به نام خدا

تعریف پروژه عملی

ترم تحصیلی ۲-۱۴۰۲

درس: اصول طراحی کامپایلر

استاد: دکتر محمدهادی علائیان

مقدمه و طرح مسئله

مبهم‌سازی کد (Code Obfuscation) یک فرآیند در مهندسی نرم‌افزار است که با هدف پنهان سازی منطق برنامه انجام می‌شود، به گونه‌ای که خوانایی کد به شدت کاهش یافته ، استخراج هرگونه معنا و درک کردن ساختار و چگونگی رفتار کد برای انسان‌ها یا حتی برنامه‌های تجزیه‌کننده (Decompilers) تا حدود بسیاری دشوار یا غیرممکن شود. این کار معمولاً برای حفاظت از حقوق مالکیت فکری یا جلوگیری از تجزیه و تحلیل و تغییرات غیرمجاز در نرم‌افزار انجام می‌گیرد.

همانطور که برایتان قابل تصور است، ساخت این ابزار بصورت کامل امری بسیار سنگین است که در ابعاد یک پروژه ی دانشجویی نمیگنجد اما با محدود کردن فضای مسئله و در نظر نگرفتن حالات خاص برای شما نسخه ای سبک تر تعبیه شده تا در عین جذابیت و چالش بر انگیز بودن موضوع، پیاده سازی آن بیش از حد درگیر کننده نباشد.

```
class GFG {

    // Function to print N Fibonacci Number
    static void Fibonacci(int N)
    {
        int num1 = 0, num2 = 1;

        int counter = 0;

        // Iterate till counter is N
        while (counter < N) {

            // Print the number
            System.out.print(num1 + " ");

            // Swap
            int num3 = num2 + num1;
            num1 = num2;
            num2 = num3;
            counter = counter + 1;
        }
    }

    // Driver Code
    public static void main(String args[])
    {
        // Given Number N
        int N = 10;

        // Function Call
        Fibonacci(N);
    }
}
```

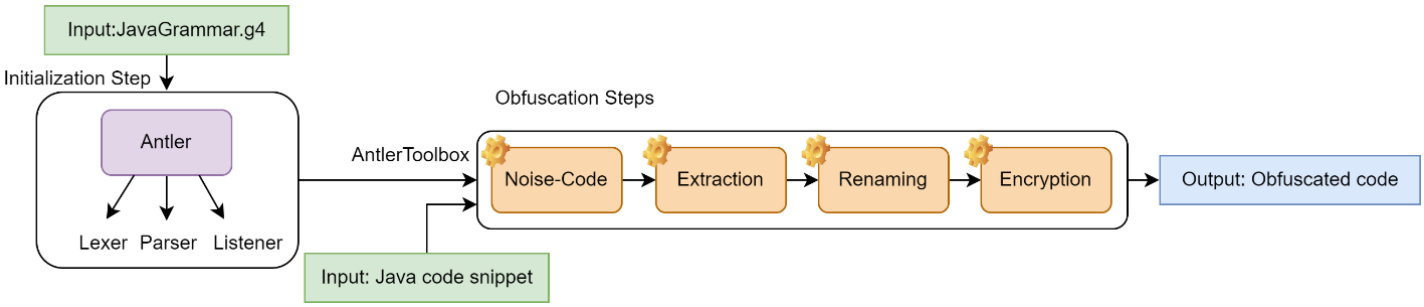
```
static void IuMfb8biNFO(int HJSrK5gW6Hl) {

    String ScOrwOED2Ll = MyCipher.getInstance()
            .encode(-Float.parseFloat(MyCipher.getInstance().decode
                ("152f1824b0710d1e74dd9eed234e87d36218ac9a")));
    String WWf12OOxTrn = "4e59e8cd306ab752b508dad476bad45e1c5307";
    String ATCklqmbaUD = MyCipher.getInstance()
            .encode(-Integer.parseInt(MyCipher.getInstance().decode("dd8d0eb46e")));
    String H3yhWhjj55O = "28", Fsa5TuZdnWq = "e5";

    String EVoFr8zRUVP = "f5839a4c236c0c6b49ea";
    String[] Vc53iQ4KkNs = new String[Integer.parseInt(MyCipher.getInstance().decode("b5"))];
    for (String EewoOjz5IwB = "28"; Integer.parseInt(MyCipher.getInstance().decode(EewoOjz5IwB)) < Integer
            .parseInt(MyCipher.getInstance().decode("b5")); EewoOjz5IwB = MyCipher.getInstance()
                    .encode(Integer.parseInt(MyCipher.getInstance().decode(EewoOjz5IwB))
                            + Integer.parseInt(MyCipher.getInstance().decode("e5")))) {
        Vc53iQ4KkNs[Integer.parseInt(MyCipher.getInstance().decode(EewoOjz5IwB))] = MyCipher.getInstance()
                .decode("ae82d7554677");
    }
    ArrayList<String> Tiyo4OBeZSR = new ArrayList<String>();
    Tiyo4OBeZSR.add(WWf12OOxTrn);
    Tiyo4OBeZSR.add(ATCklqmbaUD);
    Tiyo4OBeZSR.add(ScOrwOED2Ll);
    ZSRTiyo4OBe(MyCipher.getInstance().decode(EVoFr8zRUVP), Vc53iQ4KkNs);
    String SgB6m0gkwOi = "28";

    while (Integer.parseInt(MyCipher.getInstance().decode(SgB6m0gkwOi)) < HJSrK5gW6Hl) {

        System.out.print(Integer.parseInt(MyCipher.getInstance().decode(H3yhWhjj55O)) + " ");

        String KLVsbzW1Dik = MyCipher.getInstance()
                .encode(Integer.parseInt(MyCipher.getInstance().decode(Fsa5TuZdnWq))
                        + Integer.parseInt(MyCipher.getInstance().decode(H3yhWhjj55O)));
        H3yhWhjj55O = MyCipher.getInstance().encode(Integer.parseInt(MyCipher.getInstance().decode
            (Fsa5TuZdnWq)));
        Fsa5TuZdnWq = MyCipher.getInstance().encode(Integer.parseInt(MyCipher.getInstance().decode
            (KLVsbzW1Dik)));
        SgB6m0gkwOi = MyCipher.getInstance().encode(Integer.parseInt(MyCipher.getInstance().decode
            (SgB6m0gkwOi))
                + Integer.parseInt(MyCipher.getInstance().decode("e5")));
    }
}

public static float Z83PSHfLWib(float IQQMOq7PEHm, short GyA2xOEjr26) {

    String EuLlICGrrNR = MyCipher.getInstance()
            .encode(IQQMOq7PEHm * GyA2xOEjr26
                    + (float) Math.pow(IQQMOq7PEHm, Integer.parseInt(MyCipher.getInstance().decode("b5")))
                    - (float) Math.pow(GyA2xOEjr26, Integer.parseInt(MyCipher.getInstance().decode("b5"))));

    return Long.parseLong(MyCipher.getInstance().decode(EuLlICGrrNR));
}
```

نمونه ای از کد مبهم شده

مبهم سازی کد **زبان جاوا** در این صورت مسئله انتخاب شده که ابزار اصلی برای رسیدن به این هدف کتابخانه ی **انتلر ۴** و بستر پیاده سازی این پروژه **پایتون** است.

مبهم‌سازی کد می‌تواند شامل تکنیک‌ها و روش‌های متفاوتی باشد اما روش ما برای این پروژه از ۳ فاز اصلی و یک فاز امتیازی تشکیل شده است:

۱. استخراج (Extraction)	فاز اول
۲. تغییر نام (Renaming)	فاز دوم
۳. رمزنگاری (Encryption)	فاز سوم
٤. تولید Noise-code	فاز سوم (امتیازی)



روش کلی در نظر گرفته شده نیازمند چندین پیمایش متوالی درکد ورودی است که با در دست داشتن فایل گرامری زبان جاوا با فرمت G4، انتلر فایل های مورد نیاز (لکسر، پارسر و پارسرلیسنر) برای قدم های پایه و اساسی ما را فراهم میکند.

فاز اول – استخراج

بسیاری از تکنیک ها و روش هایی که در فاز های آینده استفاده خواهیم کرد نیازمند استخراج و کسب اطلاعات هستند. مسلما برای اقداماتی مانند تغییر نام و رمزنگاری، هر تغییر نابجا و کورکورانه که موجب سینتکس ارور در کد خروجی میشود، به هیچ عنوان قابل پذیرش نیست. به همین منظور ساخت یک دیتابیس غنی از شناسه ها و ویژگی های آنها دستمان را برای فاز های بعدی باز تر می گذارد و قدرت تصمیم گیری ای منطقی را به ما خواهد داد.

شناسه یا identifier از جزئی ترین گرامر رول های جاواست که تمام حالات در نظر گرفته شده در این صورت مسئله را شامل میشود و شما باید با استفاده از روش های گفته شده در جلسات حل تمرین، استفاده از توابع enterRule یا با استفاده از صفات هر توکن و همچنین با اشراف کامل به قواعد گرامری و ارتباطات آنها، اطلاعات خواسته شده که به شرح زیر هست را استخراج کنید:

۱. نام

۲. اسم فایل

۳. شماره خط

٤. تایپ:
شماره ی قانون این شناسه در جدول پارسر (رول ایندکس)

٥. مقدار:
اگر این شناسه دارای مقدار است مقدار آن و در غیر این صورت null ذخیره شود

٦. VarOrReturnType:
اگر شناسه ی ما از تایپ variableDeclaration بود، تایپ نرمال آن ذخیره شود (Integer, String, ...)
اگر شناسه ی ما از تایپ methodDeclaration بود، تایپ نرمال خروجی آن ذخیره شود
اگر شناسه ی ما از تایپ classDeclaration بود، نام خودش ذخیره شود

نکات تکمیلی

قابل ذکر است که در این صورت مسئله از شما خواسته شده تا صرفا اطلاعات تایپ های زیر از زیرمجموعه های قانون گرامری شناسه را استخراج نمایید:

- classDeclaration با ایندکس گرامری ۷
- enumDeclaration با ایندکس گرامری ۱۱
- methodDeclaration با ایندکس گرامری ۲۰
- variableDeclaratorID با ایندکس گرامری ۳۸
  - String
  - Double
  - Int
  - Float
  - Char
  - Long
  - Short
  - Boolean

- enumConstant با ایندکس گرامری ۱۳
- interfaceDeclaration با ایندکس گرامری ۱۵
- constantDeclarator با ایندکس گرامری ۳۱
- interfaceCommonBodyDeclaration با ایندکس گرامری ۳۵

مثالی ساده برای استخراج خواسته شده:

```java
public class TestClass {
    public static void main(String[] args) {

        String testStr = "Test";
        int testInt = 1;
    }
}
```

Name	Value	FileName	Line	type	VarOrReturnType
TestClass	-	TestClass.java	۳	ClassDeclaration	TestClass
main	-	TestClass.java	٤	MethodDeclaration	void
testStr	“test”	TestClass.java	٦	VariableDeclaration	String
testInt	۱	TestClass.java	۷	VariableDeclaration	int

ارزیابی

فاز اول پروژه

لطفا قبل از ارسال پروژه به نکات زیر توجه کنید:

۱. فایل ارسالی شما تنها یک فایل فشرده شده شامل کد و نتایج پروژه باشد. (.zip/.rar)

۲. نام فایل شما باید به صورت روبرو باشد:

YourFullName_YourStudentID

۳. توجه شود، ارائه به صورت مجازی برگزار خواهد شد، تمامی افراد گروه در روز ارائه باید حضور داشته باشند، در صورت عدم حضور هر یک از اعضا، نمره به آن شخص تعلق نخواهد گرفت.

🔔 مهلت ارسال در سامانه vc:

جمعه ــ ۲۱ اردیبهشت ــ ۱۴۰۳

موفق باشید 😊