

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS



Lenguajes de Programación

Práctica 6 - Punto Extra

Johann Ramón Gordillo Guzmán

418046090

Punto extra para la sexta práctica, presentado como parte del curso de **Lenguajes de Programación** impartido por la profesora **M.I. Karla Ramírez Pulido**.

29 de Mayo del 2020

Link al código fuente: <https://github.com/JohannGordillo/>

1. Preguntas

1. Encuentra dos lenguajes que hagan diferente manejo de tipos entre sí en las condicionales `if` y `cond` (o similares), donde expliques si el lenguaje permite diferentes tipos en las múltiples expresiones que se pueden evaluar, dependiendo de si se cumple o no la condicional. Adicionalmente muestra tres instrucciones que ejemplifiquen la explicación anterior.

Respuesta.

■ Python.

En Python tenemos el condicional *if*, acompañado de *else* y *elif*.

En este lenguaje, la condicional debe evaluarse a un valor booleano (True o False), y las expresiones asociadas al condicional pueden evaluarse a valores de distintos tipos (enteros, cadenas, objetos, etc...).

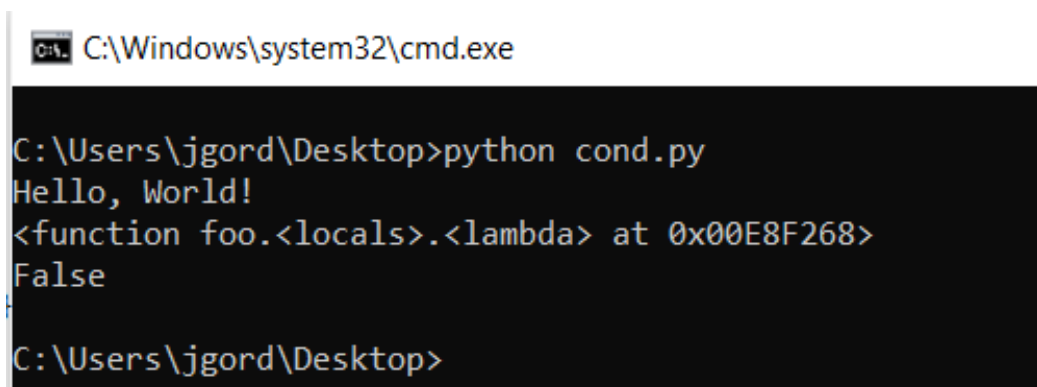
Python es un lenguaje de propósito general, dinámicamente tipificado, con tipado fuerte, y además es multiparadigma.

Como Python es dinámicamente tipificado, la verificación de tipos se lleva a cabo en tiempo de ejecución.

Un ejemplo de condicional en Python es:

```
1 def foo(bar):
2     """No docstring today."""
3     if bar < 1729:
4         return "Hello, World!"
5     elif bar == 1729:
6         return (lambda x: x)
7     else:
8         return False
9
10 for x in [1, 1729, 1000000000000000]:
11     print(foo(x))
```

Al ejecutarlo:



```
C:\Windows\system32\cmd.exe

C:\Users\jgord\Desktop>python cond.py
Hello, World!
<function foo.<locals>.<lambda> at 0x00E8F268>
False
C:\Users\jgord\Desktop>
```

- Haskell.

En Haskell tenemos el condicional *if then else*.

En este lenguaje, la condicional debe evaluarse a un valor booleano (True o False), y las expresiones asociadas al condicional pueden evaluarse a valores del mismo tipo.

Haskell es un lenguaje de propósito general, estáticamente tipificado, con tipado fuerte, y de paradigma funcional.

Como Haskell es estáticamente tipificado, la verificación de tipos se lleva a cabo en tiempo de compilación.

Si intentamos ejecutar el siguiente código:

```
1 foo = if True then 5 else "six"
```

Obtenemos un error:

```
prueba.hs:1:20: error:
    * No instance for (Num [Char]) arising from the literal `5'
    * In the expression: 5
      In the expression: if True then 5 else "six"
      In an equation for `foo': foo = if True then 5 else "six"
1 | foo = if True then 5 else "six"
  |                      ^
Failed, no modules loaded.
Prelude> _
```

Pero si intentamos ejecutar el siguiente código:

```
1 foo = if True then 5 else 6
```

No obtendremos errores:

```
Prelude> :l prueba.hs
[1 of 1] Compiling Main                ( prueba.hs, interpreted )
Ok, one module loaded.
*Main>
```

...

2. Bibliografía

- Ramírez, K. (2020).
Notas del curso de Lenguajes de Programación.
Facultad de Ciencias - UNAM
Ciudad de México, México.
- Krishnamurthi, S. (2017).
Programming Languages: Application and Interpretation.
Estados Unidos de América.