ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΜΕΤΑΓΛΩΤΤΙΣΤΕΣ
Προαιρετικές Ασκήσεις

Σημασιολογική Ανάλυση και Παραγωγή Κώδικα

Ημερομηνία Παράδοσης: όπως αναφέρεται στο compus
Ποσοστό στη τελική βαθμολογία: 10*% (1 Μονάδα στο τελικό βαθμό)*

1. A JVM Lang

Μια απλή γλώσσα προγραμματισμού δέχεται αριθμητικές εκφράσεις σε επιθεματική μορφή και κάποιες απλές εντολές. Η μορφή των αριθμητικών εκφράσεων είναι για παράδειγμα:

(3 4 +) = 3 + 4
(3.0 4.5 +) = 3.0 + 4.5
(6 4 + 5 *) = (6 + 4)*5
(x y 2 + +) = x + (y + 2)

Όπως φαίνεται παραπάνω οι εκφράσεις μπορούν να περιέχουν και μεταβλητές. Υπάρχουν οι ακόλουθες εντολές:

- **Εντολή ανάθεσης**, για παράδειγμα (x 0) που δίνει στη μεταβλητή x την τιμή 0, (x 3 4 +), που δίνει στο x την τιμή 7 κλπ.
- **Εντολή εκτύπωσης**, για παράδειγμα (print 3) που τυπώνει στην οθόνη το 3, (print x) που τυπώνει στην οθόνη την τιμή του x , (print 3 4 +) που τυπώνει στην οθόνη την τιμή 7 κλπ.

Παράδειγμα προγράμματος:

```
start simple3
(print (3 4 +))
(print (3 4 + 7 *))
(x (22 8 + 2 *) )
(print x)
end
```

Παρατηρήσεις:

- Κάθε εντολή περικλείεται σε *παρενθέσεις*. Οι εκφράσεις μπορούν να περικλείονται σε παρενθέσεις προαιρετικά.
- Η γλώσσα ΔΕΝ *υποστηρίζει type coersion* οπότε δεν επιτρέπονται πράξεις μεταξύ αριθμών διαφορετικών τύπων.
- Η γλώσσα υποστηρίζει ρητή μετατροπή τύπου (*type casting),* με τα σύμβολα int και float.
- Οι μεταβλητές *ΔΕΝ δηλώνονται*, αλλά ο τύπος τους συνάγεται από την πρώτη εντολή ανάθεσης στην οποία συμμετέχουν. Για παράδειγμα έστω η ακόλουθη εντολή, η οποία είναι η πρώτη ανάθεση του x:

```
...
(x 10)
...
```

συνάγεται ότι η μεταβλητή x είναι ακέραια και η μεταβλητή προστίθεται στον πίνακα συμβόλων. Αν η μεταβλητή υπάρχει και σε επόμενη εντολή ανάθεσης θα πρέπει να γίνει έλεγχος τύπων. Για παράδειγμα:

```
...
(x 1.0)
...
(x 3.0 4.0 +)              (1)
(x (float 3 2 +))          (2)
```

Η περίπτωση (1) πετυχαίνει γιατί τα δύο ορίσματα έχουν ίδιο τύπο (**real**), ενώ στην

περίπτωση (2), απαιτείται η ρητή μετατροπή τύπου, με την χρήση του keyword **float**.
Έστω όμως το ακόλουθο παράδειγμα:

```
...
(y 10)
...
(y 3 4 +)                    (3)
(y 3.0 2.0 +)                (4)
```

Η περίπτωση (3) παραπάνω επιτυγχάνει στον έλεγχο τύπων, ενώ η περίπτωση (4) *αποτυγχάνει* γιατί απαιτεί ρητή μετατροπή τύπου (casting), όπως φαίνεται στο ακόλουθο παράδειγμα:

```
(y (int 3.0 2.0 +))
```

Τις λεκτικές μονάδες της γλώσσας, αλλά και την σύνταξη θα τις εξάγετε από τα παραδείγματα μεταγλώττισης που ακολουθούν.

Ο τελικός κώδικας μεταγλώττισης είναι JVM assembly (jasmin). Παρακάτω δίνονται μερικά παραδείγματα μετάφρασης.

ΠΑΡΑΔΕΙΓΜΑ 1

start simple1 (**print** (3 4 +)) end	.class public simple1 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush 3 sipush 4 iadd getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V return .end method

Παρατηρήσεις στο παραπάνω πρόγραμμα:

- Το όνομα του προγράμματος γίνεται όνομα της κλάσης σε JVM assembly.
- Οι εντολές .limit locals 20 και .limit stack 20 καθορίζουν το μέγεθος της στοίβας των τελεστέων και τον αριθμό των τοπικών μεταβλητών. Για ευκολία μην το αλλάζετε. (θα θεωρήσουμε ότι πάντα το 20 είναι αρκετό και θα γίνει μεταγλώττιση σε μικρά προγράμματα).
- Ένας ακέραιος τοποθετείται στη στοίβα με την εντολή sipush (απλοποιημένη μορφή JVM).
- iadd: πρόσθεση των δύο κορυφαίων ακεραίων της στοίβας και τοποθέτηση του αποτελέσματος πάλι στη στοίβα.
- Το επόμενο κομμάτι τυπώνει την ακέραια τιμή που υπάρχει στην κορυφή της στοίβας (int μεταβλητή - (println(I))
  getstatic java/lang/System/out Ljava/io/PrintStream;
  swap
  invokevirtual java/io/PrintStream/println(I)V

ΠΑΡΑΔΕΙΓΜΑ 2

start simple2 (print (*3 4 + 7 **)) (print (**3.0 4.0 + 7.0 ***)) end	..class public simple2 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 *sipush 3* *sipush 4* *iadd* *sipush 7* *imul* getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V **ldc 3.0** **ldc 4.0** **fadd** **ldc 7.0** **fmul** getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(F)V return .end method return .end method

Παράδειγμα μεταγλώττισης μιας περισσότερο πολύπλοκης αριθμητικής έκφρασης, καθώς και εκφράσεων που περιέχουν πραγματικούς. Ένας πραγματικός προστίθεται στη στοίβα με lcd, και οι αντίστοιχες πράξεις είναι fadd, fmul, κλπ. Οι εντολές iadd fadd παίρνουν τα ορίσματα τους από την στοίβα και τοποθετούν το αποτέλεσμα στη στοίβα (stack machine).

ΠΑΡΑΔΕΙΓΜΑ 3

start simple3 (x 1) (print x) **(print (x++))** (print x) (y 10) (print y) **(print (++y))** end	.class public simple3 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush 1 istore 1 iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V **iload 1** **iinc 1 1** getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V sipush 10 istore 2 iload 2 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V **iinc 2 1** **iload 2** getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V return .end method

Παρατηρήσεις στο παραπάνω πρόγραμμα:

- η μεταβλητή x μπαίνει στη θέση 1 του χώρου των μεταβλητών. Οι επόμενες μεταβλητές μπαίνουν διαδοχικά στις θέσεις 2, 3, κοκ.
  - istore 1 αποθηκεύει την τιμή της στοίβας (TOS) στην θέση της x.
  - iload 1 φορτώνει την τιμή της x στη στοίβα, όπου το 1 η θέση στον πίνακα.
  - Ομοίως για την μεταβλητή y αλλά τώρα η θέση είναι 2.
- Η εκτύπωση γίνεται όπως παραπάνω.
- Χρήση των τελεστών "++". Προσοχή πρέπει να δοθεί στο γεγονός ότι στην περίπτωση x++, πρώτα μπαίνει η τιμή της x στην στοίβα και έπειτα αυξάνεται η τιμή της μεταβλητής, ενώ στην περίπτωση ++y πρώτα αυξάνεται η τιμή και έπειτα μπαίνει στην στοίβα.

ΠΑΡΑΔΕΙΓΜΑ 4

start simple4 (y 10) (x (3 (++y) +)) (print x) end	.class public simple4 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush 10 istore 1 sipush 3 iinc 1 1 iload 1 iadd istore 2 iload 2 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V return .end method

- η μεταβλητή x μπαίνει στη θέση 1 του χώρου των μεταβλητών. Οι επόμενες μεταβλητές μπαίνουν διαδοχικά στις θέσεις 2, 3, κοκ.
- istore 1 αποθηκεύει την τιμή της στοίβας (TOS) στην θέση της x.
- iload 1 φορτώνει την τιμή της x στη στοίβα, όπου το 1 η θέση στον πίνακα.
- Η εκτύπωση γίνεται όπως παραπάνω.

ΠΑΡΑΔΕΙΓΜΑ 5

start simple5 start simple5 (x (3 **(int 4.0)** +)) (print x) end	class public simple5 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush 3 **ldc 4.0** **f2i** iadd istore 1 iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V return .end method

Ρητή μετατροπή τύπου. Η σταθερά 4.0 μετατρέπεται σε ακέραια με κλήση της εντολής f2i.

ΠΑΡΑΔΕΙΓΜΑ 6

start simple6 (x 1) (x **(int (3.0 4.0 +))**) (print x) end	.class public simple6 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush 1 istore 1 **ldc 3.0** **ldc 4.0** **fadd** **f2i** istore 1 iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V return .end method

Παράδειγμα μετατροπής τύπου μιας ολόκληρης έκφρασης για ανάθεση σε μεταβλητή.

ΠΑΡΑΔΕΙΓΜΑ 7

start simple7 (x 1.0) (x **(float 3 4 +)**) (print x) (y 1) (y (int 1 2 +)) (print y) end	.class public simple7 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 ldc 1.0 fstore 1 **sipush 3** **sipush 4** **iadd** **i2f** fstore 1 fload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(F)V sipush 1 istore 2 **sipush 1** **sipush 2** **iadd** **istore 2** iload 2 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V return .end method

Παραδείγματά ρητής μετατροπής τύπου. Η ακέραια έκφραση (3 4 +) μετατρέπεται αυτόματα σε πραγματική, για να δοθεί ως τιμή στην μεταβλητή x, η οποία είναι πραγματική.
Η έκφραση (1 2 +) είναι ήδη ακέραια, οπότε η μετατροπή σε ακέραιο αγνοείται και απλά τυπώνεται ένα Warning (δες παρακάτω).

ΠΑΡΑΔΕΙΓΜΑ 8

start simple8 (y -1) (z -1.0) (y **(y abs)**) (print y) (z **(z abs)**) (print z) end	.class public simple8 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush -1 istore 1 ldc -1.0 fstore 2 **iload 1** **invokestatic java/lang/Math/abs(I)I** **istore 1** iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V **fload 2** **invokestatic java/lang/Math/abs(F)F** **fstore 2** fload 2 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(F)V return .end method

Παραδείγματά κλήσης συνάρτησης abs της βιβλιοθήκης java/lang/Math. Η abs συνάρτηση ορίζεται τόσο για float, όσο και για int, οπότε ανάλογα με τον τύπο του ορσίσματος θα πρέπει να κληθεί η αντίστοιχη static method της java.lang.Math

ΠΑΡΑΔΕΙΓΜΑ 9

start simple9 (y -1) (print y) (y **(12 5 max)**) (print y) end	.class public simple9 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush -1 istore 1 iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V **sipush 12** **sipush 5** **invokestatic java/lang/Math/max(II)I** **istore 1** iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V return .end method

Παραδείγματά κλήσης συνάρτησης max της βιβλιοθήκης java/lang/Math. Η γλώσσα υποστηρίζει

τόσο την συνάρτηση max, όσο και την συνάρτηση min της βιβλιοθήκης. Η max (και η min) συνάρτηση ορίζεται τόσο για float, όσο και για int, οπότε ανάλογα με τον τύπο του ορίσματος θα πρέπει να κληθεί η αντίστοιχη static method της java.lang.Math.

ΠΑΡΑΔΕΙΓΜΑ 10

start simple10 (y -1) (z -2.0) (y (12 **12 5 max** + 10 (**int z abs**) min +)) (print y) (z (12.0 **z -1.0 min** +)) (print z) end	.class public simple10 .super java/lang/Object .method public static main([Ljava/lang/String;)V .limit locals 20 .limit stack 20 sipush -1 istore 1 ldc -2.0 fstore 2 sipush 12 **sipush 12** **sipush 5** **invokestatic java/lang/Math/max(II)I** iadd sipush 10 **fload 2** **invokestatic java/lang/Math/abs(F)F** **f2i** invokestatic java/lang/Math/min(II)I iadd istore 1 iload 1 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(I)V ldc 12.0 **fload 2** **ldc -1.0** **invokestatic java/lang/Math/min(FF)F** fadd fstore 2 fload 2 getstatic java/lang/System/out Ljava/io/PrintStream; swap invokevirtual java/io/PrintStream/println(F)V return .end method

Παραδείγματά κλήσης συνάρτησης max/min της βιβλιοθήκης java/lang/Math, με μετατροπές τύπων και εφαρμογή τόσο σε float όσο και σε int.

ΣΦΆΛΜΑΤΑ

Το πρόγραμμά σας θα πρέπει να παράγει σφάλματα(errors) και προειδοποιήσεις (warnings) όπου απαιτείται, όπως φαίνεται στο ακόλουθο παράδειγμα, που περιέχεται στο αρχείο simple_error.

start simple7	
(x 1.0 2)	ERROR: syntax error, unexpected ')'. on line 2.
(y 1)	Warning: value is already int, in line 4.
(y (int 2 3 +))	Variable z NOT initialised, in line 5. ERROR: Variable fault. on line 5.
(y (z 100 +))	ERROR: Type missmatch. on line 6.
(y (10 20.0 +))	ERROR: Type missmatch. on line 6.
(w 1.0)	Warning: value is already real, in line 8.

(w (float 3.0)) (print x) (print y) end	Variable x NOT initialised, in line 9. ERROR: Variable fault. on line 9. Errors found 5. No Code Generated.

Να αναπτύξετε ένα πρόγραμμα σε flex/bison το οποίο να υλοποιεί την παραπάνω γλώσσα. Η υλοποίηση σας θα δέχεται ένα πρόγραμμα γραμμένο στη παραπάνω γλώσσα και να το μεταγλωττίζει σε JVM assembly που υποστηρίζεται από το jasmin. Αν όλα είναι σωστά τότε το τελικό πρόγραμμα που θα βγάλει ο μεταγλωττιστής μπορεί να μεταγλωττιστεί από το jasmin και να παραχθεί ένα class αρχείο "JAVA executable".

ΠΑΡΑΔΟΣΗ - Οδηγίες

Θα πρέπει να παραδώσετε ένα αρχείο coursework2_2019.zip, που θα περιέχει::

- Μια μικρή αναφορά σε μορφή pdf, σχολιασμένο τον κώδικα (θα περιλάβετε και τον κώδικα στην εργασία σας) που αφορά τα εργαλεία FLEX/BISON. **Θα πρέπει να περιγράψετε τι αλλαγές/προσθήκες κάνατε για να υλοποιήσετε το αντίστοιχο παράδειγμα (1-10). Η δομή της αναφοράς σας πρέπει να έχει ξεκάθαρα ένα section για κάθε παράδειγμα.**
- **Το γραπτό κείμενο θα βαθμολογηθεί με 15%.**
- **ΣΤΗΝ ΠΡΩΤΗ ΣΕΛΙΔΑ Η ΑΝΑΦΟΡΑ ΣΑΣ ΝΑ ΠΕΡΙΕΧΕΙ ΤΟ ΟΝΟΜΑ ΚΑΙ ΤΟΝ ΑΜ ΣΑΣ.**
- Τα αρχεία που αντιστοιχούν στις λύσεις των ασκήσεων, **οργανωμένα σε καταλόγους όπως στο compus.**
- **ΟΛΑ ΤΑ ΟΝΟΜΑΤΑ ΤΩΝ ΑΡΧΕΙΩΝ ΠΟΥ ΘΑ ΠΑΡΑΔΟΣΕΤΕ ΝΑ ΕΙΝΑΙ ΜΕ ΑΓΓΛΙΚΟΥΣ ΧΑΡΑΚΤΗΡΕΣ.**

•

<u>Σημειώσεις</u>:

- Παραδείγματα προγραμμάτων σε Jasmin θα βρείτε στην ιστοσελίδα του compus στην ενότητα Έγγραφα. Εκεί θα βρείτε και την υλοποίηση μεθόδων για είσοδο έξοδο, οι οποίες είναι χρήσιμες για την υλοποίηση των παραπάνω ασκήσεων.
- Οι εντολές της JVM περιγράφονται εκτενώς στο Java Virtual Machine Specification. Το εν λόγω βιβλίο είναι διαθέσιμο δωρεάν στην ιστοσελίδα:
  http://docs.oracle.com/javase/specs/jvms/se14/html/index.html
  Υπάρχει κεφάλαιο που περιγράφει τις εντολές της JVM. Η πλειοψηφία των εντολών που απαιτούνται για την επίλυση των ασκήσεων έχουν παρουσιαστεί στο μάθημα.
- Ο συμβολομεταφραστής jasmin είναι διαθέσιμος από την ιστοσελίδα *http://jasmin.sourceforge.net/*. Σε ubuntu είναι **εγκαταστήσιμο package (jasmin-sable)**.
- Οι χαρακτήρες νέας γραμμής διαφέρουν από unix σε windows. Για να μην έχετε προβλήματα που θα σας ταλανίσουν πολύ, δώστε (και χρησιμοποιήστε το ακόλουθο ορισμό στο αρχείο flex που θα δημιουργήσετε

```
newline \n|\x0A|\x0D\x0A
```

- Στο σύστημα compus, θα βρείτε χρήσιμα αρχεία για την υλοποίηση των ασκήσεων. Μπορείτε να χρησιμοποιήσετε αρχεία που έχουν δοθεί σαν παραδείγματα μέσα στο μάθημα για να αναπτύξετε την λύση σας.