



# Proyecto 1

## C# to Python

### Objetivos

---

#### General:

1. El estudiante aprenda a generar un analizador léxico y sintáctico.

#### Específico:

1. El estudiante implemente tanto un analizador léxico como un sintáctico entendiendo la funcionalidad de ambos y cómo se relacionan.
2. El estudiante comprenda la funcionalidad de un traductor.
3. El estudiante debe capturar errores léxicos y sintácticos.
4. El estudiante debe generar la tabla de símbolos.

### Descripción

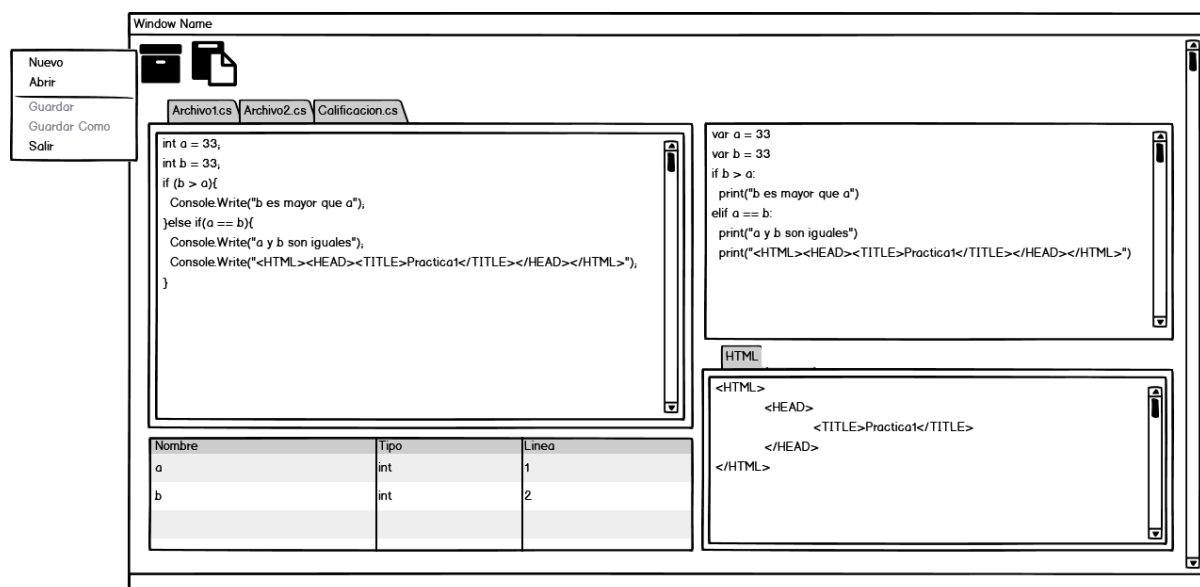
---

Debido a lo tedioso que puede llegar a ser el sobrescribir una aplicación de un lenguaje a otro se le solicita que, como estudiante de sistemas, realice un traductor de lenguaje de programación C# a Python, el cual contiene algunas de las sentencias básicas del lenguaje como lo son declaración y asignación de variables, sentencias de retorno, sentencias de condición y repetición, métodos, funciones entre otros; Del cual se espera que genere la salida del lenguaje python así como el resultado final de texto tipo HTML que contendrá el archivo a traducir.

# Área de Edición y Resultados

El área de edición y resultados será aquella en la cual se realizarán acciones como:

1. Archivo: En esta área se tendrá un menú el cual se encargará de crear nuevos documentos en blanco, abrir documentos existentes, guardar documentos o guardar como.
2. Reportes: Este sera el menu en el cual se podrán generar los reportes de salida en formato .py para la salida en python y en formato .html para la salida html de la página obtenida.
3. Lista de Ventanas: Para facilitar la traducción de varios archivos se podrán generar múltiples pestañas las cuales pueden generarse con nuevos documentos o al momento de abrir uno existente.
4. Consolas: Se tendrán 2 consolas de salida las cuales deben de contener la salida en código python y la salida del código html obtenido, respectivamente. **Ambas consolas deben de tener la restricción de no ser editables.**
5. Tabla de símbolos: En este se mostrarán todas las variables identificadas, de las cuales se desea saber: tipo,nombre,línea.



# Especificación del programa fuente

---

## Traductor de código C# a Python

Se desea una herramienta para migrar aplicaciones desarrolladas en lenguaje C# a lenguaje Python, sin necesidad de tener que reescribir el código.

Por lo cual se le ha asignado la tarea de crear una aplicación la cual reciba como entrada un archivo en lenguaje C# y devuelve como salida un archivo en lenguaje Python.

A continuación, se describe la sintaxis estándar y su conversión a código Python:

### Comentarios

El manejo de los comentarios dentro de los lenguajes pueden ser de 2 formas:

- **Comentario de una sola línea:**

<b>C#</b>	
<b>Python</b>	# Este es un comentario






- **Comentarios multilínea:**

<b>C#</b>	<pre>/* Este es    un comentario    multilínea */</pre>
<b>Python</b>	<pre>''' Este es     un comentario     multilínea '''</pre>

Los comentarios pueden venir en cualquier parte del archivo.

### Tipos de Dato

Dado que el lenguaje python es de tipado dinámico este no contiene una palabra reservada específica para cada tipo de dato por lo cual cualquier tipo de dato será sustituido por 'var'.

	C#		Python
	int		
	double		
	char		
	bool		
	string		

## Declaración y asignación de Variables

<b>C#</b>	TipoDato <b>Lista_ID</b> ; TipoDato <b>Lista_ID</b> = <b>EXPRESION</b> ;	
<b>Python</b>	id = <b>EXPRESION</b>	

**Lista\_ID:** Se podrán declarar n variables en una línea separados por coma si así se desea.

<b>Lista_ID</b>	id1, id2, id3, ..., idn	

Un id podría estar formado por letras, números y guión bajo.

## Expresión

Dentro de una expresión pueden venir operadores aritméticos, llamadas a funciones, llamadas a variables, cadenas, etc.

*Ejemplo:*

<b>C#</b>	var1 = 5+8*9;	
<b>Python</b>	var1 = 5+8*9	

## Declaración de métodos

- **C#**

void id ( <b>PARAMETROS</b> ){ <Sentencias> }
---

- **Python**

```
def id( PARAMETROS ):
    <Sentencias>
```

Pueden o no llevar parámetros.

## Declaración de funciones

- **C#**

```
<TIPO> id ( PARAMETROS ){
    <Sentencias>
}
```

- **Python**

```
def id( PARAMETROS ):
    <Sentencias>
```

Pueden o no llevar parámetros.

## Declaración del main

- **C#**

```
void main ( ){
    <Sentencias>
}
```

- **Python**

```
def main( ):
    <Sentencias>

if __name__ == "__main__":
    main()
```

No lleva parámetros.

## Parámetros

Listas de declaraciones para métodos y funciones.

<b>C#</b>	<pre>void mifuncion(int a, int b){     &lt;Sentencias&gt; }</pre>
<b>Python</b>	<pre>def mifuncion(a, b):     &lt;Sentencias&gt;</pre>

## Sentencias de Control

- **if**

<b>C#</b>	<pre>if (b &gt; a)     Console.WriteLine("b mayor que a"); if (a == b)     Console.WriteLine("a y b son iguales") }</pre>
<b>Python</b>	<pre>if b &gt; a:     print("b mayor que a") elif a == b:     print("a y b son iguales")</pre>

- **switch**

<b>C#</b>	<pre>switch(valor){     case 1:         precio = 55;         break;     case 2:         precio = 25;     case 3:         precio = 40;     default:         Console.WriteLine("No válido. Escoja 1, 2, o 3."); }</pre>
<b>Python</b>	<pre>def switch(case, precio):     switcher = {         1: precio = 55,         2: precio = 25,         3: precio = 40,         4: print("No válido. Escoja 1, 2, o 3."),     }</pre>

## Sentencias de Repetición

- for

<b>C#</b>	<pre>for (int a=0; a&lt;10; a++){     Console.Write("el valor de a es: " + a); }</pre>
<b>Python</b>	<pre>for a in range(1,10):     print("el valor de a es: ", a)</pre>

- while

<b>C#</b>	<pre>while(a &lt; 10){     Console.Write("el valor de a es: " + a); }</pre>
<b>Python</b>	<pre>while a &lt; 10 :     print("el valor de a es: ", a)</pre>

- do-while

<b>C#</b>	<pre>int a = 1; do {     Console.Write("el valor de a es: " + a); } while(a &lt; 5);</pre>
<b>Python</b>	<pre>i = 1 while True:     print("el valor de a es: ", a)     a = a + 1     if (a &lt; 5):         break</pre>

## Sentencia Imprimir

<b>C#</b>	<pre>Console.Write("el valor de a es: " + a);</pre>
<b>Python</b>	<pre>print("el valor de a es: ", a)</pre>

## Sentencia Return

- Para métodos

<b>C#</b>	return;
<b>Python</b>	return

- Para funciones

<b>C#</b>	return 5*5
<b>Python</b>	return 12+8

## Sentencia Break

La sentencia break solo se podrá utilizar dentro de las sentencias de repetición y dentro de una sentencia switch.

<b>C#</b>	break;
<b>Python</b>	break

## Sentencia Continue

La sentencia continue solo se podrá utilizar solo dentro de las sentencias de repetición.

<b>C#</b>	continue;
<b>Python</b>	continue



## Operadores para las Expresiones

- Aritméticos

Tipo Operacion	C#/Python
Suma	+
Resta	-
Multiplicación	*
División	/

- Lógicos

Tipo Operacion	C#	Python
And	&&	and
Or		or
Not	!	not

- Relacionales

Tipo Operacion	C#/Python
Mayor	>
Menor	<
Mayor igual	>=
Menor igual	<=
Igual	==
Distinto	!=

# Manejo y Recuperación de Errores

---

Para generar la traducción de código, la entrada debe estar libre de errores léxicos y sintácticos. La aplicación debe estar en la capacidad de **recuperarse de errores léxicos y sintácticos**, para poder corregir de manera eficiente los archivos.

Para los errores léxicos podrá utilizar cualquier método de recuperación, como en la práctica y proyecto realizado. **Para la recuperación de errores sintácticos se deberá utilizar el método de recuperación modo pánico.**

## Reporte de errores léxicos y sintácticos:

No.	Tipo error	Línea	Columna	Descripción
1	Léxico	12	6	El carácter '↵' no pertenece al lenguaje.
2	Sintáctico	45	10	Se esperaba... (se encontró id...)...
3	...	...	...	...

- El reporte debe ser en formato HTML y de existir errores deberá ser visualizado al momento de terminar el análisis.

# Entregables

---

1. Código fuente de la solución.
2. Manual Técnico y de Usuario.
3. Link del repositorio de versiones.

# Consideraciones

---

1. Trabajar de manera individual, utilizando como lenguaje de programación recomendado **JavaScript y la Herramienta Jison,**
2. El sistema operativo y el IDE son libres.
3. Cualquier copia total o parcial será reportada y no permitirá la continuidad en el laboratorio.
4. Los archivos de entrada serán proporcionados el día de la calificación.
5. No habrá prórroga y **entregas fuera de la fecha establecida tienen una nota de CERO.**
6. Fecha de entrega **viernes 16 de diciembre a las 23:59.**