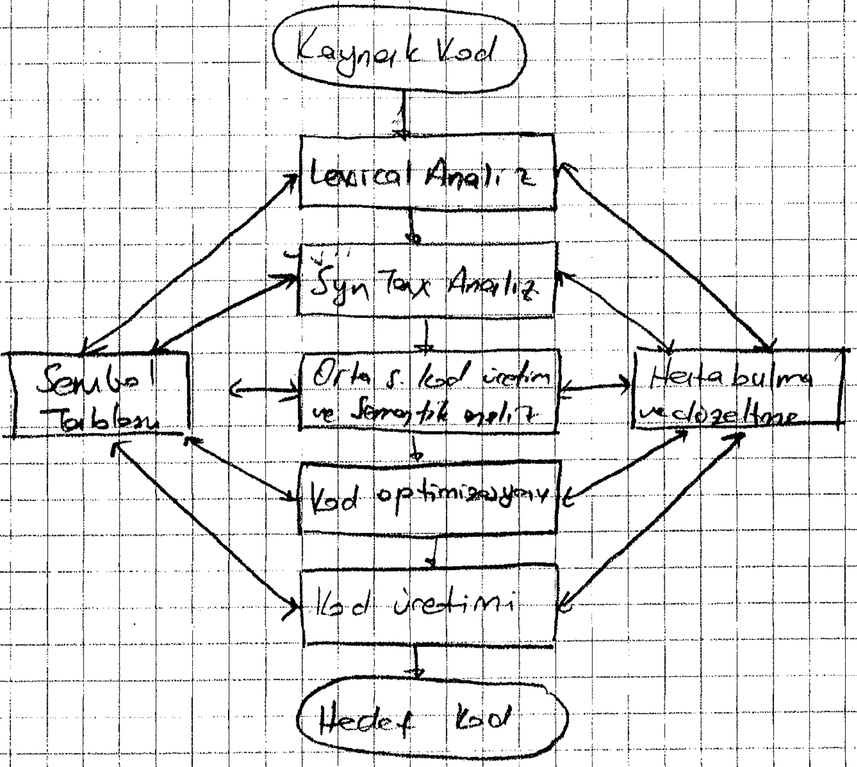
Derleyici Tasarımı

#1.Hafta

Derleyici Nedir?

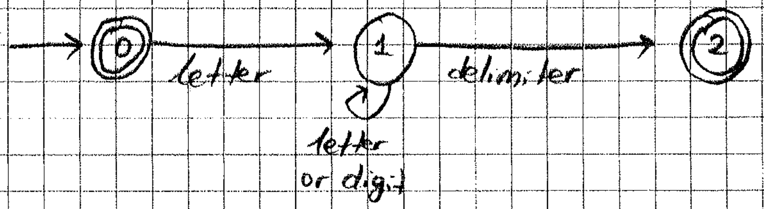
Yüksek seviyeli bir kodu makine koduna çeviren bir yazılımdır.

Derleme Aşaması



Lexical Analiz

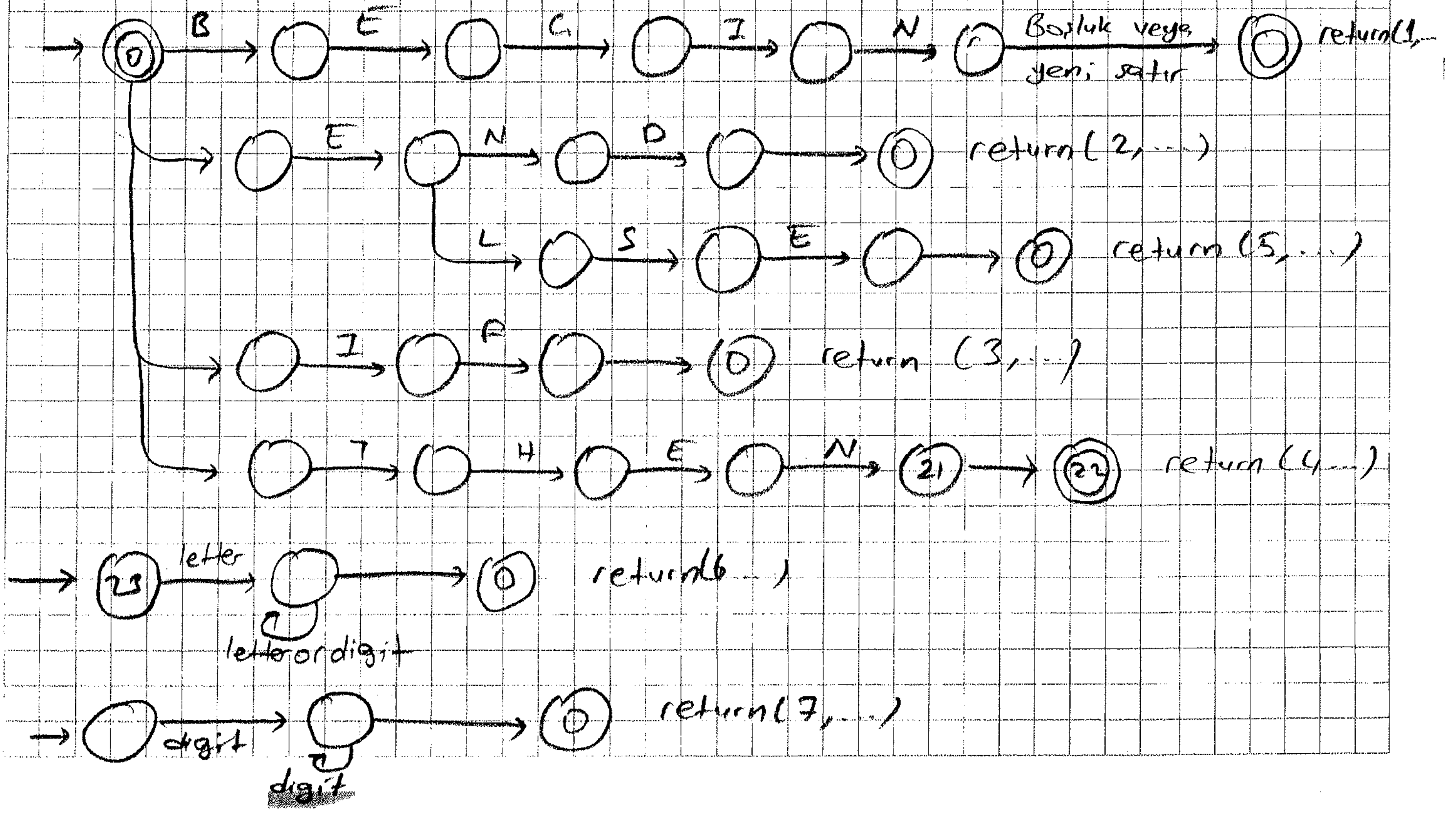
Kaynak kodu en küçük hücrelerine ayırır. Bu hücrelere token adı verilir. Lexical analiz kısmı genelde otomatalar kullanılarak oluşturulur. Örneğin bir değişken adı tanımlamak için;



otomatayı kullanabiliriz. Bunun yazılımsal kodu ise aşağıdaki gibidir.

```
state 0 : c= GETCHAR();
          if LETTER(c) then goto state 1
          else FAIL();
state 1 : c= GETCHAR();
          if LETTER(c) or DIGIT(c) then goto state 1
          else if DELIMITER(c) then goto state 2
          else FAIL();
state2  : RETRACT()
          return id, INSTALL()
```

Token	Code	Value
BEGIN	1	–
END	2	–
IF	3	–
THEN	4	–
ELSE	5	–
identifier	6	–
constant	7	–
<	8	1
<=	9	2
=	10	3
<>	11	4
>	12	5
>=	13	6



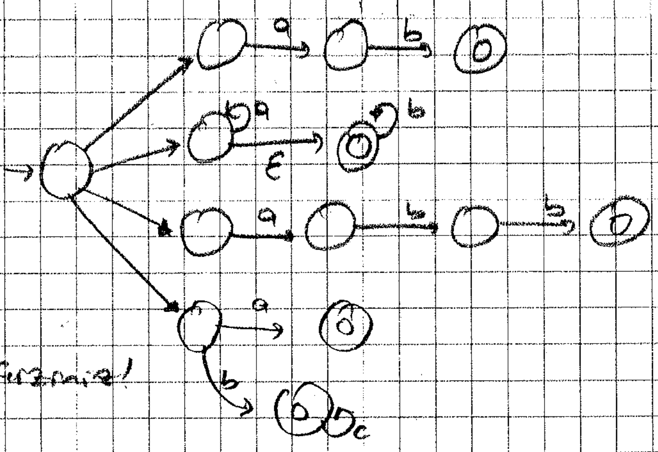
Örnek:

Token 1: $a.b$

Token 2: a^*b^*

Token 3: abb

Token 4: $a+b.c^*$



Tek bir otomata halinde çiziniz!

Bilgi

Lexical analiz kısmında alınacak hatalar Lexical hata olarak adlandırılır. Örneğin derleyici tasarımında yer almayan bir karakter tanımlanmışsa, bu bir hatadır. Başka bir hata ile karşılaşmak mümkün olmayabilir.

Semantik Analiz

$E \rightarrow E+E$ gibi bir ifade fonksiyon ile ifade edilir.

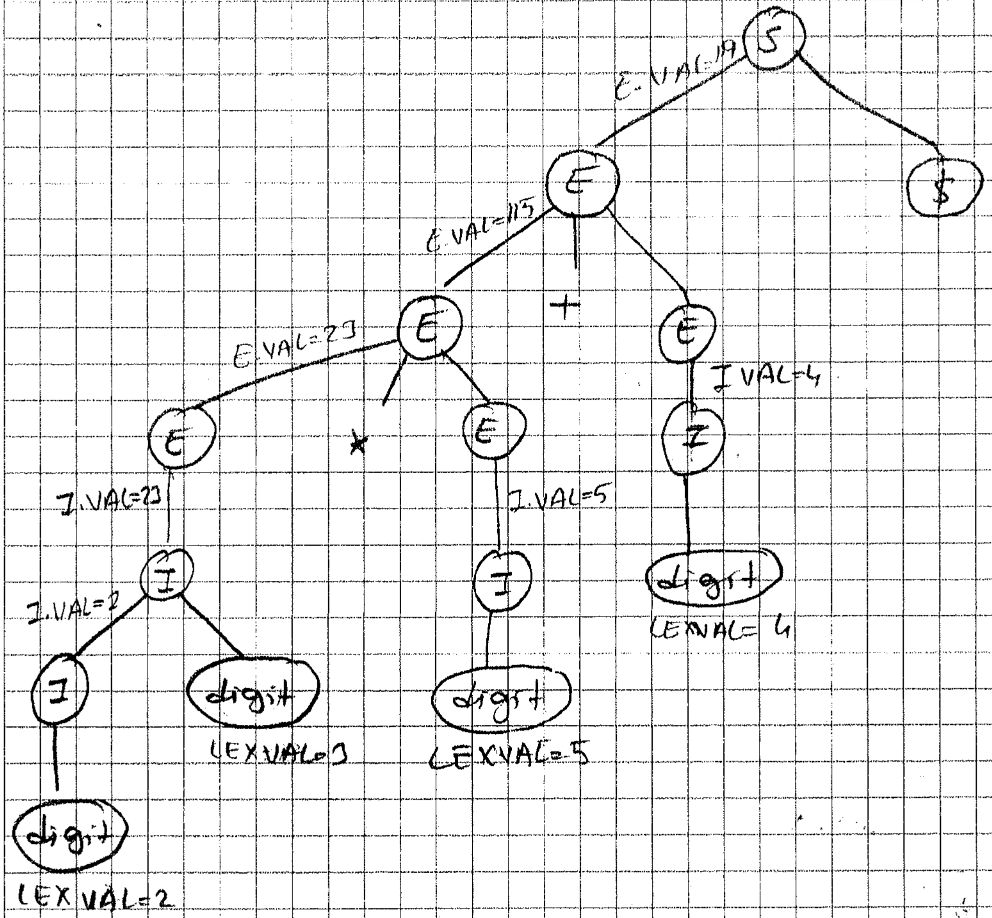
$\{ E.VAL = E.VAL^{(1)} + E.VAL^{(2)} \}$

$E \rightarrow digit \quad \{ E.VAL = digit \}$

Örnek: Bir hesap makinesi yapınız sadece toplama(+) ve çarpma(*) yapsın. Sonlandırmasını $ ile gösterin.

$S \rightarrow E\$$	print E.VAL
$E \rightarrow E+E$	$E.VAL = E.VAL^{(1)} + E.VAL^{(2)}$
$E \rightarrow E*E$	$E.VAL = E.VAL^{(1)} * E.VAL^{(2)}$
$E \rightarrow (E)$	$E.VAL = E^{(1)} VAL$
$E \rightarrow I$	$E.VAL = I.VAL$
$I \rightarrow I\ digit$	$I.VAL = 10 * I\ VAL + LEX\ VAL$
$I \rightarrow digit$	$I.VAL = LEXVAL$

Örnek: 23 * 5+4$



* Önceki yaptığımız örnek ile bunu ağaç şeklinde çizdik ve ağaçtaki gibi işlem yaptık

Orta Seviye Kod Üretimi

Kullanılan teknikler şunlardır:

* Postfix Notasyonu
* Syntax Tree
* Quadruplar
* Tripleks (3 Adres kodu)

Postfix Notasyonu

a+b → ab+

(a+b)*c → ab+c*

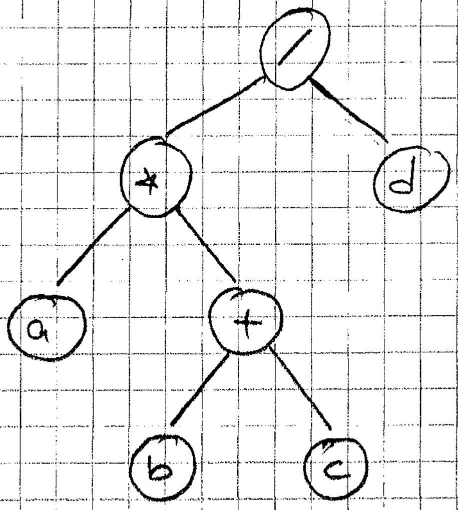
a*(b+c) → abc+*

(a+b)*(c+d) → ab+cd+*

* Bu notasyon önce değişkeni sonra opsiyonu yazar. post son anlamındadır. Yapılacak işlem sonda gösterilir.

Syntax Tree

a * (b+c) / d



* Burda yapılacak işlem kök düğümlerde değişkenler yaprakta tutulur.

Üç Adres Kodu

Karşılaşabileceğimiz deyimler şunlardır

```
1) A = B op C
2) A = op B
3) goto L
4) if A relop B goto L
5) param A
        ≡
      call P, n
6) A = B[I] , A[I] = B
7) A = addr B,  A = *B,  *A = B
```

NOT: 3 sayı aynı anda işleme tabi tutulamaz. 2'li olarak işlenir.

```
A = X + Y * Z

    T1 = Y * Z
    T2 = X + T1
    A = T2
```

Örnek: A = − B * (C + D)

```
T1 = − B
T2 = C + D
T3 = T1 + T2
A = T3
```

Örnek if A < B then 1 else 0

```
1  if A < B goto 4
2  T = 0
3  goto 5
4  T = 1
5  ...
```

örnek

```
for I = 1 step 1 until N
    do A[ I ] = 0
```

```
1   I = 1
2   if  I > N   goto 7
3   A[ I ] = 0
4   T1 = I+1
5   I = T1
6   goto 2
7   ...
```

~~Örnek~~:

LR Parsıng

Örnek: $E' \rightarrow E$

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid id$

I_0

$E' \rightarrow \bullet E$

$E \rightarrow \bullet E+T$

$E \rightarrow \bullet T$

$T \rightarrow \bullet T*F$

$T \rightarrow \bullet F$

$F \rightarrow \bullet (E)$

$F \rightarrow \bullet id$

$I_1 = Goto(I_0, E)$

$E' \rightarrow E\bullet$

$E \rightarrow E\bullet +T$

$I_2 = Goto(I_0, T)$

$E \rightarrow T\bullet$

$T \rightarrow T\bullet *F$

$I_3 = Goto(I_0, F)$

$T \rightarrow F\bullet$

$I_4 = Goto(I_0, ()$

$F \rightarrow (\bullet E)$

$E \rightarrow \bullet E+T$

$E \rightarrow \bullet T$

$T \rightarrow \bullet T*F$

$T \rightarrow \bullet F$

$F \rightarrow \bullet (E)$

$F \rightarrow \bullet id$

Devamı Arka Sayfada

$I_5 = Goto(I_0, id)$

$F \to id \bullet$

$I_6 = Goto(I_1, +)$

$E \to E + \bullet T$

$T \to \bullet T * F$

$T \to \bullet F$

$F \to \bullet (E)$

$F \to \bullet id$

$I_7 = Goto(I_1, *)$

$T \to T * \bullet F$

$F \to \bullet (E)$

$F \to \bullet id$

$I_8 = Goto(I_4, E)$

$F \to (E \bullet)$

$E \to E \bullet + T$

$I_9 = Goto(I_6, T)$

$E \to E + T \bullet$

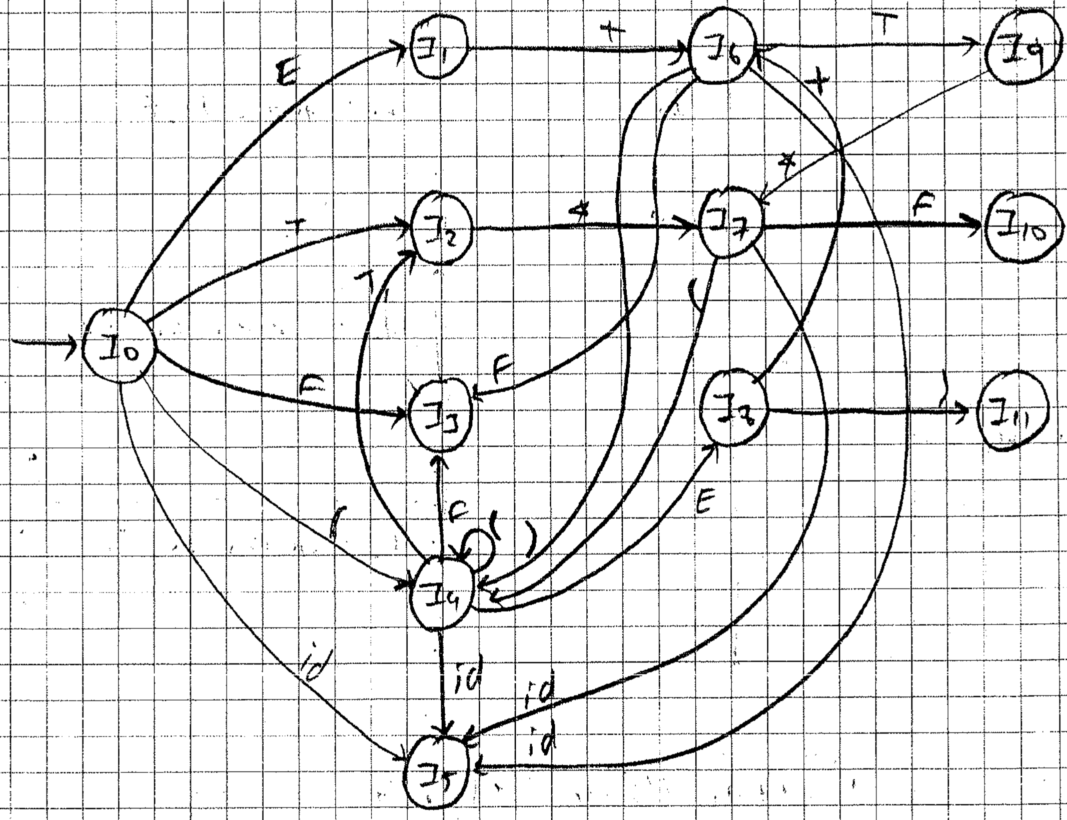
$T \to T \bullet + F$

$I_{10} = Goto(I_7, F)$

$T \to T * F \bullet$

$I_{11} = Goto(I_8,))$

$F \to (E) \bullet$



örnek: $S \to Aa$ Dilbilgisi için LR parsing yapınız!

$A \to aA \mid ba$

I_0

$S \to \bullet Aa$

$A \to \bullet aA$

$A \to \bullet ba$

$I_1 = Goto(I_0, A)$

$S \to A \bullet a$

$I_2 = Goto(I_0, a)$

$A \to a \bullet A$

$A \to \bullet aA$

$A \to \bullet ba$

$I_3 = Goto(I_0, b)$

$A \to b \bullet a$

$I_4 = Goto(I_1, a)$

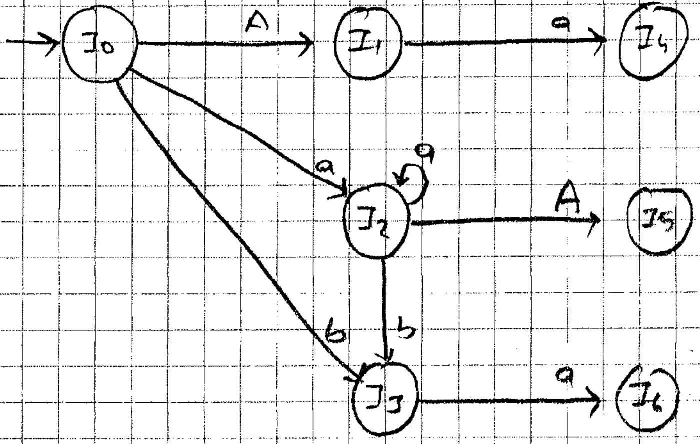
$S \to Aa \bullet$

$I_5 = Goto(I_2, A)$

$A \to aA \bullet$

$I_6 = Goto(I_3, a)$

$A \to ba \bullet$



Kontrol Akış Grafı

Örnek: while(x<y) kontrol akış grafını çıkarın!

```
while( x<y)
      if( z<t)
          then a = b+c*d
```

* Öncelikle 3 adres koduna çevirmeliyiz.

```
• 0    if x<y goto 2      B1
• 1    goto 8             B2
• 2    if z<t goto 4      B3
• 3    goto 8             B4
• 4    T1 = c*d           B5
  5    T2 = b+T1
  6    a = T2
  7    goto 0
• 8    ...                B6
```

Kontrol Akış grafı için şartlar

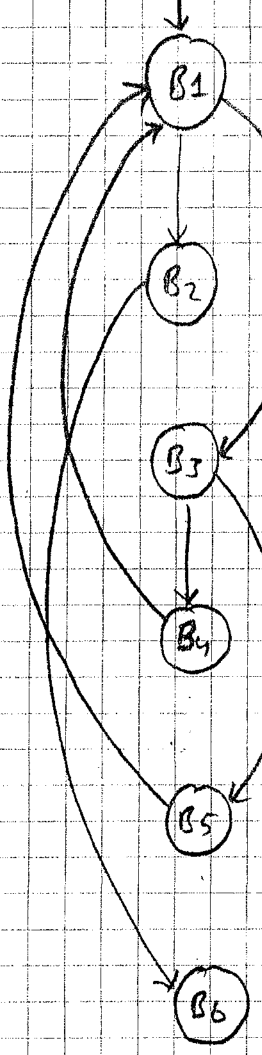
1) Kod temel bloklara ayrılmalıdır. Yani satırları gruplayacağız.

→ Bunun için Lider satırları bulmalıyız. Bulmak için;

- ilk satır Liderdir.
- Şartlı ya da şartsız bütün goto'ların hedefi liderdir.
- Şartlı goto'dan sonraki satır liderdir.

2) Bir Lider satırdan diğer Lider satıra olan kısım bir bloktur.

NOT Lider satırların soluna • işareti, bloklar □ şeklinde sağdaki Bx ise blok ismidir.



Sınavda sorulan soru

Örnekteki kodun 3 adres kodunu yazın ve kontrol akış grafını çizin!

Bilgi	Grafın çizilmesi

Örnek Sınav Soruları (2019 Vize)

1) $S \to A$ — Yandaki gramer için LR parsing çiziniz.

$A \to A+A \mid B{+}{+}$ — $w = y{+}{+} + y{+}{+}$ dizisini tanıyorsa gösteriniz!

$B \to y$

30P

2) Derleyici aşama grafiğini çiziniz! Birer cümleyle açıklayınız 25P

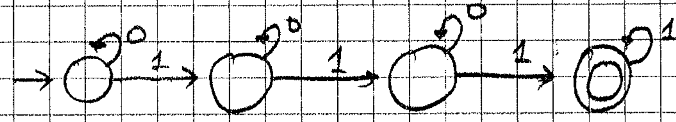
* Defterin en başında var

3)

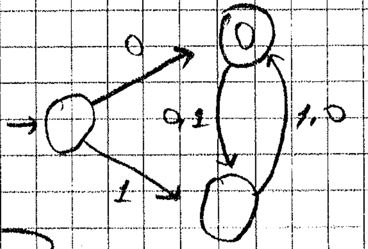
a) En az üç tane 1 içeren dizileri

b) 0 ile başlayan tek uzunluklu 1 ile başlayan çift uzunluklu dizileri tanıyan DFA'yı çizin.

a



b



4)

a) $\Sigma = \{a,b,c\}$ ise a ile biten diziler

b) $\Sigma = \{0,1,2,\dots,9\}$ ise çift sayılar

c) 5'in katı olan sayılar

için regüler ifade yazınız!

a) $(a+b+c)^* . a$

b) $(0+1+2+\dots+9)^* . (0+2+4+6+8)$

c) $(0+1+2+\dots+9)^* . (0+5)$

5) Bubble sort'un 3 adres kodunu yazın ve kontrol akış grafını çizin!

6) $S \to Aa$

$A \to aA \mid ba$

LR parsin eble ediniz. (Defterde var) (sayfa 11)

7) Kontrol akış grafını çiziniz!

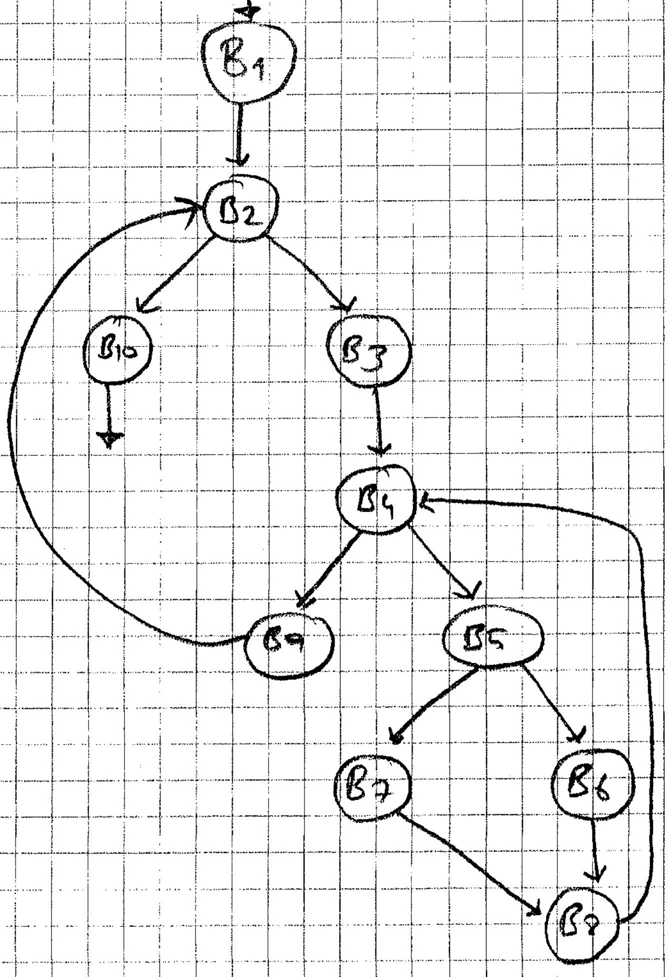
i = 1	B1
L1: if(...) goto L6	B2

t1 = 0
j = 0
	B3

L2: if(...) goto L5	B4

tmp = t5 + tr
if(...) goto L3
	B5

t1 = t5 + t1 goto L4	B6
L3: t1 = t1 + tR	B7
L4: i = j + 1 goto L2	B8
L5: i = i + 2 goto L1	B9
L6: t1 = 0	B10



8/a) ((a*(b+c)) * (d+(e+f))) + ((g+(h+i)) + (j*(k*1)))

Bunun için syntax tree çizin.

b)

```
int count = 0;
double sum = 0.0;
while (count < maxVect) {
        sum = sum + Vect[count];
        count ++;
}
```

Yandaki kodu bloklarına ayırın!

iki soru 30P