Υποπρογράμματα (Subroutines)

- Τα υποπρογράμματα είναι τμήματα κώδικα που συγκεντρώνουν ένα σύνολο από εντολές το οποίο θα εκτελέσει πλήρως μια λειτουργία και πιθανόν να επιστρέψει και τιμή.
- Τα υποπρογράμματα είναι απαραίτητα στοιχεία του δομημένου προγραμματισμού. Δεδομένου ότι κάθε υποπρόγραμμα μπορεί να γραφτεί, να ελέγχει και να δοκιμαστεί ανεξάρτητα από τα υπόλοιπα, είναι ευκολότερη η υλοποίηση και η επαλήθευση ενός προγράμματος που αποτελείται από υποπρογράμματα.
- Η διάσπαση ενός προγράμματος σε υποπρογράμματα κάνει το πρόγραμμα ευκολοκατανόητο γιατί το κάθε υποπρογράμματα μπορεί να διαβαστεί και να κατανοηθεί ξεχωριστά.

Υποπρογράμματα (Subroutines)

- Ένα υποπρόγραμμα μπορεί να κληθεί από διάφορα σημεία του προγράμματος, με διαφορετικές αν χρειάζεται πραγματικές παραμέτρους.
- Μπορεί επίσης για λόγους αυτονομίας να χρησιμοποιεί τοπικές μεταβλητές οι οποίες υπάρχουν μόνο όσο χρόνο εκτελείται το υποπρόγραμμα αυτό.
- Το πρόγραμμα αποτελείται από το κύριο τμήμα το οποίο καλεί διάφορα υποπρογράμματα τα όποια με την σειρά τους πιθανόν να καλούν και αλλά υποπρογράμματα.

Υποπρογράμματα (Subroutines)

- Κάθε υποπρόγραμμα έχει συγκεκριμένο όνομα, η αρχή του δηλώνεται με την οδηγία **PROC** και το πέρας του με την οδηγία **ENDP**. Το όνομα του υποπρογράμματος είναι ετικέτα που αντιπροσωπεύει την λογική δ/νση της αρχής του κώδικα του υποπρογράμματος.
- Η ύπαρξη σωρού απαραίτητη για την λειτουργία των υποπρογραμμάτων.
- Κάθε υποπρόγραμμα καλείται με την εντολή **CALL** και το όνομα του Π.χ **CALL** `emfanish__char`
- Κάθε υποπρόγραμμα επιστέφει στο κυρίως πρόγραμμα ή στο υποπρόγραμμα που το κάλεσε με την εντολή **RET**.
- Κάθε υποπρόγραμμα δηλώνεται **NEAR** ή **FAR** και έχει τουλάχιστον μια εντολή **RET**.

ΕΝΤΟΛΕΣ CALL ΚΑΙ RET

Η θεμελιώδης λειτουργία των υποπρογραμμάτων επιτυγχάνεται με τον συνδυασμό των εντολών **CALL** και **RET** και με την βοήθεια του σωρού, κατά τον εξής τρόπο:

- Η εντολή CALL αποθηκεύει την δ/νση της εντολής μετά την CALL στο σωρό, και μεταφερεί με άλμα τον έλεγχο του προγράμματος στην δ/νση του υποπρογράμματος της CALL.
- Μόλις το υποπρόγραμμα τελειώσει, εκτελεί μια εντολή RET, η οποία ανακαλεί από τον σωρό την δ/νση επιστροφής και την φορτώνει στον μετρητή προγράμματος για να συνεχιστεί η εκτέλεση του προγράμματος στην εντολή μετά την CALL που κάλεσε το υποπρόγραμμα.

Εντολή CALL

Υπάρχουν δυο μορφές κλήσεων με την εντολή CALL, η ενδοτμηματική και η εξωτμηματική. Υπάρχουν δυο μορφές της εντολής επιστροφής RET.

Όταν το υποπρόγραμμα που βρίσκεται η CALL είναι NEAR πρόκειται για ενδοτμηματική CALL (near ptr call), αλλιώς αν είναι FAR τότε η CALL είναι εξωτμηματική (far ptr call) σε άλλο code segment.

Στην ενδοτμηματική κλήση μεταφέρεται στην κορυφή του σωρού το περιεχόμενο του IP και μειώνεται ο δείκτης σωρού SP κατά 2. Ακολούθως συνεχίζεται η εκτέλεση του προγράμματος στην δ/νση του υποπρογράμματος η οποία όμως πρέπει να βρίσκεται στο ίδιο τμήμα κώδικα.

Εντολή CALL

- Στην εξωτμηματική κλήση μεταφέρεται πρώτα στην κορυφή του σωρού το περιεχόμενο του καταχωρητή τμήματος κώδικα CS και μειώνεται ο δείκτης σωρού SP κατά 2 και ακολούθως μεταφέρεται στην κορυφή του σωρού το περιεχόμενο του IP μειώνεται πάλι ο δείκτης σωρού SP.
- Η εκτέλεση του προγράμματος συνεχίζεται στην δ/νση του υποπρογράμματος η οποία μπορεί να βρίσκεται σε διαφορετικό τμήμα κώδικα από το τμήμα εντολής CALL.
- Στην περίπτωση αυτή πριν από το όνομα του υποπρογράμματος πρέπει να μπεί η έκφραση **FAR PTR** αν δεν υπάρχουν οι δηλώσεις NEAR ή FAR π.χ.

```
CALL FAR PTR emfanish__char
```

Εντολή RET

- Υπάρχουν δυο μορφές της εντολής επιστροφής RET. Όταν το υποπρόγραμμα που βρίσκεται η RET είναι NEAR πρόκειται για ενδοτμηματική RET (RETN), αλλιώς αν είναι FAR τότε η RET είναι εξωτμηματική (RETF).
- Στην ενδοτμηματική RET μεταφέρεται από την κορυφή του σωρού το περιεχόμενο στον IP και αυξάνεται ο δείκτης σωρού SP κατά 2. Ακολούθως συνεχίζεται η εκτέλεση του προγράμματος στην μετά την CALL εντολή.

Εντολή `RET`

- Στην εξωτμηματική `RET` μεταφέρεται από την κορυφή του σωρού το περιεχόμενο του IP αυξάνεται ο δείκτης σωρού SP κατά 2 και ακολούθως μεταφέρεται από νέα κορυφή σωρού το περιεχόμενο στον CS αυξάνεται πάλι ο δείκτης σωρού SP κατά 2 και συνεχίζεται η εκτέλεση του προγράμματος.
- Όπως φαίνεται από τα παραπάνω πρέπει ο συνδυασμός των εντολών `CALL` και `RET` να γίνεται με προσοχή ούτως ώστε και οι δυο να είναι του ίδιου τύπου δηλαδή ή ενδοτμηματικές ή εξωτμηματικές.

ΣΩΡΟΣ (STACK)

Καταχωρητής τμήματος σωρού SS(Stack Segment)

Ο καταχωρητής SS περιέχει την δ/νση από όπου αρχίζει το τμήμα μνήμης της σωρού (STACK).

Δείκτης βάσης BP (Base Pointer)

Χρησιμοποιείται για προσπέλαση δεδομένων στο σωρό (τοπικές μεταβλητές, παράμετροι υποπρογραμμάτων).

Δείκτης σωρού SP (Stack Pointer)

Δείχνει την πρώτη ελεύθερη θέση στο σωρό. Δηλαδή το σημείο μέχρι το οποίο έχει γεμίσει ο σωρός

Π.χ. αν SS = 1ABCh και SP = 100h, τότε η φυσική δ/νση της κορυφής του σωρού είναι:

SS * 10h + SP = 1ABCh * 10h + 100h = 1ABC0h + 100h = 1ACC0h

```
ASSUME CS:KODIKAS, DS:DEDOMENA, SS:SOROS
KODIKAS SEGMENT PUBLIC
    MAIN PROC NEAR
        MOV AX,DEDOMENA          ; Apokatastash tou DS
        MOV DS,AX                ; Apokatastash tou DS
        .....
        CALL DISPLAY-HEX         ; Kaloume tin Display_hex
        MOV AH,4CH               ; Eksodos sto leitoyrgiko systhma
        INT 21H
    MAIN ENDP
    DISPLAY-HEX PROC NEAR
        .....
        CALL ONE-DIGIT  ; Kaloume tin One_digit
        .....
        RET                      ; Epistrefo stin thesi apo opou klithike h yporoutina
    DISPLAY-HEX ENDP
    ONE-DIGIT PROC NEAR
        .....
    TELOS:      RET              ; Epistrefo stin thesi apo opou klithike h yporoutina
    ONE-DIGIT ENDP
KODIKAS ENDS
DEDOMENA SEGMENT
    ......
DEDOMENA ENDS
SOROS SEGMENT STACK
    DB 256 DUP(0)
SOROS ENDS
    END MAIN
```

Πέρασμα παραμέτρων

- Η ενέργεια με την οποία μεταβιβάζουμε δεδομενα διαφορετικά κάθε φορά στα υποπρογράματα για να τα καθοδηγήσουμε τι να κάνουν, λέγεται πέρασμα παραμέτρων.
- Υπάρχουν δύο μέθοδοι για πέρασμα παραμέτρων:
  1. Πέρασμα μέσω καταχωρητών
  2. Πέρασμα μέσω σωρού
- Η μέθοδος του περάσματος μέσω καταχωρητών είναι πιο απλή και χρησιμοποιείται σε προγράμματα αποκλειστικά σε γλώσσα ASSEMBLY
- Η μέθοδος του περάσματος μέσω σωρού είναι σχετικά πιο δυσχερής μέθοδος, λιγότερο ευέλικτη και απαιτεί περισσότερο κώδικα. Την χρησιμοποιούν όλες οι γλώσσες 3^{ης} γενιάς και άνω.

Πέρασμα μέσω καταχωρητών

Στην περίπτωση αυτή επιλέγονται οι κατάλληλοι καταχωρητές σαν παράμετροι οι οποίοι βέβαια είναι οι ίδιοι σε κάθε κλήση του υποπρογράμματος.

Γι' αυτόν τον λόγο πρέπει πριν από τον κώδικα κλήσης του υποπρογράμματος, με κατάλληλα σχόλια, να αναγράφονται όλοι οι καταχωρητές που χρησιμοποιούνται σαν παράμετροι.

Π.χ.

```
MOV DL,AL   ;O DL GIA PERASMA
CALL EMFANISH_CHAR
```

Πέρασμα μέσω σωρού

- Στη περίπτωση αυτής της μεθόδου περάσματος, το πρόγραμμα που καλεί μεταφέρει στον σωρό με εντολές `PUSH` τις παραμέτρους, ενώ το υποπρόγραμμα ανακτά από το σωρό τις παραμέτρους με τη βοήθεια του καταχωρητή δείκτη BP.
- Μετά την επιστροφή από το υποπρόγραμμα (μετά την εντολή `CALL`) πρέπει να αποκαθίσταται ο σωρός.
- Η χρήση της μεθόδου είναι υποχρεωτική εφόσον το υποπρόγραμμα πρόκειται να συνδεθεί και να κληθεί από πρόγραμμα γλώσσας 3^{ης} γενιάς.

Π.χ.

```
LEA AX,MINIMA
PUSH AX ;O AX MPAINEI STI SORO GIA PERASMA
CALL EMFANISI_MINIMA
```