Έντυπο Υποβολής – Αξιολόγησης ΓΕ

Ο φοιτητής συμπληρώνει την ενότητα «Υποβολή Εργασίας» και αποστέλλει το έντυπο σε δύο μη συρραμμένα αντίγραφα (ή ηλεκτρονικά) στον Καθηγητή-Σύμβουλο. Ο Καθηγητής-Σύμβουλος συμπληρώνει την ενότητα «Αξιολόγηση Εργασίας» και στα δύο αντίγραφα και επιστρέφει το ένα στο φοιτητή μαζί με τα σχόλια επί της ΓΕ, ενώ κρατά το άλλο για το αρχείο του μαζί με το γραπτό σημείωμα του Συντονιστή, εάν έχει δοθεί παράταση.

*Σε περίπτωση ηλεκτρονικής υποβολής του παρόντος εντύπου, το όνομα του ηλεκτρονικού αρχείου θα πρέπει να γράφεται υποχρεωτικά με λατινικούς χαρακτήρες και να ακολουθεί την κωδικοποίηση του παραδείγματος: Π.χ., το όνομα του αρχείου για τη 2η ΓΕ του φοιτητή ΙΩΑΝΝΟΥ στη ΔΕΟ13 θα πρέπει να γραφεί: «**ioannou_ge2_deo13.doc**».*

ΥΠΟΒΟΛΗ ΕΡΓΑΣΙΑΣ

Ονοματεπώνυμο φοιτητή	ΜΠΑΤΣΑΛΗΣ ΕΥΑΓΓΕΛΟΣ

ΚωδικόςΘΕ	ΠΛΗ21	Ονοματεπώνυμο Καθηγητή - Σύμβουλου	ΒΑΡΖΑΚΑΣ ΠΑΝΑΓΙΩΤΗΣ
Κωδικός Τμήματος	ΠΛΗ21-ΗΛΕ43	Καταληκτική ημερομηνία παραλαβής	13 Μαϊου 2020
Ακ. Έτος	2019-20	Ημερομηνία αποστολής ΓΕ από το φοιτητή	13 Μαΐου 2020
α/α ΓΕ	4η	Επισυνάπτεται (*σε περίπτωση που έχει ζητηθεί*) η άδεια παράτασης από το Συντονιστή;	

Υπεύθυνη Δήλωση Φοιτητή*: Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τη συγκεκριμένη Θεματική Ενότητα..*

ΑΞΙΟΛΟΓΗΣΗ ΕΡΓΑΣΙΑΣ

Ημερομηνία παραλαβής ΓΕ από το φοιτητή	
Ημερομηνία αποστολής σχολίων στο φοιτητή	
Βαθμολογία (αριθμητικά, ολογράφως)	

Υπογραφή
Φοιτητή

Υπογραφή
Καθηγητή-Συμβούλου

Επίλυση της Άσκησης 1

Η λογική σκέψη πάνω σε αυτή την άσκηση είναι ως εξής:
Το πρόγραμμά μας θα ελέγχει 16ψήφιο αριθμό δηλαδή 3 bit. Γνωρίζουμε ότι στον 8085 κάθε θέση μνήμης μπορεί και αποθηκεύει 1 bit άρα και θα χρειαστούμε 3 θέσεις μνήμης και εφόσον βάση της εκφώνησης ξεκινάει από την 2050h άρα για την αποθήκευση του 6ψήφιου αριθμού το πρόγραμμα θα χρειαστεί και την 2051h και την 2052h

Έπειτα έναν από τους τρόπους που θα χρησιμοποιήσω ώστε να μπορέσω να ολοκληρώσω το πρόγραμμα χωρίς περιττές εντολές είναι με την εντολή CMP αντί της αφαίρεσης με μάσκα διότι όπως ανέφερα θεώρησα ότι θα χρειαστούν λιγότερες εντολές. Αφού πρώτα όμως βάση διδακτέας ύλης και τεχνικών προγραμματισμού που εντόπισα από παλαιότερες εργασίες είναι με την αντιμετάθεση. Άρα θα κυλήσω 4 φορές προς τα δεξιά ώστε να αλλάξουν θέσεις η μια από τις δύο φορτωμένες θέσεις μνήμης ώστε να επιτευχθεί η σύγκριση και αναλόγως θα γίνει και το jump.

Ο έλεγχος γενικά θα γίνει πρώτα με τις δύο άκρες. Αν το περιεχόμενο της θέσης μνήμης 2050h και 2051 είναι JZ τότε το πρόγραμμα θα συνεχίσει στην επόμενη σύγκριση όπου θα επιτευχθεί το ίδιο και για την μεσαία θέση μνήμης την 2051h. Αν τυχόν στα Jump δεν γίνει η σύγκριση που συγκρίνω με το 0 τότε θα περάσει και στο ζητούμενο της άσκησης όπου και θα αποθηκεύσει το αντίστοιχο στοιχείο στη θέση μνήμης

LDA 2050h	;Load στον A το περιεχόμενο της μνήμης 2050h (τα δύο πρώτα ψηφία του 16αδικού αριθμού)
MOV B,A	;αποθήκευση το περιεχόμενο του A στον B
LDA 2052h	;Load στον A το περιεχόμενο της μνήμης 2052h (τα δύο τελευταία ψύφία του 16αδικού αριθμού)
RLC	;ολίσθηση 1/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
RLC	;ολίσθηση 2/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
RLC	;ολίσθηση 3/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
RLC	;ολίσθηση 4/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
CMP B	;Σύγκριση ψηφίων μεταξύ θέσης μνήμης τα δύο πρώτα με τα δύο τελευταία
JNZ LFS	;Jump στο LFS αν δεν είναι όμοια. LFS για LevelFailSucceed
LDA 2051h	;Load στον A το περιεχόμενο της μνήμης 2051h (τα δύο μεσαία ψηφία του 16αδικού αριθμού)
MOV B,A	;αποθήκευση το περιεχόμενο του A στον B
RLC	;ολίσθηση 1/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
RLC	;ολίσθηση 2/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
RLC	;ολίσθηση 3/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
RLC	;ολίσθηση 4/4 (Γίνεται ολίσθηση 4 φορές ώστε να γίνει αντιμετάθεση του αριθμού)
CMP B	;Σύγκριση ψηφίων μεταξύ θέσης μνήμης τα δύο πρώτα με τα δύο τελευταία
JNZ LFS	;Jump στο LFS αν δεν είναι όμοια. LFS για LevelFailSucceed
MVI A,01h	;Load στον A την τιμή 01
STA 3050h	;αποθήκευση το περιεχόμενο του A στη θέση μνήμης 3050h
JMP LS1	;Jump στο LS1 γιατερματισμό. LS για LevelSucceed
LFS: MVI A,0FFh	;Load στον A την τιμή FF
STA 3050h	;αποθήκευση το περιεχόμενο του A στη θέση μνήμης 3050h
LS1: hlt	;Τερματισμός προγράμματος

Επίλυση της Άσκησης 2
Υπό-ερώτημα 1

Η λογική σκέψη που έχω κάνει ώστε να επιλύσω αυτή την άσκηση είναι ως εξής: Για αρχή θα προσθέσω τις τιμές της επαλήθευσης της άσκηση (A) στους καταχωρητές ζεύγη HL και DE έπειτα θα προσθέσω πρώτα τα περιεχόμενα από τους καταχωρητές L+E και θα χρησιμοποιήσω την μετατροπή σε BCD της γλώσσας assembly DAA. Στη συνέχεια θα αποθηκεύσω το αποτέλεσμα βάση ζητούμενου της άσκησης. Έπειτά θα προσθέσω τους H+D αλλά με τη χρήση της εντολή ADC που προσθέτει το carry διότι μπορεί να έχει προκύψει κρατούμενο που πρέπει να υπολογιστεί.

LXI H, 4163h	;Loads τιμή 4-ψήφιο BCD αριθμό στον καταχωρητή HL
LXI D, 4738h	;Loads τιμή 4-ψήφιο BCD αριθμό στον καταχωρητή DE
MOV A, E	;Loads το A την τιμή LSB του καταχωρητη DE τον E από τον Ζεύγος τον LowOrder
ADD L	;Άθροισμα την τιμή LSB του καταχωρητη HL τον L με τον E
DAA	;Μετατροπή DAA |27|*****| 4|Decimal Adjust Accumulator| Βάση Τυπολογίου
STA 2300h	;Αποθήκευση αποτελέσματος στη θέση μνήμης 2300h
MOV A, H	;Loads το A την τιμή MSB του καταχωρητη HL τον H από τον Ζεύγος τον HighOrder
ADC D	;Άθροισμα την τιμή MSB του καταχωρητη DE τον D με τον H από το ζεύγος HL
DAA	Μετατροπή DAA |27|*****| 4|Decimal Adjust Accumulator| Βάση Τυπολογίου
STA 2301h	;Αποθήκευση αποτελέσματος στη θέση μνήμης 2301h
HLT	;Εντολή τερματισμού Προγράμματος

Επίλυση της Άσκησης 2
Υπό-ερώτημα 2

Το πρόγραμμα αυτό που κάνει είναι να υπολογίζει το γινόμενο δύο αριθμών του 1 byte που βρίσκονται στη θέση μνήμης 2200h και 2201h. Μοιάζει με τον πολλαπλασιασμό των χωρικών (αλά ρωσικά) μόνο που κάνει αριστερές ολισθήσεις για λόγους ευχέρειας ώστε να γλιτώσουμε διπλασιασμούς των μηδενικών.

Σε μια loop που αρχικοποιεί και το γινόμενο σε 0 διπλασιάζει το γινόμενο 8 φορές (με μετρητή επαναλήψεων)και ελέγχει αν το ψηφίο του 2ου αριθμού έχει carry 0 ή 1, αν είναι 0 πάει στο SKIP όπου και γίνεται μείωση του μετρητή των επαναλήψεων. Αν είναι 1 το Carry προσθέτει τον πρώτο αριθμό στο γινόμενο (2200h) και αποθηκεύει το αποτέλεσμα με την εντολή SHLD στις θέσεις μνήμης 2300h και 2301h

LXI H, 2200	;Φορτώνει στην HL το περιεχόμενο της μνήμης 2200h
MOV E, M	;Μεταφέρει τον πρώτο από τους δύο αριθμούς στον καταχωρητή E
MVI D, 00H	;Γίνεται μηδενισμός του D
Με τις από πάνω εντολές επιτυγχάνει το πρόγραμμα να αποθηκεύσει τον πρώτο αριθμό στους καταχωρητές DE. Από ότι έχω καταλάβει ο αριθμός είναι 1 byte και χρησιμοποιούμε ζεύγος ώστε να μπορεί να αποθηκευτεί το γινόμενο που να είναι μεγαλύτερος από 1 byte	
INX H	;Αυξάνεται η τιμή του HL κατά 1 και δείχνει τη διεύθυνση θέσης μνήμης του 2ου αριθμού
MOV A, M	;μεταφέρει το περιεχόμενο του HL στον συσσωρευτή A
Σε αυτό το σημείο φορτώνεται στο πρόγραμμα και ο δεύτερος αριθμός και μετά τις αρχικοποιήσεις ξεκινάει ο βρόγχος Loop	
LXI H, 0000	;Αρχικοποιεί στην τιμή 0 το ζεύγος HL
MVI B, 08H	;Αρχικοποιεί τον μετρητή του Loop στο 8 για 8 επαναλήψεις
K1: DAD H	;Εδώ γίνεται ο διπλασιασμός του HL δηλαδή HL+HL και ξεκινάει η λούπα με την ετικέτα K1
RAL	;Όπως και αναφέραμε γίνεται ολίσθηση αριστερή
JNC SKIP	;Γίνεται έλεγχος του γινομένου JumpNotCarry που σημαίνει όταν είναι άρτιος δηλαδή το carry να έχει την τιμή 0 τότε πάει στο SKIP
DAD D	;Το carry έχει τη τιμή 1 τότε το πρόγραμμα συνεχίζεται και γίνεται πρόσθεση στο γινόμενο HL ο πρώτος αριθμός δηλαδή ο DE
SKIP: DCR B	;Πριν το τέλος της Λούπας μειώνεται κατά 1 η τιμή του βρόγχου επανάληψης
JNZ K1	;γίνεται έλεγχος και αν δεν είναι 0 τότε γίνεται jump στην ετικέτα K1 θεωρείται τέλος της λούπας και ελέγχει τον μετρητή βρόγχου επανάληψης
SHLD 2300H	;Η εντολή αυτή αποθηκεύει το γινόμενο των δύο αριθμών μόλις ολοκληρωθεί το πρόγραμμα διαδοχικά στη θέση μνήμης 2300h και 2301h
HLT	;Εντολή τερματισμού προγράμματος

Επίλυση της Άσκησης 3
Υπό-ερώτημα α)

- Η ετικέτα K2 αντιστοιχεί στην διεύθυνση HEX της μνήμης 8012 όπου είναι η διεύθυνση της εντολής INX H με το περιεχόμενο 23
- Η ετικέτα K8 αντιστοιχεί στην διεύθυνση HEX της μνήμης 8008 όπου είναι η διεύθυνση της εντολής SUB B με το περιεχόμενο 80

Επίλυση της Άσκησης 3
Υπό-ερώτημα β)

MVI B,01 MVI C,00	;Αρχικοποιεί στον καταχωρητή B την τιμή 1 ;Αρχικοποιεί στον καταχωρητή C την τιμή 0
Σε αυτό το σημείο βλέπουμε κάποια αρχικοποίηση που γίνεται στους καταχωρητές B και C	
LXI H,5000	;Αρχικοποίηση του ζεύγους καταχωρητή HL να δείχνει το περιεχόμενο της τιμή 5000h
MOV A,M	;Μεταφορά του περιεχομένου της θέσης μνήμης 5000h στον συσσωρευτή A
Σε αυτό το σημείο το πρόγραμμα παίρνει την τιμή από τη θέση μνήμης 5000 και μεταφέρει το περιεχόμενο στον A. Γενικότερα μας δείχνει από παίρνει το αποτέλεσμα.	
K8:SUB B	;Ξεκινάει βρόχος με την ετικέτα K8 όπου και αφαιρεί την τιμή A – B
JC K2	;γίνεται jump όταν υπάρχει carry στην ετικέτα K2 και όπως γνωρίζουμε παίρνουμε carry από την αφαίρεση όταν έχουμε αρνητικό αριθμό (< 0)
Σε αυτό το σημείο το πρόγραμμα ξεκινάει έναν βρόγχο όπου και αφαιρεί τον B από το A και αμέσως μετά γίνεται ο έλεγχος carry όπου και όταν είναι μικρότερος του 0 γίνεται jump στο K2. Διαφορετικά το πρόγραμμα συνεχίζει	
INR B	;Αυξάνεται ο B κατά 1 φορά
INR B	;Αυξάνεται ο B κατά 1 φορά
INR C	;Αυξάνεται ο C κατά 1 φορά
JMP K8	;είναι το τέλος του βρόγχου όπου και για να συνεχίσει η αναδρομικά γίνεται Jump στην ετικέτα K8 όπου είναι και η αρχή του βρόγχου
K2:INX H	;Ετικέτα K2 όπου και σε αυτό το σημείο έρχεται μετά από το τέλος του βρόγχου όπου και το ζεύγος καταχωρητή HL δείχνει πλέον την τιμή 5001h
MOV M,C	;Γίνεται αποθήκευση του περιεχομένου C στην θέση μνήμης HL που είναι πλέον ο 5001h
Σε αυτό το σημείο βλέπουμε πού αποθηκεύονται τα δεδομένα στην θέση μνήμης 5001 και αποθηκεύονται από τον καταχωρητή C	
HLT	;Εντολή τερματισμού προγράμματος

Αυτό που καταλαβαίνω το πρόγραμμα ότι διαχειρίζεται είναι ως εξής:

- Ο Β αρχικοποιείται με 1 και κάθε επανάληψη του βρόγχου προστίθεται +2 δηλαδή B=1,3,5,7,…..
- Ο C αρχικοποιείται με 0 και κάθε επανάληψη του βρόγχου προστίθεται +1 δηλαδή C=1,2,3,4……
- Α έχει τον αρχικό αριθμό που τον παίρνει από τη θέση μνήμης 5000 και κάθε επανάληψη του βρόγχου και κάθε επανάληψη αφαιρείται ο Β δηλαδή A= A – B μέχρι όπου όταν ο Α γίνει μικρότερος από το 0
- Και τέλος αποθηκεύει το αποτέλεσμα του μετρητή C στη θέση μνήμης 5001

Σαν πρόγραμμα και έπειτα από αρκετές δοκιμές συνειδητοποίησα ότι αυτό που επιτυγχάνεται είναι ότι ο μετρητής έχει το ακέραιο αποτέλεσμα μιας τετραγωνικής ρίζας
Παράδειγμα:
για A=9 το αποτέλεσμα του μετρητή C είναι 3
για A=11 το αποτέλεσμα του μετρητή C είναι 3(αντί του 3.3)
για A=16 το αποτέλεσμα του μετρητή C είναι 4
για A=25 το αποτέλεσμα του μετρητή C είναι 5
για A=30 το αποτέλεσμα του μετρητή C είναι 5(αντί του 5.47)

Επίλυση της Άσκησης 3
Υπό-ερώτημα γ)

Διεύθυνση	Περιεχόμενο	Μνημονικό	Εντολές
8000	06		MVI B,01
8001	01		
8002	0E		MVI C,00
8003	00		
8004	21		LXI H,5000
8005	00		
8006	50		
8007	7E		MOV A,M
8008	90	K8	K8:SUB B
8009	DA		JC K2
800A	12		
800B	80		
800C	04		INR B
800D	04		INR B
800E	0C		INR C
800F	C3		JMP K8
8010	08		
8011	80		
8012	23	K2	K2:INX H
8013	71		MOV M,C
8014	76		HLT

Επίλυση της Άσκησης 4

Λόγω του περιορισμένου αριθμών καταχωρητών που μπορώ να χρησιμοποιήσω θα επιλέξω τον HL για να μπορώ να μεταφέρομαι μεταξύ των μνημών 3001 → 2001 και αντίστροφα ανάλογα με το πλήθος του περιεχομένου της διεύθυνσης 3000 ώστε να καταγράφεται το bit ισοτιμίας. Όταν θα έχω άρτιους άσσους στον δυαδικό του περιεχόμενο της κάθε θέσης μνήμης 3001 έως 300X ανάλογα με το πλήθος του περιεχομένου της 3000 τότε θα καταγράφεται στην θέση μνήμης το bit 1 ισοτιμίας αντίστοιχα 2001 έως 300X διαφορετικά θα καταγράφεται το 0.

Επειδή βάση της εκφώνησης δεν μπορώ να χρησιμοποιήσω τη σημαία P parity θα χρησιμοποιήσω ολίσθηση με carry ώστε να μπορέσει το πρόγραμμα να κάνει το έλεγχο και επειδή είναι 8 bit η θέση μνήμης θα κάνει βρόχο με 8 επαναλήψεις ώστε να ελεγχθεί όλο το περιεχόμενο.

Θα έχει έναν εμφωλευμένο βρόχο: ο πρώτος βρόγχος θα περιέχει τις επαναλήψεις που θα χρειαστούν ώστε να μεταφερθούν όλα τα στοιχεία στο πρόγραμμα και να ελεγχθούν για την ισοτιμία και ο εμφωλευμένος βρόχος θα περιέχει τις εσωτερικές επαναλήψεις ώστε να ολοκληρωθούν οι ολισθήσεις.

Θα χρησιμοποιήσω τον καταχωρητή B ώστε να έχω το πλήθος των στοιχείων και να μπορώ να έχω τον βρόγχο επανάληψης που θα μειώνεται κατά 1 ως όταν μηδενιστεί. Είναι αυτό που σε κάθε βρόγχο θα προχωράει στην επόμενη θέση μνήμης ώστε να εισαχθούν τα στοιχεία

Θα χρησιμοποιήσω τον καταχωρητή C όπου θα περιέχει τις επαναλήψεις των ολισθήσεων δηλαδή 8 ώς ότου να μηδενιστεί που σημαίνει και τέλος εμφωλευμένου βρόγχου και προχωράει στο επόμενο στοιχείο της επόμενης θέσης μνήμης.

Θα χρησιμοποιήσω τον καταχωρητή D όπου και θα περιέχει το πλήθος των άσσων και όταν τελειώσει ο βρόγχος ανάλογα με τον αν θα είναι άρτιος η περιττός σε βρόγχο θα γράφει τα στοιχεία στην αντίστοιχη θέση μνήμης όπου είναι και το ζητούμενο της εκφώνησης το bit της ισοτιμίας.

LDA 3000h	; Loads στον A το πλήθος των στοιχείων του προγράμματος
MOV B, A	;Αντιγραφή το πλήθος των στοιχείων στο B
LXI H, 3001h	;Το πρώτο στοιχείο από την αρχική διεύθυνση μνήμης
LOS: MOV A, M	;Loads το πρώτο στοιχείο στον A και ετικέτα αρχικού βρόγχου επανάληψης
MVI C, 08h	;Αρχικοποίηση του πλήθους της ολίσθησης με 8 επαναλήψεις (8bit)
MVI D, 00h	;Γίνεται αρχικοποίηση του πλήθους των άσσων του βρόγχου που θα εξεταστεί παρακάτω
LIS: RRC	;Γίνεται ολίσθηση με carry του A
JNC LISJ1	;Αν το carry δεν είναι 1 τότε γίνεται jump στο LISJ1 (LoopInternalJump1)
INR D	;αύξηση του D κατά 1 που είναι ο αριθμός των άσσων
LISJ1: DCR C	;Μείωση κατά 1 του πλήθους του μετρητή βρόγχου
JNZ LIS	;Jump κατά τον έλεγχο του C μέχρι να ολοκληρωθούν οι 8 επαναλήψεις
MOV A, D	;Loads πλήθος των άσσων
RRC	; Έλεγχος το πλήθος των άσσων αν είναι άρτιοι οι περιττοί
JC LOSJ1	;Αν είναι περιττός jump στον LOSJ1 (LoopOutSideJump)

MOV A, H
SUI 10h
MOV H, A
	;μεταφορά του HL στον A ώστε να μειώσω τον H κατά 10 ώστε να δείχνει στην ;αντίστοιχη θέση μνήμης του 20χχ αντί του 30χχ

MVI M, 00h	;Γράφει το αποτέλεσμα του parity την τιμή 00 στη θέση μνήμης HL
JMP LOSJ2	;άλμα στην ετικέτα LOSJ2 για να μην ώστε ο βρόγχος να λειτουργήσει σαν if/else
LOSJ1: MOV A, H SUI 10h MOV H, A	;μεταφορά του HL στον A ώστε να μειώσω τον H κατά 10 ώστε να δείχνει στην ;αντίστοιχη θέση μνήμης του 20χχ αντί του 30χχ
MVI M, 01h	;Γράφει το αποτέλεσμα του parity την τιμή 01 στη θέση μνήμης HL
LOSJ2: MOV A, H ADI 10h MOV H, A	;μεταφορά του HL στον A ώστε να αυξήσω τον H κατά 10 ώστε να δείχνει στην ;αντίστοιχη θέση μνήμης του 30χχ αντί του 20χχ
INX H	;Αυξάνω τον HL κατά 1 ώστε να προχωρήσω στο επόμενο στοιχείο ως προς έλεγχο
DCR B	;αφαιρώ 1 από τον μετρητή επαναλήψεων του βρόγχου που πήραμε από τη θέση 3000 ώς όταν μηδενιστεί και τερματιστεί
JNZ LOS	;κάνει jump στην αρχή του βρόγχου μέχρι να γίνει 0
HLT	Εντολή τερματισμού προγράμματος