دانشجویان محترم لطفاً به نکته‌های زیر توجه کرده و آن‌ها را رعایت کنید:

- تحویل پروژه فقط از طریق سامانه آموزش مجازی دانشگاه امکان پذیر است.
- انجام پروژه به صورت انفرادی می‌باشد، درصورت اثبات کپی برداری، نمره طرفین، 100- خواهد بود.
- تحویل پروژه بعد از مهلت مشخص شده نمره‌ای نخواهد داشت.
- حتما فایل ارسالی به شکل ZIP باشد و نام گذاری فایل به صورت StudentID-Firstname-Lastname-AntlrProject.zip باشد.
- در کنار گرامر خود حتما یک یا چند نمونه تست کیس برای زبانی که گرامر آن را نوشته اید قرار دهید تا قابلیت های مختلف گرامر را نمایش دهد.
- ZIP ارسالی باید شامل فایل گرامر (.g4) و تست کیس آماده شده (.txt) باشد.

شرح پروژه

در این پروژه قصد داریم ساختار یک زبان برنامه‌نویسی را تعریف و به وسیله Antlr طراحی کنیم و درخت کدهای نوشته شده به این زبان را رسم کنیم.

ساختار زبان:

۱. در ابتدای برنامه یک یا چند دستور Import می‌آید.

۲. هر کلاس در برنامه می‌تواند شامل توابع و تعریف متغیرها باشد. کلاس‌ها حتما باید یک تابع سازنده (Constructor) با نام همان کلاس داشته باشد.

۳. دستورات توسط line new از یکدیگر جدا می‌شوند به این معنی که هر خط معادل یه دستور است.

۴. بلاک‌ها باید توسط tab پیاده‌سازی شوند.

۵. قوانین نام‌گذاری:

- نام‌ها حداقل شامل دو کاراکتر هستند.

- با رقم یا کاراکترهای ’$’ و ’_’ شروع نمی‌شوند.
- نام‌ها نمی‌توانند کلمه کلیدی باشند. (کلمات با رنگ سبز)
- متشکل از حروف کوچک و بزرگ لاتین، ارقام و کاراکترهای ’$’ و ’_’ هستند.
- Dot یا نقطه نشان دهنده صدا زدن تابع یا دسترسی به یک متغیر از یک کلاس است و باید پیاده‌سازی شود.
- حرف اول نام کلاس‌ها باید capital باشد.

۶. کامنت گذاری به دو صورت single-line و multi-line توسط کاراکترهای دلخواه باید پیاده سازی شود و خطوط کامنت شده نباید در درخت کد ترسیم شود.

در ادامه ساختار دستوراتی که باید پیاده‌سازی شود به همراه مثال آورده شده است. قسمت‌های داخل [ ] ممکن است در کد نوشته شود یا نشود.

- **Import کردن:**

```
use lib-name[::mod-name]

use lib-name[::mod-name] as name
```

```
extern crate rand
use rand::Rng
use rand::Rng as RandomNumberGenerator
```

- **تعریف متغیر:**

```
var variable-name [<type>] ] <value>]

var <name-1>, <name-2>, …,<name-n>[<type>] =
<value-1>, <value-2>, …,<value-n>

<name> =: <value>

var-name-1, var-name-2, …,var-name-n =: value-1, value-2,…,value-n

var ( var-1-name var-type [=value-1]
var-2-name var-type [=value-2] … )
```

```
var student1 string = ”John”
var student2, student3 = ”Jane”, ”Alex”
x =: 2
var (a int
b string = ”hello”)
var(number int)
var(count int = 8)
```

- **تعریف آرایه:**

(در این بخش منظور از براکت ها ([ ]) بودن یا نبودن در کد نیست بلکه منظور خود براکت ها است که در مثال‌ها مشاهده می‌شود.)

```
var <name> = [<size>] <type> <values>

<name> =: [<size>] <type> <values>
```

```
var arr1 = [3] int {1, 2, 3}
arr2 =: [5] int {4, 5, 6, 7, 8}
var arr1 = [...] int {1, 2, 3}
arr2 =: [...] int {4, 5, 6, 7, 8}
```

نکته: سه نقطه (...) در قسمت سایز آرایه نشان دهنده مشخص نبودن طول آرایه است و برنامه باید آن را بپذیرد.

- **دستورات شرط:**

```
if <condition> :
    <code>

elif <condition> :
    <code>

else:
    <code>
```

• دستورات حلقه:

```
while <condition> :
        <code>

for <name> in <iterator-name>:
        <code>

for <name> in range (<value> or <variable-name>):
        <code>
```

```
while true:
        i++
for x in fruits:
        print(x)
for x in range (6):
        print(x)
for x in range :(grades)
        print(x)
```

• ساختار switch case:

```
match <variable-name> :
        case <value> :
                <code>
                [break]
        [default:
                <code>
                [break]
        ]
```

```
match argument:
        case :0
                return ”zero”
        case :1
                return ”one”
        case default:
                return ”something”
```

- **تعریف کلاس:**
(this یک کلمه کلیدی است.)

```
class <name> [ extends <name> ] :
        public/private <Constructor> ([<parameters>]):
                <code>
```

```
class Car extends Vehicle :
        private Car (int speed):
                this.speed = speed
```

- **تعریف توابع:**

```
public/private [static] <return-type> <name> ([parameters]):
        <code>
        [return <expression>]
```

```
public static void main(String[] args):
        int max = Math.max(x1, x2)
        System.out.println(myCar.modelYear + ” ” + myCar.modelName)
```

- **فراخوانی توابع:**

```
<function-name> ([parameters])
```

```
MyCar.showCars()
```

- **کار با فایل:**
(method می‌تواند r به معنای خواندن و یا w به معنای نوشتن باشد.)

```
<name> = open (”<name>.txt” , ”<method>”)
<name>.write(”<text>”)
<name>.close()
```

```
file = open(”prices.txt”, ”r”)
file.write(”Hello, World!”)
```

```
file.close()
```

• عملگرها و اولویت‌ها:

برنامه شما باید بتواند عملگرهای زیر را با اولویت بندی داده شده تشخیص دهد.

۱. ( )
۲. **
۳. + - (unary, e.g. -a)
۴. ### (unary, e.g. a++ or ++a)
۵. % // / *
۶. + - (binary, e.g. a-b)
۷. < < > >
۸. & |□ (bitwise operators)
۹. <> =! == ===
۱۰. =< => < >
۱۱. && || !
۱۲. = ##= (e.g. =// or =+)