

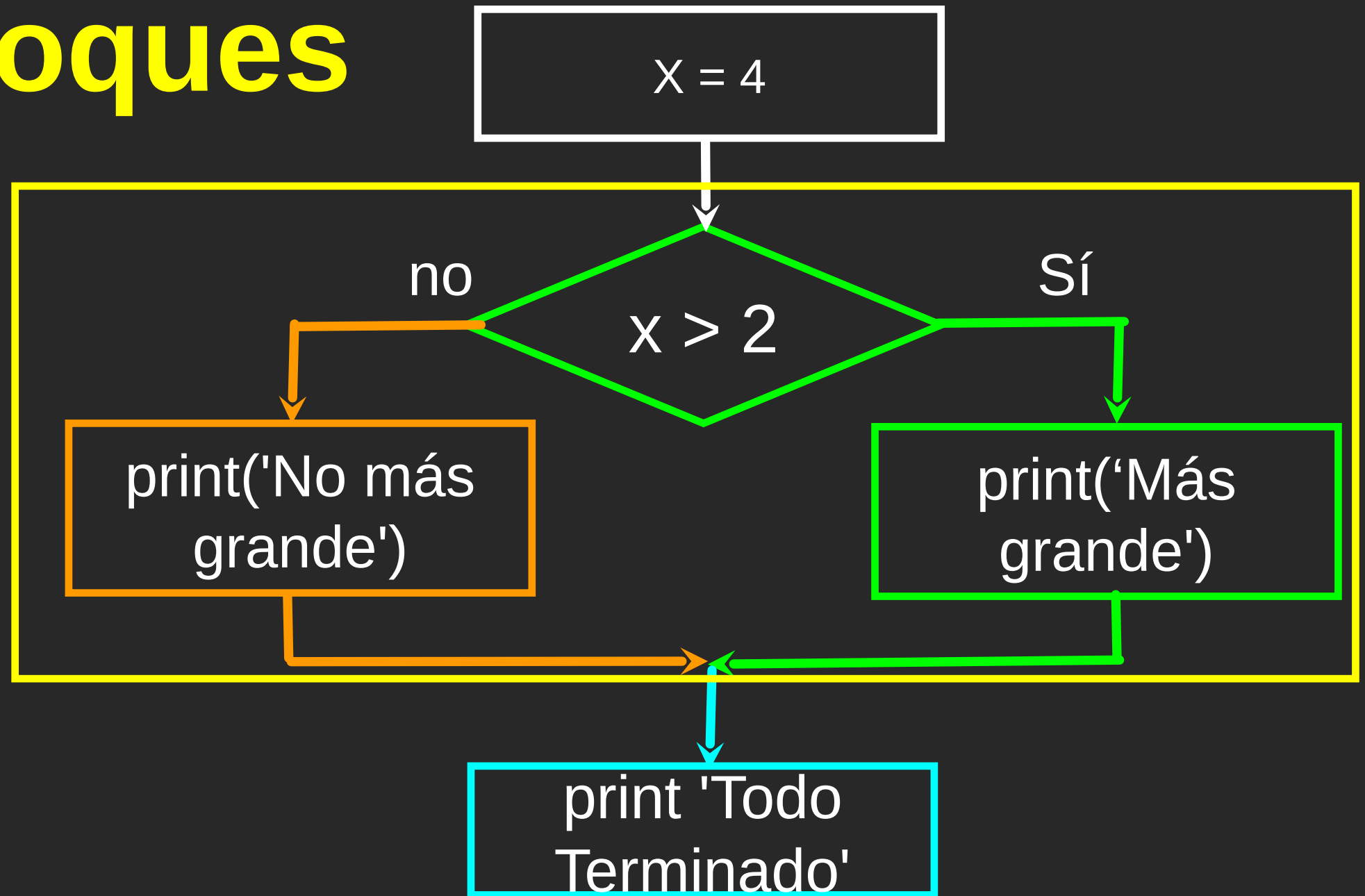


Más Patrones de Ejecución Condicional



Visualizar Bloques

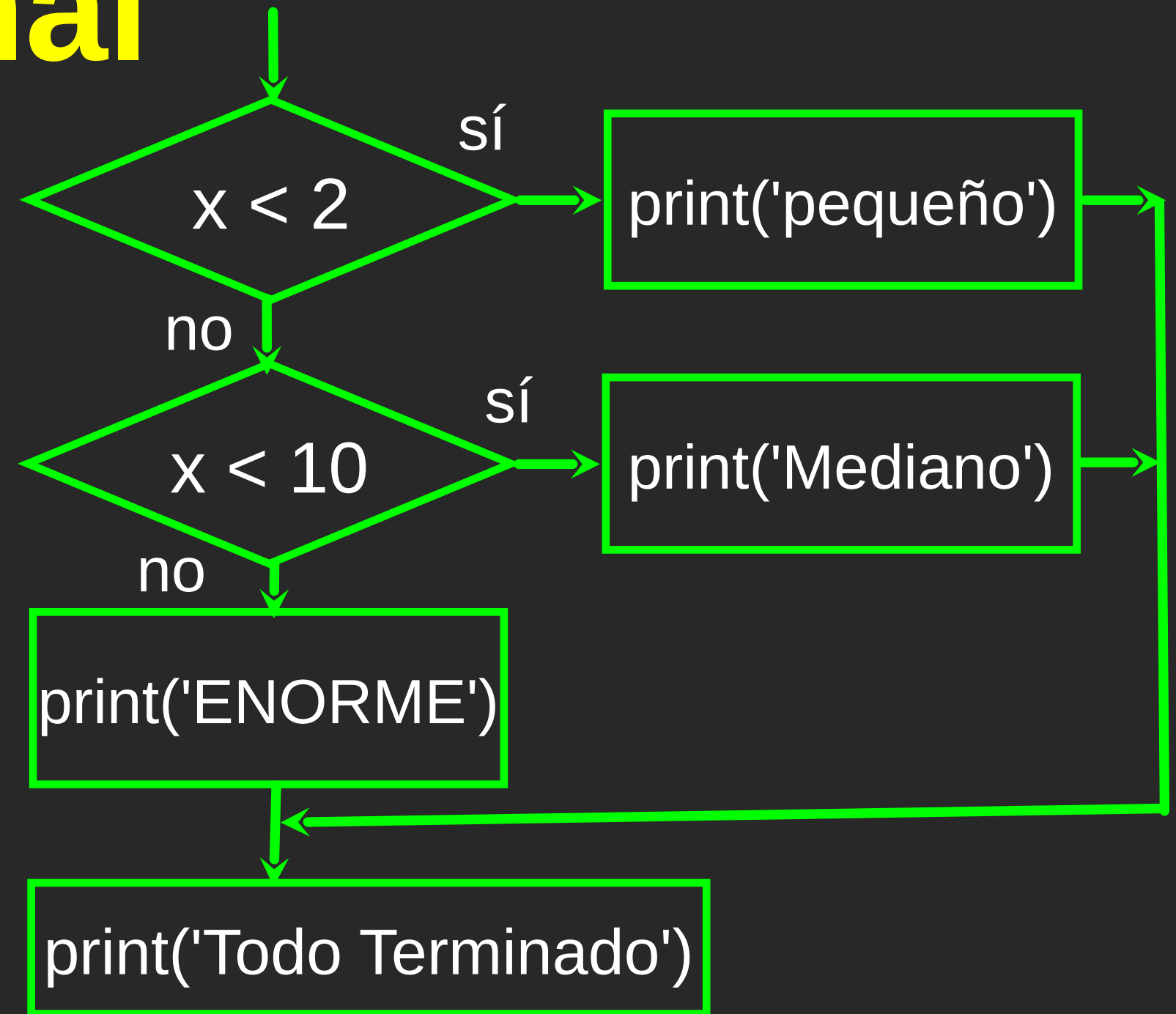
```
x = 4  
  
if x > 2 :  
    print('Más grande')  
else :  
    print('Más pequeño')  
  
print 'Todo Terminado'
```





Multidireccional

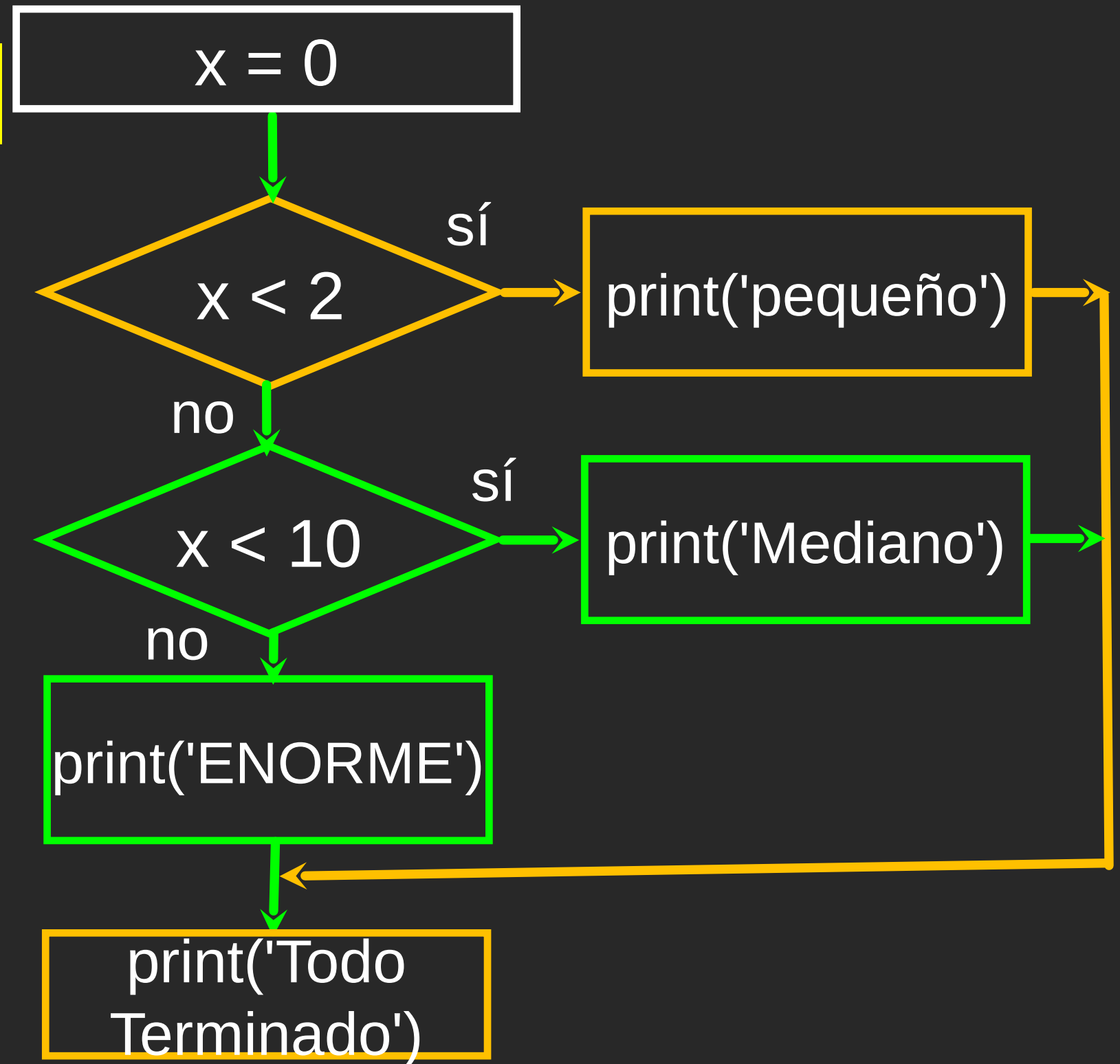
```
if x < 2 :  
    print('Pequeño')  
elif x < 10 :  
    print('Mediano')  
else :  
    print('ENORME')  
print('Todo terminado')
```





Multidireccional

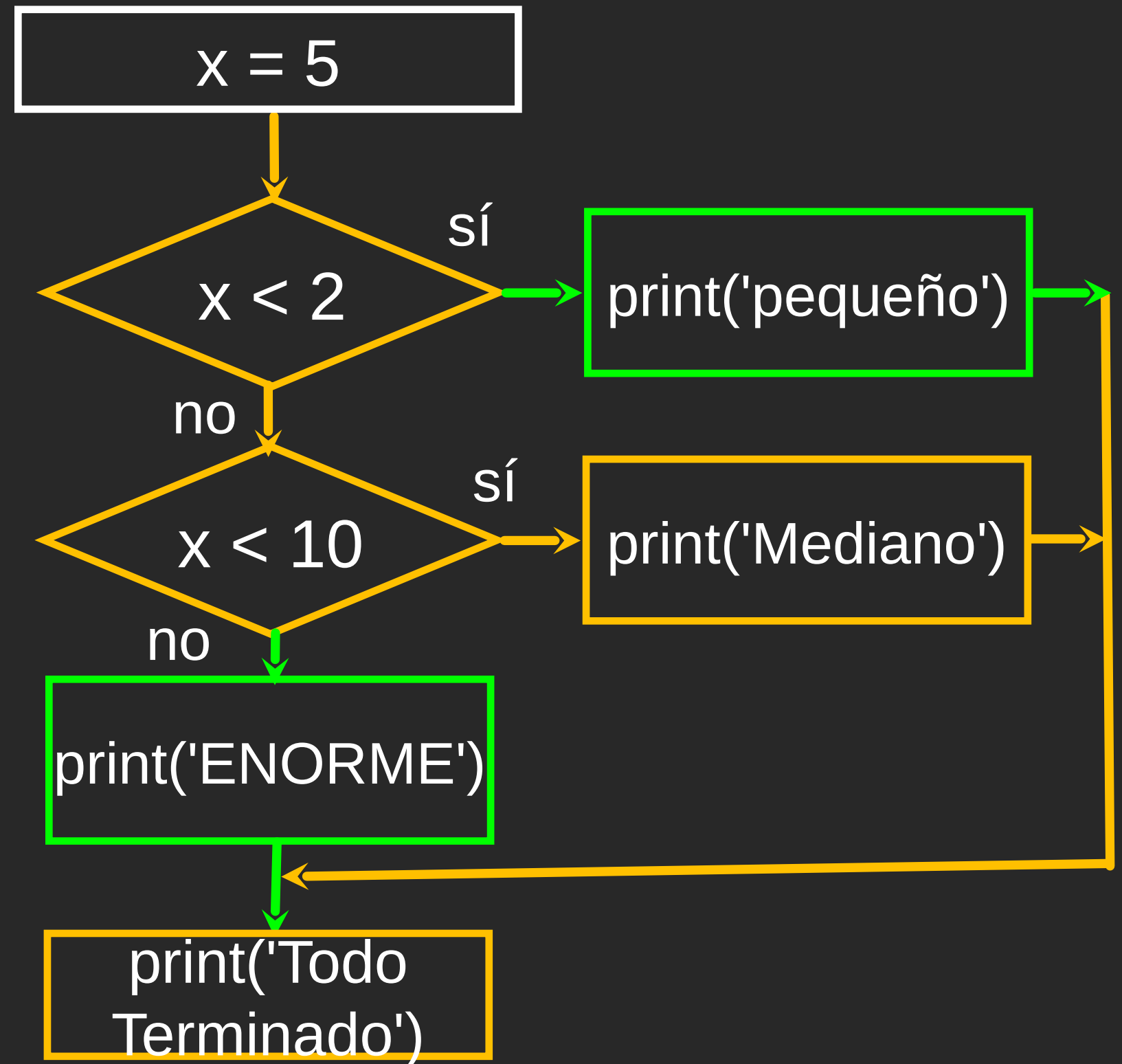
```
x = 0
if x < 2 :
    print('pequeño')
elif x < 10 :
    print('Mediano')
else :
    print('ENORME')
print('Todo
terminado')
```





Multidireccional

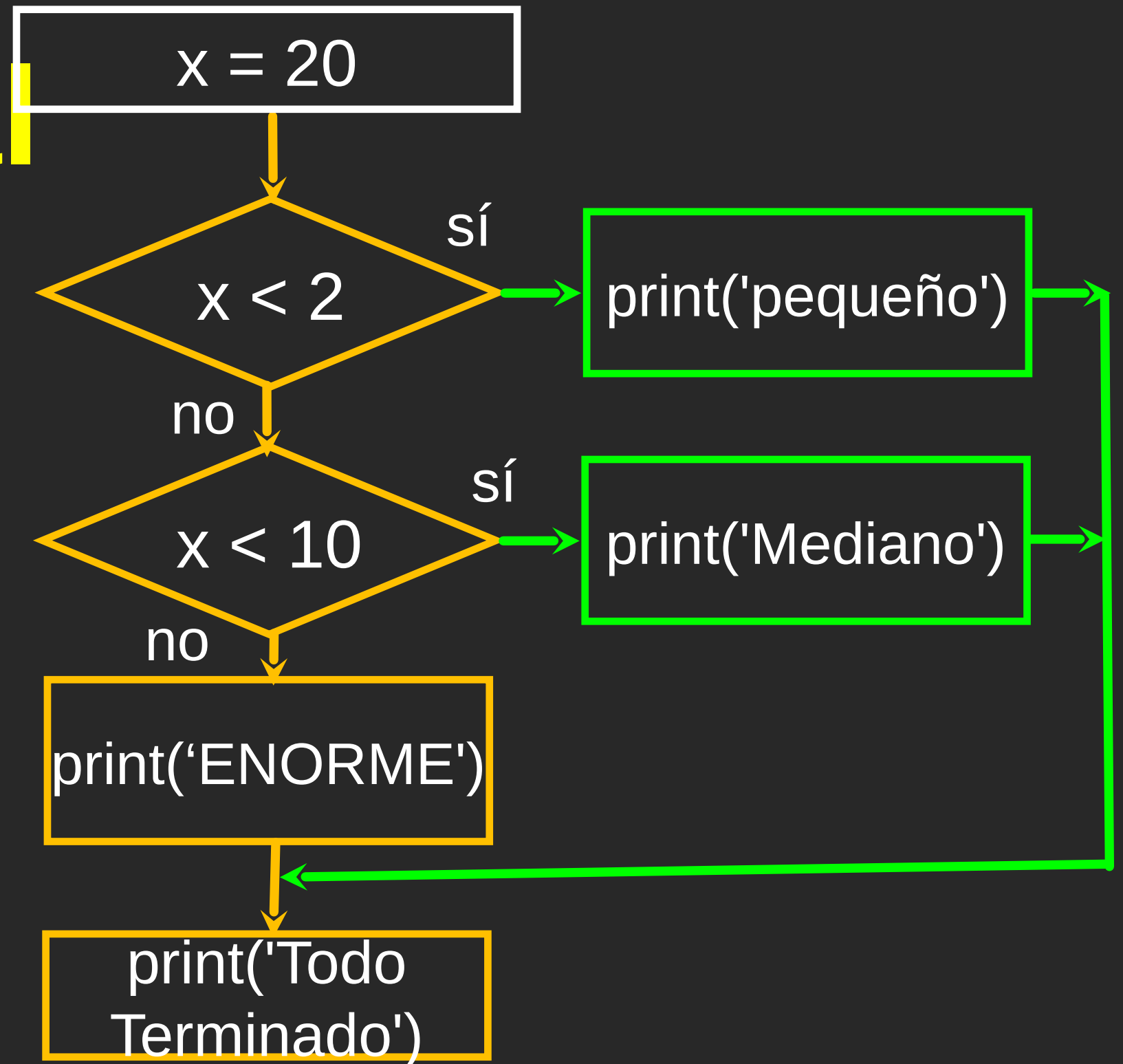
```
x = 5
if x < 2 :
    print('pequeño')
elif x < 10 :
    print('Mediano')
else :
    print('ENROME')
print('Todo
terminado')
```





Multidireccional

```
x = 20
if x < 2 :
    print('pequeño')
elif x < 10 :
    print('Mediano')
else :
    print('ENORME')
print('Todo
terminado')
```





Multidireccional

```
# No Else
x = 5
if x < 2 :
    print('Pequeño')
elif x < 10 :
    print('Mediano')

print 'Todo terminado'
```

```
if x < 2 :
    print('pequeño')
elif x < 10 :
    print('Mediano')
elif x < 20 :
    print('Grande')
elif x < 40 :
    print('Enorme')
elif x < 100:
    print('Gigante')
else :
    print('Descomunal')
```

Enigmas Multidireccionales

¿Cuál es el que nunca se imprimirá independientemente del valor de x?

```
if x < 2 :  
    print('Debajo de 2')  
elif x >= 2 :  
    print('Dos o más')  
else :  
    print('Otro')
```

```
if x < 2 :  
    print('Debajo de 2')  
elif x < 20 :  
    print('Debajo de 20')  
elif x < 10 :  
    print('Debajo de 10')  
else :  
    print('Otro')
```

La Estructura `try / except`

- Usted rodea una sección peligrosa del código con `try` y `except`
- Si el código en `try` funciona – `except` es omitido
- Si el código en `try` falla – pasa a la sección `except` 5

```
$ python3 notry.py
```

```
Traza de rastreo (llamada más reciente a la último): Archivo "notry.py", línea 2, in <module> istr = int(astr)ValueError: invalid literal for int() with base 10: 'Hola Bob'
```

```
$ cat notry.py
astr = 'Hola Bob'
istr = int(astr)
print('Primero', istr)
astr = '123'
istr = int(astr)
print('Segundo', istr)
```

Todo Terminado





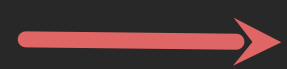
```
$ python3 notry.py
```

```
Trazas de rastreo (llamada más reciente a lo último): Archivo "notry.py", línea 2, in <module> istr = int(astr)ValueError: invalid literal for int() with base 10: 'Hola Bob'
```

Todo Terminado



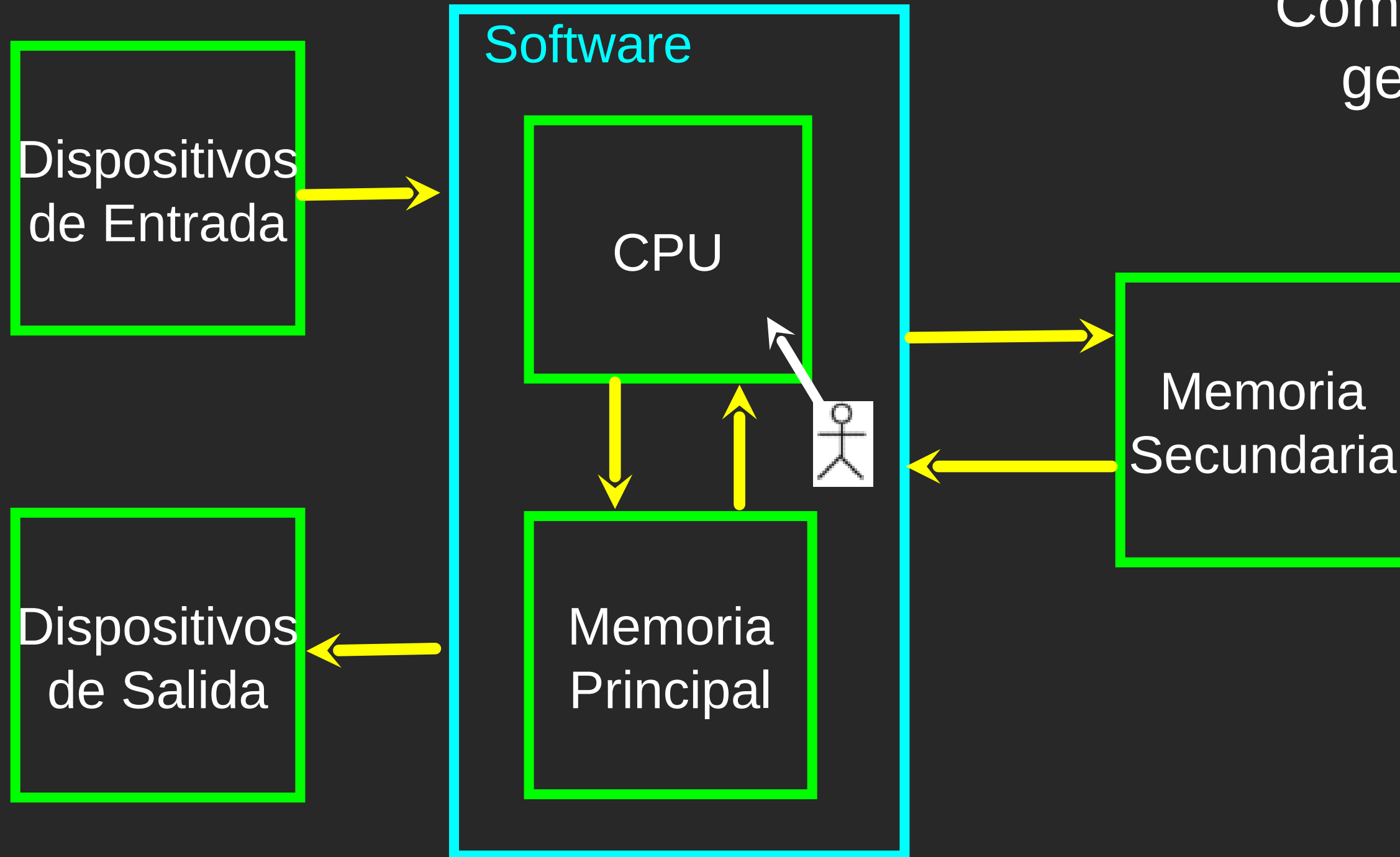
El programa se detiene aquí



```
$ cat notry.py
astr = 'Hola Bob'
istr = int(astr)
```

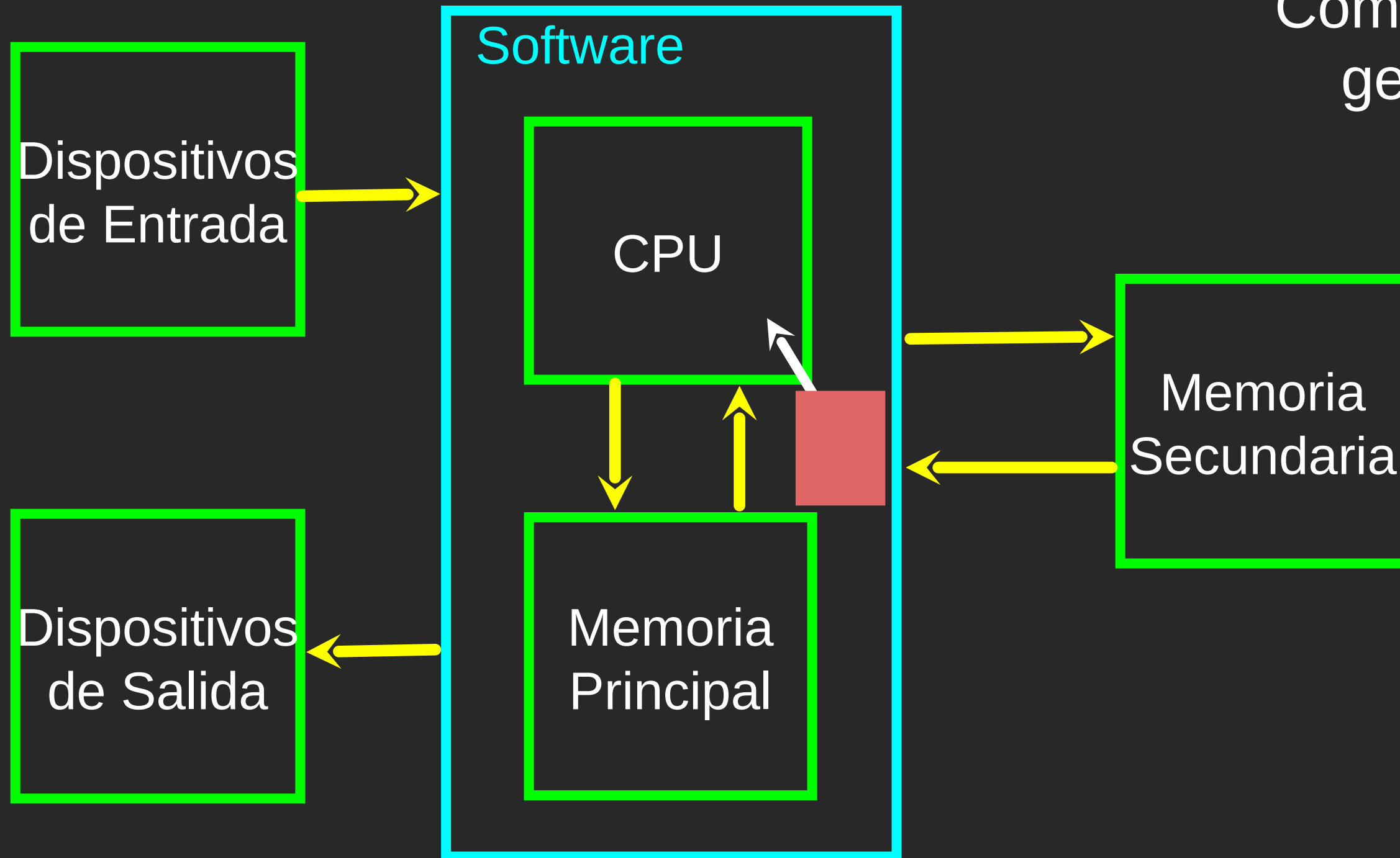


Computadora genérica





Computadora genérica





```

astr = 'Hola Bob'
try:
    istr = int(astr)
except:
    istr = -1

print('Primero', istr)
    
```



```

astr = '123'
try:
    istr = int(astr)
except:
    istr = -1

print('Segundo', istr)
    
```



Cuando la primera conversión falla
– simplemente cae en except
(excepción): clausula, y el
programa continúa.

```

$ python tryexcept.py
Primero -1
Segundo 123
    
```

Cuando la segunda conversión es
exitosa – solo omite except
(excepción): clausula, y el
programa continúa.

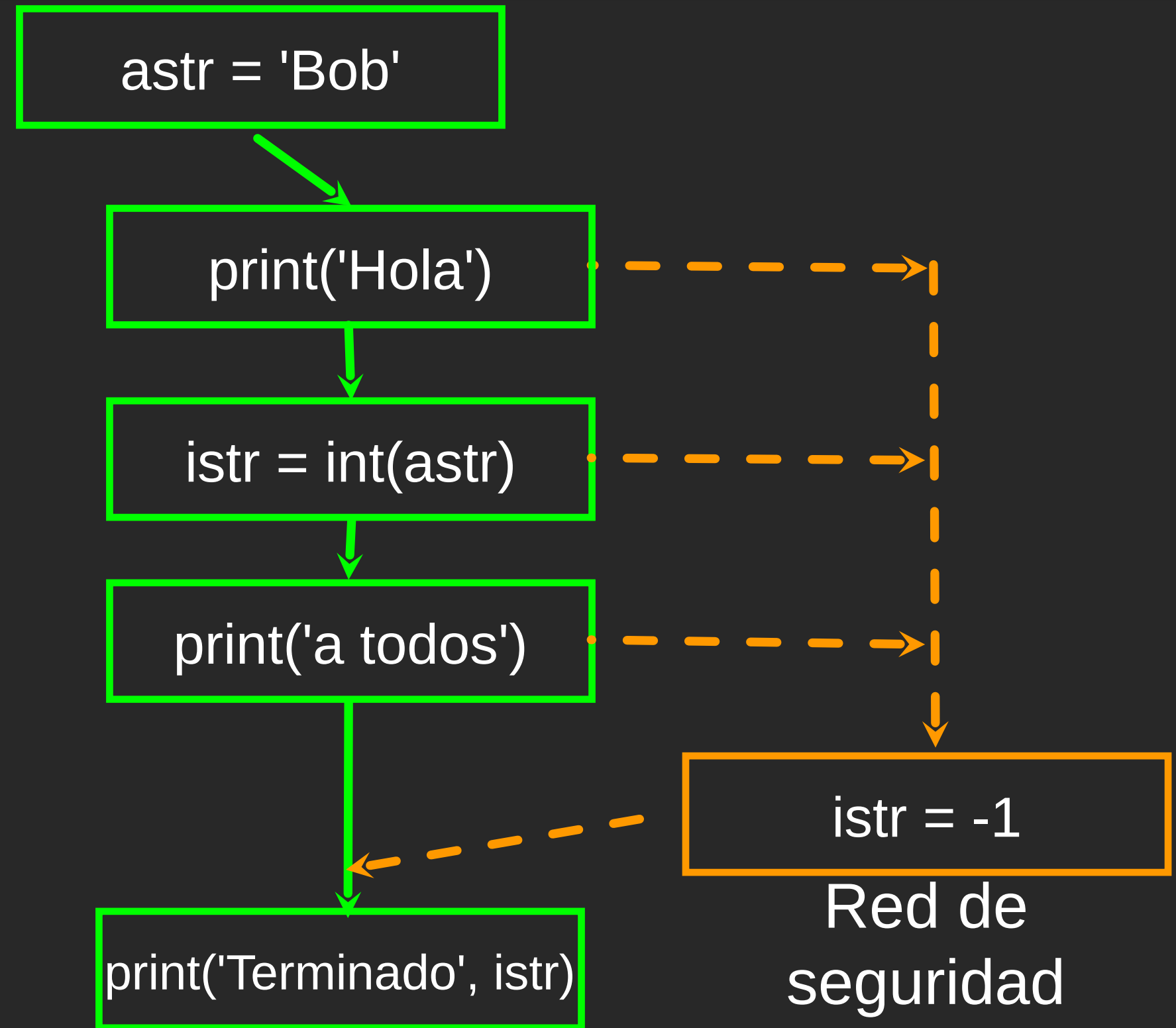


try / except

```

astr = 'Bob'
try:
    print('Hola')
    istr = int(astr)
    print('a todos')
except:
    istr = -1

print('Terminado',
istr)
    
```





Muestra de try / except

```
rawstr = input('Ingresar un número:')
try:
    ival = int(rawstr)
except:
    ival = -1

if ival > 0 :
    print('Buen trabajo')
else:
    print('No es un número')
```

```
$ python3 trynum.py
Ingresar un número:42
Buen trabajo
$ python3 trynum.py
Ingresar un
número:cuarenta-y-dos
No es un número
$
```



Ejercicio

Reescriba su cálculo del salario para darle al empleado 1,5 veces la tarifa por hora para las horas trabajadas que excedan las 40 horas.

```
Ingresar Horas: 45  
Ingresar Tarifa: 10
```

```
Salario: 475.0
```

$$475 = 40 * 10 + 5 * 15$$



Ejercicio

Reescriba su programa de salarios usando try y except de modo que su programa maneje input (entradas) no numéricas de forma correcta.

```
Ingresar Horas: 20
```

```
Ingresar Tarifa: nueve
```

```
Error, por favor, ingresar un valor  
numérico
```

```
Ingresar Horas: cuarenta
```

```
Error, por favor, ingresar un valor  
numérico
```



Síntesis

- Operadores de comparación
`==` `<=` `>=` `>` `<` `!=`
- Indentación
- Decisiones Unidireccionales
- Decisiones Bidireccionales:
`if:` y `else:`
- Decisiones Anidadas
- Decisiones Multidireccionales usando `elif`
- `try` / `except` para compensar errores



Agradecimientos / Colaboraciones



Estas diapositivas están protegidas por derechos de autor 2010-Charles R. Severance (www.dr-chuck.com) de la Facultad de Información de la Universidad de Michigan, y se ponen a disposición bajo licencia de Creative Commons Attribution 4.0. Por favor, conserve esta última diapositiva en todas las copias del documento para cumplir con los requisitos de atribución de la licencia. Si realiza algún cambio, siéntase libre de agregar su nombre y el de su organización a la lista de colaboradores en esta página cuando republique los materiales.

Desarrollo inicial: Charles Severance, Facultad de Información de la Universidad de Michigan

... Ingrese nuevos colaboradores y traductores aquí