

ΛΥΣΕΙΣ

3^η 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:

68000 Memory

\$ Address:	From:\$00000000	To:\$00000000	Bytes:\$00000000
00400400	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F		
00400400:	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F		
00400410:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400420:	41 F9 00 40 04 00 43 F9 00 40 04 10 10 3C 00 0F		
00400430:	12 18 B2 3C 00 0A 65 00 00 04 5E 01 06 01 00 30		
00400440:	12 C1 51 C8 FF EC FF FF FF FF FF FF FF FF FF		
00400450:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		

Η εικόνα μετά την εκτέλεση του προγράμματος:

```

68000 Memory
$ Address: From:$00000000 To:$00000000 Bytes:$00000000
00400400 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00400400: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00400410: 30 31 32 33 34 35 36 37 38 39 41 42 43 44 45 46
00400420: 41 F9 00 40 04 00 43 F9 00 40 04 10 10 3C 00 0F
00400430: 12 18 B2 3C 00 0A 65 00 00 04 5E 01 06 01 00 30
00400440: 12 C1 51 C8 FF EC FF FF FF FF FF FF FF FF FF
00400450: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

```

Άσκηση 2:

Να γραφτεί ένα πρόγραμμα που:

A. Θα αποθηκεύει στις θέσεις μνήμης από \$400800 - \$4008FF τους αριθμούς FF, FE, FD, FC, ... έως το 0.

B. Στη συνέχεια θα μεταφέρει τα περιεχόμενα των θέσεων αυτών από την θέση \$4006FF και κάτω με αντίστροφη φορά (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.

Απάντηση:

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:

```

68000 Memory
$ Address: From:$00000000 To:$00000000 Bytes:$00000000
00400400 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00400400: 42 80 20 7C 00 40 09 00 22 7C 00 40 07 00 11 00
00400410: 52 00 0C 00 00 00 66 F6 13 18 52 00 66 FA FF FF
00400420: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

```

Η εικόνα μετά την εκτέλεση του προγράμματος:

```

68000 Memory
$ Address: From:$00000000 To:$00000000 Bytes:$00000000
004005F0: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
004005F0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00400600: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00400610: 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
00400620: 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
00400630: 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F
00400640: 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F
00400650: 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F
00400660: 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F
00400670: 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F
00400680: 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F
00400690: 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F
004006A0: A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF
004006B0: B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF
004006C0: C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF
004006D0: D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF
004006E0: E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF
004006F0: F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
00400700: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00400710: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
...
004007F0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
00400800: FF FE FD FC FB FA F9 F8 F7 F6 F5 F4 F3 F2 F1 F0
00400810: EF EE ED EC EB EA E9 E8 E7 E6 E5 E4 E3 E2 E1 E0
00400820: DF DE DD DC DB DA D9 D8 D7 D6 D5 D4 D3 D2 D1 D0
00400830: CF CE CD CC CB CA C9 C8 C7 C6 C5 C4 C3 C2 C1 C0
00400840: BF BE BD BC BB BA B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
00400850: AF AE AD AC AB AA A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
00400860: 9F 9E 9D 9C 9B 9A 99 98 97 96 95 94 93 92 91 90
00400870: 8F 8E 8D 8C 8B 8A 89 88 87 86 85 84 83 82 81 80
00400880: 7F 7E 7D 7C 7B 7A 79 78 77 76 75 74 73 72 71 70
00400890: 6F 6E 6D 6C 6B 6A 69 68 67 66 65 64 63 62 61 60
004008A0: 5F 5E 5D 5C 5B 5A 59 58 57 56 55 54 53 52 51 50
004008B0: 4F 4E 4D 4C 4B 4A 49 48 47 46 45 44 43 42 41 40
004008C0: 3F 3E 3D 3C 3B 3A 39 38 37 36 35 34 33 32 31 30
004008D0: 2F 2E 2D 2C 2B 2A 29 28 27 26 25 24 23 22 21 20
004008E0: 1F 1E 1D 1C 1B 1A 19 18 17 16 15 14 13 12 11 10
004008F0: 0F 0E 0D 0C 0B 0A 09 08 07 06 05 04 03 02 01 00
00400900: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

```

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 σημαντικότερο 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:

68000 Memory

```

$ Address: From:$00000000 To:$00000000 Bytes:$00000000
00400400 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00400400: 11 44 33 26 87 65 58 12 FF FF FF FF FF FF FF FF
00400410: 41 F9 00 40 04 04 43 F9 00 40 04 08 44 FC 00 EF
00400420: 10 3C 00 03 C3 08 51 C8 FF FC FF FF FF FF FF FF
00400430: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

```

Η εικόνα μετά την εκτέλεση του προγράμματος:

68000 Memory

```

$ Address: From:$00000000 To:$00000000 Bytes:$00000000
00400400 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00400400: 11 44 33 26 99 09 91 38 FF FF FF FF FF FF FF FF
00400410: 41 F9 00 40 04 04 43 F9 00 40 04 08 44 FC 00 EF
00400420: 10 3C 00 03 C3 08 51 C8 FF FC FF FF FF FF FF FF
00400430: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

```

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 υποθεθεί 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:

68000 Memory

From: \$00000000 To: \$00000000 Bytes: \$00000000

\$ Address: 00400400

00400400:	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00400410:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400420:	10	3C	00	5F	32	3C	D0	4E	24	3C	87	65	43	21	48	C1
00400430:	D4	81	3E	7C	7F	E8	2F	02	02	7C	FF	F2	00	7C	00	05
00400440:	36	1F	28	3C	78	9A	EC	91	48	C3	D6	84	23	C3	00	40
00400450:	04	00	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400460:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400470:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400480:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400490:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
004004A0:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

68000 Stack

00FFFFFFD0: FF FF FF FF
 00FFFFFFD4: FF FF FF FF
 00FFFFFFD8: FF FF FF FF
 00FFFFFFDC: FF FF FF FF
 00FFFFFFE0: FF FF FF FF
 00FFFFFFE4: FF FF FF FF
 00FFFFFFE8: FF FF FF FF
 00FFFFFFEC: FF FF FF FF
 00FFFFFFF0: FF FF FF FF
 00FFFFFFF4: FF FF FF FF
 00FFFFFFF8: FF FF FF FF
 00FFFFFFFC: FF FF FF FF
 00000000: FF FF FF FF

Η εικόνα μετά την εκτέλεση του προγράμματος:

68000 Memory

From: \$00000000 To: \$00000000 Bytes: \$00000000

\$ Address: 00400400

00400400:	78	9A	73	F6	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400410:	10	3C	00	5F	32	3C	D0	4E	24	3C	87	65	43	21	48	C1
00400420:	D4	81	3E	7C	7F	E8	2F	02	02	7C	FF	F2	00	7C	00	05
00400430:	36	1F	28	3C	78	9A	EC	91	48	C3	D6	84	23	C3	00	40
00400440:	04	00	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400450:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400460:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400470:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400480:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400490:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
004004A0:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
004004B0:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

68000 Stack

00007FB6: FF FF FF FF
 00007FBA: FF FF FF FF
 00007FBE: FF FF FF FF
 00007FC2: FF FF FF FF
 00007FC6: FF FF FF FF
 00007FCA: FF FF FF FF
 00007FCE: FF FF FF FF
 00007FD2: FF FF FF FF
 00007FD6: FF FF FF FF
 00007FDA: FF FF FF FF
 00007FDE: FF FF FF FF
 00007FE2: FF FF 87 65
 00007FE6: 13 6F FF FF
 00007FEA: FF FF FF FF
 00007FEE: FF FF FF FF

Registers

D0=	0000005F	D4=	789AEC91	A0=	00000000	A4=	00000000	T	S	INT	XNZVC	Cycles	152
D1=	FFFFFFD04E	D5=	00000000	A1=	00000000	A5=	00000000	SR=	0010000000010000	Clear Cycles			
D2=	8765136F	D6=	00000000	A2=	00000000	A6=	00000000	US=	00FF0000				
D3=	789A73F6	D7=	00000000	A3=	00000000	A7=	00007FE6	SS=	00007FE6	PC=	00400446		

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 ισχύει 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:

68000 Memory

\$ Address:	From:\$	To:\$	Bytes:\$													
00400400	00000000	00000000	00000000													
00400400:	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00400400:	4F	46	4E	49	0D	04	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400410:	3E	7C	7F	EE	1F	3C	00	0D	41	F9	00	40	04	00	43	F9
00400420:	00	40	04	05	10	18	B0	3C	00	0D	67	00	00	06	1F	00
00400430:	60	F2	10	1F	B0	3C	00	0D	67	00	00	06	10	C0	60	F2
00400440:	4E	71	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400450:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400460:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400470:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400480:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400490:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

68000 Stack

00FFFFFFD0:	FF	FF	FF	FF
00FFFFFFD4:	FF	FF	FF	FF
00FFFFFFD8:	FF	FF	FF	FF
00FFFFFFDC:	FF	FF	FF	FF
00FFFFFFE0:	FF	FF	FF	FF
00FFFFFFE4:	FF	FF	FF	FF
00FFFFFFE8:	FF	FF	FF	FF
00FFFFFFEC:	FF	FF	FF	FF
00FFFFFFF0:	FF	FF	FF	FF
00FFFFFFF4:	FF	FF	FF	FF
00FFFFFFF8:	FF	FF	FF	FF
00FFFFFFFC:	FF	FF	FF	FF
00000000:	FF	FF	FF	FF

Η εικόνα μετά την εκτέλεση του προγράμματος:

68000 Memory

\$ Address:	From:\$	To:\$	Bytes:\$													
00400400	00000000	00000000	00000000													
00400400:	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00400400:	4F	46	4E	49	0D	49	4E	46	4F	FF	FF	FF	FF	FF	FF	FF
00400410:	3E	7C	7F	EE	1F	3C	00	0D	41	F9	00	40	04	00	43	F9
00400420:	00	40	04	05	10	18	B0	3C	00	0D	67	00	00	06	1F	00
00400430:	60	F2	10	1F	B0	3C	00	0D	67	00	00	06	10	C0	60	F2
00400440:	4E	71	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400450:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400460:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400470:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400480:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400490:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00400420:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

68000 Stack

00007FBE:	FF	FF	FF	FF
00007FC2:	FF	FF	FF	FF
00007FC6:	FF	FF	FF	FF
00007FCA:	FF	FF	FF	FF
00007FCE:	FF	FF	FF	FF
00007FD2:	FF	FF	FF	FF
00007FD6:	FF	FF	FF	FF
00007FDA:	FF	FF	FF	FF
00007FDE:	FF	FF	FF	FF
00007FE2:	FF	FF	49	FF
00007FE6:	4E	FF	46	FF
00007FEA:	4F	FF	0D	FF
00007FEE:	FF	FF	FF	FF

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:

68000 Memory

\$ Address:	From:\$00000000	To:\$00000000	Bytes:\$00000000
00400400	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F		
00400400:	4F 46 4E 49 0D 04 FF FF FF FF FF FF FF FF FF		
00400410:	3A 7C 7F EE 1B 3C 00 0D 41 F9 00 40 04 00 43 F9		
00400420:	00 40 04 05 10 18 B0 3C 00 0D 67 00 00 06 1B 00		
00400430:	60 F2 10 1D B0 3C 00 0D 67 00 00 06 10 C0 60 F2		
00400440:	4E 71 FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400450:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400460:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400470:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400480:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400490:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
004004A0:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		

68000 Stack

00FFFFFFD0:	FF FF FF FF
00FFFFFFD4:	FF FF FF FF
00FFFFFFD8:	FF FF FF FF
00FFFFFFDC:	FF FF FF FF
00FFFFFFE0:	FF FF FF FF
00FFFFFFE4:	FF FF FF FF
00FFFFFFE8:	FF FF FF FF
00FFFFFFEC:	FF FF FF FF
00FFFFFFF0:	FF FF FF FF
00FFFFFFF4:	FF FF FF FF
00FFFFFFF8:	FF FF FF FF
00FFFFFFFC:	FF FF FF FF
00000000:	FF FF FF FF
00000004:	FF FF FF FF

Η εικόνα μετά την εκτέλεση του προγράμματος:

68000 Memory

\$ Address:	From:\$00000000	To:\$00000000	Bytes:\$00000000
00400400	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F		
00400400:	4F 46 4E 49 0D 49 4E 46 4F FF FF FF FF FF FF FF		
00400410:	3A 7C 7F EE 1B 3C 00 0D 41 F9 00 40 04 00 43 F9		
00400420:	00 40 04 05 10 18 B0 3C 00 0D 67 00 00 06 1B 00		
00400430:	60 F2 10 1D B0 3C 00 0D 67 00 00 06 10 C0 60 F2		
00400440:	4E 71 FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400450:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400460:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400470:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400480:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
00400490:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		
004004A0:	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF		

68000 Stack

00007FBE:	FF FF FF FF
00007FC2:	FF FF FF FF
00007FC6:	FF FF FF FF
00007FCA:	FF FF FF FF
00007FCE:	FF FF FF FF
00007FD2:	FF FF FF FF
00007FD6:	FF FF FF FF
00007FDA:	FF FF FF FF
00007FDE:	FF FF FF FF
00007FE2:	FF FF FF FF
00007FE6:	FF FF FF 49
00007FEA:	4E 46 4F 0D
00007FEE:	FF FF FF FF
00007FF2:	FF FF FF FF