Type-Casting

```
# implicit type conversion
a = 7
print(type(a))
b = 3.0
print(type(b))
c = a+b
print(c)
print(type(c))
```

```
<class 'int'>
<class 'float'>
10.0
<class 'float'>
```

```
# explicit type conversion

a = 7
b = 3.5
c = int(a+b)
print(type(a))
print(type(b))
print(c)
print(type(c))
```

```
<class 'int'>
<class 'float'>
10
<class 'int'>
```

```
# int()

a = 8.5
i_1 = int(a)
print(i_1)
print(type(i_1))
#b = "1.5"
#i_2 = int(b)
#print(type(i_2))  # --- this will throw an error
c = "10"
i_3 = int(c)
print(i_3)
print(type(i_3))
d = "abc"
i_4 = int(d)
print(type(i_4))
```

```
8
<class 'int'>
10
<class 'int'>
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-9-db6489b5d349> in <cell line: 15>()
     13 print(type(i_3))
     14 d = "abc"
---> 15 i_4 = int(d)
     16 print(type(i_4))
     17

ValueError: invalid literal for int() with base 10: 'abc'
```

Next steps: Explain error

```
# float()

a = 8
f_1 = float(a)
print(f_1)
print(type(f_1))
b = "10.5"
f_2 = float(b)
print(f_2)
print(type(f_2))
c = "10"
f_3 = float(c)
print(f_3)
print(type(f_3))
d = "abc"
f_4 = float(d)
print(type(f_4))
```

```
8.0
<class 'float'>
10.5
<class 'float'>
10.0
<class 'float'>
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-12-e66e9fef2d4a> in <cell line: 16>()
     14 print(type(f_3))
     15 d = "abc"
---> 16 f_4 = float(d)
     17 print(type(f_4))

ValueError: could not convert string to float: 'abc'
```

Next steps: Explain error

```
# str()

a = 8
print(type(a))
s_1 = str(a)
print(s_1)
print(type(s_1))
b = 10.5
print(type(b))
s_2 = str(b)
print(s_2)
print(type(s_2))
```

```
<class 'int'>
8
<class 'str'>
<class 'float'>
10.5
<class 'str'>
```

Operators

Arithmetic Operators

```
a = 8
b = 5
```

```
print("+", a+b)
print("-", a-b)
print("*", a*b)
print("/", a/b)
print("//", a//b)
print("%", a%b)
print("**", a**b)
print(type(a%b))
```

```
+ 13
- 3
* 40
/ 1.6
// 1
% 3
** 32768
<class 'int'>
```

```
2**5
```

```
32
```

```
7//2
```

```
3
```

```
-7//2
```

```
-4
```

```
12.5%7.5
```

```
5.0
```

```
20.5%7.5
```

```
5.5
```

```
-7%5
```

```
3
```

```
-21%8
```

```
3
```

```
-27%7
```

```
1
```

```
a = 2.5
b = 5.2

print("+", a+b)
print("-", a-b)
print("*", a*b)
print("/", a/b)
print("//", a//b)
print("%", a%b)
print("**", a**b)
```

```
+ 7.7
- -2.7
* 13.0
```

```
/ 0.4807692307692307
// 0.0
% 2.5
** 117.29730800599916
```

Comparison Operator

```
a = 8
b = 5

print("==", a==b)
print("!=", a!=b)
print(">=", a>=b)
print("<=", a<=b)
print(">", a>b)
print("<", a<b)
print(type(a<b))
```

```
== False
!= True
>= True
<= False
> True
< False
<class 'bool'>
```

```
a = 5
b = 5

print("==", a==b)
print("!=", a!=b)
print(">=", a>=b)
print("<=", a<=b)
print(">", a>b)
print("<", a<b)
print(type(a<b))
```

```
== True
!= False
>= True
<= True
> False
< False
<class 'bool'>
```

```
print(0==0.0)
print(1==1.0)
print(True==1)
print(True==1.0)
print(False==0)
print(False==0.0)
```

```
True
True
True
True
True
True
```

Logical Operators

```
# and, or, not

x = 5
y = 10

print(x>4 and y<12)
print(x>4 and y<10)
print(x>6 and y<10)
print(x>6 and y<12)
print(x>4 or y<12)
print(x>4 or y<10)
print(x>6 or y<10)
print(x>6 or y<12)
```

```
True
False
False
False
True
True
False
True
```

```
a = 8
b = 5
c = 12

print(a and b and c)
```

```
12
```

```
a = 8
b = 5
c = 0

print(a and b and c)
```

```
0
```

```
a = 8
b = 0
c = 12

print(a and b and c)
```

```
0
```

```
a = 0
b = 5
c = 12

print(a and b and c)
```

```
0
```

```
a = 8
b = 5
c = 12

print(a or b or c)
```

```
8
```

```
a = 0
b = 5
c = 12

print(a or b or c)
```

```
5
```

```
a = 0
b = 0
c = 12

print(a or b or c)
```

```
12
```

```
a = 0
b = 0
c = 0

print(a or b or c)
```

```
0
```

```
a = -1
b = -2
c = -3

print(a or b or c)
print(a and b and c)
```

```
-1
-3
```

```
not(a or b or c)
```

```
False
```

```
a = 10
b = 5
c = 12
d = 0

print(a and c and b) # q1
print(d or a or c) # q2
print(not(a and d and b)) # q3
print(not(d or d or a)) # q4
```

```
5
10
True
False
```

Membership Operator

Start coding or generate with AI.