Πανεπιστήμιο Ιωαννίνων
Πολυτεχνική Σχολή
Τμήμα Μηχανικών Η/Υ και Πληροφορικής
"Μεταφραστές"
Διδάσκων: Γ. Μανής
Φεβρουάριος 2020

Προγραμματιστική άσκηση:

Η γλώσσα προγραμματισμού minimal++

Η minimal++ είναι μια μικρή γλώσσα προγραμματισμού φτιαγμένη με βάση τις ανάγκες της προγραμματιστικής άσκησης του μαθήματος. Παρόλο που οι προγραμματιστικές της ικανότητες είναι μικρές, η εκπαιδευτική αυτή γλώσσα περιέχει πλούσια στοιχεία και η κατασκευή του μεταγλωττιστή της έχει να παρουσιάσει αρκετό ενδιαφέρον, αφού περιέχονται σε αυτήν πολλές εντολές που χρησιμοποιούνται από άλλες γλώσσες, καθώς και κάποιες πρωτότυπες. Η minimal++ υποστηρίζει συναρτήσεις και διαδικασίες, μετάδοση παραμέτρων με αναφορά και τιμή, αναδρομικές κλήσεις και άλλες ενδιαφέρουσες δομές. Επίσης, επιτρέπει φώλιασμα στη δήλωση συναρτήσεων κάτι που λίγες γλώσσες υποστηρίζουν (το υποστηρίζει η Pascal, δεν το υποστηρίζει η C).

Από την άλλη όμως πλευρά, η minimal++ δεν υποστηρίζει βασικά προγραμματιστικά εργαλεία όπως η δομή *for*, ή τύπους δεδομένων όπως οι πραγματικοί αριθμοί και οι συμβολοσειρές. Οι παραλήψεις αυτές έχουν γίνει ώστε να απλουστευτεί η διαδικασία κατασκευής του μεταγλωττιστή, μία απλούστευση όμως που έχει να κάνει μόνο με τη μείωση των γραμμών κώδικα και όχι με τη δυσκολία κατασκευής του ή την εκπαιδευτική αξία της άσκησης.

Παρακάτω παρουσιάζεται μία περιγραφή της γλώσσας:

Λεκτικές μονάδες

Το αλφάβητο της minimal++ αποτελείται από:

- τα μικρά και κεφαλαία γράμματα της λατινικής αλφαβήτου («A»,…,«Z» και «a»,…,«z»),
- τα αριθμητικά ψηφία («0»,…,«9»),
- τα σύμβολα των αριθμητικών πράξεων («+», «-», «*», «/»),
- τους τελεστές συσχέτισης «<», «>», «=», «<=», «>=», «<>»,
- το σύμβολο ανάθεσης «:=»,
- τους διαχωριστές («;», «,», «:»)
- καθώς και τα σύμβολα ομαδοποίησης («(»,«)»,«[»,«]»,«{»,«}»)

- και διαχωρισμού σχολίων («/*»,«*/»,«//»).

Τα σύμβολα «[» και «]» χρησιμοποιούνται στις λογικές παραστάσεις όπως τα σύμβολα «(» και «)» στις αριθμητικές παραστάσεις.

Οι δεσμευμένες λέξεις είναι:

program	***declare***				
if	***then***	***else***			
while	~~***doublewhile***~~	~~***loop***~~	~~***exit***~~		
forcase	~~***incase***~~	***when***	***default***		
not	***and***	***or***			
function	***procedure***	***call***	***return***	***in***	***inout***
input	***print***				

Οι λέξεις αυτές δεν μπορούν να χρησιμοποιηθούν ως μεταβλητές. Οι σταθερές της γλώσσας είναι ακέραιες σταθερές που αποτελούνται από προαιρετικό πρόσημο και από μία ακολουθία αριθμητικών ψηφίων.

Τα αναγνωριστικά της γλώσσας είναι συμβολοσειρές που αποτελούνται από γράμματα και ψηφία, αρχίζοντας όμως από γράμμα. Ο μεταγλωττιστής λαμβάνει υπόψη του μόνο τα τριάντα πρώτα γράμματα. Οι λευκοί χαρακτήρες (tab, space, return) αγνοούνται και μπορούν να χρησιμοποιηθούν με οποιονδήποτε τρόπο χωρίς να επηρεάζεται η λειτουργία του μεταγλωττιστή, αρκεί βέβαια να μην βρίσκονται μέσα σε δεσμευμένες λέξεις, αναγνωριστικά, σταθερές. Το ίδιο ισχύει και για τα σχόλια, τα οποία πρέπει να βρίσκονται μέσα στα σύμβολα /* και */ ή να βρίσκονται μετά το σύμβολο // και ως το τέλος της γραμμής. Απαγορεύεται να ανοίξουν δύο φορές σχόλια, πριν τα πρώτα κλείσουν. Δεν υποστηρίζονται εμφωλευμένα σχόλια.

Μορφή προγράμματος

program *id*
{
    declarations
    subprograms
    sequence of statements
}

Τύποι και δηλώσεις μεταβλητών

Ο μοναδικός τύπος δεδομένων που υποστηρίζει η minimal++ είναι οι ακέραιοι αριθμοί. Οι ακέραιοι αριθμοί πρέπει να έχουν τιμές από –32767 έως 32767. Η δήλωση γίνεται με την εντολή ***declare***. Ακολουθούν τα ονόματα των αναγνωριστικών χωρίς καμία άλλη δήλωση, αφού γνωρίζουμε ότι πρόκειται για ακέραιες μεταβλητές και χωρίς να είναι αναγκαίο να βρίσκονται στην ίδια γραμμή. Οι μεταβλητές χωρίζονται μεταξύ τους με κόμματα. Το τέλος της δήλωσης αναγνωρίζεται με το ελληνικό ερωτηματικό. Επιτρέπεται να έχουμε περισσότερες των μία συνεχόμενες χρήσεις της ***declare***.

Τελεστές και εκφράσεις

Η προτεραιότητα των τελεστών από τη μεγαλύτερη στη μικρότερη είναι:

(1) Μοναδιαίοι λογικοί: «not»

(2) Πολλαπλασιαστικοί: «*», «/»

(3) Μοναδιαίοι προσθετικοί: «+», «-»

(4) Δυαδικοί προσθετικοί: «+», «-»

(5) Σχεσιακοί «=», «<», «>», «<>», «<=», «>=»

(6) Λογικό «and»,

(7) Λογικό «or»

Δομές της γλώσσας

Εκχώρηση

Id := expression

Χρησιμοποιείται για την ανάθεση της τιμής μιας μεταβλητής ή μιας σταθεράς, ή μιας έκφρασης σε μία μεταβλητή.

Απόφαση if

if *(condition)*
statements[1]
*[**else***
statements[2]*]*

Η εντολή απόφασης ***if*** εκτιμάει εάν ισχύει η συνθήκη *condition* και εάν πράγματι ισχύει, τότε εκτελούνται οι εντολές *statements*[1] που το ακολουθούν. Το ***else*** δεν αποτελεί υποχρεωτικό τμήμα της εντολής και γι' αυτό βρίσκεται σε αγκύλη. Οι εντολές *statements*[2] που ακολουθούν το ***else*** εκτελούνται εάν η συνθήκη *condition* δεν ισχύει.

Επανάληψη while

while **(***condition*)
statements

Η εντολή επανάληψης **while** επαναλαμβάνει συνεχώς τις εντολές *statements*, όσο η συνθήκη *condition* ισχύει. Αν την πρώτη φορά που θα αποτιμηθεί η *condition*, το αποτέλεσμα της αποτίμησης είναι ψευδές, τότε οι *statements* δεν εκτελούνται ποτέ.

~~**Επανάληψη loop**~~

~~***loop***~~
~~*statements*~~

~~Η εντολή επανάληψης **loop** επαναλαμβάνει για πάντα τις εντολές *statements*. Έξοδος από το βρόχο γίνεται μόνο όταν κληθεί η εντολή **exit**~~

Επανάληψη forcase

forcase
(***when:*** *(condition)****:*** *statements*[1] *)**
default: *statements*[2]

Η δομή επανάληψης ***forcase*** ελέγχει τις *condition* που βρίσκονται μετά τα ***when***. Μόλις μία από αυτές βρεθεί αληθής, τότε εκτελούνται οι *statements*[1] που ακολουθούν. Μετά ο έλεγχος μεταβαίνει στην αρχή της ***forcase.*** Αν καμία από τις ***when*** δεν ισχύει, τότε ο έλεγχος μεταβαίνει στη ***default*** και εκτελούνται οι *statements*[2]. Στη συνέχεια ο έλεγχος μεταβαίνει έξω από την ***forcase***.

~~**Επανάληψη incase**~~

~~***incase***~~
~~*(**when:** (condition)**:** statements)**~~

~~Η δομή επανάληψης ***incase*** ελέγχει τις *condition* που βρίσκονται μετά τα ***when***, εξετάζοντας τες κατά σειρά. Για κάθε μία από αυτές που η αντίστοιχη *condition* ισχύει, εκτελούνται οι *statements* που ακολουθούν το σύμβολο **":"**. Θα εξεταστούν όλες οι *condition* και θα εκτελεστούν όλες οι *statements* των οποίων οι *condition* ισχύουν. Αφότου εξεταστούν όλες οι ***when***, ο έλεγχος μεταβαίνει έξω από τη δομή ***incase*** εάν καμία από τις *statements* δεν~~

~~έχει εκτελεστεί ή μεταβαίνει στην αρχή της **incase**, έαν έστω και μία από τις *statements* έχει εκτελεστεί.~~

~~Επανάληψη doublewhile~~

~~***doublewhile*** *(condition)*~~
~~*statements*[1]~~
~~***else***~~
~~*statements*[2]~~

~~Την πρώτη φορά που ο έλεγχος εισέρχεται στον βρόχο **doublewhile**, αποφασίζεται μέσα από την *condition* αν η εκτέλεση θα μεταβεί στο *statements*[1] (*true*) ή αν θα μεταβεί στο *statements*[2] (*false*). Από το *statements*[1] φεύγει, όταν η συνθήκη σταματήσει να είναι *true*. Από το *statements*[2] φεύγει όταν η συνθήκη σταματήσει να είναι *false*. Και στις δύο αυτές περιπτώσεις ο έλεγχος μεταφέρεται έξω από την δομή. Δηλαδή, δεν είναι ποτέ δυνατόν σε μία εκτέλεση της **doublewhile** ο έλεγχος να περάσει και από την *statements*[1] και από την *statements*[2].~~

Επιστροφή τιμής συνάρτησης

return *(expression)*

Χρησιμοποιείται μέσα σε συναρτήσεις για να επιστραφεί το αποτέλεσμα της συνάρτησης.

Έξοδος δεδομένων

print *(expression)*

Εμφανίζει στην οθόνη το αποτέλεσμα της αποτίμησης του *expression*

Είσοδος δεδομένων

input *(id)*

Ζητάει από τον χρήστη να δώσει μία τιμή μέσα από το πληκτρολόγιο

Κλήσης διαδικασίας

call *function_name(actual_parameters)*

Καλεί μία διαδικασία

Έξοδος από βρόχο loop

exit

Εκτελεί έξοδο από βρόχο **loop**

Υποπρογράμματα

Η minimal++ υποστηρίζει συναρτήσεις.

function *id (formal_pars)*
{
declarations
subprograms
statements
}

Η *formal_pars* είναι η λίστα των τυπικών παραμέτρων. Οι συναρτήσεις μπορούν να φωλιάσουν η μία μέσα στην άλλη και οι κανόνες εμβέλειας είναι όπως της PASCAL. Η επιστροφή της τιμής μιας συνάρτησης γίνεται με την **return.**

Η κλήση μιας συνάρτησης, γίνεται από τις αριθμητικές παραστάσεις σαν τελούμενο. π.χ.

D = a + f(**in** x)

όπου f η συνάρτηση και x παράμετρος που περνάει με τιμή.

Οι διαδικασίες συντάσσονται ως εξής:

procedure *id (formal_pars)*
{
declarations
subprograms
statements
}

Η κλήση μιας διαδικασίας, γίνεται με την **call**. π.χ.

call f(**inout** x)

όπου *f* η διαδικασία και *x* παράμετρος που περνάει με αναφορά.

Μετάδοση παραμέτρων

Η minimal++ υποστηρίζει δύο τρόπους μετάδοσης παραμέτρων:

- με σταθερή τιμή. Δηλώνεται με τη λεκτική μονάδα **in**. Αλλαγές στην τιμή της δεν επιστρέφονται στο πρόγραμμα που κάλεσε τη συνάρτηση.
- με αναφορά. Δηλώνεται με τη λεκτική μονάδα **inout**. Κάθε αλλαγή στη τιμή της μεταφέρεται αμέσως στο πρόγραμμα που κάλεσε τη συνάρτηση.

Στην κλήση μίας συνάρτησης οι πραγματικοί παράμετροι συντάσσονται μετά από τις λέξεις κλειδιά **in** και **inout**, ανάλογα με το αν περνάνε με τιμή ή αναφορά.

Κατάληξη

Τα αρχεία της minimal++ έχουν κατάληξη .min

Ζητούμενο:

Να κατασκευάσετε σε γλώσσα Python, έναν πλήρως λειτουργικό μεταγλωττιστή της γλώσσας minimal++. Ο μεταγλωττιστής πρέπει να παράγει ως τελική γλώσσα την γλώσσα assembly του επεξεργαστή MIPS.

Η εργαστηριακή άσκηση θα παραδοθεί σε τέσσερις φάσεις:

α) λεκτική συντακτική ανάλυση

β) παραγωγή ενδιάμεσου κώδικα

γ) σημασιολογική ανάλυση και πίνακας συμβόλων

δ) τελικός κώδικας και αναφορά ελέγχου ορθής λειτουργίας.