

**sys.getsizeof(object[, default])**Return the size of an object in bytes. The object can be any type of object. All built-in objects will return correct results, but this does not have to hold true for third-party extensions as it is implementation specific.Only the memory consumption directly attributed to the object is accounted for, not the memory consumption of objects it refers to.If given, default will be returned if the object does not provide means to retrieve the size. Otherwise a TypeError will be raised.getsizeof() calls the object’s \_\_sizeof\_\_ method and adds an additional garbage collector overhead if the object is managed by the garbage collector.

if we take a look at instances of these classesThe default argument is 0 and these two instances have the same amount of memory usage for this argument.

n CPython implementation, every object ([source](https://github.com/python/cpython/blob/f7d72e48fb235684e17668a1e5107e6b0dab7b80/Include/object.h#L104)) begins with a reference count and a pointer to the type object for that object. That's 16 bytes.

|  |  |  |
| --- | --- | --- |
| **Bytes** | **type** | **scaling notes** |
| 28 | int | +4 bytes about every 30 powers of 2 |
| 37 | bytes | +1 byte per additional byte |
| 49 | str | +1-4 per additional character (depending on max width) |
| 48 | tuple | +8 per additional item |
| 64 | list | +8 for each additional |
| 224 | set | 5th increases to 736; 21nd, 2272; 85th, 8416; 341, 32992 |
| 240 | dict | 6th increases to 368; 22nd, 1184; 43rd, 2280; 86th, 4704; 171st, 9320 |
| 24 | float |  |
| 28 | boolean |  |
| 56 | class inst | Has a\_\_dict\_ attr, same scaling as dict above |
| 888 | class def | with slots |
| 1056 | class def | No\_slots |
| 136 | func def | does not include default args and other attrs |