UNIVERSIDAD SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA

Organización Lenguajes y Compiladores 1
Escuela de vacaciones
Diciembre de 2018
Ing. Mario Bautista
Tutor Académico
Luis Paz

PRÁCTICA 1: C# TRANSLATE

Contenido

PRÁCTICA 1: C# Translate 1
OBJETIVOS 2
Objetivo General 2
Objetivo Específico: 2
DESCRIPCIÓN DE LA PRÁCTICA 2
INTERFAZ DE C# Translate 3
Descripción de la interfaz: 3
Esquema de la Solución: 4
DESCRIPCIÓN DEL PSEUDO CÓDIGO 5
1. Clases: 5
2. Variables: 5
3. Constructor: 6
4. Métodos y Funciones 7
5. Sentencias If, If-else 8
6. Sentencia While , Do while 9
7. Sentencia For 9
8. Sentencia Switch: 10
9. Sentencias return,break: 12
10. Sentencia print: 12
11. Expresiones 12
CONSIDERACIONES 13
1. Herramientas a utilizar: 13
2. Restricciones: 13

3. Entregables: 13
4. Fecha de Entrega: 13

OBJETIVOS

Objetivo General

- Que el estudiante aprenda y comprenda el uso y manejo de la herramientas orientadas a generar analizadores léxicos y sintácticos.

Objetivo Específico:

- Que el estudiante genere archivos ejecutables de C# a partir de un lenguaje de alto nivel.
- Que el estudiante comprenda el funcionamiento básico de las dos primeras fases de análisis de un compilador.
- Que el estudiante resuelva problemas a través de herramientas generadoras de scanners y parsers.
- Que el estudiante puede obtener información y transformarla por medio de la fase de análisis léxico y sintáctico de un compilador.
- Que el estudiante consolide el conocimiento de la fase de análisis léxico y sintáctico y los diversos componentes que estas implican.

DESCRIPCIÓN DE LA PRÁCTICA

La empresa Xtrem Programing solicita sus servicios como desarrollador para poder crear un programa capaz de realizar la traducción de pseudo código en alto nivel al lenguaje C#, de forma que los archivos generados por la traducción sean funcionales y pueda hacerse uso de estos inmediatamente después de la traducción sin realizar ninguna alteración a los mismos.

Actualmente la empresa Xtreem Programing captura pseudo código por medio de una herramienta similar a “captcha” que se utiliza en diferentes validaciones de autenticidad, este pseudo código recolectado es agrupado y en diferentes archivos con extensión (**.psc**) los cuales por medio de un algoritmo generan la estructura de un clase en pseudo código, la empresa actualmente necesita que este pseudo código sea traducido al lenguaje C# y la tradición resultante sea totalmente funcional.

Los archivos en pseudo código, contienen la estructura de una clase, en la cual contienen declaraciones de variables, asignaciones de variables, Sentencias de control, ciclos, expresiones aritméticas, lógicas, relacionales, declaración de métodos y funciones. De forma que los archivos

resultantes de la traducción serán agregados a proyectos ya existentes en dicho lenguaje y se podrá instanciar estos (crear objetos) y hacer uso de sus variables métodos y funciones.

INTERFAZ DE C# TRANSLATE

Descripción de la interfaz:

La interfaz gráfica solicitada a los estudiantes se presenta a continuación:

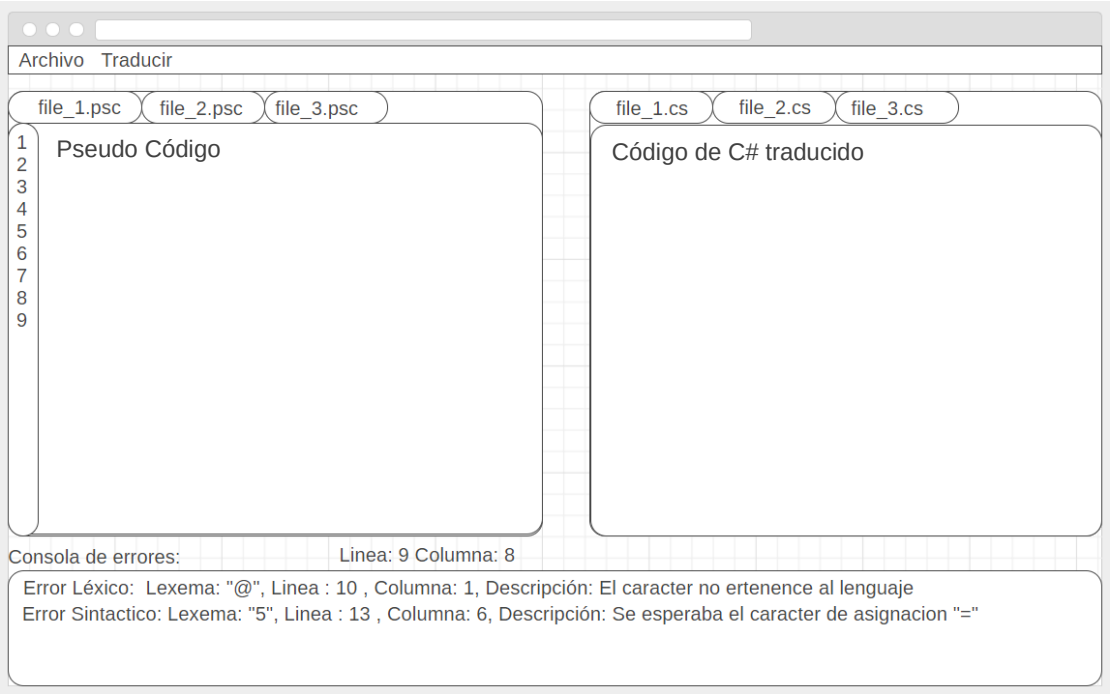

Imagen No. 1: Interfaz gráfica sugerida

Los botones (o menú) de la aplicación deberán contar con las siguientes funcionalidades:

1. **Archivo:** Creará una nueva pestaña en blanco en el editor.
   1.1 Abrir: Abrirá un cuadro de diálogo para seleccionar un archivo con extensión XML y mostrará su contenido en una nueva pestaña.
   1.2 Cerrar pestaña: Esta opción eliminará la pestaña de la interfaz gráfica.
   1.3 Exportar Archivo: Esta opción permitirá guardar los archivos (.cs) resultantes de la traducción en las diferentes directorios del ordenador.

2. **Traducir:** Analizara el contenido de la pestaña actual.

3. **Consola de errores:** En este panel se podrán encontrar los errores encontrados en la fase de análisis léxico y sintáctico, por lo que se debe visualizar el número de fila y

columna donde se encontró el error, el tipo de error, el lexema que lo causo y su descripción.

4. **Líneas y Columnas:** Se debe mostrar la posición del puntero así como el número de línea en el editor de texto.

Esquema de la Solución:

Imagen No. 2: Esquema de Funcionamiento.

1. Se tendrá un archivo de entrada con extensión (.psc) el cual será cargado a la aplicación C# translate.
2. El traductor procederá a compilar el archivo y a realizar el procesamiento del pseudo código.
3. Si existen errores deben ser mostrados en la consola de errores con las características correspondientes.
4. Si no existe ningún error se procede a generar el archivo (.cs) de forma correcta, y se desplegara en la aplicación como resultado de la traducción, el cual podrá ser exportado y posteriormente ejecutado en cualquier proyecto de C#.

DESCRIPCIÓN DEL PSEUDO CÓDIGO

1. Clases:

En pseudo código se tendran un archivo en el cual puede estar contenida más de una clase, de forma que la traducción de los nombres de las clases deben conservarse, el resto de la estructura de la clase puede cambiar. Ejemplo:

Código PSC:

```
Conteiner  Carro[

        //Sentencias en pseudo código
]
```

Código CS:

```
class Carro {

        //Sentencia en C# traducidas
}
```

2. Variables:

En pseudo código se podrán declarar variables de forma global, estas tendrán visibilidad y podrán ser declaradas con o sin asignación, variables locales las cuales no tendrán visibilidad y estas podrán ser declaradas con o sin asignación y también únicamente asignadas, tendrán la siguiente estructura:

Aclaración: Lo contenido dentro de "< >" indica que es obligatorio, lo contenido dentro de "¡ !" indica que es opcional (puede o no venir).

GLOBALES:

Código PSC:

```
¡visibilidad!  <tipo> <nombre variable > ¡= Expresión ! < ; >

Pub Str cadena = "Hola mundo" ;
Pri Str cadena1 ;
Pro Num var = 5*5;
Bool var1 = 5 Gt 7 ;
Pub Carro car =  inst Carro();
Dec var2 = 5.5 ;
Num entero ;
```

Código CS:

```
public String cadena = “Hola mundo” ;
private String cadena1 ;
protected int  var = 5*5;
Boolean var1 = 5 > 7 ;
public Carro car =  new Carro();
Double var2 = 5.5 ;
int entero;
```

LOCALES:

Código PSC:

```
<tipo> <nombre variable > ¡= Expresión ! < ; >
<nombre variable>  < = > < Expresión>  < ; >
<prop> < . > <nombre variable>  < => < Expresión>  < ; >

Str cadena = “Hola mundo” ;
cadena1 = “Hola mundo 2 solo asignado” ;
Num var = 5*5;
Bool var1 = 5 GT 7 ;
Carro car =  inst Carro();
Dec var2 = 5.5 ;
prop.entero = 55 + 10 ;
```

Código CS:

```
String cadena = “Hola mundo” ;
cadena1 = “Hola mundo 2 solo asignado”;
int  var = 5*5;
Boolean var1 = 5 > 7 ;
Carro car =  new Carro();
Double var2 = 5.5 ;
this.entero =  55 + 10;
```

3. Constructor:

El constructor es el “método” que se ejecuta cuando se instancia un objeto, en este se pueden ejecutar otras sentencias o únicamente inicializar las variables.

Código PSC:

```
Carro() :pub [
        //Sentencias en pseudo código
]

Carro( Num par1, Str cad, Dec  par3, Carro car, Bool var4) :Pub [
        //Sentencias en pseudo código
]
```

Código CS:

```
public Carro () {
        //Sentencia en C# traducidas
}

public Carro (int par1, String cad, Double par3, Carro car, Boolean var4) {
        //Sentencia en C# traducidas
}
```

4. Métodos y Funciones

El pseudo código, contendrá métodos y funciones los cuales tienen una visibilidad y también = pueden o no, tener parámetros, la forma de estos es la siguiente.

Código PSC:

```
<visbilidad> <nombre> ( ! Parametros ! ) : <tipo> [
        //sentencias en psudo código
]

Pub calcular() :vac [
        //Sentencias en pseudo código
]

Pro suma(num a, dec b):num [
        ret a + b ;
]

Pri Carro( num par1, str cad, dec  par3, Carro car, bool var4) :Carro [
        //Sentencias en pseudo código
        ret inst Carro(); //sentencia de retorno
]
```

Código CS:

```
public void calcular () {
        //Sentencia en C# traducidas
}

protect int suma(int a, Double b){
        return a+b;
}

public Carro (int par1, String cad, Double par3, Carro car, Boolean var4) {
        //Sentencia en C# traducidas
        return new Carro();
}
```

5. Sentencias If, If-else

Código PSC:

```
<  sif ( Condición ) [
        //sentencias en psudo código
] > ¡ sifnot [
        //sentencias en pseudo código
] !

sif ( 55 Gt 10 or a Eq b)[
        //sentencias en pseudo código
]

sif ( trus or fals  and trus or not fals)[
        //sentencias en psudo código
]sifnot[
        //sentencias en psudo código
]
```

Código CS:

```
if ( 55 > 10 or a == b){
        //sentencias en psudo código
}

if ( true || false  && true ||  ! false){
        //sentencias en psudo código
}else{
        //sentencias en psudo código
}
```

6. Sentencia While , Do while

Código PSC:

```
whs ( 55 Gt 10 or a Eq b)[
        //sentencias en psudo código
]

hc[
        //sentencias en psudo código
] whs ( 55 Lt 21) [
        //sentencias en psudo código
]
```

Código CS:

```
while ( 55 > 10 || a == b){
        //sentencias en psudo código
}

do{
        //sentencias en psudo código
} while ( 55 < 21) {
        //sentencias en psudo código
}
```

7. Sentencia For

Código PSC:

```
<sfr> ( <Num| Dec> <nombre var> = <exp> ; <Condición> ; < inc | de> )[
        //sentencias en psudo código
]

sfr(Num i =0 ; i Lt 10 ; inc) [
        //sentencias en psudo código
]

sfr(Dec i =20 ; i Gt 10 ; dec) [
        //sentencias en psudo código
]
```

Código CS:

```
for(int i =0 ; i < 10 ; i++) {
        //sentencias en psudo código traducidas
}

for(Double i =20 ; i > 10 ; i--) {
        //sentencias en psudo código traducidas
}
```

8. Sentencia Switch:

Código PSC:

```
<select> (<Exp>)[
        cas <Str| Num |Dec> :
                //sentencias en psudo código
        ¡
        cas < Str| Num |Dec> :
                //sentencias en pseudo código
        def :
                //sentencias en pseudo código
        !
]

select (“Hola”) [
        cas “hi” :
                //sentencias en psudo código
]

select( 5.5 * 20.6) [
        cas 2.5 :
                print(“hola”);
                brk;
        cas 10.5 :
                //sentencias en psudo código
        cas 25.5 :
                //sentencias en psudo código
        def :
                //sentencias en psudo código
]
```

Código CS:

```
switch(“Hola”) {
        case “hi” :
                //sentencias traducidas
}

switch( 5.5 * 20.6) [
        case 2.5 :
                Cosole.WriteLine(“hola”);
                break;
        case 10.5 :
                //sentencias traducidas
        case 25.5 :
                //sentencias traducidas
        default:
                //sentencias traducidas
]
```

9. Sentencias return,break:

PSC	CS
<ret> ¡ <exp> ! ;	return exp ; return;
<brk> ;	Break ;

10. Sentencia print:

PSC	CS
print (<exp>) ;	Console.WriteLine(<exp>);

11. Expresiones

Operadores aritméticos básicos: +, -, *, /.

Operadores de comparación:

PSC	CS
Lt,Lte,Gt,Gte,Eq,Eqs	<,<=,>,>=,==, !=

Operadores lógicos:

PSC	CS
and, or, not	`&&, ||, !`

CONSIDERACIONES

1. Herramientas a utilizar:

Para el desarrollo de esta práctica se utilizarán las siguientes herramientas.

- El lenguaje a utilizar es C#.
- El IDE a utilizar queda a discreción del estudiante.
- Para todo lo que concierne a la traducción y el análisis léxico y/o sintáctico se debe realizar con **irony**.

2. Restricciones:

Para el correcto desarrollo de esta práctica deben respetarse los siguientes lineamientos

- El análisis de los archivos .PSC es Case-Sensitive.
- Copias parciales o totales de prácticas o proyectos tendrán una nota de 0 puntos y los estudiantes serán reportados a escuela de ciencias y sistemas.
- NO habrá prórroga de ningún tipo.
- La calificación se realizará a partir de los entregables enviados por el estudiante.

3. Entregables:

Los entregables para esta práctica son:

- Aplicación funcional
- Código fuente(todo el proyecto)
- Gramáticas utilizadas en un archivo PDF que sea entendible.

4. Fecha de Entrega:

Entrega el 10 de diciembre de 2018 antes de las 11:59 PM