Karadeniz Teknik Üniversitesi
Bilgisayar Mühendisliği Bölümü
2019-2020 Güz Yarıyılı
Simgesel Matematik ve Programlama Final Sınavı
Tarih: 9 Ocak 2020 Perşembe **Süre:** 120 dakika

1. Bazı programlama dillerinde liste oluşturmak için (matematiğin *set comprehension* notasyonundan esinlenerek geliştirilen) "*list comprehension*" denilen bir sözdizim yapısı desteklenmiştir. Örneğin, bu yapının genel biçimi Haskell dilinde aşağıdaki gibidir.

```
[ expr | qualifier1, qualifier2, ... ]
```

Burada `expr` bir *expression*, `qualifier` ise bir *generator* yada *filter* olabilir; *generator* `var<-list` veya `var<-[a..b]` biçimindeki bir liste ifadesini, *filter* ise bir boolean ifadeyi temsil eder.

İçerisinde basit bir (tamsayıları, değişken isimlerini, parantezleri ve `+`, `-`, `*`, `/`, `%`, `^` işleçlerini içerebilen) `expr` ve bir *generator* bulunan Haskell *list comprehension* ifadelerini değerlendirmek için bir yorumlayıcı yazınız. Bu yorumlayıcının değerlendirmesi gereken bazı ifade örnekleri aşağıdaki tabloda gösterilmiştir (`..` notasyonunun aritmetik dizi ürettiğine dikkat ediniz).

girdi ifadesi	*değerlendirme sonucu*
`[x | x<-[3..8]]`	`[3,4,5,6,7,8]`
`[a^2 | a<-[0,1,2,3,4,5]]`	`[0,1,4,9,16,25]`
`[m%3 | m<-[1..6]]`	`[1,2,0,1,2,0]`
`[2*t+1 | t<-[5,6,7,8,9]]`	`[11,13,15,17,19]`
`[n*(n+1)/2 | n<-[1..8]]`	`[1,3,6,10,15,21,28,36]`

a) Bu biçimde ifade üretebilen bir LL(1) grameri geliştiriniz. (25p)
b) Gramere ait sözdizim sınıflarını nesneye dayalı bir programlama dilinde yazınız. (25p)
c) Girdi verisinin sözdizim analizini yapabilecek bir ayrıştırıcıyı (parser), sözdizim ağacı (*syntax tree*) oluşturan ifadeleri de ekleyerek JavaCC notasyonunda yazınız. (25p)
d) Değerlendirme işlemlerini yapabilen bir Visitor tasarlayınız ve kodlayınız. (25p)