

# Casos de Prueba



Jasson Moya Álvarez  
Crisia Piedra Chaves  
Compiladores e Intérpretes  
Profesor: Erika Marín Shumann

## Contenido

Contenido .....	1
Introducción .....	2
Casos de Prueba .....	2

# Introducción

Este documento presenta los casos de uso que se presentaron en la realización del parser para el proyecto de crear un compilador de similar funcionamiento al lenguaje de programación Python, este proyecto es parte fundamental para cumplir con las etapas respectivas de un compilador.

## Casos de Prueba

1. **Resultados esperados:** Leer correctamente la declaración de las variables, estas pueden ser de tipo o int, float, list, string, boolean y char

**Resultado obtenido:** EXITOSO.

Ya que se permite llevar a cabo la creación de diferentes variables con el tipo previamente definido. Al ejecutar un archivo con estas declaraciones de variable este no presenta ningún error.

```
run-single:
HAY 0 ERRORES
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 int x
2 float y
3 list z
4 string p
5 boolean q
6 char hola
7
```

Se puede además crear variables seguidas por coma por ejemplo:

```
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 3 seconds)
```

```
1 int g, y, x
2 float a, b, c
3 boolean g ,k
4 char z, t, y, y
5
```

Si se introduce algún tipo de variable que no se encuentre en la gramática, este mostrará los siguientes resultados

```
Syntax Error: x. Linea: 1. Columna: 6.
Syntax Error: z. Linea: 3. Columna: 8.
Syntax Error: q. Linea: 5. Columna: 11.
```

```
1 inti x
2 float y
3 listi z
4 string p
5 booleanp q
6 char hola
7
```

## 2. Resultados esperados: Leer expresiones.

**Resultados obtenidos:** EXITOSO

Las expresiones se crean perfectamente, si se encontrará un error este se mostrará en pantalla con su respectiva línea y columna.

```
Compiling 1 source file to C:\Users\Crisia\
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 int y = (a+6)
2 float z = t(8) // a % g
3 int a = 6 + (4 + (5 * 7) - 56) * (4 - 5) + (5)
4
```

## 3. Resultados esperados: Declaración de funciones y llamado de funciones.

**Resultados obtenidos:** EXITOSO.

Si se ejecuta el siguiente código estos darán error, ya que no cumplen con la gramática previamente establecida

```
1 def funcion3( x, y): x= g+6
2 ;
3 def funcion4(int, int): x= g+6
4 ;
5 def funcion1( int x : x= g+6
6 ;
7 def funcion4 : x= g+6
8 ;
9 x= funcion(x, 5)
10 g= funcion X,5) # error de parentesis
11 g= funcion(X,5 # error de parentesis
12 g= funcion(X 5) # falta coma
13
```

```
Syntax Error: x. Linea: 1. Columna: 15.
Syntax Error: ,. Linea: 3. Columna: 17.
Syntax Error: :. Linea: 5. Columna: 21.
Syntax Error: :. Linea: 7. Columna: 14.
Syntax Error: X. Linea: 10. Columna: 12.
Syntax Error:
. Linea: 11. Columna: 15.
Syntax Error: 5. Linea: 12. Columna: 14.
BUILD SUCCESSFUL (total time: 1 second)
```

Una declaración de una función y un llamado de la misma se ejecutan de manera correcta en el siguiente ejemplo:

```
Updating property file: C:\Users\Crisia\Doc
Compiling 1 source file to C:\Users\Crisia\
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 def funcion (): while (n>=0): print(n)
2     n = n-1
3     ;
4     a = input("Digite un numero")
5     cuentaAbajo(a)
6     x=funcion1(x,5)
7     ;
8
```

3. **Resultados esperados:** Definición de las estructuras de las funciones print e input

**Resultados obtenidos:** EXITOSO.

Se validaron dichas funciones según la gramática de python

En este ejemplo se muestran los inputs y prints de forma correcta los que no son correctos se documentaron para demostrar el ejemplo el ejemplo

```
Compiling 1 source file to C:\Users\Crisia\
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 def funcion (): while (n>=0): print(n)
2     #input
3     input()
4     input(X)
5     input(123)
6     input(x+x)
7     #input("digite un valor", x)
8     print(213+ 518- 345)
9     print(C)
10    print(C- 123)
11    #print()
12    #print
13    print("El valor es", x)
14    print("El resultado de la suma es ", suma(a,b,c))
15    print("hola")
16    ;
17    ;
18
19
```

Si los inputs y prints no cumplen con la gramática correcta este mostrará los siguientes resultados.

```
Compiling 1 source file to C:\Users\Crisia\
compile-single:
run-single:
Syntax Error:
. Linea: 2. Columna: 7.

Syntax Error: ,. Linea: 7. Columna: 28.

Syntax Error:
. Linea: 12. Columna: 11.

BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 def funcion (): while (n>=0): print(n)
2     input
3     input()
4     input(X)
5     input(123)
6     input(x+x)
7     input("digite un valor", x)
8     print(213+ 518- 345)
9     print(C)
10    print(C- 123)
11    print()
12    print
13    print("El valor es", x)
14    print("El resultado de la suma es ", suma(a,b,c))
15    print("hola")
16    ;
17    ;
18
```

#### 4. Resultados esperados: Inicialización con declaración de variables

**Resultados obtenidos:** EXITOSO.

La prueba es exitosa ya que el programa principal puede iniciar de diferentes formas y una de ellas es con la declaración de variables

```
Compiling 1 source file to C:\Users\Crisia\Documents\Python\
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
1  int x
2  int y
3  int z
4  float a
5  boolean c = True
6  while(x<5): print("Entra al ciclo while")
7  ;
8
```

De no cumplir con estas reglas de la gramática como en el siguiente ejemplo se mostraran los siguientes resultados.

```
Updating property file. C:\Users\Crisia\Documents\Python\
Compiling 1 source file to C:\Users\Crisia\Documents\Python\
compile-single:
run-single:
Syntax Error: int. Linea: 3. Columna: 1.
Syntax Error: def. Linea: 5. Columna: 1.
BUILD SUCCESSFUL (total time: 3 seconds)
1  int x
2  x = X + c
3  int y
4  w = a+ v + b
5  def funcion (int y): w = a+ v + b
6  ;
7
```

#### 5. Resultados esperados: Asignaciones de variables

**Resultados obtenidos:** EXITOSO.

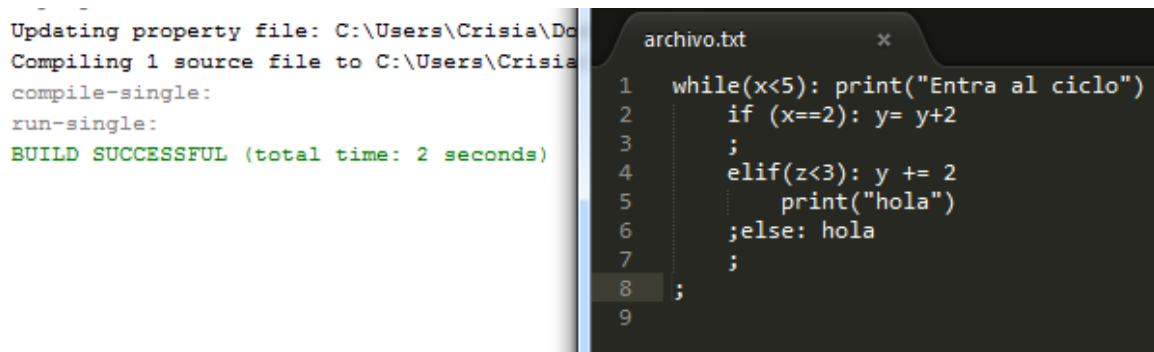
```
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
1  int g, y, x
2  float a, b, c
3  a = a * b
4  b = 2 % 3 & 345 +8*9+5-3
5
```

Si los resultados no son los correctos, se mostrará un mensaje de error, donde se muestre la línea y la columna de donde se encuentra el error.

## 6. Resultados esperados: Declaración del Ciclo while

**Resultados obtenidos:** EXITOSO.

La declaración del ciclo while se logró con gran éxito como el ejemplo que se muestra a continuación:



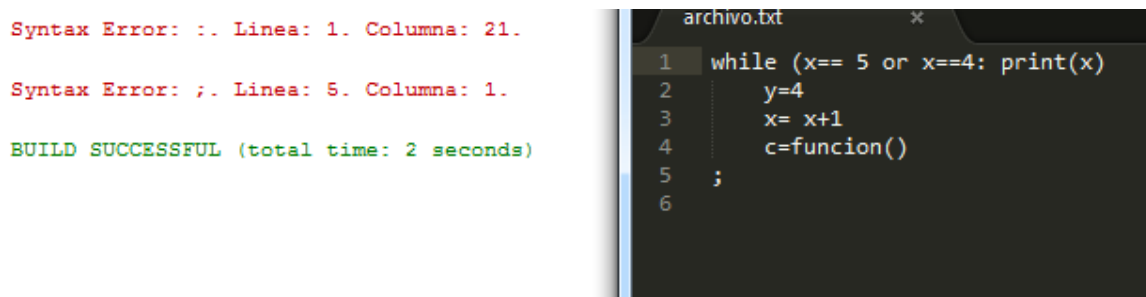
The image shows a terminal window on the left and a code editor on the right. The terminal displays the following output:

```
Updating property file: C:\Users\Crisia\Do
Compiling 1 source file to C:\Users\Crisia
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

The code editor shows a file named 'archivo.txt' with the following Python code:

```
1 while(x<5): print("Entra al ciclo")
2     if (x==2): y= y+2
3     ;
4     elif(z<3): y += 2
5         print("hola")
6     ;else: hola
7     ;
8 ;
9
```

De no contar con la gramática establecida este mostrará los siguientes resultados:



The image shows a terminal window on the left and a code editor on the right. The terminal displays the following output:

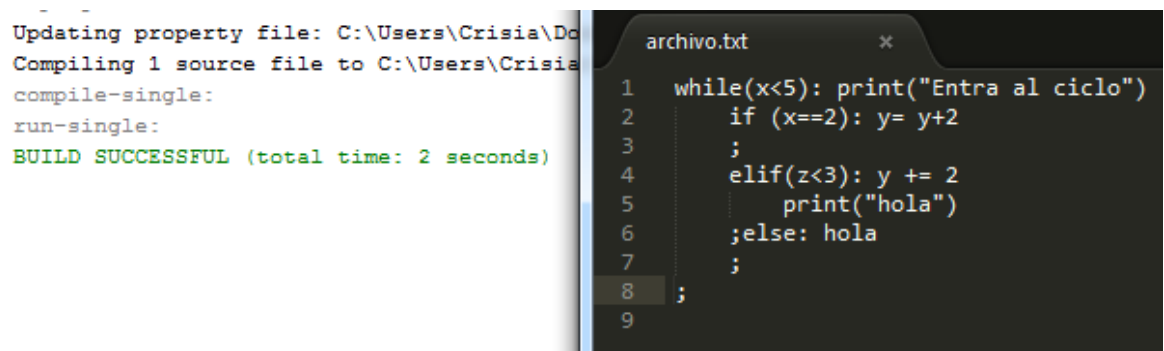
```
Syntax Error: :. Linea: 1. Columna: 21.
Syntax Error: ;. Linea: 5. Columna: 1.
BUILD SUCCESSFUL (total time: 2 seconds)
```

The code editor shows a file named 'archivo.txt' with the following Python code:

```
1 while (x== 5 or x==4: print(x)
2     y=4
3     x= x+1
4     c=funcion()
5 ;
6
```

## 7. Resultados esperados: Declaracion if, elif y else.

**Resultados obtenidos:** EXITOSO.



The image shows a terminal window on the left and a code editor on the right. The terminal displays the following output:

```
Updating property file: C:\Users\Crisia\Do
Compiling 1 source file to C:\Users\Crisia
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

The code editor shows a file named 'archivo.txt' with the following Python code:

```
1 while(x<5): print("Entra al ciclo")
2     if (x==2): y= y+2
3     ;
4     elif(z<3): y += 2
5         print("hola")
6     ;else: hola
7     ;
8 ;
9
```

## 8. Resultados esperados: Definición del ciclo for

**Resultados obtenidos:** EXITOSO.

```
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 def cuentaAbajo(int n): C=0
2     for a in range(2,7):print(a)
3     ;
4     a = input("Digite un numero")
5     cuentaAbajo(a)
6     ;
7
```

En caso de presentar errores en el for, se pueden presentar los siguientes resultados:

```
run-single:
Syntax Error: in. Línea: 1. Columna: 6.

Syntax Error: range. Línea: 4. Columna: 8.

Syntax Error:
. Línea: 9. Columna: 1.

Syntax Error: 7. Línea: 10. Columna: 18.

Syntax Error: :. Línea: 13. Columna: 20.

Syntax Error: 2. Línea: 16. Columna: 16.

BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 for in range(2,7): print(a)
2 ;
3
4 for a range(2,7): print(a)
5 ;
6
7 for a in range(2): print(a)
8 ;
9
10 for a in range(2 7): print(a)
11 ;
12
13 for a in range(2, 7: print(a)
14 ;
15
16 for a in range 2, 7): print(a)
17 ;
18
```



## 9. Resultados esperados: Try except y finally

**Resultados obtenidos: EXITOSO.**

```
run-single-  
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 def esNumero(int x): x=2  
2     try: print(x)  
3         ;except error: print(0)  
4     ;  
5  
6
```

El finally no es obligatorio en caso de que pueda venir se muestra de la siguiente manera

```
run-single:  
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 def esNumero(int x): x=2  
2     try: print(x)  
3         ;except error: print(0)  
4         ;finally: c.Clase()  
5             x= input("Ingrese valor")  
6     ;  
7  
8
```

Si se presenta algún error en el try, except o finally este puede mostrar la siguiente información

```
compile-single-  
run-single:  
Syntax Error: except. Linea: 3. Columna: 2.  
  
Syntax Error:  
. Linea: 6. Columna: 3.  
  
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 def esNumero(int x): x=2  
2     try: print(x)  
3     except : print(0)  
4     ;finally: c.Clase()  
5         x= input("Ingrese valor")  
6     ;  
7  
8
```

Como se mostró en el ejemplo anterior se puede observar como hace falta el uso del ; en el try y la excepción en el except.

## 10. Resultados esperados: Declaración de clases

### Resultados obtenidos: EXITOSO.

Para llevar a cabo la declaración de las clases, solo debe de existir una única clase en el archivo de no ser así esta daría error

```
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 class clasePrueba : int x
2     float y
3     print("hola")
4     if (True): x=1
5     ;
6
7 ;
8
```

```
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 class clasePrueba : int x
2     float y
3     x= funcion1()
4     print("hola")
5     if (True): x=1
6     ;
7
8 ;
9
```

Un posible error puede ser el siguiente:

```
Compile single.
run-single:
Syntax Error: :. Linea: 1. Columna: 8.
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 class : int x
2     float y
3     print("hola")
4     if (True): x=1
5     ;
6
7 ;
8
```

## 11. Resultados esperados: Llamado a una clase

**Resultados obtenidos:** EXITOSO.

```
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1  class clasePrueba : int x
2      float y
3      x= funcion1()
4      print("hola")
5      if (True): x=1
6      c.Classe()
7      ;
8
9  ;
10
```

Al obtener algún error este mostrará el siguiente resultado:

```
run-single:
Syntax Error:
. Linea: 6. Columna: 10.

BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1  class clasePrueba : int x
2      float y
3      x= funcion1()
4      print("hola")
5      if (True): x=1
6      c.Classe
7      ;
8
9  ;
10
```

## 12. Resultados esperados: Llamado a una función

**Resultados obtenidos:** EXITOSO.

```
Compiling 1 source file to C:\Users\Crisia\I
compile-single:
run-single:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
archivo.txt
1  class clasePrueba : int x
2      float y
3      x= funcion1()
4      print("hola")
5      if (True): x=1
6      clase.funcion() #Llamada de una funcion
7      ;
8
9  ;
10
```

Si el llamado a una función no es el correcto este mostrará los siguientes resultados

```
compile-single:
run-single:
Syntax Error:
. Linea: 6. Columna: 17.

BUILD SUCCESSFUL (total time: 2 seconds)
```

```
1 class clasePrueba : int x
2     float y
3     x= funcion1()
4     print("hola")
5     if (True): x=1
6     clase.funcion(x #error en llamado a una funcion
7     ;
8
9 ;
10
```