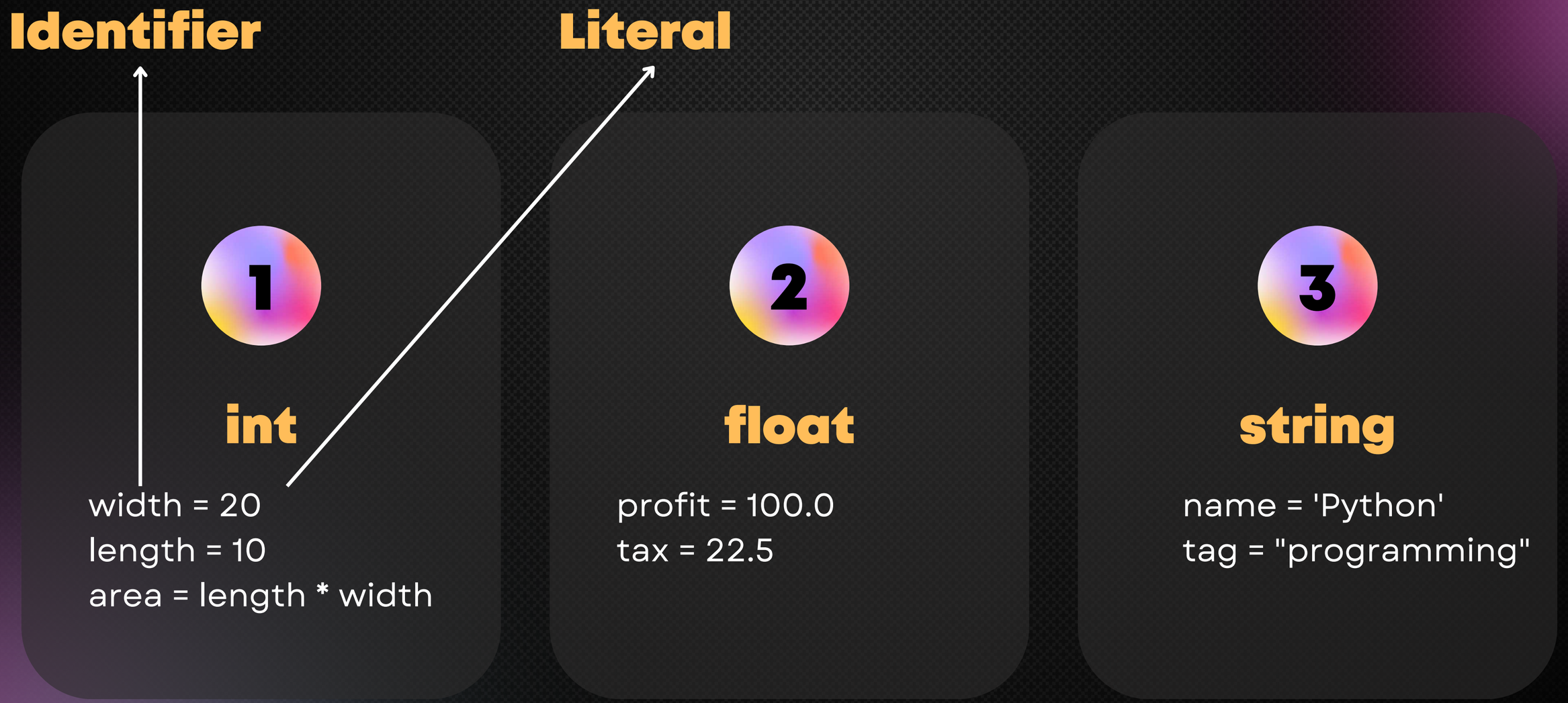
# Python 002

Data types
1. Numeric ( int, float )
2. Booleans
3. Strings
4. Datetime

Line Joining ( implicit, explicit )
Comments
Type Casting
Built_ins


Identifier
Literal
1
int
width = 20
length = 10
area = length * width
2
float
profit = 100.0
tax = 22.5
3
string
name = 'Python'
tag = "programming"

Identifiers

Delimiters

What is a statement composed of in Python?

Keywords

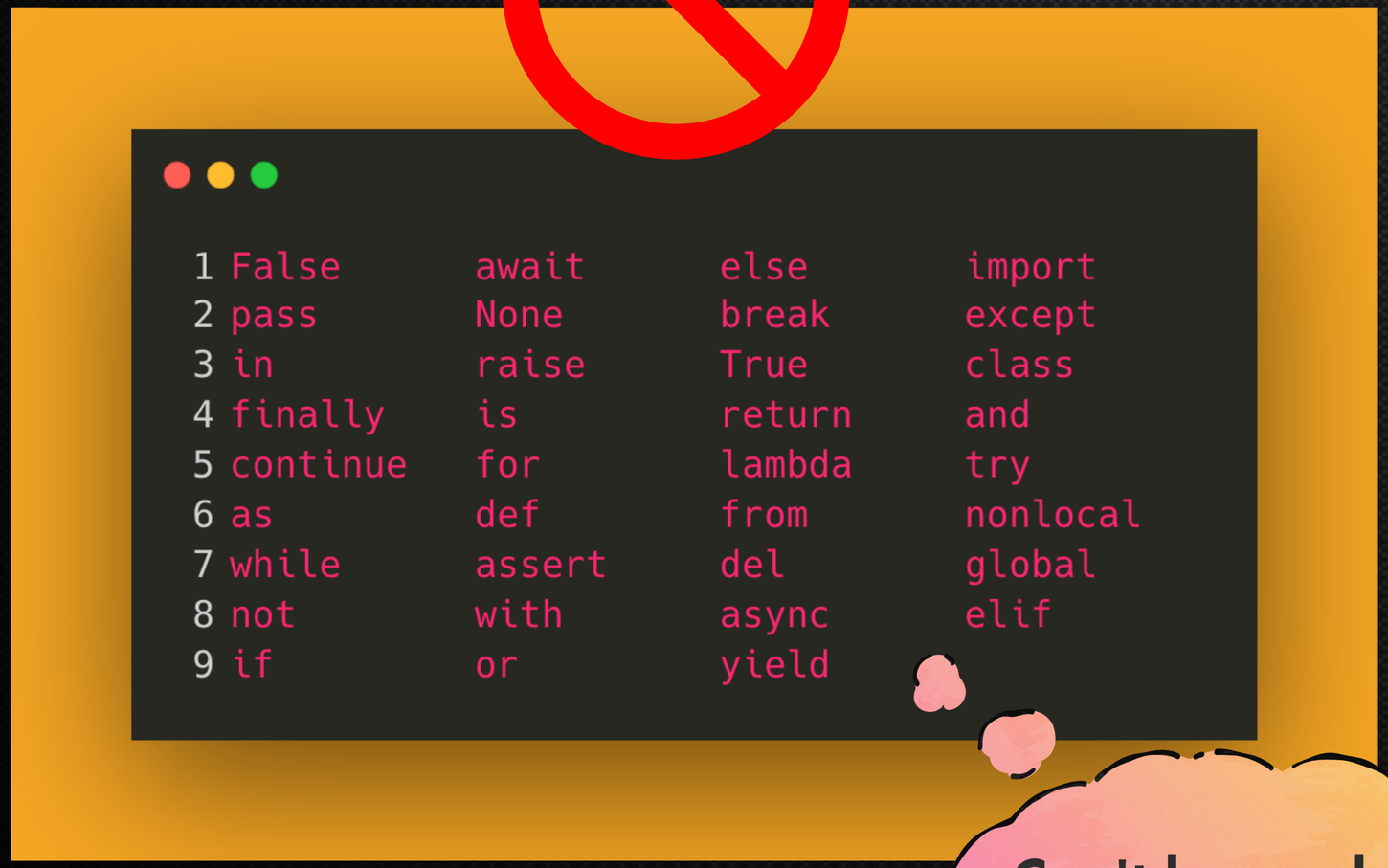
Operators

Literals

# Identifiers

Names used to identify variables, functions, or other entities in Python

- Variable identifier: x, count, name
- Function identifier: calculate_sum, print_message
- Class identifier: MyClass, Circle, Rectangle



```
1 False      await      else       import
2 pass       None       break      except
3 in         raise      True       class
4 finally    is         return     and
5 continue   for        lambda     try
6 as         def        from       nonlocal
7 while      assert     del        global
8 not        with       async      elif
9 if         or         yield
```

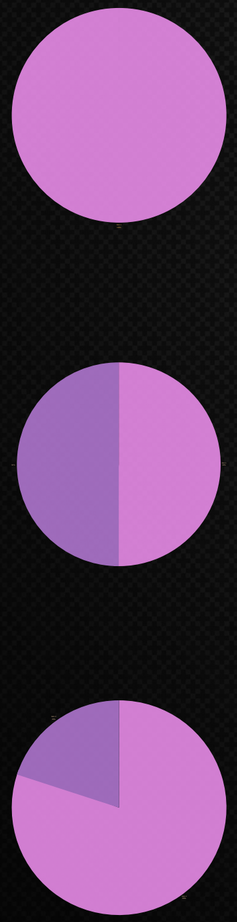
# Keywords

Reserved words with predefined meanings in Python programming

Can't be used as identifiers for naming

# Delimiters

Characters used to separate or group elements in Python code



**Parenthesis ( )**

**Brackets [ ]**

**Comma ' , '**

# Literals

Fixed values used to represent data in Python code

- Numeric Literals: 42, 3.14, 0b1010 (binary), 0o777 (octal)
- String Literals: "Hello, World!", 'Python is fun!'
- Boolean Literals: True, False
- None: (representing absence of a value)

# Operators

Symbols for performing operations on data in Python

- **Arithmetic operators**: +, -, *, /, %
- **Comparison operators**: ==, !=, <, >, <=, >=
- **Assignment operators**: =, +=, -=, *=, /=
- **Logical operators**: and, or, not

## this is a comment

```
# this is a comment
```

## this is a docstring

```
"""
this is used to generated helpful documentation of function, class, module
"""
```

# Implicit Line Joining

```
result = (1 + 2 +
          3 + 4 +
          5 + 6)
```

Done by Packages / Extensions automatically ( if you have those installed in your IDE )

Improves Readability

# Explicit Line Joining

```
names = ['Alice', \
'Bob', 'Charlie', \
      'Dave', 'Eve']
```

- Done using a backslash (\) at the end of a line
- indicate that the line should be continued to the next line
- Used to split a long line of code for improved readability

# Type Casting

Magic!
Convert Int to Float


int ----- float

# Magic Mantras

bool ------ int
int(bool_value)

int ----- float
float(int_value)

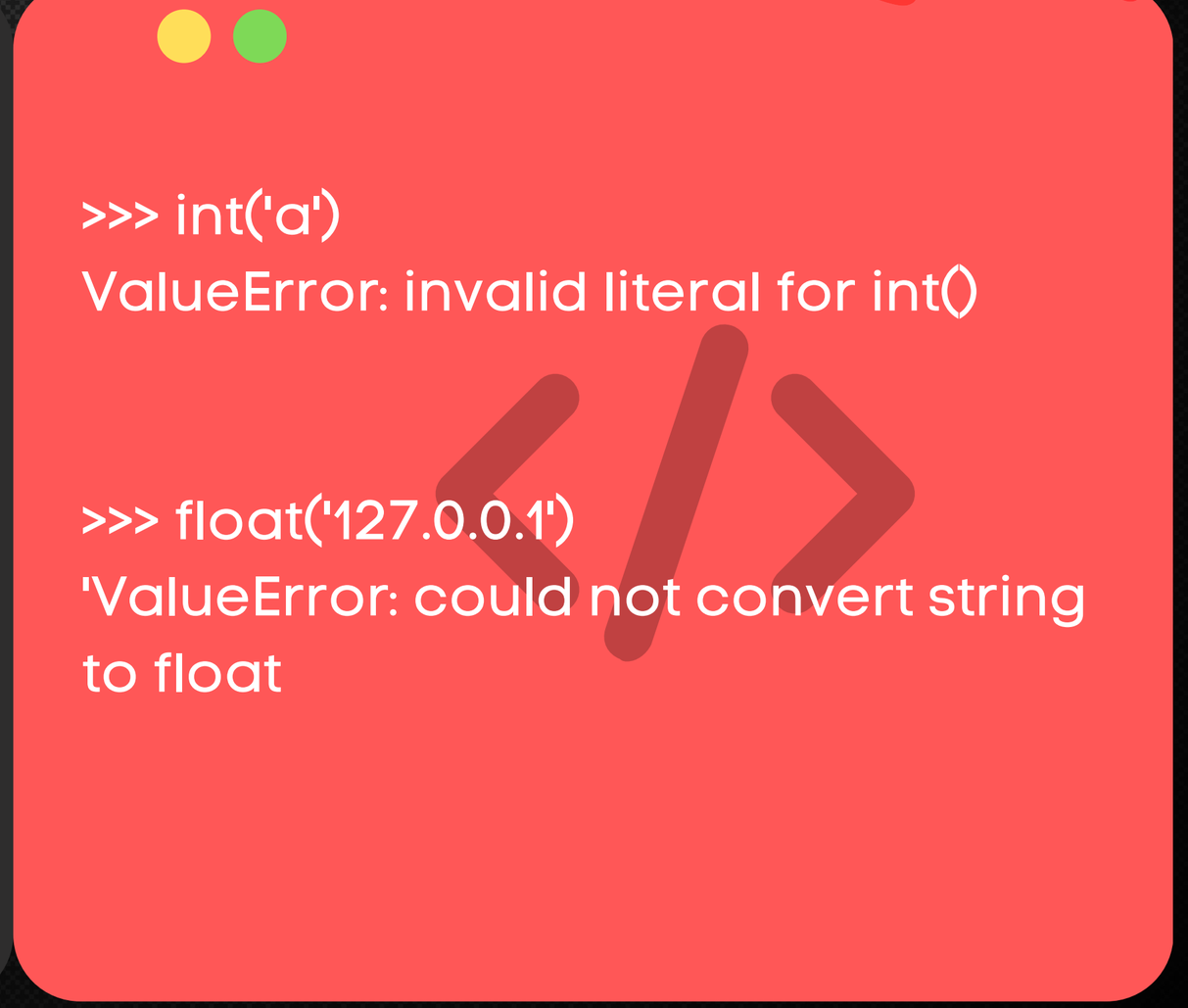

string ------ int
int(string_value)

float ------ int
int(float_value)

float ------ string
string(float_value)

# How magic works

```
>>> int(3.11)
3
>>> bool(-1)
True
>>> str(1.2)
'1.2'
>>> str(True)
'True'
>>> float(32)
32.0
```

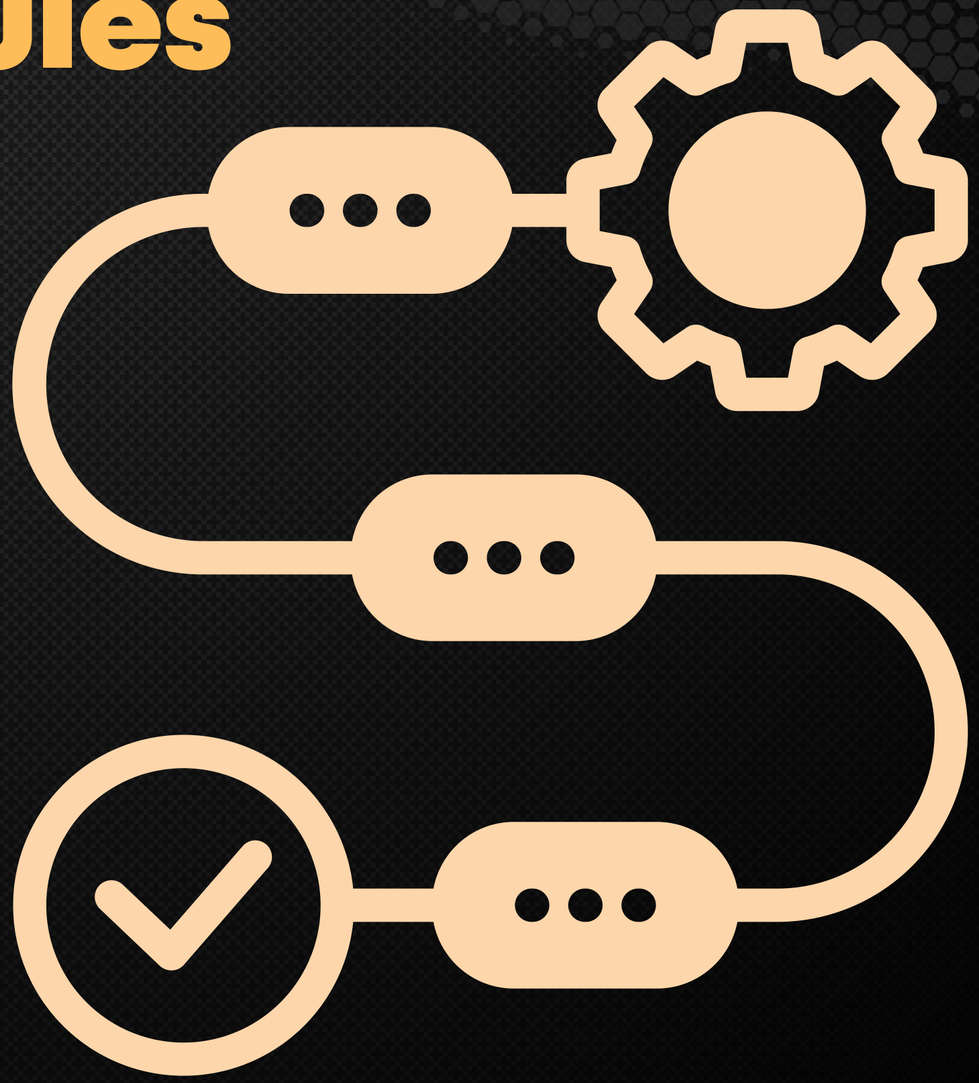


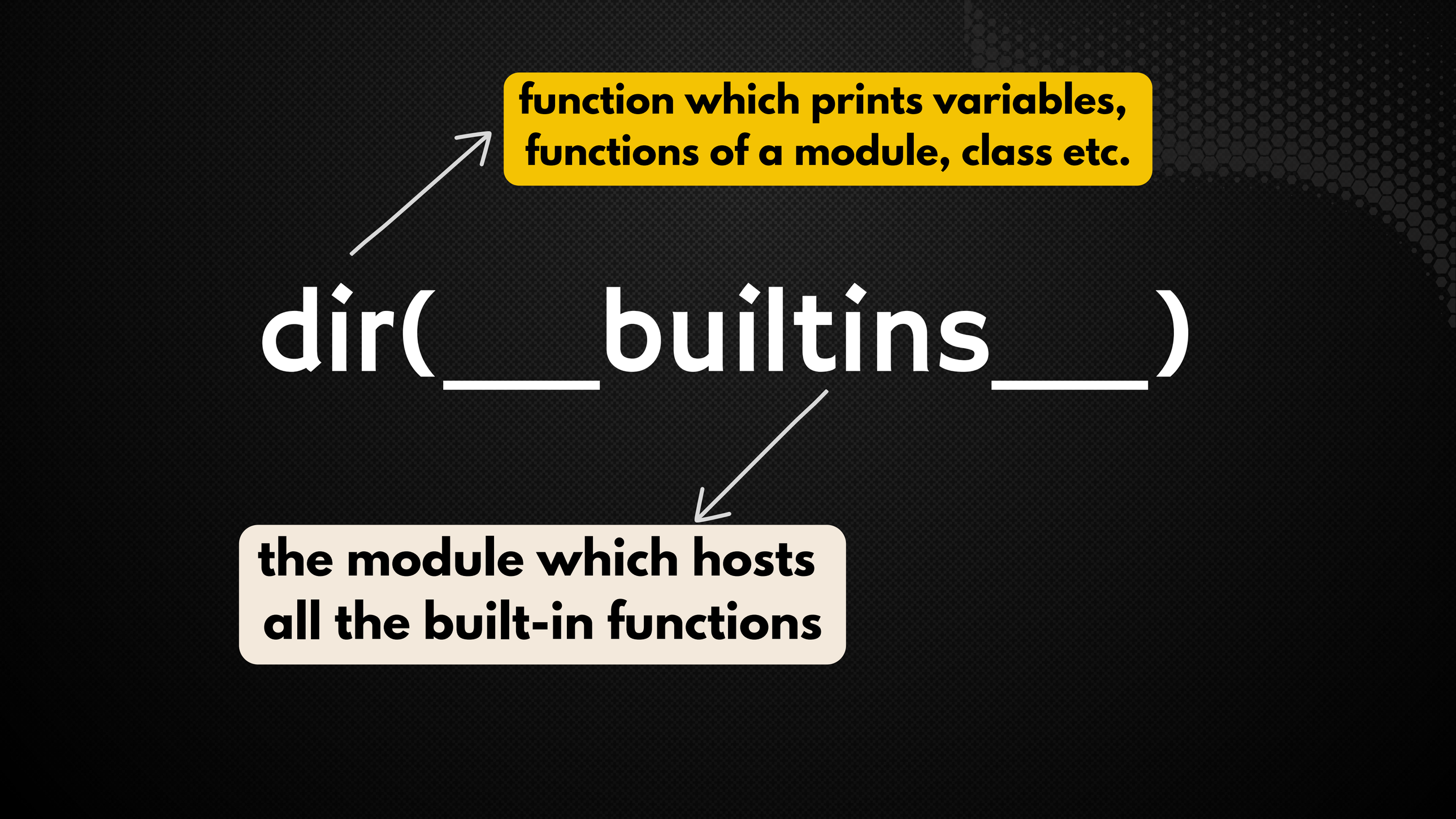
```
>>> int('a')
ValueError: invalid literal for int()


>>> float('127.0.0.1')
'ValueError: could not convert string
to float
```

# Built-in Modules

Python provides a set of built-in functions that are readily available for use without the need for importing any modules




function which prints variables,
functions of a module, class etc.
dir(__builtins__)
the module which hosts
all the built-in functions

```
1 abs, all, any, ascii, bin, bool, breakpoint, bytearray, bytes, callable,
2 chr, classmethod, compile, complex, delattr, dict, dir, divmod, enumerate,
3 eval, exec, filter, float, format, frozenset, getattr, globals, hasattr, hash,
4 help, hex, id, input, int, isinstance, issubclass, iter, len, list, locals,
5 map, max, memoryview, min, next, object, oct, open, ord, pow, print, property,
6 range, repr, reversed, round, set, setattr, slice, sorted, staticmethod, str,
7 sum, super, tuple, type, vars, zip, import
```

# Built-ins

# Type Function

## type(obj)

built-in function in Python that returns the type of an object.

type(32) = <class 'int'>

## Code Example

```
x = 10
print(type(x))
# Output: <class 'int'>

print(type("Hello"))
# Output: <class 'str'>
```

# Range Function

## range(start, end)

built-in function in Python that generates a sequence of numbers

range(stop)
range(start, stop)
range(start, stop, step)

## Code Example

```
for num in range(5):
    print(num)  # Output: 0 1 2 3 4

for num in range(2, 6):
    print(num)  # Output: 2 3 4 5

for num in range(1, 10, 2):
    print(num)  # Output: 1 3 5 7 9
```

# Any Function

## any(bool, bool, expression )

built-in function in Python that returns True if at least one element in an iterable object is true, and False otherwise

## Code Example

```
any([False, False, True])
# T rue

values = (-1, 0, -3, 5)
print(any(val > 0 for val in values))  #
Output: True
```

# Built-in Modules

## Datetime

The datetime module in Python provides classes and functions for working with dates and times.

IMPORT

**import datetime**

CURRENT TIME

**current_time = datetime.datetime.now()**

PRINT

**print(current_time)**

# time module

The time module in Python provides functions for working with time-related functionality, specifically dealing with time in terms of seconds

# Running time

```
import time
start_time = time.time()

# Code to measure the running time
# ...
end_time = time.time()

running_time = end_time - start_time
print(running_time)
```

# math module

The math module in Python provides functions for mathematical calculations.

## Square Root

```
import math
number = 16
square_root = math.sqrt(number)
print(square_root)  # Output: 4.0
```

## Factorial

```
import math
number = 5
factorial = math.factorial(number)
print(factorial)  # Output: 120
```

## Ceil

```
import math
number = 3.7
rounded_number = math.ceil(number)
print(rounded_number)  # Output: 4
```

# random module

The random module in Python provides functions for generating random numbers

## Example 1

```
import random
random_number = random.random()
print(random_number)  # Output: a random float between 0 and 1
```

## Example 2

```
import random
random_integer = random.randint(1, 10)
print(random_integer)  # Output: a random integer between 1 and 10
```

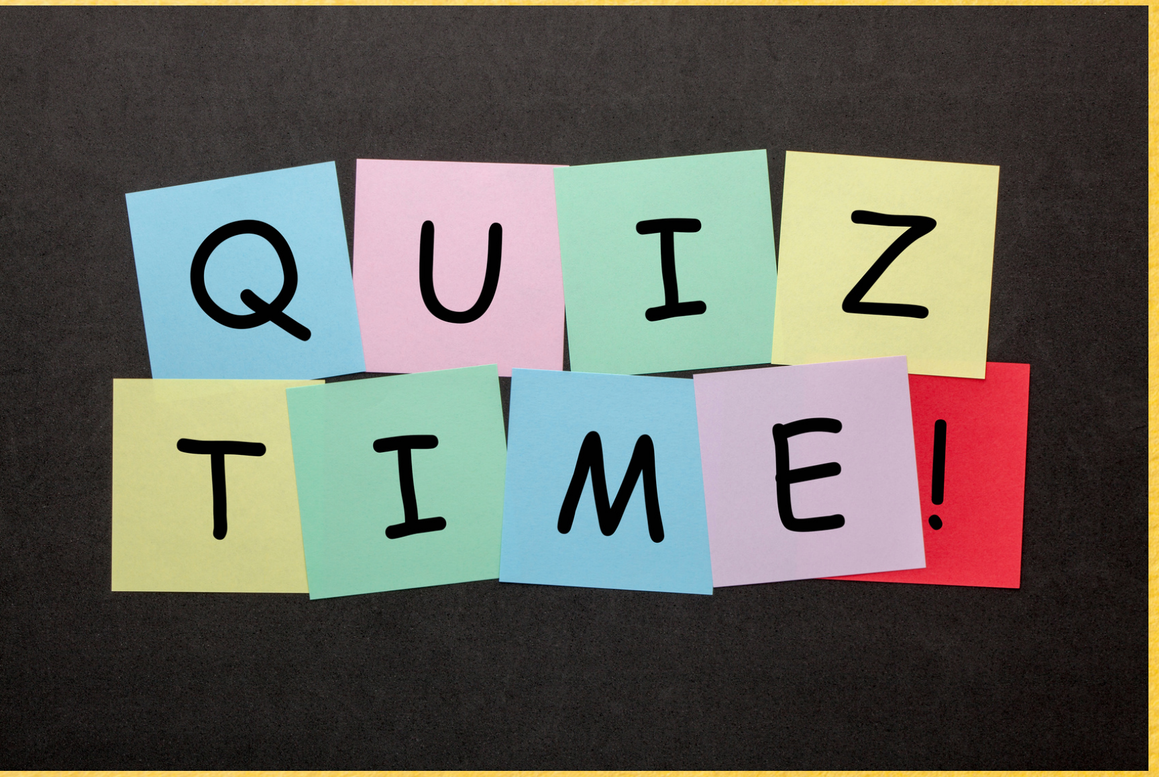
## Example 3

```
import random
random_float = random.uniform(0.5, 1.5)
print(random_float)
```

function to print all the functions available inside of a python module.

```
dir(module_name)
```

name of the module
e.g. dir(time) or dir(math)




Thank you! :)