Αρχές Γλωσσών Προγραμματισμου

Χρήστος Νομικός

Τμήμα Μηχανικών Η/Υ και Πληροφορικής
Πανεπιστήμιο Ιωαννίνων

2015

Περιεχόμενα

Περιεχόμενα

Υποπρογράμματα

Ένα υποπρόγραμμα συνίσταται από μία σειρά εντολών που υλοποιουν μία συγκεκριμένη λειτουργία, η οποία εκτελείται περισσότερες από μία φορές κατά τη διάρκεια εκτέλεσης του προγράμματος, για διαφορετικά δεδομένα κάθε φορά.

Τα υποπρογράμματα αποτελούν ένα μηχανισμό αφαίρεσης: όταν καλούμε ένα υποπρόγραμμα μας ενδιαφέρει μόνο το ποια λειτουργία υλοποιεί και όχι το πώς την υλοποιεί.

Υποπρογράμματα

Όταν καλείται ένα υποπρόγραμμα, η εκτέλεση του καλούντος προγράμματος ή υποπρογράμματος αναστέλλεται μέχρι την ολοκλήρωση του καλούμενου υποπρογράμματος.

Όταν ολοκληρωθεί η εκτέλεση του υποπρογράμματος, η ροή εκτέλεσης του προγράμματος μεταφέρεται αμέσως μετά από το σημείο από το σημείο κλήθηκε το υποπρόγραμμα.

Υποπρογράμματα

Συνήθως ένα υποπρόγραμμα έχει ένα πλήθος παραμέτρων, οι οποίες χρησιμοποιούνται για να καθοριστούν κατά την κλήση του τα δεδομένα πάνω στα οποία θα εκτελέσει τη λειτουργία του, καθώς και για να ρυθμιστεί ο ακριβής τρόπος λειτουργίας του.

Επίσης είναι δυνατόν κάποιες παράμετροι να χρησιμοποιούνται για επιστροφή στο καλούν τμήμα του κώδικα αποτελεσμάτων ή άλλων πληροφοριών σχετικών με την εκτέλεση του υποπρογράμματος.

Υποπρογράμματα

Ονομάζουμε τυπικές παραμέτρους τα συμβολικά ονόματα που χρησιμοποιούνται μέσα στο υποπρόγραμμα για να αναφερθούμε στα δεδομένα τα οποία έχουν μεταβιβαστεί από το καλούν τμήμα κώδικα στο υποπρόγραμμα.

Ονομάζουμε πραγματικές παραμέτρους τις μεταβλητές ή γενικότερα τις παραστάσεις που χρησιμοποιεί το τμήμα του κώδικα το οποίο καλεί το υποπρόγραμμα ώστε να καθορίσει τα δεδομένα που θέλει να περάσει στο υποπρόγραμμα.

Υποπρογράμματα

Παράδειγμα

Εστω το παρακάτω πρόγραμμα Pascal:

```
program what(output);
const MAX = ...; ASCENDING = 1; DESCENDING = -1; ...
var S:array[1..MAX] of real;  i : integer;  ...

    procedure swap(var a,b:real; mode:integer);
    var tmp:real;
    begin
        if mode * a > mode * b then
           begin
              tmp := a; a := b; b := tmp
           end
    end;

begin (* of program what *)
... swap(S[i],S[i+1],ASCENDING); ...
end.
```

Υποπρογράμματα

Στο παραπάνω πρόγραμμα το υποπρόγραμμα swap έχει τρεις παραμέτρους. Οι δύο πρώτες χρησιμοποιούνται τόσο για να περάσουν δεδομένα προς το υποπρόγραμμα, όσο και για να επιστραφούν αποτελέσματα από αυτό. Η τρίτη καθορίζει τον τρόπο λειτουργίας του προγράμματος (συγκεκριμένα το κατά πόσο η διάταξη των τιμών των a και b μετά την εκτέλεση του υποπρογράμματος θα είναι αύξουσα ή φθίνουσα).

Οι a, b και mode είναι οι τυπικές παράμετροι του υποπρογράμματος swap.

Οι S[i], S[i+1] και ASCENDING είναι οι πραγματικές παράμετροι στην κλήση του swap.

Υποπρογράμματα

Διακρίνομε δύο κατηγορίες υποπρογραμμάτων:

- Οι συναρτήσεις είναι υποπρογράμματα τα οποία επιστρέφουν μία τιμή και η κλήση τους μπορεί να εμφανίζεται μέσα σε μία παράσταση.
- Οι διαδικασίες είναι υποπρογράμματα τα οποία δεν επιστρέφουν τιμή (μπορούν ωστόσο να επιστρέφουν δεδομένα στο καλούν τμήμα κώδικα μέσω των παραμέτρων) και ο σκοπός τους είναι να προκαλέσουν αλλαγές στο περιβάλλον αναφοράς (μεταβλητές που δηλωνονται σε εξωτερικά μπλοκ), είσοδο-έξοδο δεδομένων, κλπ.

Υποπρογράμματα

Ένα υποπρόγραμμα έχει τα παρακάτω συστατικά:

- όνομα: χρησιμοποιείται για να μπορεί να κληθεί το υποπρόγραμμα
- λίστα παραμέτρων: για κάθε τυπική παράμετρο περιέχει το όνομα, τον τύπο, και τον τρόπο σύνδεσης με την αντίστοιχη πραγματική παράμετρο κατά την κλήση.
- σώμα: περιλαμβάνει τις δηλώσεις των τοπικών μεταβλητών καθώς και τις εντολές που υλοποιούν την προκαθορισμένη για το υποπρόγραμμα λειτουργία.
- τύπο αποτελέσματος (μόνο για συναρτήσεις)

Υποπρογράμματα

Επικεφαλίδα του υποπρογράμματος ονομάζεται η πρώτη γραμμή του, η οποία περιλαμβάνει το είδος, το όνομα, τη λίστα των τυπικών παραμέτρων και τον τύπο του αποτελέσματος.

Η επικεφαλίδα ενός υποπρογράμματος περιέχει όλη την πληροφορία που χρειάζεται για να μπορέσουμε να καλέσουμε το υποπρόγραμμα.

Παράδειγμα

Στο πρόγραμμα του προηγούμενου παραδείγματος, η επικεφαλίδα του υποπρογράμματος είναι: procedure swap(var a,b:real; mode:integer).

Η λέξη procedure υποδηλώνει ότι το υποπρόγραμμα είναι διαδικασία.

Η λέξη swap είναι το όνομα του υποπρογράμματος.

Η λίστα παραμέτρων είναι (var a,b:real; mode:integer). Οι παράμετροι a και b είναι τύπου real και περνουν με αναφορά (αυτό καθορίζεται από τη λέξη var που προηγείται του ονόματος των παραμετρων), ενώ η παράμετρος mode είναι τύπου integer και περνάει με τιμή (που είναι ο προκαθορισμένος τρόπος περάσματος παραμέτρων στην Pascal).

Υποπρογράμματα

Μία συνάρτηση μπορεί να επιστρέφει τιμή με έναν από τους παρακάτω τρόπους:

- με εντολή επιστροφής τιμής (όπως είναι η return των Python, C, Java)
- με εντολή ανάθεσης, στο αριστερό μερος της οποίας τοποθετείται το όνομα της συνάρτησης (όπως γίνεται στην PASCAL).

Παράδειγμα

Η παρακάτω συνάρτηση Python αποφασίζει αν ένας ακέραιος αριθμός είναι πρώτος:

```
def prime(n):
        if n < 2:
                return False
        k = 2
        while k*k <= n:
                if n % k == 0:
                        return False
                else:
                        k=k+1
        return True
```

Αν ο αριθμός είναι μικρότερος του 2, τότε η prime τερματίζει άμεσα επιστρέφοντας την τιμή False με την εντολή return.

Σε αντίθετη περίπτωση εκτελείται ένας βρόχος while που ελέγχει αν το n διαιρείται απο κάποιον ακέραιο ανάμεσα στο 2 και το $\sqrt{n}$. Αν βρεθεί ένα τέτοιος ακέραιος, τότε η prime τερματίζει άμεσα επιστρέφοντας την τιμή False.

Αν η prime βγεί ομαλά από το βρόχο while, τότε ο n είναι πρώτος αριθμός και επιστρέφεται η τιμή True.

Παράδειγμα

Η παρακάτω συνάρτηση Pascal επίσης αποφασίζει αν ένας ακέραιος αριθμός είναι πρώτος:

```
function prime(n: integer): boolean;
var k: integer; maybePrime: boolean;
begin
     maybePrime := n >= 2;
     k := 2;
     while maybePrime and (k*k <= n) do
           if n mod k = 0 then
                maybePrime := False
           else
                k := k+1;
     prime := maybePrime
end;
```

Υποπρογράμματα

Η Pascal δεν έχει εντολή παρόμοια με τη return και η επιστροφή του αποτελέσματος γίνεται με ανάθεση του στην prime. Το μέχρι στιγμής αποτέλεσμα αποθηκεύται στη μεταβλητή maybePrime.

Αν ο αριθμός είναι μικρότερος του 2, τότε η μεταβλητή maybePrime αρχικοποιείται σε False αποτρέποντας την εκτέλεση του βρόχου while.

Σε αντίθετη περίπτωση η maybePrime αρχικοποιείται σε True και εκτελείται ένας βρόχος while που ελέγχει αν το n διαιρείται απο κάποιον ακέραιο ανάμεσα στο 2 και το $\sqrt{n}$. Αν βρεθεί ένα τέτοιος ακέραιος, τότε η maybePrime γίνεται False, τερματίζοντας έτσι το βρόχο.

Μετά την ολοκλήρωση του βρόχου εκτελείται η εντολή

prime := maybePrime

μέσω της οποίας επιστρέφεται το τελικό αποτέλεσμα.

Υποπρογράμματα

Ορισμένες γλώσσες επιτρέπουν στα υποπρογράμματα να δέχονται ως τυπικές παραμέτρους άλλα υποπρογράμματα (Pascal, Python, Haskell, Lisp)

Επίσης ορισμένες γλώσσες επιτρέπουν τον ορισμό ενός υποπρογράμματος μέσα σε άλλο (Pascal, Python, Haskell)

Υποπρογράμματα

Το περιβάλλον αναφοράς ενός υποπρογράμματος περιλαμβάνει:

- τις τυπικές παραμέτρους
- τις τοπικές μεταβλητές
- τις μη τοπικές μεταβλητές

Περιεχόμενα

Πέρασμα παραμέτρων

Οι παράμετροι ενός υποπρογράμματος χρησιμοποιούνται για ανταλλαγή δεδομένων ανάμεσα στο τμήμα του κώδικα που καλεί το υποπρόγραμμα και στο υποπρόγραμμα.

Οι τυπικές παράμετροι συνδέονται με τις πραγματικές παραμέτρους κατά την κλήση του υποπρογράμματος. Η διαδικασία αυτή ονομάζεται πέρασμα παραμέτρων και μπορεί να γίνει με διάφορους εναλλάκτικούς τρόπους.

Πέρασμα παραμέτρων

Ο τρόπος του περάσματος παραμέτρων αποτελεί το πρωτόκολλο επικοινωνίας ανάμεσα στο υποπρόγραμμα και το τμήμα του κώδικα που το καλεί, το οποίο καθορίζει το πώς μεταφέρονται δεδομένα προς και από το υποπρόγραμμα μέσω των παραμέτρων.

Εκτός από τις διαφορές που έχουν στον τρόπο υλοποίησής τους, οι εναλλακτικοί τρόποι περάσματος παραμέτρων συνεπάγονται διαφορές στη συμπεριφορά του προγράμματος, σε συνδυασμό και με το μοντέλο που χρησιμοποιείται για τις μεταβλητές (μοντέλο αναφορών ή μοντέλο τιμών).

Πέρασμα παραμέτρων

Ορισμένες γλώσσες προγραμματισμού υποστηρίζουν μόνο έναν τρόπο περάσματος παραμέτρων (C, Python), ενώ άλλες υποστηρίζουν περισσότερους (Pascal, Ada, Java).

Στη δεύτερη περίπτωση η γλώσσα μπορεί να παρέχει τη δυνατότητα στον προγραμματιστή να επιλέξει τον τρόπο περάσματος για κάθε παράμετρο χωριστά (Pascal, Ada).

Στη συνέχεια περιγράφονται οι κυριότεροι τρόποι περάσματος παραμέτρων.

Πέρασμα παραμέτρων

Πέρασμα παραμέτρων με τιμή:

- Η τυπική παράμετρος αποθηκεύται τοπικά στο εγγράφημα δραστηριοποίησης του υποπρογράμματος.
- Η αντίστοιχη πραγματική παράμετρος αποτιμάται κατά την κλήση του υποπρογράμματος και η τιμή της αποθηκεύται στη θέση μνήμης της τυπικής παραμέτρου.
- Απο αυτό το σημείο και μετά δεν υπάρχει καμία άλλη αλληλεπίδραση ανάμεσα στην τυπική και την πραγματική παράμετρο.

Πέρασμα παραμέτρων

Οποιαδήποτε αλλαγή γίνει στην τιμή της τυπικής παραμέτρου δεν επηρεάζει την πραγματική παράμετρο.

Αντίστοιχα, αν αλλάξει με κάποιο τρόπο η τιμη της πραγματικής παραμέτρου (αυτό μπορεί να γίνει αν η πραγματική παράμετρος είναι μεταβλητή και είναι ορατή στο υποπρόγραμμα) δεν αλλάζει η τιμή της τυπικής παραμέτρου.

Πέρασμα παραμέτρων

Το πέρασμα παραμέτρων με τιμή είναι ο μοναδικός τρόπος περάσματος παραμέτρων στη C, ενώ υποστηρίζεται από την Pascal την Ada και τη Modula. Επίσης η Java τον εφαρμόζει για τους ενσωματωμένους τύπους που χρησιμοποιούν το μοντέλο τιμών.

Πέρασμα παραμέτρων

Παράδειγμα

Εστω το παρακάτω πρόγραμμα Pascal (τό πέρασμα της παραμέτρου k στη διαδικασία p γίνεται με τιμή, που είναι ο προκαθορισμένος τρόπος περάσματος παραμέτρων στην Pascal):

```
program CallByValueExample(output);
var n:integer;

    procedure p(k:integer);
        begin
            k:=k+5;
        end;

begin
  n:=10; p(n); writeln(n)
end.
```

Πέρασμα παραμέτρων

Η εκτέλεση του προγράμματος συνοψίζεται στον παρακάτω πίνακα:

	n	k	έξοδος
αρχή			
n:=10	10		
p(n)			
πέρασμα παραμέτρων		10	
k:=k+5		15	
επιστροφή από p			
writeln(n)			10
τέλος			

Πέρασμα παραμέτρων

Η τιμή της πραγματικής παραμέτρου n δεν αλλάζει από την εκτέλεση της διαδικασίας p. Η τυπική παράμετρος k αποθηκεύται τοπικά και η αλλαγή στην τιμή της δεν επηρεάζει την n.

Το πρόγραμμα τυπώνει την τιμή 10, που είναι η τιμή που έχει ανατεθεί στην n πριν από την κλήση της p.

Πέρασμα παραμέτρων

Παράδειγμα

Προσθέτουμε στο πρόγραμμα του προηγούμενου παραδείγματος την εντολή n:=n+8 μέσα στη διαδικασία p:

```
program CallByValueExample2(output);
var n:integer;

    procedure p(k:integer);
        begin
            k:=k+5;
            n:=n+8;
        end;

begin
  n:=10; p(n); writeln(n)
end.
```

Πέρασμα παραμέτρων

Το πρόγραμμα τυπώνει την τιμή 18. Συνεπώς η τιμή της πραγματικής παραμέτρου είναι διαφορετική μετά την εκτέλεση του υποπρογράμματος, παρότι το πέρασμα γίνεται με τιμή!

Ωστόσο η αλλαγή στην τιμή της n δε σχετίζεται με το μηχανισμό περάσματος παραμέτρων και δεν οφείλεται στο ότι η n είναι πραγματική παράμετρος.

Η αλλαγή γίνεται επειδή η n είναι ορατή μέσα στη διαδικασία p και συνεπώς η p μπορεί να αλλάξει την τιμή της. Η τιμή της n θα άλλαζε ακόμη και αν το πρόγραμμα καλούσε την p με κάποια άλλη πραγματική παράμετρο.

Πέρασμα παραμέτρων

Πέρασμα παραμέτρων με αναφορά:

- Χρησιμοποιήται από γλώσσες που υποστηρίζουν το μοντέλο τιμών για τις μεταβλητές.
- Κατά την κλήση του υποπρογράμματος το καλούν τμήμα του κώδικα περνάει στο υποπρόγραμμα την αναφορά της πραγματικής παραμέτρου (δηλαδή τη θέση μνήμης στην οποία είναι αποθηκευμένη η πραγματική παράμετρος), η οποία γίνεται και αναφορά της τυπικής παραμέτρου.
- Κατ΄ αυτόν τον τρόπο η πραγματική και η τυπική παράμετρος είναι άρρηκτα συνδεδεμένες με το ίδιο αντικείμενο καθ΄ όλη τη διάρκεια εκτέλεσης του υποπρογράμματος και συνεπώς κάθε χρονική στιγμή μέχρι και την ολοκλήρωση του υποπρογράμματος έχουν ακριβώς την ίδια τιμή.

Πέρασμα παραμέτρων

Οποιαδήποτε αλλαγή γίνει στην τιμή της τυπικής παραμέτρου επηρεάζει αυτόματα την τιμή της πραγματικής παραμέτρου.

Αντίστοιχα, αν αλλάξει με κάποιο τρόπο η τιμη της πραγματικής παραμέτρου (αυτό μπορεί να γίνει αν η πραγματική παράμετρος είναι μεταβλητή και είναι ορατή στο υποπρόγραμμα) αλλάζει αυτόματα και την τιμή της τυπικής παραμέτρου.

Πέρασμα παραμέτρων

Το πέρασμα παραμέτρων με αναφορά:

- χρησιμεύει ώστε να επιστρέφονται περισσότερα από ένα αποτελέσματα από το υποπρόγραμμα.
- επιτρέπει να περνούν στο υποπρόγραμμα αντικείμενα που καταλαμβάνουν πολλές θέσεις μνήμης (π.χ. πίνακες, εγγραφές κ.λ.π) χωρίς να χρειάζεται αντιγραφή τους (που συνεπάγεται κόστος σε χρόνο και μνήμη).

Ορισμένες γλώσσες παρέχουν στο προγραμματιστή τη δυνατότητα να περνάει μεγάλα αντικείμενα με αναφορά στο υποπρόγραμμα, προστατεύοντάς τα ωστόσο ώστε να μην είναι δυνατό να τροποποιηθούν (π.χ χρήση του READONLY στη Modula 3).

Πέρασμα παραμέτρων

Το πέρασμα παραμέτρων με αναφορά είναι ο μοναδικός τρόπος περάσματος παραμέτρων στη Fortran, ενώ υποστηρίζεται από την Pascal, τη C++, τη Modula και την PHP.

Στη C το αντίστοιχο αποτέλεσμα μπορεί να επιτευχθεί με χρήση δεικτών.

Πέρασμα παραμέτρων

Παράδειγμα

Εστω το παρακάτω πρόγραμμα Pascal (η λέξη var που προηγείται της τυπικής παραμέτρου k στην επικεφαλίδα της διαδικασίας p υποδηλώνει ότι το πέρσμα της k γίνεται με αναφορά):

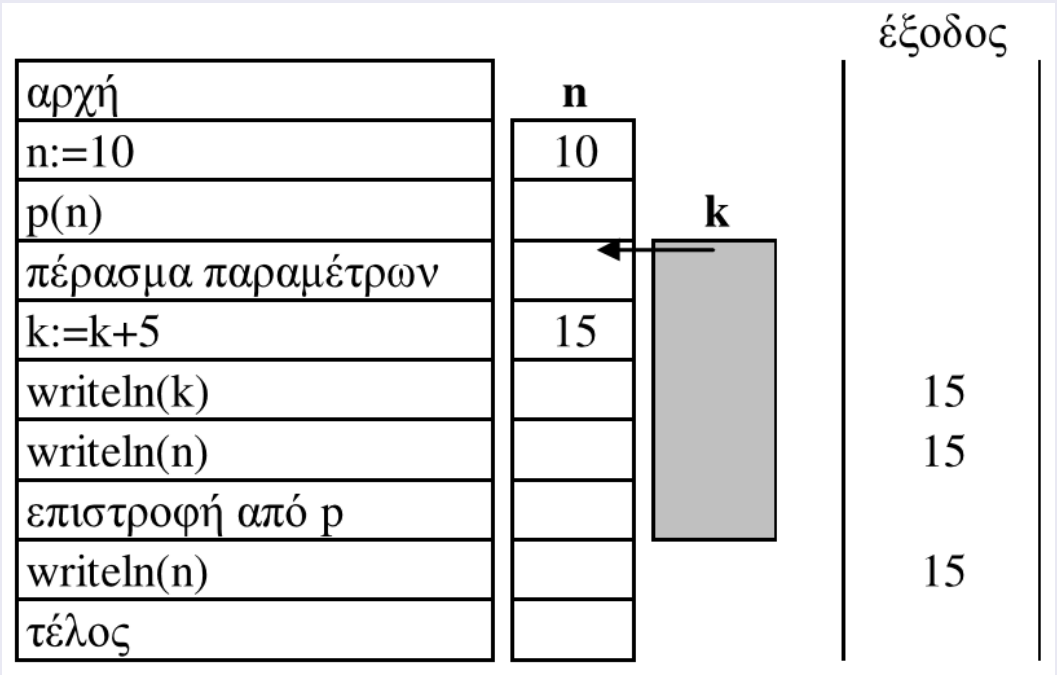
```
program CallByReferenceExample(output);
var n:integer;

    procedure p(var k:integer);
        begin
            k:=k+5;
            writeln(k);
            writeln(n);
        end;

begin
  n:=10; p(n); writeln(n)
end.
```

Πέρασμα παραμέτρων

Η εκτέλεση του προγράμματος συνοψίζεται στον παρακάτω πίνακα:



Πέρασμα παραμέτρων

Η τιμή της πραγματικής παραμέτρου n αλλάζει όταν εκτελείται η εντολή k:=k+5 εντός της διαδικασίας p. Αυτό συμβαίνει επειδή η τυπική παράμετρος k και η πραγματική παράμετρος n είναι δύο διαφορετικά ονόματα για την ίδια θέση μνήμης. Συνεπώς κατά τη διάρκεια εκτέλεσης της p με πραγματική παράμετρο την n, η μεταβλητή n και η τυπική παράμετρος k έχουν την ίδια τιμή.

Ως επακόλουθο, η τιμή της n μετά την εκτέλεση της p είναι 15.

Πέρασμα παραμέτρων

Πέρασμα παραμέτρων με τιμή-αποτέλεσμα:

- Η τυπική παράμετρος αποθηκεύται τοπικά στο εγγράφημα δραστηριοποίησης του υποπρογράμματος.
- Η αντίστοιχη πραγματική παράμετρος αποτιμάται κατά την κλήση του υποπρογράμματος και η τιμή της αποθηκεύται στη θέση μνήμης της τυπικής παραμέτρου.
- Απο αυτό το σημείο μέχρι και την ολοκλήρωση του υποπρογράμματος δεν υπάρχει καμία αλληλεπίδραση ανάμεσα στην τυπική και την πραγματική παράμετρο.
- Μετά την ολοκλήρωση του υποπρογράμματος η τιμή της τυπικής παραμέτρου αντιγράφεται στην πραγματική παράμετρο (η οποία θα πρέπει να είναι μεταβλητή ή κάποια έκφραση που προσδιορίζει μία θέση στη μνήμη στην οποία μπορεί να αποθηκευτεί τιμή).

Πέρασμα παραμέτρων

Οι αλλαγές που γίνονται στην τιμή της τυπικής παραμέτρου επηρεάζουν την πραγματική παράμετρο μόνο μετά την ολοκλήρωση του υποπρογράμματος. Κατά τη διάρκεια εκτέλεσης του υποπρογράμματος η τυπική και η πραγματική παράμετρος μπορεί να έχουν διαφορετικές τιμές.

Το πέρασμα παραμέτρων με τιμή-αποτέλεσμα υποστηρίζεται από την Ada για βαθμωτούς τύπους.

Πέρασμα παραμέτρων

Παράδειγμα

Τροποποιούμε το πρόγραμμα του προηγούμενου παραδείγματος, χρησιμοποιώντας τη λέξη inout (επεκτείνοντας καταχρηστικά τη σύνταξη της Pascal) για να υποδηλώσουμε ότι το πέρασμα της παραμέτρου k στη διαδικασία p γίνεται με τιμή-αποτέλεσμα (σημειώνεται ότι το πέρασμα με τιμή αποτέλεσμα δεν υποστηρίζεται από τη Pascal):

Πέρασμα παραμέτρων

```
program CallByValueResultExample(output);
var n:integer;

    procedure p(inout k:integer);
        begin
            k:=k+5;
            writeln(k);
            writeln(n);
        end;

begin
  n:=10; p(n); writeln(n)
end.
```

Πέρασμα παραμέτρων

Η εκτέλεση του προγράμματος συνοψίζεται στον παρακάτω πίνακα:

	n	k	έξοδος
αρχή			
n:=10	10		
p(n)		k	
πέρασμα παραμέτρων		10	
k:=k+5		15	
writeln(k)			15
writeln(n)			10
επιστροφή από p	15		
writeln(n)			15
τέλος			

Πέρασμα παραμέτρων

Η τιμή της πραγματικής παραμέτρου n δεν αλλάζει όταν εκτελείται η εντολή k:=k+5 εντός της διαδικασίας p. Αυτό συμβαίνει επειδή η τυπική παράμετρος k και η πραγματική παράμετρος n είναι αποθηκευμένες σε διαφορετικές θέσεις μνήμης.

Ωστόσο η τιμή της k αντιγράφεται στην n μετά την ολοκλήρωση της εκτέλεσης της p. Συνεπώς, η τιμή της n μετά την εκτέλεση της p είναι 15.

Πέρασμα παραμέτρων

Από τα δύο τελευταία παραδείγματα φαίνεται ότι το πέρασμα παραμέτρων με τιμή-αποτέλεσμα διαφέρει από το πέρασμα με αναφορά (οι τιμές που τυπώνονται από τη διαδικασία p είναι διαφορετικές στις δύο περιπτώσεις).

Ωστόσο για το συγκεκριμένο πρόγραμμα, η τιμή της πραγματικής παραμέτρου που προκύπτει μετά την εκτέλεση διαδικασίας p είναι η ίδια και στις δύο περιπτώσεις (και διαφέρει από την τιμή που είχε πριν από την κλήση της p).

Αν η πραγματική παράμετρος είναι μία μεταβλητή ορατή στο υποπρόγραμμα και αυτό αλλάζει την τιμή της, τότε οι τιμές της πραγματικής παραμέτρου μετά την εκτέλεση του υποπρογράμματος ενδέχεται να είναι διαφορετικές για τους δύο αυτούς τρόπους περάσματος παραμέτρων.

Πέρασμα παραμέτρων

Παράδειγμα

Εστω το παρακάτω πρόγραμμα Pascal:

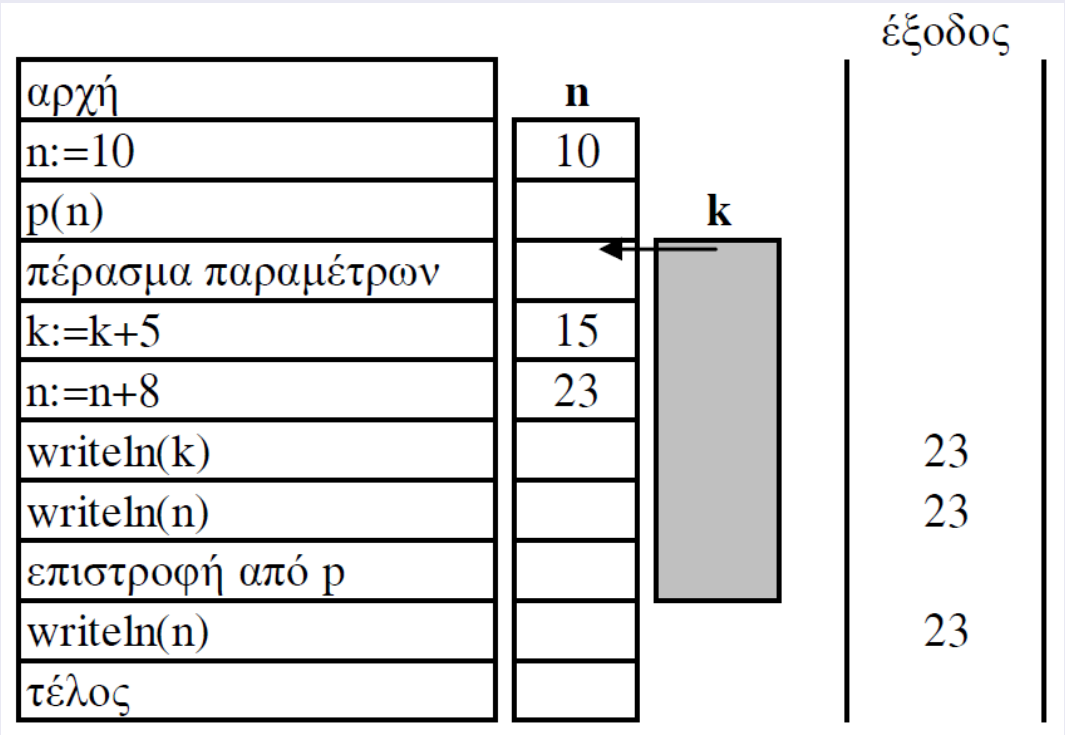
```
program CallByReferenceExample2(output);
var n:integer;

    procedure p(var k:integer);
        begin
            k:=k+5;
            n:=n+8;
            writeln(k);
            writeln(n);
        end;

begin
  n:=10; p(n); writeln(n)
end.
```

Πέρασμα παραμέτρων

Η εκτέλεση του προγράμματος συνοψίζεται στον παρακάτω πίνακα:



Πέρασμα παραμέτρων

Επειδή στο πέρασμα με αναφορά η μεταβλητή n και η τυπική παράμετρος k έχουν την ίδια αναφορά, οι δύο εντολές ανάθεσης μέσα στην p επηρεάζουν την ίδια θέση μνήμης.

Μετά την εκτέλεσή τους οι n και k έχουν και οι δύο τιμή 23, η οποία είναι και η τιμή της n μετά την εκτέλεση της p.

Πέρασμα παραμέτρων

Αν το πέρασμα της παραμέτρου k στη διαδικασία p γίνει με τιμή αποτέλεσμα, τότε η εκτέλεση του προγράμματος θα είναι αυτή που φαίνεται στον παρακάτω πίνακα:

	n	**k**	έξοδος
αρχή			
n:=10	10		
p(n)			
πέρασμα παραμέτρων		10	
k:=k+5		15	
n:=n+8	18		
writeln(k)			15
writeln(n)			18
επιστροφή από p	15		
writeln(n)			15
τέλος			

Πέρασμα παραμέτρων

Επειδή στο πέρασμα με τιμή-αποτέλεσμα η μεταβλητή n και η τυπική παράμετρος k έχουν διαφορετική αναφορά, οι δύο εντολές ανάθεσης μέσα στην p επηρεάζουν διαφορετικές θέσεις μνήμης.

Μετά την εκτέλεσή της p, τη τιμή 15 που είναι η τελική τιμή της k αντιγράφεται στην n.

Πέρασμα παραμέτρων

Πέρασμα παραμέτρων με αποτέλεσμα:

- Η τυπική παράμετρος αποθηκεύται τοπικά στο εγγράφημα δραστηριοποίησης του υποπρογράμματος.
- Κατά τη διάρκεια εκτέλεσης του υποπρογράμματος δεν υπάρχει καμία αλληλεπίδραση ανάμεσα στην τυπική και την πραγματική παράμετρο.
- Μετά την ολοκλήρωση του υποπρογράμματος η τιμή της τυπικής παραμέτρου αντιγράφεται στην πραγματική παράμετρο (η οποία θα πρέπει να είναι μεταβλητή ή κάποια έκφραση που προσδιορίζει μία θέση στη μνήμη στην οποία μπορεί να αποθηκευτεί τιμή).

Πέρασμα παραμέτρων

Το πέρασμα παραμέτρων με αποτέλεσμα υποστηρίζεται από την Ada για βαθμωτούς τύπους.

Πέρασμα παραμέτρων

Πέρασμα παραμέτρων με μοίρασμα:

- Χρησιμοποιήται από γλώσσες που υποστηρίζουν το μοντέλο αναφορών για τις μεταβλητές.
- Κατά την κλήση του υποπρογράμματος το καλούν τμήμα του κώδικα περνάει στο υποπρόγραμμα την αναφορά του αντικειμένου με το οποίο συνδέεται η πραγματική παράμετρος (ή του αντικειμένου που προκύπτει από την αποτίμηση της πραγματικής παραμέτρου, αν αυτή είναι συνθετη παράσταση) και η τυπική παράμετρος συνδέεται με αυτό το αντικείμενο.
- Κατ΄ αυτόν τον τρόπο η πραγματική και η τυπική παράμετρος στην αρχή της εκτέλεσης του υποπρογράμματος είναι συνδεδεμένες με το ίδιο αντικείμενο.

Πέρασμα παραμέτρων

Αν στο διάστημα κατά το οποίο η πραγματική παράμετρος και η τυπική παράμετρος είναι συνδεδεμένες με το ίδιο αντικείμενο το αντικείμενο αυτό τροποποιηθεί, τότε η αλλαγή αυτή επηρεάζει και τόσο την τυπική όσο και την πραγματική παράμετρο. Αυτό μπορεί να συμβεί μόνο στην περίπτωση που το εν λόγω αντικείμενο είναι μεταλλάξιμο.

Σημειώνεται ότι η τροποποίηση μπορεί να γίνει είτε μέσω της τυπικής παραμέτρου, είτε μέσω της παραγματικής παραμέτρου (αν είναι ορατή στο υποπρόγραμμα) είτε μέσω κάποιας άλλου ψευδώνυμου που μπορεί να έχει το αντικείμενο.

Πέρασμα παραμέτρων

Αν η τυπική παράμετρος συνδεθεί με άλλο αντικείμενο (π.χ. μέσω εντολής ανάθεσης) τότε αυτό δεν επηρεάζει την πραγματική παράμετρο: το υποπρόγραμμα δεν μπορεί να συνδέσει την πραγματική παράμετρο με διαφορετικό αντικείμενο.

Μετά από το σημείο δεν υπάρχει συσχετισμός ανάμεσα στην πραγματική και την τυπική παράμετρο.

Πέρασμα παραμέτρων

Παρότι ο τρόπος υλοποίησης του περάσματος με μοίρασμα και του πέρασματος με αναφορά είναι παρόμοιοι (το καλούν τμήμα κώδικα περνάει στο υποπρόγραμμα μία αναφορά), εντούτοις διαφέρουν ως προς το αποτέλεσμα, λόγω της αλληλεπίδρασης με την εντολή ανάθεσης, η οποία λειτουργεί διαφορετικά στο μοντέλο αναφορών από ότι στο μοντέλο τιμών.

Πέρασμα παραμέτρων

Αν η πραγματική παράμετρος είναι συνδεδεμένη με ένα μη μεταλλάξιμο αντικείμενο, τότε η τιμή της δεν μπορεί να αλλάξει από το υποπρόγραμμα: το αντίκείμενο δεν μπορεί να τροποποιηθεί και το υποπρόγραμμα δεν μπορεί να συνδέσει τη μεταβλητή μέ άλλο αντικείμενο.

Επίσης αν η τυπική παράμετρος χρησιμοποιηθεί στα αριστερά του τελεστή ανάθεσης σε κάποια εντολή, τότε αλλάζει η τιμή της, καθώς αυτή συνδέεται με κάποιο άλλο αντικείμενο, αλλά η τιμή της πραγματικής παραμέτρου παραμένει η ίδια.

Πέρασμα παραμέτρων

Το πέρασμα παραμέτρων με μοίρασμα χρησιμοποιείται από την Clu, την Python, τη Ruby, τη Smalltalk και τη Java.

Στις γλώσσες που χρησιμοποιούν μοντέλο τιμών για τις μεταβλητές, το αντίστοιχο αποτέλεσμα μπορεί να επιτευχθεί περνώντας μεταβλητές τύπου δείκτη με τιμή.

Πέρασμα παραμέτρων

Παράδειγμα

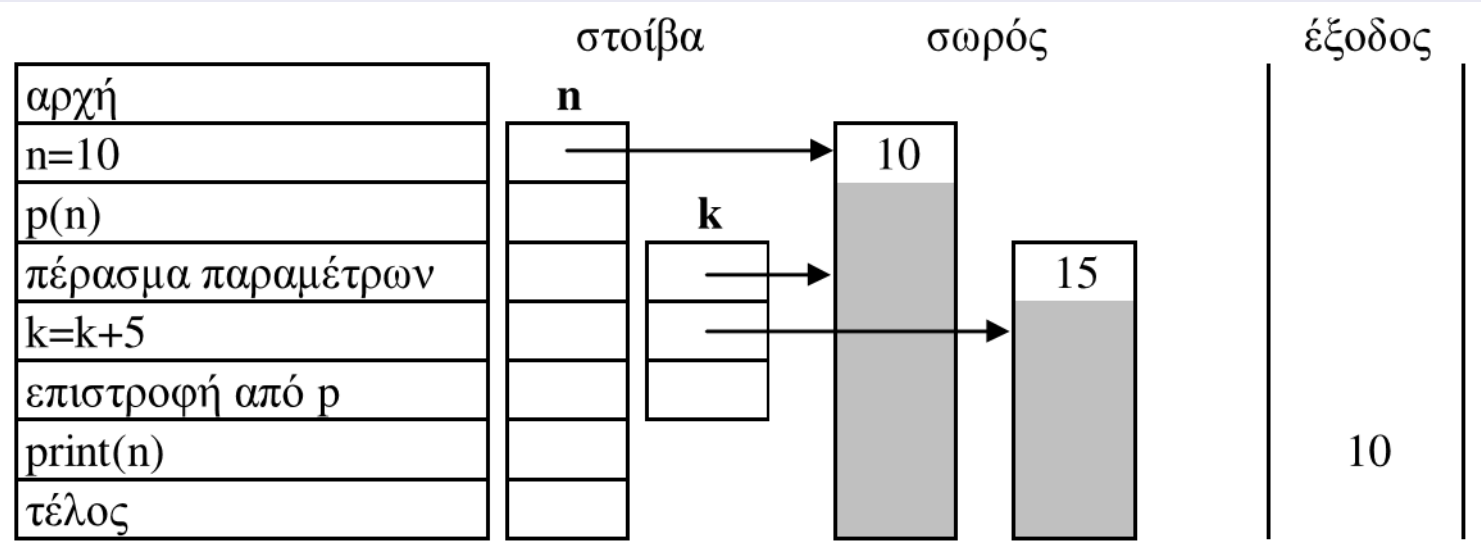
Εστω το παρακάτω πρόγραμμα Python:

```
def p(k):
        k=k+5

n = 10
p(n)
print(n)
```

Πέρασμα παραμέτρων

Η εκτέλεση του προγράμματος συνοψίζεται στον παρακάτω πίνακα:



Πέρασμα παραμέτρων

Μετά το πέρασμα παραμέτρων στο υποπρόγραμμα p η μεταβλητή n και η τυπική παράμετρος k είναι συνδεδεμένες με το ίδιο αντικείμενο που έχει τιμή 10.

Η εντολή k=k+5 αναθέτει στην k την τιμή 15. Αυτό γίνεται συνδέοντας την k με ένα διαφορετικό αντικείμενο. Το αντικείμενο με το οποίο είναι συνδεδεμένη η n εξακολουθεί να έχει τιμή 10.

Μετά την ολοκλήρωση της εκτέλεσης του υποπρογράμματος p η μεταβλητή n έχει την ίδια τιμή που είχε και πριν την κλήση του υποπρογράμματος. Επειδή η p είναι συνδεδεμένη με ένα μη-μεταλλάξιμο αντικείμενο (ακέραιο) η τιμή της δεν μπορεί να αλλάξει μέσω του περάσματος παραμέτρων με μοίρασμα.

Πέρασμα παραμέτρων

Παράδειγμα

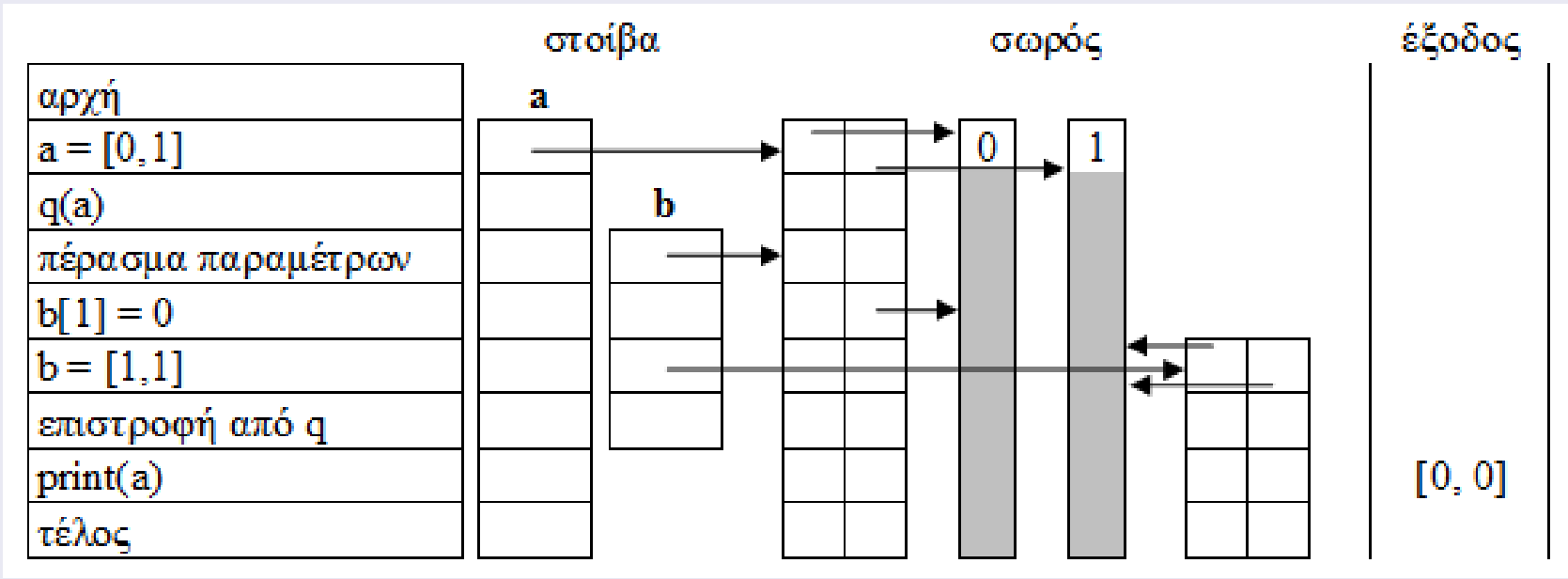
Εστω το παρακάτω πρόγραμμα Python:

```
def q(b):
        b[1] = 0
        b = [1,1]

a = [0,1]
q(a)
print(a)
```

Πέρασμα παραμέτρων

Η εκτέλεση του προγράμματος συνοψίζεται στον παρακάτω πίνακα:



Πέρασμα παραμέτρων

Μετά το πέρασμα παραμέτρων στο υποπρόγραμμα q η μεταβλητή a και η τυπική παράμετρος b είναι συνδεδεμένες με το ίδιο αντικείμενο που είναι η λίστα [0,1].

Η εντολή b[1] = 0 τροποποιεί την παραπάνω λίστα, αλλάζοντας την τιμή της b και της a. Αυτό είναι εφικτό καθώς οι λίστες στην Python είναι μεταλλάξιμα αντικείμενα.

Η εντολή b = [1,1] συνδέει την b με μία διαφορετική λίστα. Ωστόσο η λίστα με την οποία είναι συνδεδεμένη η a παραμένει η ίδια.

Μετά την ολοκλήρωση της εκτέλεσης του υποπρογράμματος q η μεταβλητή a έχει την τιμή [0,0]: είναι συνδεδεμένη με το ίδιο αντικείμενο όπως και πριν την κλήση, ωστόσο αυτό το αντικείμενο έχει τροποποιηθεί εσωτερικά.

Πέρασμα παραμέτρων

Πέρασμα παραμέτρων με όνομα:

- Αποτελεί έναν τρόπο περάσματος παραμέτρων που προσπαθεί να μιμηθεί τη λειτουργία των μακροεντολών.
- Κατά την κλήση του υποπρογράμματος οι πραγματικές παράμετροι ενσωματώνονται στο σώμα του υποπρογράμματος και δεν αποτιμούνται άμεσα.
- Η αποτίμηση γίνεται (χρησιμοποιώντας το περιβάλλον αναφοράς του καλούντος) κάθε φορά που χρειάζεται η τιμή της τυπικής παραμέτρου.

Πέρασμα παραμέτρων

Το πέρασμα παραμέτρων με όνομα:

- έχει το ίδιο αποτέλεσμα όπως το με πέρασμα με τιμή, αν η πραγματική παράμετρος είναι μία σταθερή τιμή.
- έχει το ίδιο αποτέλεσμα όπως το πέρασμα με αναφορά, αν η πραγματική παράμετρος είναι μεταβλητή.
- έχει διαφορετικό αποτέλεσμα από τους άλλους τρόπους περάσματος παραμέτρων, αν η πραγματική παράμετρος είναι στοιχείο πίνακα, ή σύνθετη παράσταση που περιέχει μεταβλητές.

Πέρασμα παραμέτρων

Το πλεονέκτημα του περάσματος παραμέτρων με όνομα είναι ότι αποφεύγει την άσκοπη αποτίμηση των πραγματικών παραμέτρων, στην περίπτωση που αυτές δεν χρησιμοποιούνται από το υποπρόγραμμα.

Ωστόσο το μειονέκτημα του είναι ότι αν η πραγματική παράμετρος χρησιμοποιείται σε πολλά σημεία του υποπρογράμματος τότε αποτιμάται κάθε φορά από την αρχή. Επίσης παρουσιάζει δυσκολίες στην υλοποίηση.

Το πέρασμα παραμέτρων με όνομα χρησιμοποιήθηκε από την Algol 60

Πέρασμα παραμέτρων

Οκνηρή αποτίμηση (πέραμα παραμέτρων με ανάγκη):

- Είναι παραλλαγή του περάσματος με όνομα και χρησιμοποιείται από τις συναρτησιακές γλώσσες.
- Η αποτίμηση μίας πραγματικής παραμέτρου γίνεται την πρώτη φορά που θα χρειαστεί η τιμή της για το υπολογισμό του αποτελέσματος της συνάρτησης και στη συνέχεια αποθηκεύεται.
- Αν η τιμή της πραγματικής παραμέτρου χρειαστεί ξανά, τότε χρησιμοποιείται η αποθηκευμένη τιμή και δεν γίνεται εκ νέου αποτίμηση.

Το πέρασμα παραμέτρων με ανάγκη χρησιμοποιείται στη Haskell και στη Miranda.

Περιεχόμενα

Εγγραφήματα δραστηριοποίησης και ακολουθίες κλήσεις

Κάθε φορά που καλείται ένα υποπρόγραμμα κατά την διάρκεια της εκτέλεσης ενός προγράμματος, εισάγεται στην κορυφή της στοίβας το εγγράφημα δραστηριοποίησης του υποπρογράμματος.

Το εγγράφημα δραστηριοποίησης διαγράφεται από τη στοίβα όταν ολοκληρωθεί η εκτέλεση του υποπρογράμματος.

Εγγραφήματα δραστηριοποίησης και ακολουθίες κλήσεις

Ο καταχωρητής δείκτη στοίβας, περιέχει κάθε χρονική στιγμή τη διεύθυνση της κορυφής της στοίβας.

Ο καταχωρητής δείκτη βάσης, περιέχει κάθε χρονική στιγμή τη διεύθυνση ενός σημείου αναφοράς στο εγγράφημα δραστηριοποίησης του τρέχοντος υποπρογράμματος.

Επειδή το εγγράφημα δραστηριοποίησης ενδέχεται να περιέχει αντικείμενα μεταβλητού μεγέθους, είναι απαραίτητη η χρήση και των δύο παραπάνω δεικτών.

Εγγραφήματα δραστηριοποίησης και ακολουθίες κλήσεις

Το εγγράφημα δραστηριοποίησης περιέχει:

- τις τυπικές παραμέτρους του υποπρογράμματος
- τις τοπικές μεταβλητές
- τις τιμές των καταχωρητών πριν την κλήση
- τη διεύθυνση επιστροφής
- το στατικό δείκτη
- τον αποθηκευμένο δείκτη στοίβας
- τον αποθηκευμένο δείκτη βάσης (δυναμικό δείκτη, δείκτη βάσης του καλούντος)

Εγγραφήματα δραστηριοποίησης και ακολουθίες κλήσεις

Για τη δέσμευση μνήμης στη στοίβα και την αρχικοποίηση του εγγραφήματος δραστηριοποίησης κατά την κληση του υποπρογράμματος καθώς και για την επαναφορά της πληροφορίας που έχει αποθηκευτεί στο εγγράφημα δραστηριοποίησης και την αποδεύσμευση της μνήμης στη στοίβα μετά την ολοκλήρωση του υποπρογράμματος, χρειάζεται να εκτελεστεί κώδικας ο οποίος χωρίζεται σε:

- ακολουθία κλήσης (εκτελείται απο τον καλούντα)
- πρόλογος (εκτελείται απο τον καλούμενο)
- επίλογος (εκτελείται απο τον καλούμενο)

Εγγραφήματα δραστηριοποίησης και ακολουθίες κλήσεις

Ορισμένες ενέργειες που αναφέραμε πρέπει υποχρεωτικά να συμπεριλαμβάνονται στην ακολουθία κλήσης (πέρασμα παραμέτρων, υπολογισμός στατικού δείκτη, αποθήκευση διεύθυνσης επιστροφής).

Αντίστοιχα, ορισμένες ενέργειες πρέπει υποχρεωτικά να συμπεριλαμβάνονται στον πρόλογο/επίλογο (μεταφορά της διεύθυνσης επιστροφής στον μετρητή προγράμματος).

Εγγραφήματα δραστηριοποίησης και ακολουθίες κλήσεις

Ωστόσο υπάρχουν ενέργειες που μπορούν να γίνουν είτε στην ακολουθία κλήσης είτε στον πρόλογο/επίλογο (π.χ. αποθήκευση δείκτη στοίβας, αποθήκευση καταχωρητών).

Η μεταφορά ενεργειών από την ακολουθία κλήσης στον πρόλογο/επίλογο συνεπάγεται εξοικονόμηση χώρου (ο αντίστοιχος κώδικας υπάρχει μία μόνο φορά πριν και μετά τον κώδικα του υπορογράμματος και όχι σε κάθε σημείο από το οποίο καλείται το υποπρόγραμμα).

Περιεχόμενα

Ενσωμάτωση υποπρογραμμάτων

Σε ορισμένες υλοποιήσεις γλωσσών, επιλέγεται κατά την μετάφραση ενός προγράμματος αντί του μηχανισμού κλήσης υποπρογράμματος που βασίζεται στη χρήση της στοίβας, να γίνεται ενσωμάτωση του κώδικα του υποπρογράμματος σε κάθε σημείο από το οποίο καλείται το υποπρόγραμμα.

Το πλεονέκτημα της ενσωμάτωσης είναι ότι πολλές ενέργειες που απαιτούνται για τη διαχείριση της στοίβας μπορούν να παραληφθούν.

Επιπλέον μπορούν να εφαρμοστούν ορισμένες πρόσθετες βελτιώσεις στον παραγόμενο κώδικα.

Ενσωμάτωση υποπρογραμμάτων

Το μειονέκτημα της ενσωμάτωσης είναι ότι αυξάνει το μέγεθος του μεταφρασμένου προγράμματος (καθώς ο κώδικας του υποπρογράμματος επαναλαμβάνεται σε πολλά σημεία).

Συνεπώς συνίσταται μόνο για μικρά υποπρογράμματα.

Επίσης δεν μπορεί να εφαρμοστεί σε αναδρομικά υποπρογράμματα.

Ενσωμάτωση υποπρογραμμάτων

Σε πολλές υλοποιήσεις που υποστηρίζουν ενσωμάτωση, ο μεταφραστής είναι αυτός που αποφασίζει ποια υποπρογράμματα θα ενσωματωθούν.

Στην Ada ο προγραμματιστής μπορεί να προτείνει ποια υποπρογράμματα θα ενσωματωθούν.

Περιεχόμενα

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Ορισμένες γλώσσες προγραμματισμού επιτρέπουν τον ορισμό προκαθορισμένων τιμών για κάποιες από τις παραμέτρους (Python, C++, Ada, Common Lisp, Fortran 90).

Οι παράμετροι που έχουν προκαθοριμένες τιμές μπορούν να παραλείπονται κατά την κλήση της συνάρτησης.

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Στην Python οι παράμετροι με προκαθοριμένες τιμές θα πρέπει ακολουθούν τις παραμέτρους χωρίς προκαθορισμένη τιμή στην επικεφαλίδα του υποπρογράμματος.

Κατά την κλήση του υποπρογράμματος δίνονται υποχρεωτικά τιμές για τις παραμέτρους χωρίς προκαθορισμένη τιμή και προαιρετικά τιμές για ένα αρχικό πλήθος από τις παραμέτρους με προκαθοριμένη τιμή.

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Παράδειγμα

Εστω το παρακάτω υποπρόγραμμα Python:

```
def totalCost(price, quantity = 1, vat = 23, discount = 0):
        return price*quantity*(1+vat/100)*(1-discount/100)
```

όλες οι παρακάτω κλήσεις είναι έγκυρες:

```
totalCost(35.50)
totalCost(35.50, 3)
totalCost(35.50, 3, 13)
totalCost(35.50, 3, 13, 10)
```

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Κατά την κλήση ενός υποπρογράμματος η αντιστοιχία των πραγματικών παραμέτρων με τις τυπικές παραμέτρους γίνεται με βάση τη θέση τους: η πρώτη πραγματική παράμετρος αντιστοιχεί στην πρώτη τυπική παράμετρο, η δεύτερη πραγματική παράμετρος στη δεύτερη τυπική παράμετρο κ.λ.π.

Οι πραγματικές παράμετροι που χρησιμοποιούν τον παραπάνω κανόνα ονομάζονται πραγματικές παράμετροι θέσης.

Ορισμένες γλώσσες προγραμματισμού επιτρέπουν την αντιστοίχιση πραγματικών παραμέτρων με τυπικές παραμέτρους, χρησιμοποιώντας τα ονόματα των τυπικών παραμέτρων κατά την κλήση του υποπρογράμματος (Python, Ada, Common Lisp, Fortran 90, Modula 3).

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Η χρήση επώνυμων πραγματικών παραμέτρων κατά την κλήση ενός υποπρογράμματος έχει τα παρακάτω πλεονεκτήματα:

- Επιτρέπει να αλλάξει η σειρά εμφάνισης των ορισμάτων στην κλήση.
- Επιτρέπει να δοθεί τιμή σε κάποια προκαθορισμένη παράμετρο, χωρίς να δοθεί τιμή στις προκαθοριμένες παραμέτρους που βρίσκονται αριστερά της στην επικεφαλίδα του υποπρογράμματος.
- Βελτιώνει την αναγνωσιμότητα του κώδικα: αν τα ονομάτα των παραμέτρων αντανακλούν το ρόλο τους μέσα στο υποπρόγραμμα τότε με την παρουσία των ονομάτων κατά την κλήση τεκμηριώνεται καλύτερα το πρόγραμμα.

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Στην Python η χρήση επώνυμων παραμέτρων υπακούει στους παρακάτω περιορισμούς:

- Οι επώνυμες παράμετροι θα πρέπει να έπονται των παραμέτρων θέσης κατά την κλήση του υποπρογράμματος.
- Όλες οι τυπικές παράμετροι χωρίς προκαθορισμένη τιμή θα πρέπει να καθορίζονται είτε από πραγματικές παραμέτρους θέσης είτε από επώνυμες πραγματικές παραμέτρους.
- Η τιμή κάθε τυπικής παραμέτρου θα πρέπει να καθορίζεται το πολύ μία φορά κατά την κλήση.

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Παράδειγμα

Μπορούμε κατά την κλήση του υποπρογράμματος totalCost του προηγούμενου παραδείγματος να αλλάξουμε τη σειρά των παραμέτρων χρησιμοποιώντας επώνυμες παραμέτρους:

```
totalCost(quantity = 4, price = 10.25, discount = 5, vat = 13)
```

Επίσης μπορούμε να καθορίσουμε την τιμή της παραμέτρου discount, διατηρώντας τις προκαθορισμένες τιμές των παραμέτρων quantity και vat:

```
totalCost(35.50, discount = 10)
totalCost(discount = 10, price = 35.50)
```

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Ορισμένες γλώσσες προγραμματισμού επιτρέπουν τον ορισμό υποπρογραμμάτων που δέχονται μεταβλητό (και οσοδήποτε μεγάλο) πλήθος ορισμάτων (Python, C, Lisp).

Οι πραγματικές παράμετροι πέραν του ελάχιστου υποχρεωτικού πλήθους παραμέτρων συγκεντρώνονται σε έναν πίνακα, μία λίστα ή μία πλειάδα την οποία μπορεί να επεξεργαστεί το υποπρόγραμμα.

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Στην Python για να ορίσουμε ένα υποπρόγραμμα με μεταβλητό πλήθος παραμέτρων, χρησιμοποιούμε μια παράμετρο, του ονόματος της οποίας στην επικεφαλίδα του υποπρογράμματος προηγείται το σύμβολο *. Η χρήση των *-παραμέτρων ακολουθεί τους παρακάτω κανόνες:

- Υπάρχει το πολύ μία *-παράμετρος σε κάθε υποπρόγραμμα.
- Η *-παράμετρος μπορεί να βρίσκεται σε οποιαδήποτε θέση στη λίστα των τυπικών παραμέτρων. Ωστόσο όσες τυπικές παράμετροι ακολουθούν θα πρέπει να πάρουν τιμή κατά την κλήση από επώνυμες πραγματικές παραμέτρους. Συνήθως η *-παράμετρος τοποθετείται στο τέλος της λίστας των τυπικών παραμέτρων.
- Αν η *-παράμετρος βρίσκεται στην i-οστή θέση της λίστας των τυπικών παραμέτρων, τότε η τιμή της κατά την κλήση του υποπρογράμματος είναι μία πλειάδα που περιέχει όλες τις πραγματικές παραμέτρους θέσης, από τη θέση i και μετά.

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Παράδειγμα

Παρακάτω φαίνεται η επικεφαλίδα ενός υποπρογράμματος σε Python με μεταβλητό πλήθος παρμέτρων:

```
def f(a, b, *z, c, d = 10):
```

Αν το υποπρόγραμμα κληθεί με τον παρακάτω τρόπο:

```
f(1, 2, 3, 4, 5, 6, 7, c=8)
```

τότε η παράμετρος z έχει τιμή (3, 4, 5, 6, 7).

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Παράδειγμα

Το παρακάτω υποπρόγραμμα Python βρίσκει το μέγιστο κοινό διαρέτη ενός οποιουδήποτε πλήθους θετικών αριθμών:

```
def gcd(*numbers):
        n = numbers[-1]
        if n<1:
            return "error: non-positive argument"
        for i in range(len(numbers)-1):
            m = numbers[i]
            if m<1:
                    return "error: non-positive argument"
            while m != n:
                    if m > n:
                            m = m-n
                    else:
                            n = n-m
        return n
```

Προκαθορισμένες, επώνυμες και μεταβλητού πλήθους παράμετροι

Όλες οι παρακάτω κλήσεις είναι έγκυρες:

```
>>> gcd(24,10)
>>> gcd(32,16,12)
>>> gcd(100,24,25,30,17)
>>> gcd(5)
```

Περιεχόμενα

Υπερφόρτωση και πολυμορφισμός

Υπερφόρτωση ονομάζουμε το φαινόμενο δύο υποπρογράμματα που ορίζονται μέσα στο ίδιο μπλοκ να έχουν το ίδιο όνομα, αλλά να λειτουργούν για διαφορετικό πλήθος ή/και τύπο παραμέτρων.

Η υπερφόρτωση είναι μία συντακτική ευκολία η οποία, βοηθάει ώστε το όνομα ενός υποπρογράμματος να περιγράφει όσο το δυνατόν καλύτερα τη λειτουργία που εκτελεί.

Η Java, η C++ και η ADA υποστηρίζουν υπερφόρτωση υποπρογραμμάτων.

Υπερφόρτωση και πολυμορφισμός

Ένα υποπρόγραμμα ονομάζεται γενικό ή πολυμορφικό αν περιέχει τυπικές παραμέτρους οι οποίες επιτρέπεται να έχουν διαφορετικούς τύπους, σε διαφορετικές κλήσεις του υποπρογράμματος.

Τα πολυμορφικά προγράμμα επιτρέπουν λειτουργίες που εκτελούνται με τον ίδιο τρόπο για διαφορετικούς τύπους δεδομένων να υλοποιούνται από ένα μοναδικό υποπρόγραμμα.

Η Haskell, η C++ και η ADA υποστηρίζουν πολυμορφικά υποπρογράμματα.

Υπερφόρτωση και πολυμορφισμός

Η υπερφόρτωση και ο πολυμορφισμός μοιάζουν στο ότι επιτρέπουν να καλούμε υποπρογράμματα με το ίδιο όνομα για διαφορετικούς τύπους παραμέτρων.

Ωστόσο ο τρόπος που επιτυγχάνεται αυτό είναι τελείως διαφορετικός στις δύο περιπτώσεις:

- Στο πολυμορφισμό υπάρχει ένα μοναδικό υποπρόγραμμα το οποίο λειτουργεί για πολλούς τύπους
- Στην υπερφόρτωση υπάρχουν πολλά διαφορετικά υποπρογράμματα που έχουν κοινό όνομα, το καθένα από τα οποία λειτουργεί για συγκεκριμένο πλήθος και τύπους παραμέτρων. Οι λειτουργίες που εκτελούν τα διάφορα υποπρογράμματα ενδέχεται να είναι τελειώς διαφορετικές.