

Conversión de tipos (casting)

Edison Achalma

Escuela Profesional de Economía, Universidad Nacional de San Cristóbal de Huamanga

Resumen

Este abstract será actualizado una vez que se complete el contenido final del artículo.

Tabla de contenidos

Introduction	2
0.1 Motivación	2
0.2 Intuición	2
0.3 Modelo mental	2
0.4 Desarrollo	2
0.4.1 1. <code>int()</code> - Convertir a Entero	2
0.4.2 2. <code>float()</code> - Convertir a Decimal	3
0.4.3 3. <code>str()</code> - Convertir a Texto	3
0.4.4 4. <code>bool()</code> - Convertir a Booleano	3
0.4.5 5. Conversión entre Tipos Numéricos	4
0.4.6 6. Casos de uso común	4
0.5 Ejemplos	5
0.6 Errores comunes	6
0.7 Buenas prácticas	6
0.8 Ejercicio	7
0.9 Síntesis	8
0.10 Qué sigue	8
1 Publicaciones Similares	8

Edison Achalma  <https://orcid.org/0000-0001-6996-3364>

El autor no tiene conflictos de interés que revelar. Los roles de autor se clasificaron utilizando la taxonomía de roles de colaborador (CRediT; <https://credit.niso.org/>) de la siguiente manera: Edison Achalma: conceptualización, metodología, análisis formal, investigación, recursos, curación de datos, redacción, visualización, supervisión, administración del proyecto

La correspondencia relativa a este artículo debe dirigirse a Edison Achalma, Escuela Profesional de Economía, Universidad Nacional de San Cristóbal de Huamanga, Portal Independencia N° 57, Ayacucho, AYA 5001, Perú, Email: elmer.achalma.09@unsch.edu.pe

Conversión de tipos (casting)

0.1 Motivación

En programación, a menudo necesitas convertir datos de un tipo a otro. Por ejemplo, cuando recibes entrada del usuario (siempre es texto), pero necesitas hacer cálculos matemáticos. O cuando quieres mostrar un número como parte de un mensaje de texto. El casting te permite controlar explícitamente qué tipo de dato estás usando, evitando errores y asegurando que tu código funcione correctamente.

0.2 Intuición

Imagina que tienes diferentes moldes para galletas: uno para círculos (int), uno para estrellas (float), y otro para letras (str). Tienes masa (datos) y puedes usar un molde para darle la forma que necesitas. El casting es como elegir el molde adecuado: `int()` convierte en entero, `float()` en decimal, `str()` en texto. Pero ten cuidado: no todos los moldes funcionan con todos los tipos de masa.

0.3 Modelo mental

Python usa **funciones constructoras** para crear objetos de tipos específicos:

1. `int()`: convierte a entero (elimina decimales, requiere texto numérico).
2. `float()`: convierte a decimal (puede convertir enteros y texto numérico).
3. `str()`: convierte a texto (funciona con casi cualquier cosa).
4. `bool()`: convierte a booleano (reglas específicas de verdadero/falso).

El casting es explícito: tú decides cuándo y cómo convertir.

0.4 Desarrollo

0.4.1 1. `int()` - Convertir a Entero

Convierte números (enteros, decimales) y cadenas de texto numéricas a enteros:

```
# Desde enteros
x = int(1)      # 1
y = int(-5)     # -5

# Desde floats (trunca, no redondea)
z = int(2.8)    # 2 (pierde el .8)
w = int(-3.9)   # -3 (trunca hacia cero)

# Desde strings (solo si representan números enteros)
a = int("10")   # 10
b = int("-5")   # -5

# Error: no se puede convertir texto no numérico
# c = int("Hola") # ValueError
```

0.4.2 2. *float()* - Convertir a Decimal

Convierte enteros, otros decimales y cadenas numéricas a floats:

```
# Desde enteros
x = float(1)      # 1.0
y = float(-5)     # -5.0

# Desde floats (no cambia)
z = float(2.8)    # 2.8

# Desde strings numéricos
a = float("3")    # 3.0
b = float("3.14") # 3.14
c = float("-5.5") # -5.5
d = float("1e3")  # 1000.0 (notación científica)

# Error: texto no numérico
# e = float("Hola") # ValueError
```

0.4.3 3. *str()* - Convertir a Texto

Convierte prácticamente cualquier cosa a texto:

```
# Desde enteros
x = str(5)        # "5"
y = str(-3)       # "-3"

# Desde floats
z = str(3.14)     # "3.14"
w = str(2.0)      # "2.0"

# Desde booleanos
a = str(True)     # "True"
b = str(False)    # "False"

# Desde None
c = str(None)     # "None"

# Desde estructuras de datos
d = str([1, 2, 3]) # "[1, 2, 3]"
e = str({"a": 1})  # '{"a": 1}'
```

0.4.4 4. *bool()* - Convertir a Booleano

Convierte valores a True o False según reglas específicas:

```
# Falsy (valores que son False)
print(bool(0))      # False
print(bool(0.0))    # False
```

```
print(bool(""))      # False
print(bool([]))      # False
print(bool({}))      # False
print(bool(None))    # False

# Truthy (valores que son True)
print(bool(1))       # True
print(bool(-5))      # True
print(bool("Hola"))  # True
print(bool(" "))     # True (texto con espacio)
print(bool([1, 2]))  # True
print(bool({"a": 1})) # True
```

0.4.5 5. Conversión entre Tipos Numéricos

```
# Entero → Flotante
x = 5
y = float(x)  # 5.0

# Flotante → Entero (trunca)
a = 3.99
b = int(a)    # 3 (pierde decimales)

# String numérico → Entero
edad = "25"
edad_int = int(edad)  # 25

# String numérico → Flotante
precio = "19.99"
precio_float = float(precio)  # 19.99

# Entero → String (para mostrar)
total = 100
mensaje = "El total es: " + str(total)  # "El total es: 100"
```

0.4.6 6. Casos de uso común

```
# 1. Entrada de usuario (siempre es string)
edad_input = "25"  # Simula input()
edad = int(edad_input)  # Convierte a entero para operaciones
print(edad + 5)  # 30

# 2. Mostrar números en mensajes
total = 42.50
print("El total es: " + str(total))  # "El total es: 42.5"
```

```
# 3. Operaciones mixtas
x = "10"
y = "5"
print(int(x) + int(y)) # 15 (convertir antes de sumar)

# 4. Validación de datos
def es_numero(texto):
    try:
        int(texto)
        return True
    except ValueError:
        return False
```

0.5 Ejemplos

Mínimo — Conversiones básicas:

```
# Conversiones numéricas
print(int(3.14)) # 3
print(float(5)) # 5.0
print(str(42)) # "42"
```

Progresivo — Conversiones en operaciones:

```
# Sin casting (error)
# print("Edad: " + 25) # TypeError

# Con casting (correcto)
edad = 25
print("Edad: " + str(edad)) # "Edad: 25"

# Mejor aún: f-strings
print(f"Edad: {edad}") # "Edad: 25"
```

Real — Sistema de registro de usuario:

```
# Simulación de entrada (todo es texto)
nombre = "Carlos"
edad_texto = "28"
altura_texto = "1.75"

# Casting para operaciones
edad = int(edad_texto)
altura = float(altura_texto)

# Cálculos
edad_futura = edad + 10
imc = 70 / (altura ** 2) # Peso fijo de 70kg
```

```
# Casting para mostrar
print("=== Registro de Usuario ===")
print("Nombre: " + nombre)
print("Edad: " + str(edad) + " años")
print("Altura: " + str(altura) + " m")
print("Edad en 10 años: " + str(edad_futura))
print(f"IMC estimado: {imc:.2f}") # F-strings sin casting necesario
```

0.6 Errores comunes

Error	Síntoma	Solución
Texto no numérico a int	ValueError: invalid literal	Verifica que el texto sea un número válido
Texto no numérico a float	ValueError: could not convert string to float	Solo strings numéricos ("3.14", "5")
Conversión de complex	TypeError: can't convert complex to int	No se puede convertir complex directamente
Casting innecesario	Código menos legible	Usa f-strings en lugar de str() para mostrar
Olvidar convertir input	Suma concatenada en lugar de suma matemática	Convierte input() a int/float antes de operar

0.7 Buenas prácticas

1. Siempre convierte entrada de usuario:

```
edad = int(input("Edad: ")) # input siempre retorna str
```

2. Usa f-strings para mostrar datos (evita casting manual):

```
# Menos elegante
print("Edad: " + str(edad) + " años")

# Más elegante y legible
print(f"Edad: {edad} años")
```

3. Valida antes de convertir:

```
valor = input("Número: ")
if valor.isdigit(): # Solo dígitos
    numero = int(valor)
else:
    print("No es un número válido")
```

4. Usa try/except para manejar errores de conversión:

```
try:
    edad = int(input("Edad: "))
except ValueError:
    print("Error: debes ingresar un número")
```

5. **Convierte lo más tarde posible:** mantén los datos en su tipo original hasta que necesites otro.

0.8 Ejercicio

Básico — Conversiones de tipos

Escribe un programa que:

1. Cree las siguientes variables:
 - `numero_texto` = "10"
 - `numero_decimal` = 3.14
 - `texto_numerico` = "5.5"
 - `valor_booleano` = True
2. Realice las siguientes conversiones:
 - Convierta `numero_texto` a entero
 - Convierta `numero_decimal` a entero (observa qué pasa)
 - Convierta `texto_numerico` a float
 - Convierta `valor_booleano` a entero
3. Muestre los resultados y sus tipos

Solución (no mires hasta haberlo intentado):

```
# Datos originales
numero_texto = "10"
numero_decimal = 3.14
texto_numerico = "5.5"
valor_booleano = True

# Conversiones
entero_desde_texto = int(numero_texto)
entero_desde_decimal = int(numero_decimal)  # Trunca a 3
float_desde_texto = float(texto_numerico)
entero_desde_booleano = int(valor_booleano)

# Resultados
print("=== Conversiones ===")
print(f"'{numero_texto}' → {entero_desde_texto} ({type(entero_desde_texto)})")
print(f"'{numero_decimal}' → {entero_desde_decimal} ({type(entero_desde_decimal)})")
print(f"'{texto_numerico}' → {float_desde_texto} ({type(float_desde_texto)})")
```

```
print(f"{valor_booleano} → {entero_desde_booleano} ({type(entero_desde_booleano)})")

# Nota: los booleanos son subclase de int en Python
print(f"\n¿True es int? {isinstance(True, int)}") # True
```

0.9 Síntesis

- **int()**: convierte a entero (trunca decimales, requiere números válidos).
- **float()**: convierte a decimal (funciona con enteros y texto numérico).
- **str()**: convierte a texto (funciona con casi todo).
- **bool()**: convierte a booleano (reglas de truthy/falsy).
- **F-strings**: alternativa moderna a **str()** para mostrar datos.
- **Validación**: siempre verifica antes de convertir entrada de usuario.
- **Errors comunes**: **ValueError** al convertir texto no numérico.

0.10 Qué sigue

- **Relacionado**: tipos de datos, formateo de strings, validación de entrada
- **Siguiente tema**: Strings — aprenderás a trabajar con texto y todas sus operaciones y métodos.

1 Publicaciones Similares

Si te interesó este artículo, te recomendamos que explores otros blogs y recursos relacionados que pueden ampliar tus conocimientos. Aquí te dejo algunas sugerencias:

1. [!\[\]\(4f6d8a8b127300a02d56d34d01423d15_img.jpg\) Instalacion De Anaconda](#)
2. [!\[\]\(7e3d1ad67bf2d7a17700a66d1a313f91_img.jpg\) Configurar Entorno Virtual Python Anaconda](#)
3. [!\[\]\(6aaf22e5a325c32ef2122c2939c64f9c_img.jpg\) 01 Introducion A La Programacion Con Python](#)
4. [!\[\]\(27421322d686d4980ae2b7f101ee89ba_img.jpg\) 02 Variables Expresiones Y Statements Con Python](#)
5. [!\[\]\(c45bffa8c48916027d3ccd561ed26723_img.jpg\) 03 Objetos De Python](#)
6. [!\[\]\(bac5dbca87c84b327a183d989114aa0a_img.jpg\) 04 Ejecucion Condicional Con Python](#)
7. [!\[\]\(9b39af2adf53a7d5459150a09631e679_img.jpg\) 05 Iteraciones Con Python](#)
8. [!\[\]\(f144f3b86155f30ed377aa0443a661b7_img.jpg\) 06 Funciones Con Python](#)
9. [!\[\]\(dadd32e946c6a58b99e1a6c81bb4ec46_img.jpg\) 07 Dataframes Con Python](#)
10. [!\[\]\(09e9f8ca86ff29f5796eb28c2464c49f_img.jpg\) 08 Prediccion Y Metrica De Performance Con Python](#)
11. [!\[\]\(9c3f4ce44a934e929ff02b8691b078d5_img.jpg\) 09 Metodos De Machine Learning Para Clasificacion Con Python](#)
12. [!\[\]\(63927fb89eb2a2f2ff6a82f7c3a02628_img.jpg\) 10 Metodos De Machine Learning Para Regresion Con Python](#)
13. [!\[\]\(f79d645836f935ba69cec1ddfafe4c27_img.jpg\) Visualizacion De Datos Con Python](#)

Esperamos que encuentres estas publicaciones igualmente interesantes y útiles. ¡Disfruta de la lectura!