
Ο κώδικας είναι:

```
            ORG   $400400
HEXNUMS DC.B $00,$01,$02,$03,$04,$05,$06,$07,$08,$09,$0A,$0B,$0C,$0D,$0E,$0F
ASCIICHR    DS.B  16
            ORG   $400420
HEXASCII    LEA   HEXNUMS,A0
            LEA   ASCIICHR,A1
            MOVE.B #$0F,D0
LOOP        MOVE.B (A0)+,D1
            CMP.B  #10,D1
            BCS   LETER
            ADDQ.B #7,D1
LETER       ADDI.B #$30,D1
            MOVE.B D1,(A1)+
            DBRA  D0,LOOP
            END   $400420
```

Η εικόνα πριν την εκτέλεση του προγράμματος:



Η εικόνα μετά την εκτέλεση του προγράμματος:



<u>Άσκηση 2</u>:

Να γραφτεί ένα πρόγραμμα που:

Α. Θα αποθηκεύει στις θέσεις μνήμης από **$400800 - $4008FF** τους αριθμούς <u>FF, FE, FD, FC, … έως το 0</u>.

Β. Στη συνέχεια θα μεταφέρει τα περιεχόμενα των θέσεων αυτών από την θέση **$4006FF** <u>και κάτω με αντίστροφή φορά</u> (δηλαδή στο $4006FF θα έχουμε το FF… κλπ.).

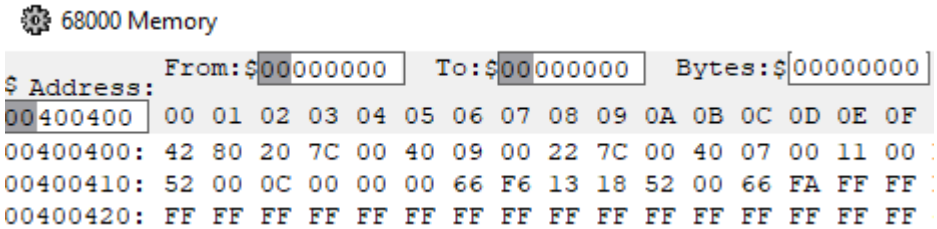
Να γράψετε το πρόγραμμα και να επαληθεύσετε τα αποτελέσματα με το Easy68K.

Να δοθεί η αρχική και τελική κατάσταση των καταχωρητών και της μνήμης στο Easy68K.

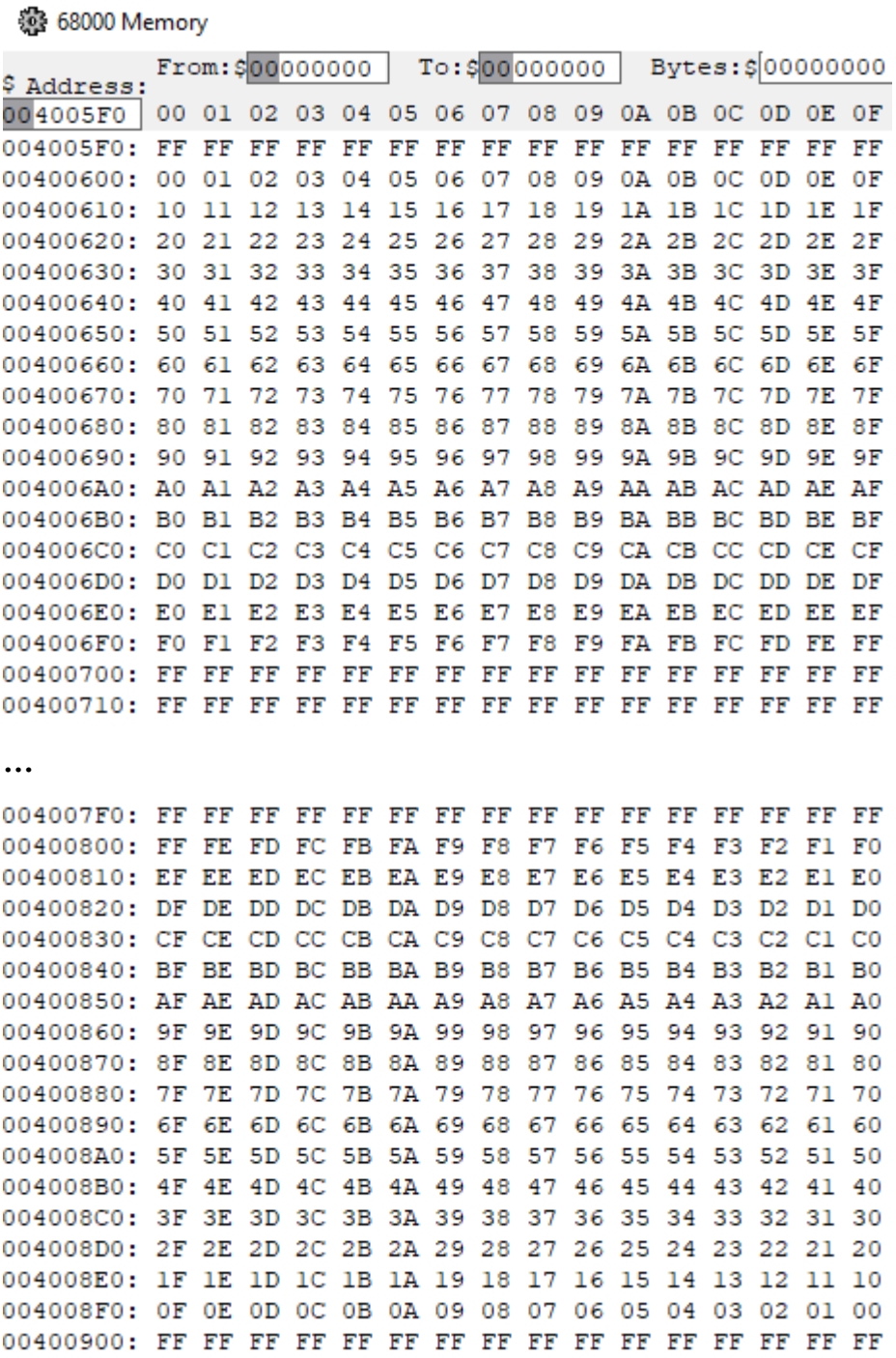
<u>Απάντηση:</u>
Ο κώδικας είναι:

```
            ORG     $400400
ROUTINE     CLR.L   D0
            MOVEA.L     #$400900,A0
            MOVEA.L     #$400700,A1
LOOP        MOVE.B D0,-(A0)
            ADDQ.B #1,D0
            CMPI.B #0,D0
            BNE     LOOP
LOOP1       MOVE.B (A0)+,-(A1)
            ADDQ.B #1,D0
            BNE     LOOP1
            END     $400400
```

Η εικόνα πριν την εκτέλεση του προγράμματος:



Η εικόνα μετά την εκτέλεση του προγράμματος:



Άσκηση 3:

Η παρακάτω υπορουτίνα φαινομενικά προσθέτει δύο (2) **BCD** αριθμούς οκτώ ψηφίων (4 byte) που είναι αποθηκευμένοι στις θέσεις μνήμης **$400400 - $400403** και **$400404 - $400407** και να αποθηκεύει το αποτέλεσμα στις θέσεις μνήμης **$400404 - $400407**.

Αυτή όμως η υπορουτίνα είναι γραμμένη με λάθος τρόπο.

Ζητείται: Να εντοπίσετε το σφάλμα στην υπορουτίνα και να το διορθώσετε. Να εξηγήσετε το σκεπτικό σας.

```
            ORG   $400400
NUM1        DC.L  $11443326
NUM2        DC.L  $87655812
            ORG   $400410
ADDBCD      LEA   NUM1+4,A0
            LEA   NUM2+4,A1
            MOVE  #$EF,CCR
            MOVE.B     #04,D0
LOOP        ABCD  -(A0),-(A1)
            SUBQ.B     #1,D0
            BNE   LOOP
            END   $400410
```

Να γράψετε το πρόγραμμα και να επαληθεύσετε τα αποτελέσματα με το Easy68K.

Να δοθεί η αρχική και τελική κατάσταση των καταχωρητών και της μνήμης στο Easy68K.

Απάντηση:

Η εντολή ABCD προσθέτει δύο BCD αριθμούς (π.χ. 011010012=69BCD 001101112=37BCD) και το δείκτη επέκτασης X που προέκυψε από την πρόσθεση των προηγούμενων δύο αριθμών BCD.

Επομένως, είναι σημαντικότατο ο δείκτης επέκτασης X που προκύπτει από την πρόσθεση των δύο προηγούμενων ψηφίων BCD να μεταφερθεί «ως έχει» στην πρόσθεση των επόμενων ψηφίων BCD.

Στην εν λόγω υπορουτίνα χρησιμοποιείται στον έλεγχο των επαναλήψεων η εντολή SUBQ.B η οποία μεταξύ άλλων επηρεάζει, και άρα μπορεί να αλλάζει, το δείκτη επέκτασης X και επομένως να δίνει λάθος αποτέλεσμα.

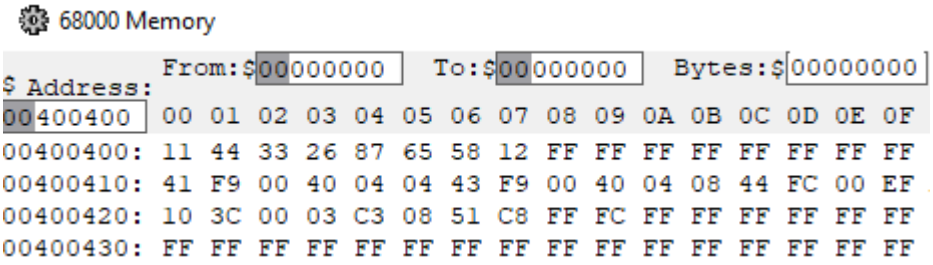
Ως διόρθωση προτείνεται να χρησιμοποιηθεί η εντολή DBRA D0,LOOP η οποία όταν εκτελείται δεν επηρεάζει το δείκτη επέκτασης X.

Επειδή όμως η εντολή αυτή σταματά την διακλάδωση όταν το περιεχόμενο το μετρητή επαναλήψεων γίνεται -1 ο μετρητής επαναλήψεων πρέπει να φορτώνεται με τον αριθμό των επαναλήψεων -1. Στην προκειμένη περίπτωση με τον αριθμό 3.

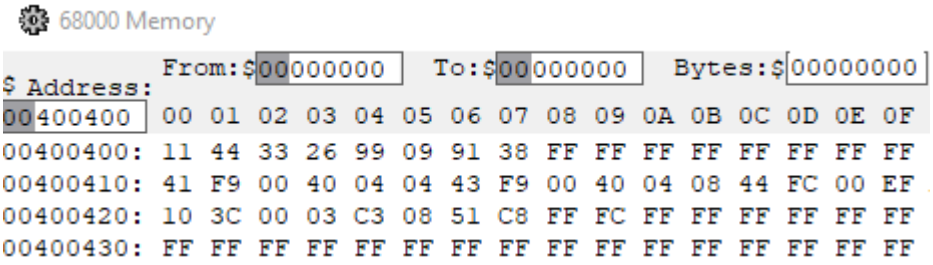
Ο κώδικας είναι:

```
                ORG     $400400
NUM1            DC.L    $11443326
NUM2            DC.L    $87655812
                ORG     $400410
ADDBCD          LEA     NUM1+4,A0
                LEA     NUM2+4,A1
                MOVE    #$EF,CCR
                MOVE.B      #03,D0
LOOP            ABCD    -(A0),-(A1)
                DBRA    D0,LOOP
                END     $400410
```

Η εικόνα πριν την εκτέλεση του προγράμματος:



Η εικόνα μετά την εκτέλεση του προγράμματος:



Άσκηση 4:

Να γραφτεί μια σειρά εντολών που:
1. Θα τοποθετεί στους καταχωρητές **D0, D1, D2** τους αριθμούς **$5F, $D04E, $87654321** αντίστοιχα.
2. Θα προσθέτει τα περιεχόμενα των **D1** και **D2**, θα αρχικοποιεί τον δείκτη του σωρού και θα σπρώχνει το αποτέλεσμα στο σωρό.
3. Θα κάνει τους δείκτες του καταχωρητή κατάστασης είναι **C=1, N=0 και Z=1,** χωρίς να επηρεάζει τους υπόλοιπους δείκτες.
4. Θα ανακαλεί από το σωρό στον **D3** την περισσότερο σημαντική λέξη του αθροίσματος των **D1** και **D2**, θα τοποθετεί στον **D4** τον αριθμό **$789AEC91** και θα προσθέτει τον **D4** στον **D3**.

Αν υποτεθεί ότι πριν από την εκτέλεση των εντολών ο δείκτης σωρού δείχνει **$07FE8** και ότι οι δείκτες του καταχωρητή κατάστασης είναι X=0, N=0, Z=0, V=0, C=0 να καταγραφεί ο τρόπος με τον οποίο αλλάζουν οι δείκτες καθώς και ο σωρός και ο δείκτης σωρού.

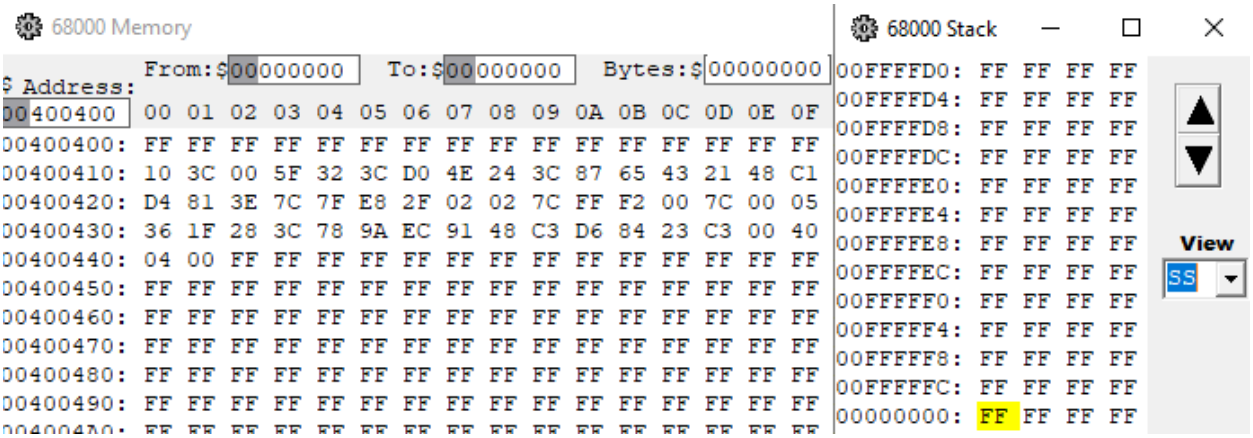
Να γράψετε το πρόγραμμα και να επαληθεύσετε τα αποτελέσματα με το Easy68K.

Απάντηση:

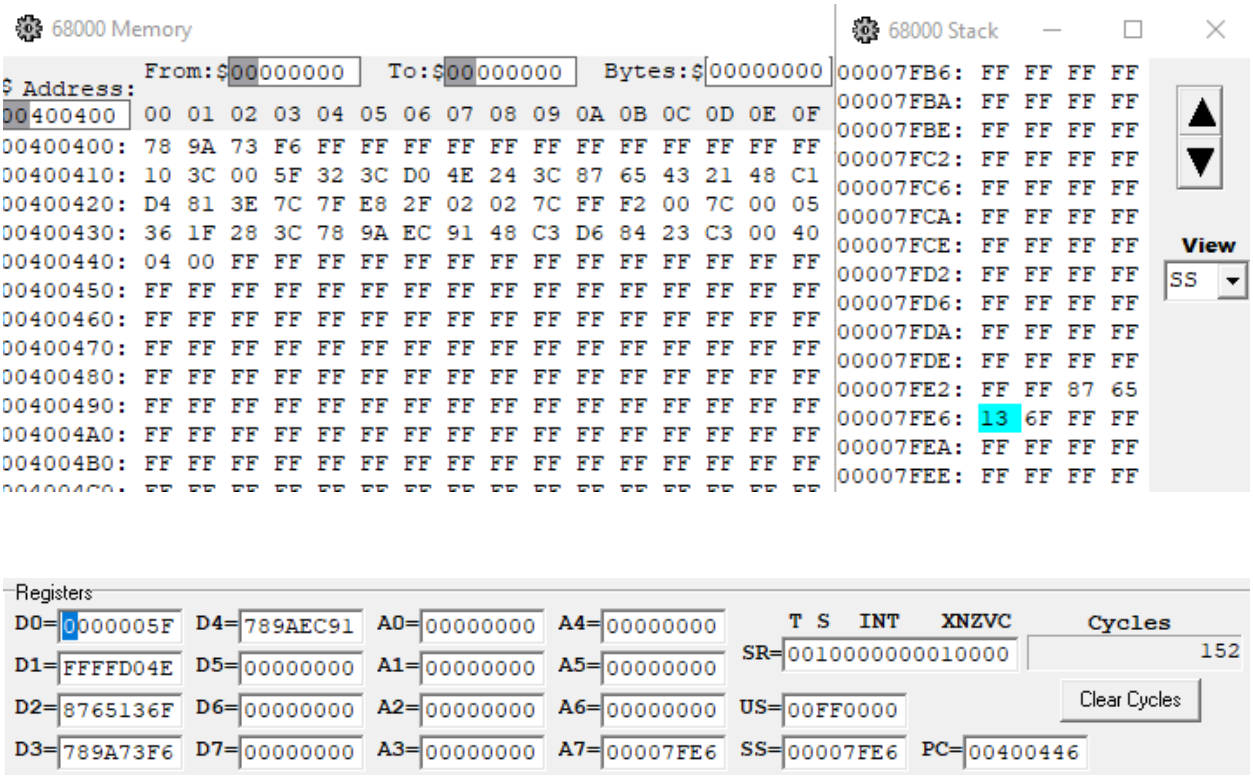
Ο κώδικας είναι:

```
          ORG $400400
INITSP    EQU $07FE8          ;Equate INITSP=$07FE8
SUM       DS.L 1
          ORG $400410
START     MOVE.B #$5F,D0
          MOVE.W #$D04E,D1
          MOVE.L #$87654321,D2
          EXT.L D1
          ADD.L D1,D2
          MOVEA #INITSP,SP    ;Initialize SP
          MOVE.L D2,-(SP)
          ANDI.W #$FFF2,SR
          ORI.W #$0005,SR
          MOVE.W (SP)+,D3
          MOVE.L #$789AEC91,D4
          EXT.L D3
          ADD.L D4,D3
          MOVE.L D3,SUM
          END START
```

Η εικόνα πριν την εκτέλεση του προγράμματος:



Η εικόνα μετά την εκτέλεση του προγράμματος:



Άσκηση 5:

Να γραφτεί ένα πρόγραμμα που θα διαβάζει μια συμβολοσειρά **ASCII** χαρακτήρων και θα την αντιστρέφει με χρήση της **Hardware Stack του 68000 (A7-SP)**.

Η συμβολοσειρά θα τερματίζει με το σύμβολο ASCII **CR**.

Για παράδειγμα μπορεί να χρησιμοποιηθεί η συμβολοσειρά: **'O','F','N','I',$0D**

Στη συνέχεια να δοθεί λύση στο ίδιο πρόβλημα με τον καταχωρητή διευθύνσεων **A5**.

Να συγκριθούν οι δύο λύσεις και να αναφέρεται διαφορές.

Να γράψετε το πρόγραμμα και να επαληθεύσετε τα αποτελέσματα με το Easy68K.

Να δοθεί η αρχική και τελική κατάσταση των καταχωρητών, της μνήμης και της στοίβας στο Easy68K.

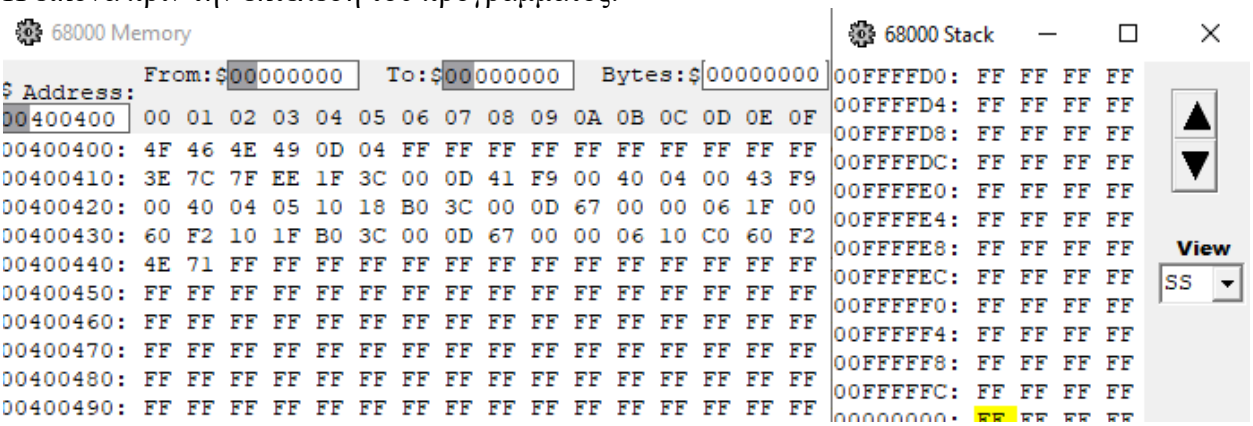
Απάντηση:

Στην άσκηση αυτή έχουμε δύο λύσεις με τον SP και τον A5.
Η μεταξύ τους διαφορά είναι στο ότι ο SP(A7) όπως ορίζεται από την αρχιτεκτονική του επεξεργαστή (προγραμματιστικό μοντέλο) είναι ο Stack Pointer και μπορεί να αποθηκεύει δεδομένα μόνο σε άρτιες θέσεις μνήμης, άρα τα δεδομένα στον σωρό αποθηκεύονται σε words και longwords.
Αυτό δεν ισχύσει στον A5 που είναι ένας σωρός χρήστη (user stack) που μπορεί να λειτουργήσει και με byte, άρα και σε περιττές θέσεις μνήμης.

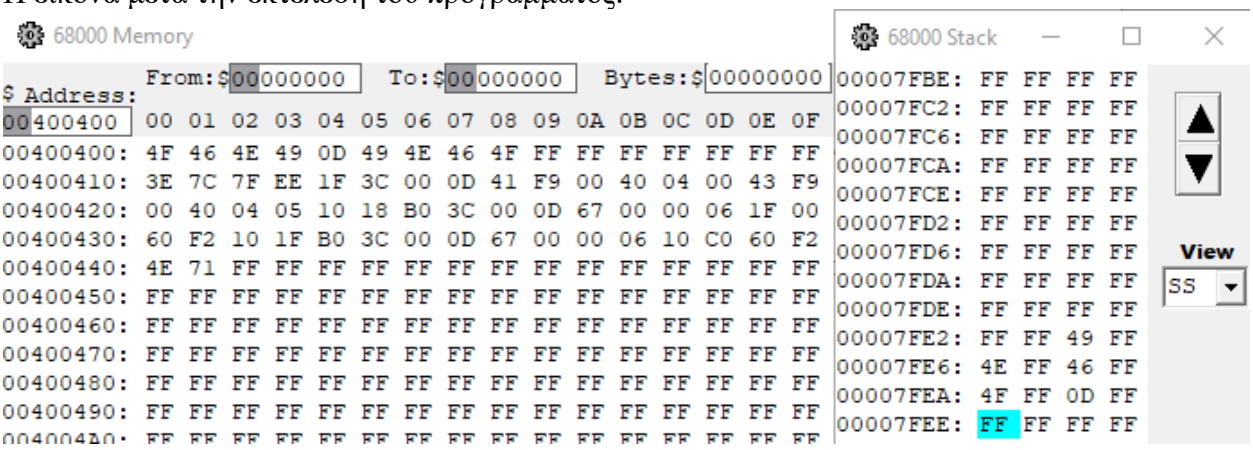
Ο κώδικας με τον SP είναι:

```
        ORG   $400400
INITSP  EQU   $07FEE ; Equate INITSP=$07FEE
CR      EQU   13
STR     DC.B 'O','F','N','I',$0D
INVSTR  DC.B 4
        ORG    $400410
START   MOVEA  #INITSP,SP ;Initialize SP
        MOVE.B #CR,-(SP)
        LEA   STR,A0
        LEA   INVSTR,A1
NPUSH   MOVE.B (A0)+,D0
        CMP.B #CR,D0
        BEQ FPOP
        MOVE.B D0,-(SP)
        BRA NPUSH
FPOP    MOVE.B (SP)+,D0
        CMP.B #CR,D0
        BEQ FIN
        MOVE.B D0,(A0)+
        BRA FPOP
FIN     NOP
        END START
```

Η εικόνα πριν την εκτέλεση του προγράμματος:



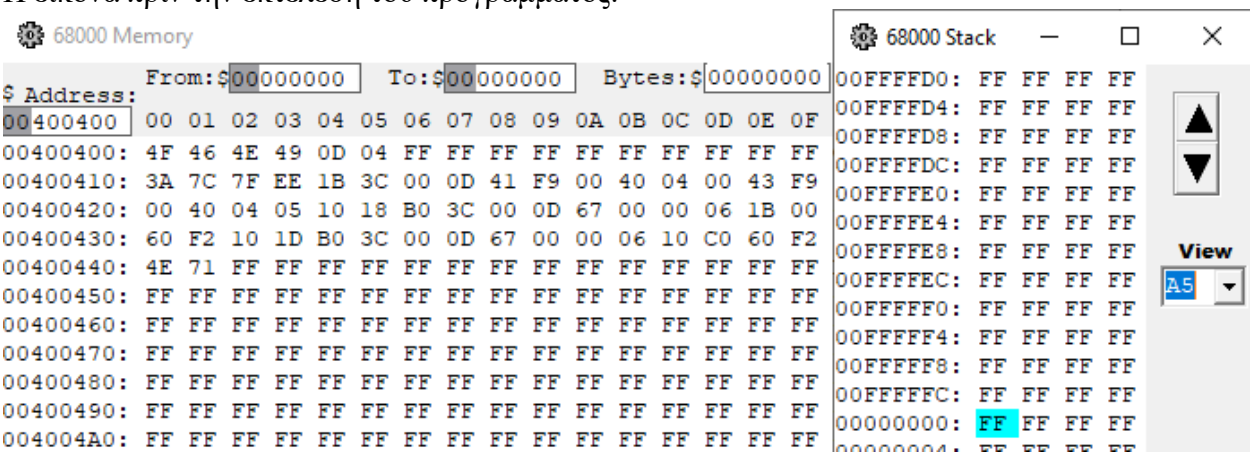
Η εικόνα μετά την εκτέλεση του προγράμματος:



Ο κώδικας με τον A5 είναι:

```
        ORG   $400400
INITSP  EQU   $07FEE ; Equate INITSP=$07FEE
CR      EQU   13
STR     DC.B  'O','F','N','I',$0D
INVSTR  DC.B  4
        ORG    $400410
START   MOVEA   #INITSP,A5 ;Initialize SP
        MOVE.B #CR,-(A5)
        LEA   STR,A0
        LEA   INVSTR,A1
NPUSH   MOVE.B  (A0)+,D0
        CMP.B #CR,D0
        BEQ FPOP
        MOVE.B D0,-(A5)
        BRA NPUSH
FPOP    MOVE.B  (A5)+,D0
        CMP.B #CR,D0
        BEQ FIN
        MOVE.B D0,(A0)+
        BRA FPOP
FIN     NOP
        END START
```

Η εικόνα πριν την εκτέλεση του προγράμματος:



Η εικόνα μετά την εκτέλεση του προγράμματος: