Pickling Without a File

The examples in the previous section showed how to serialize a Python object directly to a file on disk. But what if you don't want or need a file? You can also serialize to a `bytes` object in memory.

```
import pickle

shell = 1
print (shell)
#1
with open('entry.pickle', 'rb') as f:
     entry = pickle.load(f)

b = pickle.dumps(entry)              #①
print (type(b))                      #②
#<class 'bytes'>

entry3 = pickle.loads(b)             #③
print (entry3 == entry)              #④
#True
```

① The `pickle.dumps()` function (note the '`s`' at the end of the function name) performs the same serialization as the `pickle.dump()` function. Instead of taking a stream object and writing the serialized data to a file on disk, it simply returns the serialized data.

② Since the pickle protocol uses a binary data format, the `pickle.dumps()` function returns a bytes object.

③ The `pickle.loads()` function (again, note the `'s'` at the end of the function name) performs the same deserialization as the `pickle.load()` function. Instead of taking a stream object and reading the serialized data from a file, it takes a `bytes` object containing serialized data, such as the one returned by the `pickle.dumps()` function.

④ The end result is the same: a perfect replica of the original dictionary.