**1.1 Python 3.x Magic Methods**

c\_\_ini\_\_() Initialize the object

**1.2 Conversion**

c\_\_str\_\_() str(c)

c\_\_rep\_\_() repr(c)

c\_\_forma\_\_(spec) "{0:spec}".format(c)

c\_\_complex\_\_() complex(c)

c\_\_int\_\_() to int

c.\_\_float\_\_() to float

c.\_\_bool\_\_() to bool  
c.\_\_bytes\_\_() like \_\_str\_\_ but for byte arrays

**1.3 Iteration**

c.\_\_iter\_\_() iter(c)  
c.\_\_next\_\_() next(c)

c.\_\_reversed\_\_() reverse iterator

**1.4 Attribute access**

c.\_\_getattr\_\_(a) c.a, fallback

c.\_\_getattribute\_\_(a) c.a, unconditional

c.\_\_setattr\_\_(a, v) c.a = v

c.\_\_delattr\_\_(a) del c.a

c.\_\_dir\_\_() list attributes (for dir())  
c.\_\_dict\_\_{} access class dict directly

**1.5 Calling**

c.\_\_call\_\_(...) c(...)

**1.6 Immutable container**

c.\_\_contains\_\_(e) e in c

c.\_\_len\_\_() Length (for len())  
c.\_\_getitem\_\_(x) c[x]

**1.7 Mutable container**

c.\_\_setitem\_\_(x, v) c[x] = v

c.\_\_delitem\_\_(x) del c[x]

c.\_\_missing\_\_(x) c[x] if c not in x

**1.8 Emulating numbers**

c.\_\_add\_\_(x) c + x

c.\_\_sub\_\_(x) c - x

c.\_\_mul\_\_(x) c \* x

c.\_\_mod\_\_(x) c % x

c.\_\_floordiv\_\_(x) c // x

c.\_\_divmod\_\_(x) divmod(c, x)

c.\_\_pow\_\_(x) c \*\* x

c.\_\_rshift\_\_(x) c >> x

c.\_\_lshift\_\_(x) c << x

c.\_\_and\_\_(x) c & x

c.\_\_xor\_\_(x) c ^ x

c.\_\_or\_\_(x) c | x

c.\_\_radd\_\_(x) x + c

etc.

c.\_\_iadd\_\_(x) c += x

etc.

c.\_\_neg\_\_(x) -c

c.\_\_pos\_\_() +c

c.\_\_abs\_\_() abs(c)

c.\_\_invert\_\_() ~c

c.\_\_round\_\_(n) round(c, n)  
c.\_\_floor\_\_() used by math.floor()

c.\_\_ceil\_\_() used by math.ceil()

c.\_\_trunc\_\_() used by math.trunc()

c.\_\_index\_\_() lst[c]

c.\_\_oct\_\_() oct(c)

c.\_\_hex\_\_() hex(c)  
c.\_\_matmul\_\_() matrix multiplication with @  
 **1.9 Comparisons**

c.\_lt\_(o) c < o

c.\_\_gt\_\_(o) c > o

c.\_\_le\_\_(o) c <= o

c.\_\_ge\_\_(o) c >= o

c.\_\_eq\_\_(o) c == o

c.\_\_ne\_\_(o) c != o

**1.10 Context handlers**

c.\_\_enter\_\_(o) with c: (entry)

c.\_\_exit\_\_(o, exp\_type, exp\_val, trace) with c: (exit)

**1.11 Low-level stuff**

Read the documentation!

c.\_\_new\_\_(cls, …) object construction

c.\_\_del\_\_() clean up object

c.\_\_slots\_\_() limit attributes

c.\_\_hash\_\_() compute hash value  
c.\_\_sizeof\_\_() sys.getsizeof(c)

c.\_\_metaclass\_\_() the class of the class  
c\_\_prepare\_\_() used for metaclasses

c.\_\_instancecheck\_\_(x) isinstance(x, c)

c.\_\_subclasscheck\_\_(x) issubclass(x, c)

**1.12 Pickling**

c.\_\_getstate\_\_() pickle.dump(pkl\_file, self)  
c.\_\_setstate\_\_() data = pickle.load(pkl\_file)  
There few more magic methods for advanced pickling

**1.13 Copying**

c.\_\_copy\_\_() copy.copy() for c  
c.\_\_deepcopy\_\_(memodict={}) copy.deepcopy() for c

**1.14 Descriptor Objects**Class with one or more methods like this:

c.\_\_get\_\_(instance, owner) when value is retrieved  
c.\_\_set\_\_(instance, value) when value is changed  
c.\_\_delete\_\_(self, instance) when value is deleted  
owner is the owner class itself  
  
**1.15 Built-ins**

\_\_all\_\_[] list of default module exports  
\_\_doc\_\_ doc strings

\_\_name\_\_ module name or “\_\_main\_\_”

\_\_qualname\_\_ full name of the class  
\_\_module\_\_ module name  
\_\_defaults\_\_{} default argument values  
\_\_kwdefaults\_\_{} defaults for keyword-only parameters  
\_\_code\_\_ code object of compiled function body  
\_\_globals\_\_{} function’s global variables  
\_\_version\_\_ module version in string format