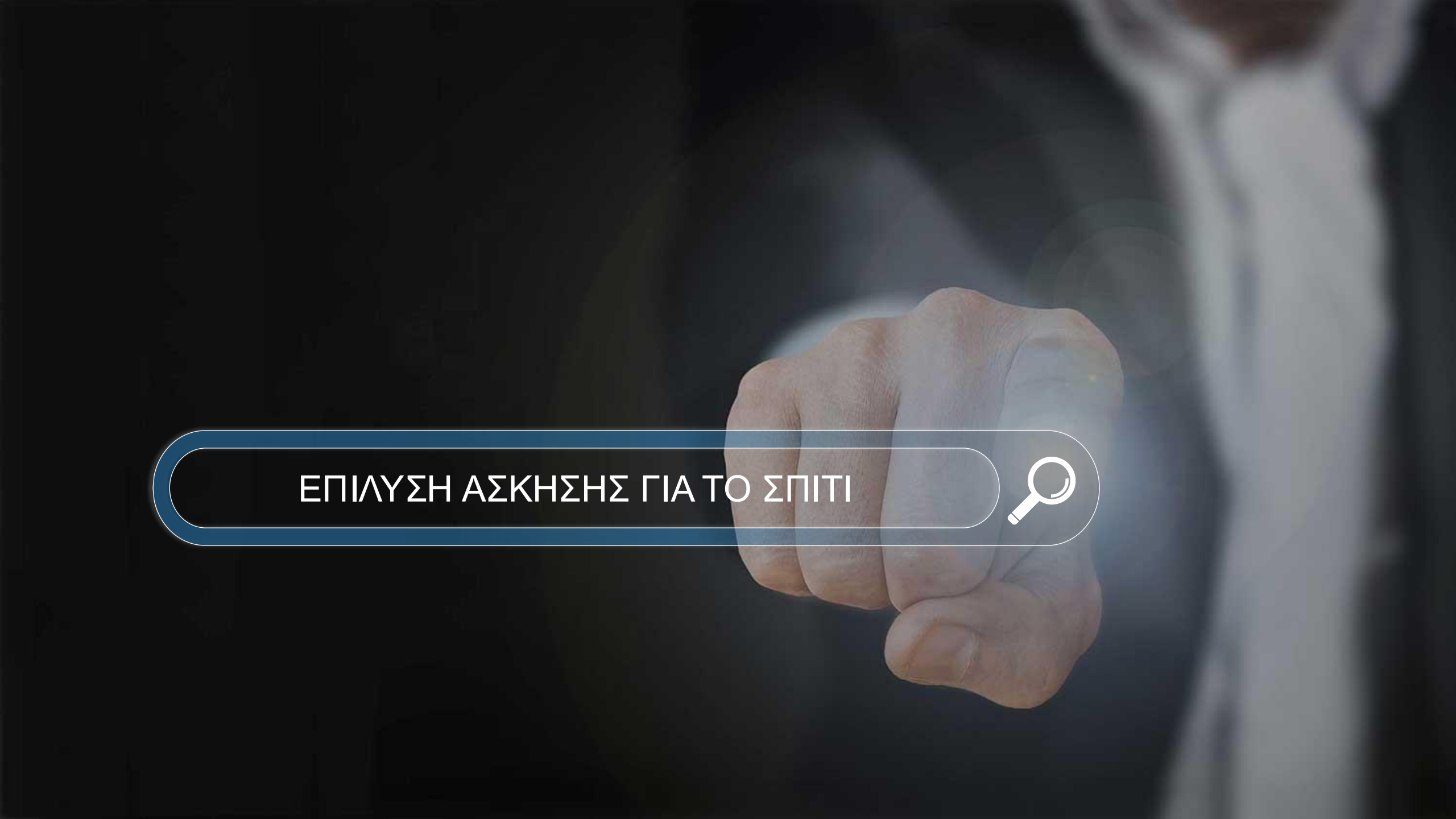

ΕΠΙΛΥΣΗ ΑΣΚΗΣΗΣ ΓΙΑ ΤΟ ΣΠΙΤΙ

ΑΣΚΗΣΗ ΣΠΙΤΙ

- **ΑΝΤΙΜΕΤΑΘΕΣΗ ΠΙΝΑΚΩΝ – ΠΑΡΑΛΛΗΛΟΙ ΠΙΝΑΚΕΣ**
- **ΣΕΙΡΙΑΚΗ ΑΝΑΖΗΤΗΣΗ ΚΑΤΑ ΓΡΑΜΜΕΣ ΚΑΙ ΣΤΗΛΕΣ ΣΕ ΔΙΣΔΙΑΣΤΑΤΟ ΠΙΝΑΚΑ**
- **ΕΥΡΕΣΗ ΠΛΗΘΟΥΣ**

Αγορά (σε €)	Άτοκες δόσεις
100 - 200	3
201 - 500	6
501 και πάνω	10

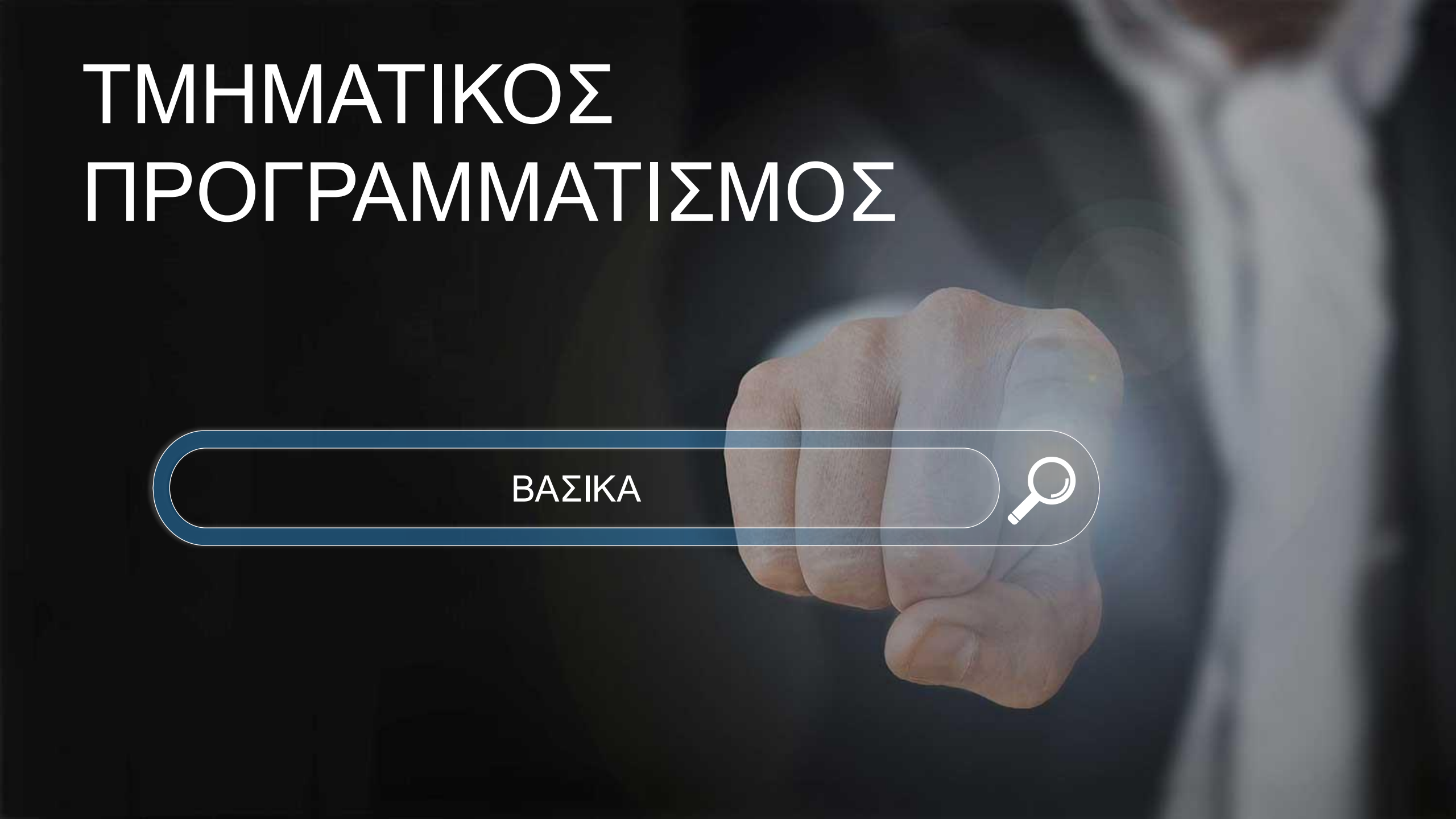
Τη στιγμή που η πωλήτρια του καταστήματος ρούχων περνά κάποιο προϊόν από το σαρωτή, η ταμειακή αναζητά στη ΒΔ τις πληροφορίες που αφορούν το προϊόν. Οι δομές είναι: 1D πίνακας Κ, με 2000 στοιχεία – όσα και τα προϊόντα – με τους κωδικούς τους, ένας παράλληλος πίνακας Τ με τις τιμές τους και παράλληλος πίνακας Π με τις σύντομες περιγραφές τους. ΝΑΠ που θα διαβάζει επαναληπτικά τον κωδικό του προϊόντος που περνά από το σαρωτή, θα εμφανίζει την περιγραφή του στην οθόνη και θα υπολογίζει το κόστος του στο συνολικό κόστος της αγοράς. Η διαδικασία θα ολοκληρώνεται όταν ως κωδικός εισαχθεί το #

Στο σημείο αυτό ο χρήστης θα ενημερώνεται ότι μπορεί να εξοφλήσει το ποσό σε άτοκες δόσεις, σύμφωνα με τον διπλανό πίνακα

Το πρόγραμμα θα πρέπει στο τέλος να εκτυπώνει το πλήθος των δόσεων, καθώς και το ποσό κάθε δόσης

```
ΠΡΟΓΡΑΜΜΑ ΚΑΤΑΣΤΗΜΑ
ΜΕΤΑΒΛΗΤΕΣ
    ΧΑΡΑΚΤΗΡΕΣ: Κ[2000], Π[2000], κωδικός
    ΑΚΕΡΑΙΕΣ: i, pos, πλήθος_δόσεων
    ΠΡΑΓΜΑΤΙΚΕΣ: Τ[2000], ποσό_δόσης
    ΛΟΓΙΚΕΣ: flag
ΑΡΧΗ
    ! Δημιουργία των 3 μονοδιάστατων πινάκων για τα προϊόντα
    ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ 2000
        ΔΙΑΒΑΣΕ Κ[i] ! Καταχώρηση των κωδικών
        ΔΙΑΒΑΣΕ Τ[i] ! Καταχώρηση των τιμών
        ΔΙΑΒΑΣΕ Π[i] ! Καταχώρηση των περιγραφών
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

    ΓΡΑΨΕ 'Δώσε κωδικό προϊόντος που σε ενδιαφέρει'
    ΔΙΑΒΑΣΕ κωδικός
    ΟΣΟ (κωδικός <> '#') ΕΠΑΝΑΛΑΒΕ
        ! Παρακάτω αναζητούμε τον κώδικό που διαβάστηκε στον αντίστοιχο πίνακα
        flag <- ΨΕΥΔΗΣ
        pos <- 0
        i <- 1

        ΟΣΟ (i <= 2000 ΚΑΙ flag = ΨΕΥΔΗΣ) ΕΠΑΝΑΛΑΒΕ
            ΑΝ Κ[i] = κωδικός ΤΟΤΕ
                flag <- ΑΛΗΘΗΣ
                pos <- i
            ΑΛΛΙΩΣ
                i <- i + 1
            ΤΕΛΟΣ_ΑΝ
        ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

        ΑΝ (flag = ΑΛΗΘΗΣ) ΤΟΤΕ
            ΓΡΑΨΕ Π[pos] ! Αν βρέθηκε το προϊόν εμφανίζουμε την περιγραφή του
            ΓΡΑΨΕ Τ[pos] ! Εμφανίζουμε και την συνολική τιμή του προϊόντος

            ΑΝ (Τ[pos] < 100) ΤΟΤΕ
                πλήθος_δόσεων <- 1
            ΑΛΛΙΩΣ_ΑΝ (Τ[pos] <= 200) ΤΟΤΕ
                πλήθος_δόσεων <- 3
            ΑΛΛΙΩΣ_ΑΝ (Τ[pos] <= 500) ΤΟΤΕ
                πλήθος_δόσεων <- 6
            ΑΛΛΙΩΣ
                πλήθος_δόσεων <- 10
            ΤΕΛΟΣ_ΑΝ

            ποσό_δόσης <- Τ[pos] / πλήθος_δόσεων ! Υπολογίζουμε το ποσό της κάθε δόσης (αν είναι 1 δόση τότε θα επιστραφεί όλο το ποσό)

            ΓΡΑΨΕ 'Το προϊόν με κωδικό ', Κ[pos], ' μπορείτε να το αποπληρώσετε σε ', πλήθος_δόσεων, ' δόσεις'
            ΓΡΑΨΕ 'Το ποσό της δόσης είναι: ', ποσό_δόσης
        ΑΛΛΙΩΣ
            ΓΡΑΨΕ 'Το προϊον με αυτό το κωδικό δεν υπάρχει...'
        ΤΕΛΟΣ_ΑΝ

        ΓΡΑΨΕ 'Δώσε άλλο κωδικό ή τερμάτισε το πρόγραμμα με το #'
        ΔΙΑΒΑΣΕ κωδικός
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

    ΓΡΑΨΕ 'Σε ευχαριστούμε για την προτίμηση!'
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

ΤΜΗΜΑΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ




ΘΕΩΡΙΑ

Η τεχνική σχεδίασης και ανάπτυξης υποπρογραμμάτων ως ένα σύνολο από απλούστερα τμήματα προγραμμάτων	Εξασφαλίζει επιτυχή και εύκολη δημιουργία σωστών προγραμμάτων	Υλοποιεί την φιλοσοφία της ιεραρχικής σχεδίασης

Πρακτικά μέσω του τμηματικού προγραμματισμού μας επιτρέπεται η διαίρεση ενός σύνθετου προβλήματος σε άλλα μικρότερα και η επίλυση καθενός από αυτά με την υλοποίηση αντίστοιχου υποπρογράμματος. Το κύριο πρόγραμμα θα παίζει τον ρόλο του συντονιστή και θα μπορεί να καλεί τα αντίστοιχα υποπρογράμματα που χρειάζεται περνώντας τα δεδομένα που αυτά χρειάζονται για την λειτουργία τους. Τα υποπρογράμματα μπορούν να επικοινωνούν και μεταξύ τους με κανόνες που θα αναλύσουμε στην συνέχεια.

ΤΙ ΕΙΝΑΙ ΤΟ ΥΠΟΠΡΟΓΡΑΜΜΑ;	Υποπρόγραμμα είναι ένα τμήμα προγράμματος που επιτελεί αυτόνομο έργο και γράφεται ξεχωριστά από το υπόλοιπο πρόγραμμα και επιτελεί κάποια αυτόνομη λειτουργία του προγράμματος. Κάθε υποπρόγραμμα δουλεύει αυτόνομα και διαθέτει δική του περιοχή μνήμης, ενώ η επαφή που έχει με άλλα τμήματα προγράμματος γίνεται αποκλειστικά και μόνο μέσω παραμέτρων. ΤΑ ΥΠΟΠΡΟΓΡΑΜΜΑΤΑ ΕΚΤΕΛΟΥΝΤΑΙ ΜΟΝΟ ΌΤΑΝ ΤΑ ΚΑΛΕΣΟΥΜΕ Έχουμε 2 ειδών υποπρογράμματα που θα αναλύσουμε. ΔΙΑΔΙΚΑΣΙΕΣ | ΣΥΝΑΡΤΗΣΕΙΣ
3 ΙΔΙΟΤΗΤΕΣ ΤΩΝ ΥΠΟΠΡΟΓΡΑΜΜΑΤΩΝ	1. Πρέπει να έχουν μία είσοδο και μια έξοδο 2. Να είναι ανεξάρτητα 3. Να μην είναι πολύ μεγάλα 4. Η είσοδος δεδομένων γίνεται πάντα στην αρχή τους
ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΤΟΥ ΤΜΗΜΑΤΙΚΟΥ ΠΡΟΓΡ/ΜΟΥ	1. Διευκόλυνση της ανάπτυξης του αλγορίθμου και του αντίστοιχου προγράμματος (Διαίρεση προβλήματος σε μικρότερα) 2. Διευκόλυνση της κατανόησης και της διόρθωσης του προγράμματος 3. Λιγότερος χρόνος και κώδικα για την **συγγραφή του προγράμματος (επαναχρησιμοποίηση υποπρογραμμάτων)** 4. Επεκτείνει τις δυνατότητες των γλωσσών προγραμματισμού
ΤΙ ΕΙΝΑΙ ΠΑΡΑΜΕΤΡΟΣ ΕΝΟΣ ΥΠΟΠΡΟΓΡΑΜΜΑΤΟΣ	Είναι μια μεταβλητή που επιτρέπει το πέρασμα τιμών από ένα υποπρόγραμμα σε ένα άλλο, επιτρέποντας την ανταλλαγή δεδομένων μεταξύ τους

ΣΥΝΑΡΤΗΣΕΙΣ

Η συνάρτηση είναι ένας από τους 2 τύπους υποπρογραμμάτων. Υπολογίζει και επιστρέφει μόνο μια τιμή με το όνομα της. Δεν χρησιμοποιείται για κανέναν άλλο λόγο(διάδραση με το χρήστη, επιστροφή πολλαπλών τιμών κτλ.)

```
ΣΥΝΑΡΤΗΣΗ [ΟΝΟΜΑ]([ΠΑΡΑΜΕΤΡΟΙ]) : [ΤΥΠΟΣ]
ΣΤΑΘΕΡΕΣ
    ...
ΜΕΤΑΒΛΗΤΕΣ
    ...
    ...
ΑΡΧΗ
    [ΕΝΤΟΛΕΣ]
    ...
    [ΟΝΟΜΑ] <- [ΕΚΦΡΑΣΗ]
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

Είσοδος – παράμετροι εισόδου - **Υποχρεωτικά**

Τύπος της τιμής που επιστρέφει η συνάρτηση

Υποχρεωτικά η συνάρτηση υπολογίζει και επιστρέφει μια τιμή στο πρόγραμμα ή την διαδικασία που την κάλεσε, περνώντας στο όνομα της συνάρτησης(σαν να είναι μεταβλητή) το αποτέλεσμα κάποιας έκφρασης ή σταθεράς

ΠΟΥ ΚΑΙ ΠΩΣ ΚΑΛΕΙΤΑΙ ΜΙΑ ΣΥΝΑΡΤΗΣΗ;

Μια συνάρτηση μπορεί να κληθεί από οποιοδήποτε τμήμα προγράμματος:

- Από το κύριο πρόγραμμα
- Από διαδικασία
- Από κάποια άλλη συνάρτηση

Η κλήση μιας συνάρτησης πραγματοποιείται με το όνομα της ακολουθούμενο από τις παραμέτρους εισόδου που χρειάζεται.

ΙΔΙΟΤΗΤΕΣ ΣΥΝΑΡΤΗΣΕΩΝ

1. Η λίστα παραμέτρων για την συνάρτηση είναι υποχρεωτική και αποτελεί είσοδο δεδομένων για τη συνάρτηση
2. Ο τύπος μιας συνάρτησης που αποτελεί τον τύπο της επιστρεφόμενης τιμής μπορεί να είναι ΑΚΕΡΑΙΑ, ΧΑΡΑΚΤΗΡΑΣ, ΠΡΑΓΜΑΤΙΚΗ, ΛΟΓΙΚΗ
3. Η εκχώρηση τιμής μέσω έκφρασης στο όνομα της συνάρτησης είναι υποχρεωτική καθώς αποτελεί την τιμή επιστροφής της συνάρτησης
4. Οι παράμετροι, οι σταθερές και οι μεταβλητές που θα χρησιμοποιηθούν εντός της συνάρτησης θα πρέπει υποχρεωτικά να δηλώνονται
5. ΔΕΝ χρησιμοποιούμε τις εντολές ΓΡΑΨΕ / ΔΙΑΒΑΣΕ στη συνάρτηση(δεν είναι ο σκοπός τους η διάδραση με το χρήστη)
6. Ως παραμέτρους εισόδου σε μια συνάρτηση μπορούμε να περνάμε κατά την κλήση της μεταβλητές αλλά και σταθερές

ΠΑΡΑΔΕΙΓΜΑ 1 - ΣΥΝΑΡΤΗΣΕΙΣ

Δημιουργήστε ένα υποπρόγραμμα(διαδικασία ή συνάρτηση), το οποίο θα δέχεται 3 πραγματικές μή μηδενικές τιμές και θα επιστρέφει τον μέσο όρο τους. Στην συνέχεια δημιουργήστε ένα πρόγραμμα το οποίο θα διαβάζει από το χρήστη 3 πραγματικές θετικές τιμές και χρησιμοποιώντας το υποπρόγραμμα που υλοποιήσατε θα εμφανίζει στον χρήστη τον μέσο όρο τους

```
ΣΥΝΑΡΤΗΣΗ ΥΠΟΛΟΓΙΣΜΟΣ_ΜΟ(x,y,z): ΠΡΑΓΜΑΤΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: x, y, z
ΑΡΧΗ
    ΥΠΟΛΟΓΙΣΜΟΣ_ΜΟ <- (x + y + z) / 3
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

Δημιουργούμε μια συνάρτηση(επιλέγουμε συνάρτηση γιατί θέλουμε τον υπολογισμό και την επιστροφή μιας τιμής). Η συνάρτηση δέχεται 3 **τυπικές παραμέτρους** για τις οποίες χρησιμοποιούμε τα ονόματα x,y,z(εμείς επιλέγουμε την ονομασία τους)

Όπως εξηγήσαμε, η επιστρεφόμενη τιμή της συνάρτησης εκχωρείται στο όνομα της, ενώ ορίζουμε και τον τύπο της επιστρεφόμενης αυτής τιμής που είναι πραγματική

Οι τυπικές παράμετροι x,y,z και (αν υπήρχαν) οι σταθερές και οι υπόλοιπες μεταβλητές θα έπρεπε να δηλωθούν τοπικά στη συνάρτηση

```
ΠΡΟΓΡΑΜΜΑ ΑΣΚΗΣΗ1
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: a,b,c,ΜΟ
    ΛΟΓΙΚΕΣ: ΕΓΚΥΡΑ
ΑΡΧΗ
    ΓΡΑΨΕ 'ΔΩΣΕ 3 ΑΡΙΘΜΟΥΣ ΤΗΣ ΑΡΕΣΚΕΙΑΣ ΣΟΥ'
    ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ
        ΕΓΚΥΡΑ <- ΑΛΗΘΗΣ
        ΔΙΑΒΑΣΕ a,b,c
        ΑΝ a<=0 Η b<=0 Η c<=0 ΤΟΤΕ
            ΕΓΚΥΡΑ <- ΨΕΥΔΗΣ
            ΓΡΑΨΕ 'Δώσε πραγματικές θετικές μη μηδενικές τιμές...'
        ΤΕΛΟΣ_ΑΝ
    ΜΕΧΡΙΣ_ΟΤΟΥ (ΕΓΚΥΡΑ)
    ΜΟ <- ΥΠΟΛΟΓΙΣΜΟΣ_ΜΟ(a,b,c)
    ΓΡΑΨΕ 'Ο μεσος όρος των 3 αριθμών είναι: ', ΜΟ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Στο κυρίως πρόγραμμα κατά τα γνωστά, διαβάζουμε από το χρήστη 3 πραγματικές τιμές πραγματοποιώντας και έλεγχο δεδομένων(να είναι θετικές πραγματικές τιμές)

Αφού ο χρήστης εισάγει έγκυρες πραγματικές τιμές τότε καλούμε την συνάρτηση με το όνομα της, περνώντας τις **πραγματικές παραμέτρους a,b,c και ταυτόχρονα εκχωρούμε την επιστρεφόμενη τιμή της συνάρτησης σε μια νέα μεταβλητή για περαιτέρω χρήση**

ΑΣΚΗΣΗ 2 - ΣΥΝΑΡΤΗΣΕΙΣ

Δημιουργήστε κατάλληλα υποπρογράμματα τα οποία:

1. Το πρώτο θα δέχεται 2 ακέραιους αριθμούς και θα επιστρέφει τον μεγαλύτερο από αυτούς
2. Υποπρόγραμμα το οποίο θα δέχεται 4 ακέραιους αριθμούς και χρησιμοποιώντας το προηγούμενο υποπρόγραμμα θα επιστρέφει τον μεγαλύτερο από αυτούς

```
ΣΥΝΑΡΤΗΣΗ ΜΕΓΙΣΤΟ2(k,z): ΑΚΕΡΑΙΑ
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: k, z, max
ΑΡΧΗ
    max <- k
    ΑΝ z > k ΤΟΤΕ
        max <- k
    ΤΕΛΟΣ_ΑΝ
    ΜΕΓΙΣΤΟ2 <- max
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

```
ΣΥΝΑΡΤΗΣΗ ΜΕΓΙΣΤΟ4(x,y,z,w): ΑΚΕΡΑΙΑ
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: x, y, z, w, max1, max2, maximum
ΑΡΧΗ
    max1 <- ΜΕΓΙΣΤΟ2(x,y)
    max2 <- ΜΕΓΙΣΤΟ2(z,w)

    maximum <- ΜΕΓΙΣΤΟ2(max1, max2)
    ΜΕΓΙΣΤΟ4 <- maximum
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

Μια συνάρτηση μπορεί να κληθεί από οποιοδήποτε άλλο τμήμα προγράμματος(κυρίως πρόγραμμα, διαδικασία, άλλη συνάρτηση). Εδώ η μια συνάρτηση καλεί την άλλη

ΑΣΚΗΣΗ 3 - ΣΥΝΑΡΤΗΣΕΙΣ

Δημιουργήστε κατάλληλο υποπρόγραμμα το οποία θα δέχεται ως παράμετρο έναν πραγματικό αριθμό και ακέραιο αριθμό που θα αντιπροσωπεύει το δεκαδικό του ψηφίο. Το υποπρόγραμμα θα υπολογίζει τον στρογγυλοποιημένο αριθμό στο ζητούμενο δεκαδικό ψηφίο

Στην συνέχεια δημιουργήστε ένα πρόγραμμα το οποίο θα διαβάζει έναν πραγματικό αριθμό και έναν ακέραιο αριθμό για τα δεκαδικά από το χρήστη και χρησιμοποιώντας το προηγούμενο υποπρόγραμμα θα εμφανίζει τον αριθμό στρογγυλοποιημένο

```
ΣΥΝΑΡΤΗΣΗ ROUNDING(k,z): ΠΡΑΓΜΑΤΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: z, integer_part
    ΠΡΑΓΜΑΤΙΚΕΣ: k, initial, decimal_part
ΑΡΧΗ
    initial <- k * 10^z
    integer_part <- A_M(initial)
    decimal_part <- initial - integer_part

    ΑΝ decimal_part >0.5 ΤΟΤΕ
        integer_part <- integer_part + 1
    ΤΕΛΟΣ_ΑΝ
    ROUNDING <- integer_part / 10 ^ z
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

```
ΠΡΟΓΡΑΜΜΑ ΑΣΚΗΣΗ1
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: number, rounded_number
    ΑΚΕΡΑΙΕΣ: floating_point
ΑΡΧΗ
    ΓΡΑΨΕ 'Δώσε πραγματικό αριθμό...'
    ΔΙΑΒΑΣΕ number
    ΓΡΑΨΕ 'Δώσε ψηφιο δεκαδικών...'
    ΔΙΑΒΑΣΕ floating_point
    rounded_number <- ROUNDING(number,floating_point)
    ΓΡΑΨΕ 'Το αποτέλεσμα της στρογγυλοποίησης είναι ', rounded_number
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

ΔΙΑΔΙΚΑΣΙΕΣ

Η διαδικασία είναι ο 2ος τύπος υποπρογράμματος που μπορεί να εκτελεί όλες τις λειτουργίες ενός προγράμματος και δεν περιορίζεται στον υπολογισμό και την επιστροφή κάποιας τιμής όπως η συνάρτηση

```
ΔΙΑΔΙΚΑΣΙΑ [ΟΝΟΜΑ]([ΠΑΡΑΜΕΤΡΟΙ])
ΣΤΑΘΕΡΕΣ
    ...
ΜΕΤΑΒΛΗΤΕΣ
    ...
    ...
ΑΡΧΗ
    [ΕΝΤΟΛΕΣ]
    ...
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Είσοδος ΚΑΙ έξοδος – παράμετροι εισόδου και εξόδου

Όπως παρατηρούμε, η διαδικασία δεν εκχωρεί στο όνομα της το αποτέλεσμα κάποιας έκφρασης καθώς έχει την δυνατότητα να επιστρέφει παραπάνω από μια τιμή μέσω των παραμέτρων εισόδου-εξόδου. Επίσης δεν δηλώνεται ο τύπος δεδομένων της ακριβώς γιατί δεν επιστρέφει μια μοναδική τιμή

ΧΡΗΣΙΜΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΔΙΑΔΙΚΑΣΙΑΣ

- Σε μια διαδικασία άλλες παράμετροι χρησιμοποιούνται για είσοδο δεδομένων, άλλες για έξοδο δεδομένων και άλλες και για τα δύο
- Σε μια διαδικασία η λίστα παραμέτρων ΔΕΝ είναι υποχρεωτική
- Η κλήση μιας διαδικασίας πραγματοποιείται με την εντολή ΚΑΛΕΣΕ ακολουθούμενη από το όνομα της διαδικασίας μαζί με τις (προαιρετικές) παραμέτρους εισόδου/εξόδου. Ενώ η λίστα παραμέτρων για τη συνάρτηση είναι υποχρεωτική, στην διαδικασία κάτι τέτοιο δεν ισχύει
- Μια διαδικασία επιστρέφει τιμές(αν το κάνει δεν είναι υποχρεωτικό) μέσω των παραμέτρων της. Η διαδικασία σε αντίθεση με την συνάρτηση μπορεί να επιστρέφει παραπάνω της μιας τιμής
- Αν η διαδικασία έχει παραμέτρους εισόδου-εξόδου, αυτές επιστρέφουν στο μέρος προγράμματος που την κάλεσε, αφού ολοκληρωθεί η εκτέλεση του κώδικα της διαδικασίας και η ροή εκτέλεσης συνεχίζεται από εκεί
- Μια διαδικασία καλείται από το κύριο πρόγραμμα ή από άλλη διαδικασία (δεν μπορεί να κληθεί από συνάρτηση)

ΠΡΟΣΟΧΗ

- Μια διαδικασία επιστρέφει **ΟΛΕΣ ΤΙΣ ΤΙΜΕΣ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΗΣ ΠΟΥ ΜΕΤΑΒΑΛΛΟΝΤΑΙ ΜΕΣΑ ΣΕ ΑΥΤΗΝ**.
- Αν στη συνάρτηση έχουν αλλάξει τιμή οι παράμετροι που έλαβε κατά την κλήση της, τότε αυτές οι αλλαγές μεταφέρονται στο μέρος του προγράμματος που την κάλεσε (παράμετροι εισόδου-εξόδου)
- Αν η συνάρτηση δεν μεταβάλει την τιμή των παραμέτρων που έλαβε, τότε αυτές ονομάζονται παράμετροι εισόδου(δεν άλλαξε την τιμή των παραμέτρων στο μέρος προγράμματος που την κάλεσε). **ΑΥΤΈΣ ΔΕΝ ΕΠΙΣΤΡΕΦΟΝΤΑΙ**
- Αν υπάρχει στη λίστα παραμέτρων κάποια, για την οποία δεν περάστηκε τιμή από το μέρος του προγράμματος που την κάλεσε, τότε αυτή ονομάζεται παράμετρος εξόδου(αρχικά στην διαδικασία είναι η τιμή της απροσδιόριστη, λαμβάνει τιμή εντός της διαδικασίας και επιστρέφεται μαζί με τις υπόλοιπες παραμέτρους που έχουν μεταβληθεί)
- Σε αντίθεση με τις συναρτήσεις, δεν επιτρέπονται σταθερές τιμές και μεταβλητές ως παράμετροι σε διαδικασίες

ΠΑΡΑΔΕΙΓΜΑ 1 - ΔΙΑΔΙΚΑΣΙΕΣ

Δημιουργήστε κατάλληλο υποπρόγραμμα που θα δέχεται την τιμή ενός προϊόντος και θα υπολογίζει και θα εκτυπώνει στην οθόνη την αξία του ΦΠΑ(24%) και την τελική τιμή του προϊόντος

```
ΔΙΑΔΙΚΑΣΙΑ Υπολογισμός_φόρου(τιμή)
ΣΤΑΘΕΡΕΣ
    ΣΦΠΑ = 0.24
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: ποσό_φόρου, τιμή, τελική
ΑΡΧΗ
    ποσό_φόρου <- τιμή * ΣΦΠΑ
    ΓΡΑΨΕ 'Το ποσό φόρου είναι ',ποσό_φόρου
    τελική <- τιμή + ποσό_φόρου
    ΓΡΑΨΕ 'Τελική τιμή με ΦΠΑ ',τελική
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Σε διαδικασία δεν δηλώνουμε κάποιο τύπο

Ούτε επιστρέφουμε κάποια τιμή μέσω του ονόματος του υποπρογράμματος στο μέρος που την κάλεσε

Εδώ καταλυτικό ρόλο στην απόφαση για υλοποίηση συνάρτησης παίζει το γεγονός ότι θα πρέπει αυτό να εκτυπώνει στην οθόνη κάποιο μήνυμα. Οι συναρτήσεις δεν περιέχουν τις εντολές ΓΡΑΨΕ, ΔΙΑΒΑΣΕ κτλ.

```
ΠΡΟΓΡΑΜΜΑ ΑΣΚΗΣΗ2
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: value
ΑΡΧΗ
    ΓΡΑΨΕ 'Δώσε Τιμή προϊόντος...'
    ΔΙΑΒΑΣΕ value
    ΚΑΛΕΣΕ Υπολογισμός_φόρου(value)
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Όταν το κυρίως πρόγραμμα καλεί την διαδικασία Υπολογισμός_φόρου:

- Περνάει την **πραγματική παράμετρο value που ορίστηκε στο κυρίως πρόγραμμα**
- Η μεταβλητή αυτή θα αναφέρεται μέσα στην διαδικασία ως τιμή(η τιμή είναι **τυπική παράμετρος της διαδικασίας)**
- Η τυπική παράμετρος τιμή δεν αλλάζει μέσα στη διαδικασία, επομένως μετά την εκτέλεση της δεν επιστρέφεται στο κυρίως πρόγραμμα(για αυτό και αποτελεί τυπική παράμετρο εισόδου καθώς χρησιμοποιείται μόνο για υπολογισμούς χωρίς να αλλάζει η τιμή της)
- Σε αυτό το παράδειγμα η διαδικασία δεν επιστρέφει καμία τιμή στο κυρίως πρόγραμμα, αφού η τιμή είναι η μοναδική παράμετρος της και αυτή όπως είπαμε είναι παράμετρος εισόδου

ΠΑΡΑΔΕΙΓΜΑ 2 - ΔΙΑΔΙΚΑΣΙΕΣ

Δημιουργήστε κατάλληλο υποπρόγραμμα που θα δέχεται 2 πραγματικούς αριθμούς και θα αντιμεταθέτει τις τιμές του. Στην συνέχεια υλοποιήστε ένα πρόγραμμα που θα ζητάει από το χρήστη 2 πραγματικούς μη μηδενικούς αριθμούς και θα χρησιμοποιεί το υποπρόγραμμα για να αντιμεταθέτει τις τιμές του

Και σε αυτή την περίπτωση θα χρησιμοποιήσουμε Διαδικασία και όχι συνάρτηση. Δεν θέλουμε να επιστρέψουμε μια μοναδική τιμή αλλά επιθυμούμε η διαδικασία να δεχθεί 2 πραγματικούς και να αλλάξει τις τιμές τους

```
ΔΙΑΔΙΚΑΣΙΑ Swap(value1, value2)
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: temp, value1, value2
ΑΡΧΗ
    temp <- value1
    value1 <- value2
    value2 <- temp
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Όταν το κυρίως πρόγραμμα καλεί την διαδικασία Swap, σταματάει την ροή εκτέλεσης του κυρίου προγράμματος και ο μεταγλωττιστής αποθηκεύει την επόμενη προς εκτέλεση εντολή(εντολή διεύθυνσης) ΓΡΑΨΕ στην στοίβα χρόνου εκτέλεσης

Περνάει στις **ΠΡΑΓΜΑΤΙΚΕΣ παραμέτρους value1, value2** της διαδικασίας αναφορές στις **ΤΥΠΙΚΕΣ παραμέτρους a,b**

```
ΠΡΟΓΡΑΜΜΑ ΑΣΚΗΣΗ3
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: a,b
    ΛΟΓΙΚΕΣ: valid
ΑΡΧΗ
    valid <- ΨΕΥΔΗΣ
    ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ
        ΓΡΑΨΕ 'Δώσε 2 θετικούς αριθμούς...'
        ΔΙΑΒΑΣΕ a,b
        ΑΝ a > 0 ΚΑΙ b > 0 ΤΟΤΕ
            valid <- ΑΛΗΘΗΣ
        ΑΛΛΙΩΣ
            ΓΡΑΨΕ 'Πρέπει να δώσεις 2 θετικούς αριθμούς...'
        ΤΕΛΟΣ_ΑΝ
    ΜΕΧΡΙΣ_ΟΤΟΥ (valid)

    ΚΑΛΕΣΕ Swap(a,b)
    ΓΡΑΨΕ 'Μετά την αντιμετάθεση οι τιμές των a, b είναι: ', a, b
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Σε αντίθεση με το προηγούμενο παράδειγμα, εδώ οι τυπικές παράμετροι value1, value2 τροποποιούνται μέσα στη διαδικασία και αυτό τις καθιστά **τυπικές παράμετροι εισόδου-εξόδου.**

Επιπλέον, από τη στιγμή που τροποποιούνται οι τιμές τους, επιστρέφονται οι νέες τιμές στο κυρίως πρόγραμμα που την κάλεσε. Αυτό σημαίνει ότι οι τιμές των value1, value2 θα επιστραφούν και θα καταχωρηθούν αντίστοιχα στις μεταβλητές a,b του κυρίως προγράμματος μετά την ολοκλήρωση της εκτέλεσης της

Δεν θα είχαμε επιστροφή παραμέτρων αν δεν προέκυπτε τροποποίηση τους(όπως στο προηγούμενο παράδειγμα)

ΠΑΡΑΔΕΙΓΜΑ 3 - ΔΙΑΔΙΚΑΣΙΕΣ

Δημιουργήστε κατάλληλο υποπρόγραμμα που θα δέχεται 2 ακέραιους αριθμούς και θα επιστρέφει τον μεγαλύτερο. Στην συνέχεια δημιουργήστε πρόγραμμα το οποίο θα διαβάζει από το χρήστη 2 ακεραίους και χρησιμοποιώντας τη διαδικασία θα εκτυπώνει την μεγαλύτερη τιμή

```
ΔΙΑΔΙΚΑΣΙΑ Find_Max(value1, value2, max)
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: value1, value2, max
ΑΡΧΗ
    max <- value1
    ΑΝ value2 > max ΤΟΤΕ
        max <- value2
    ΤΕΛΟΣ_ΑΝ
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Σε αυτή την περίπτωση ιδανικά θα χρησιμοποιούσαμε συνάρτηση καθώς πρακτικά θέλουμε να επιστρέψουμε μια τιμή(την μέγιστη) μεταξύ των παραμέτρων. Θα χρησιμοποιήσουμε Διαδικασία για να εξετάσουμε τις μεταβλητές εξόδου

Οι τυπικές παράμετροι value1, value2 είναι παράμετροι εισόδου καθώς οι τιμές τους δεν θα τροποποιηθούν μέσα στην διαδικασία και απλά χρησιμοποιούνται ως είσοδοι από το κυρίως πρόγραμμα στην διαδικασία. Από τη στιγμή που είναι παράμετροι εισόδου οι τιμές τους δεν θα επιστραφούν πίσω στο πρόγραμμα

```
ΠΡΟΓΡΑΜΜΑ ΑΣΚΗΣΗ3
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: a,b,m
ΑΡΧΗ
    ΓΡΑΨΕ 'Δώσε 2 ακεραίους'
    ΔΙΑΒΑΣΕ a,b
    ΚΑΛΕΣΕ Find_Max(a,b,m)
    ΓΡΑΨΕ 'Ο μεγαλύτερος αριθμός είναι: ', m
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Η τυπική παράμετρος max της διαδικασίας είναι **μόνο εξόδου**.

Δεν εισάγεται τιμή από το κυρίως πρόγραμμα(Το m είναι απροσδιόριστο όταν περνάει από το κυρίως πρόγραμμα και αποτελεί αναφορά της τυπικής παραμέτρου max) και λαμβάνει τιμή μέσα στη διαδικασία. Η τυπική παράμετρος max επιστρέφεται στο μέρος του προγράμματος που την κάλεσε στην θέση της πραγματικής παραμέτρου m

ΑΣΚΗΣΕΙΣ ΓΙΑ ΣΠΙΤΙ

ΑΣΚΗΣΗ 1

Να γράψετε συνάρτηση που θα δέχεται ως παράμετρο έναν αριθμό και θα ελέγχει αν είναι τετραψήφιος ή όχι

ΑΣΚΗΣΗ 2

Σύμφωνα με το νέο καθεστώς εισαγωγής στις στρατιωτικές σχολές, το όριο του ύψους είναι για τα αγόρια 1.70 μέτρα και για τα κορίτσια 1.65 μέτρα. Ταυτόχρονα, εισάγεται ο Δείκτης Μάζας Σώματος ΔΜΣ, που είναι το πηλίκο του σωματικού βάρους του υποψήφιου προς το τετράγωνο του ύψους του. Για τα αγόρια ο ΔΜΣ πρέπει να ανήκει στο διάστημα [20, 25] και για τα κορίτσια [18, 22]. Να αναπτύξετε συνάρτηση που θα δέχεται για κάποιον / κάποια υποψήφιο το φύλο, το βάρος και το ύψος και θα επιστρέφει μήνυμα “ΝΑΙ / ΟΧΙ” σχετικά με το αν γίνεται δεκτός / δεκτή ή όχι

ΑΣΚΗΣΗ 3

Να σχηματίσετε τον πίνακα τιμών του παρακάτω αλγορίθμου. Τι θα εκτυπωθεί;

```
1  ΠΡΟΓΡΑΜΜΑ Πίνακας
2  ΜΕΤΑΒΛΗΤΕΣ
3      ΑΚΕΡΑΙΕΣ: Α, Β, Γ
4  ΑΡΧΗ
5      Α <- 3
6      Β <- 13
7      Γ <- 2
8      ΓΡΑΨΕ Α,Β,Γ
9      ΚΑΛΕΣΕ Επεξεργασία(Β,Γ)
10     ΓΡΑΨΕ Α,Β,Γ
11     ΚΑΛΕΣΕ Επεξεργασία(Γ,Α)
12     ΓΡΑΨΕ Α,Β,Γ
13 ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
14
15
16 ΔΙΑΔΙΚΑΣΙΑ Επεξεργασία(κ,λ)
17 ΜΕΤΑΒΛΗΤΕΣ
18     ΑΚΕΡΑΙΕΣ: κ,λ
19 ΑΡΧΗ
20     κ <- κ DIV 2
21     λ <- λ ^ 3
22 ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

ΑΣΚΗΣΗ 4

Να αναπτύξετε διαδικασία που θα δέχεται 2 ακέραιους αριθμούς, οι οποίοι εκφράζουν τα άκρα ενός διαστήματος αριθμών και θα διαβάζει και θα επιστρέφει έναν ακέραιο αριθμό εντός του διαστήματος αυτού, πραγματοποιώντας έλεγχο ορθής καταχώρησης. Η διαδικασία να εμφανίζει και το πλήθος των λανθασμένων καταχωρήσεων