

Θεμα 3

A) i)

Όνομα Φοιτητή 1: **FOTIS TSIUMAS**

A.M. Φοιτητή 1: ΜΠΠΛ**21079**

Όνομα Φοιτητή 2: **KONSTANTINOS PETROU**

A.M. Φοιτητή 2: ΜΠΠΛ**21062**

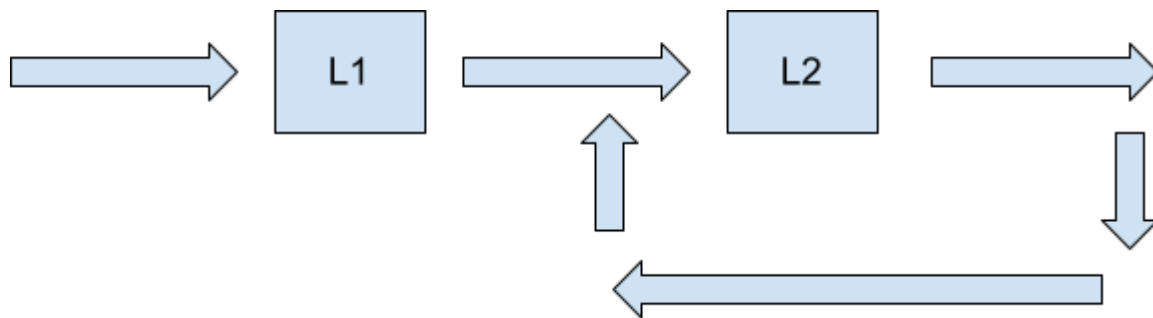
Alphabet: $\Sigma = \{\mathbf{A, E, F, I, K, M, O, P, R, S, T, U, 0, 1, 2, 6, 7, 9}\}$

A) ii)

Regular Expression: **L1L2***

όπου **L1** = {F, K, P, T} και **L2** = {A, E, I, M, O, P, R, S, T, U, 0, 1, 2, 6, 7, 9}

Συντακτικό Διάγραμμα:



B)

1. Λεκτική Ανάλυση

Κατα τη Λεκτικής Ανάλυσης διαβάζονται οι χαρακτήρες του source code και ομαδοποιούνται σε tokens/lexemes. Τα lexemes μπορεί να είναι δεσμευμένες λέξεις από τη γλώσσα (π.χ. if, for, while κτλ), operators (π.χ +, -, ==, =, ; κτλ), identifiers (π.χ. K, K2 κτλ) κ.α. Επίσης, υπάρχουν τα lexical values που έχουν εκχωρηθεί στους identifiers. Επομένως, ο λεκτικός αναλυτής παίρνει ολόκληρο τον κώδικα του θέματος 3 σαν είσοδο και η πρώτη δουλειά που κάνει είναι να τον χωρίσει στα παρακάτω lexemes (όχι με την παρακάτω σειρά):

int, if, iff, printf, "=", "==", ";", ">", "{", "}", "(", ")", "\"", K, K1, K2, K3, K4, K5, K6, K7, K8, K9, 2, 3, 4, 5, 6, 7, 8, A, A1, "K is equal to K1", "K is greater than K1"

Αφού εισαχθούν αυτά τα lexemes και αναλυθούν, δημιουργείται ένας πίνακας συμβόλων από τις παρακάτω ομάδες (όχι με την παρακάτω σειρά):

Ομάδα 1: int	Ομάδα 2: if	Ομάδα 3: iff	Ομάδα 4: printf
Ομάδα 5: ;	Ομάδα 6: =	Ομάδα 7: ==	Ομάδα 8: >
Ομάδα 9: {	Ομάδα 10: }	Ομάδα 11: (Ομάδα 12:)
Ομάδα 13: "	Ομάδα 14: id0	Ομάδα 15: id1	Ομάδα 16: id2
Ομάδα 17: id3	Ομάδα 18: id4	Ομάδα 19: id5	Ομάδα 20: id6
Ομάδα 21: id7	Ομάδα 22: id8	Ομάδα 23: id9	Ομάδα 24: id10
Ομάδα 25: id11	Ομάδα 26: 2	Ομάδα 27: 3	Ομάδα 28: 4
Ομάδα 29: 5	Ομάδα 30: 6	Ομάδα 31: 7	Ομάδα 32: 8
Ομάδα 33: K is equal to K1	Ομάδα 34: K is greater than K1		

Τέλος ο λεκτικός αναλυτής παράγει το παρακάτω κώδικα και τον περνάει στον συντακτικό αναλυτή:

```
int id0 = 2;
int id1 = 3;
int id2 = 4;
int id3 = 5;
int id4 = 4;
int id5 = 5;
int id6 = 4;
int id7 = 6;
int id8 = 7;
int id9 = 8;
if id0==id1 {
    printf("K is equal to K1");
}
iff id10 > id11 {
    printf("K is greater than K1");
}
id1 = id4;
id0 = id3;
id5 = id0;
id6 = id1;
id7 = id8;
id9 = 5;
```

2. Συντακτική Ανάλυση

Ο συντακτικός αναλυτής με την σειρά του παίρνει σαν είσοδο τον κώδικα που του δίνει ο λεκτικός αναλυτής και ξεκινάει να θέτει μια ιεραρχική δομή στην ακολουθία ομάδων. Σε αυτό το στάδιο θα βρει τα παρακάτω errors:

1. Στη γραμμή 4 του παραπάνω κώδικα, όπου μετά το `int` υπάρχει το `"id3"`, που αντιστοιχεί στο `"K3"` του `source code`, καθώς το 3 δεν υπάρχει στο αλφάβητό μας για τις μεταβλητές.
2. Στη γραμμή 5 του παραπάνω κώδικα, όπου μετά το `int` υπάρχει το `"id4"`, που αντιστοιχεί στο `"K4"` του `source code`, το 4 δεν υπάρχει στο αλφάβητό μας για τις μεταβλητές.
3. Στη γραμμή 6 του παραπάνω κώδικα, όπου μετά το `int` υπάρχει το `"id5"`, που αντιστοιχεί στο `"K5"` του `source code`, καθώς το 5 υπάρχει στο αλφάβητό μας για τις μεταβλητές.
4. Στη γραμμή 9 του παραπάνω κώδικα, όπου μετά το `int` υπάρχει το `"id8"`, που αντιστοιχεί στο `"K8"` του `source code`, καθώς το 8 υπάρχει στο αλφάβητό μας για τις μεταβλητές.
5. Στη γραμμή 11 του παραπάνω κώδικα, όπου μετά το `if` περιμένει παρένθεση `"("` και βρίσκει μεταβλητή.
6. Στη γραμμή 14 του παραπάνω κώδικα, όπου θα δει το `"iff"` και μετά το `"id10"`, που αντιστοιχεί στο `"A"` του `source code`, άρα στη θέση του `"iff"` θα περιμένει τύπο μεταβλητής καθώς το `"A"` θα το δει σαν μεταβλητή.
7. Στη γραμμή 17 του παραπάνω κώδικα, όπου θα δει το `"id4"`, που αντιστοιχεί στο `"K4"` του `source code`, καθώς δεν θα έχει αναγνωριστεί ως μεταβλητή παραπάνω και δεν θα της έχει εκχωρηθεί τιμή.
8. Στη γραμμή 18 του παραπάνω κώδικα, όπου θα δει το `"id3"`, που αντιστοιχεί στο `"K3"` του `source code`, καθώς δεν θα έχει αναγνωριστεί ως μεταβλητή παραπάνω και δεν θα της έχει εκχωρηθεί τιμή.
9. Στη γραμμή 19 του παραπάνω κώδικα, όπου θα δει το `"id5"`, που αντιστοιχεί στο `"K5"` του `source code`, καθώς δεν θα έχει αναγνωριστεί ως μεταβλητή παραπάνω και δεν θα της έχει εκχωρηθεί τιμή.
10. Στη γραμμή 21 του παραπάνω κώδικα, όπου θα δει το `"id8"`, που αντιστοιχεί στο `"K8"` του `source code`, καθώς δεν θα έχει αναγνωριστεί ως μεταβλητή παραπάνω και δεν θα της έχει εκχωρηθεί τιμή.

Λόγω των παραπάνω λαθών, το `source code` θα επιστραφεί με τα αντίστοιχα `error` ανα γραμμή στον `user`, με αποτέλεσμα να μην περάσει από την Συντακτική Ανάλυση και κατα συνέπεια να μην πάει στη Σημασιολογική Ανάλυση.

3. Σημασιολογική Ανάλυση

Η σημασιολογική ανάλυση έχει τον ρόλο του καθορισμού της έννοιας του πηγαίου κώδικα. Σε αυτό το στάδιο ελέγχονται θέματα, όπως οι εκχωρήσεις τιμών σε μεταβλητές ανάλογα με το είδος τους, η περιοχή ισχύος των μεταβλητών κτλ.

Σε αυτό το στάδιο δεν θα φτάσει καθόλου το παραπάνω πρόγραμμα, καθώς θα έχει κοπεί στο παραπάνω στάδιο.