سوال چهارم:

فایل های گرامر جاوا را که از قبل داشتیم در فولدر grammar قرار دادیم، سپس با کلیک راست روی فایل های گرامر و کلیک روی Generate ANTLR Recognization فایل های lexer, parser, listener, visitor را در فولدر gen تولید کردیم.

برای حل سوال باید فایل ورودی را بخوانیم و توکن های آن را پیمایش کنیم. در حین پیمایش درخت تجزیه شده ی فایل جاوا دو حالت داریم که چک میکنیم:

اگر توکن مورد بررسی از نوع Comment بود یعنی کامنت چند خطی بود، متن آن را بگیرد و /* و */ ابتدا و انتهای آن را حذف کند و قبل از /n ها // برای تبدیل به کامنت یک خطی بگذارد. از طرفی هر نوع کامنت دیگری بعد از آن باید حذف شود. کامنت های پشت سر هم میتوانند با /n از هم جدا شوند پس در یک حلقه چک میکنیم که اگر توکن بعدی از نوع کامنت یا /n بود متن آن را به فایل خروجی اضافه نکند و توکن بعد را بررسی کند تا زمانی که دیگر توکن بعدی ما کامنت نباشد.

اگر توکن مورد بررسی inline بود هم همان را در فایل خروجی بنویسد اما به همان روش بالا چک کند که خطوط بعدی کامنت نباشند اگر کامنت بودند آنها را نادیده بگیرد.

کد این تمرین در فایل main.py به صورت زیر نوشته شده است:

```python
from antlr4 import *
from gen.JavaLexer import JavaLexer
import os

def main(input_file, output_file):
    file_stream = FileStream(r""+input_file)
    if os.path.exists(output_file):
        os.remove(output_file)
    output_stream = open(r""+output_file, "a")
    lexer = JavaLexer(file_stream)
    token = lexer.nextToken()

    while token.type != Token.EOF:
        text = token.text
        prev = 0
        if token.type == lexer.COMMENT:
            text = "//" + text[2:-2]
            text = text.replace("\n    ", "\n")
            text = text.replace("\n", "\n//")
            text += "\n"
            prev = 0
            token = lexer.nextToken()
            t = token.text
            while token.type == lexer.LINE_COMMENT or token.type == lexer.COMMENT or token.text == "\n" or token.text == "\n    " or token.text == "\n        ":
                prev = 1
                token = lexer.nextToken()
        if token.type == lexer.LINE_COMMENT:
            text = text + "\n"
            text = text.replace("\n    ", "\n")
            prev = 0
            token = lexer.nextToken()
            t = token.text
            while token.type == lexer.LINE_COMMENT or token.type == lexer.COMMENT or token.text == "\n" or token.text == "\n    " or token.text == "\n        ":
                prev = 1
                token = lexer.nextToken()

        output_stream.write(text)
        if prev == 0:
            token = lexer.nextToken()
        prev = 0

    output_stream.close()

if __name__ == '__main__':
    input_file = "A.java"
    output_file = "B.java"

    main(input_file, output_file)
```

برای تست درست کار کردن آن در فایل A.java هر ۴ حالت برای کامنت ها را گذاشتیم و خروجی را در فایل B.java ذخیره کردیم.

فایل A.java:

```
import java.util.Random;
import java.util.ArrayList;
//first comm
//second comm
//third comm
/*sdfghjk
fcghjkl
fghjkl*/
public class A
{
    protected String name;
    protected int n1;
    protected int n2;
    /*sdfghjk
    fcghjkl345678ghjk
    cvgbhjkhgcfgyhuj
    fghjkl*/
    protected Piece(String name, int n1, int n2)
    {
        this.name = name;
        this.n1 = n1;
        this.n2 = n2;
        //first comm
        //second comm
        //third comm
    }
    /*34567fgvbnjhsdfghjk
    fcghjkl345678ghjk
    cvgbhjkhgcfgyhuj
    fghjkl*/
    //first comm
    //second comm
    //third comm
    public void M1()
    {
        this.n1 = (this.n2> this.n1) ? this.n1 : this.n2;
    }
    public void M2(int comp)
    {
        this.n2 = (comp > 0) ? this.n2 : this.n1;
    }
}
```

فایل B.java:

```
import java.util.Random;
import java.util.ArrayList;
//first comm
public class A
{
    protected String name;
    protected int n1;
    protected int n2;
```

```
    //sdfghjk
//fcghjkl345678ghjk
//cvgbhjkhgcfgyhuj
//fghjkl
protected Piece(String name, int n1, int n2)
    {
        this.name = name;
        this.n1 = n1;
        this.n2 = n2;
        //first comm
}
    //34567fgvbnjhsdfghjk
//fcghjkl345678ghjk
//cvgbhjkhgcfgyhuj
//fghjkl
public void M1()
    {
        this.n1 = (this.n2> this.n1) ? this.n1 : this.n2;
    }
    public void M2(int comp)
    {
        this.n2 = (comp > 0) ? this.n2 : this.n1;
    }
}
```