Opening Text Files

To open a file, you need to open the file in read, write or append mode.

```
file = open(filename, mode)
```

Substitute `mode` for the mode below

Mode	Description
r	This mode will be open for reading only
w	This mode will be open for writing only. If file containing that name does not exist, it will create a new one
a	This mode will append to the previous output of that file
r+	This mode will be open for both reading and writing

Reading from Text Files

To read from a file use

```
file.method()
```

Substitute `.method()` for the method you want to use on the file

Method	Description
`.read()`	This method reads the entire file
`.readline()`	This method reads one line of the file
`.readlines()`	This method returns a list of lines from the file
`.close()`	Closes file

Writing to Text Files

To write to a file use

```
file.method()
```

Substitute `.method()` for the method you want to use on the file. Substitute *`data`* for the data you want to write to the file

Method	Description
`.write(data)`	This method writes *data* to a file overwriting existing data
`.writelines(data)`	This method writes *data* as a list of strings to the file overwriting existing data
`.append(data)`	This method appends *data* to the file instead of overwriting.
`.close()`	Closes file

Opening Binary Files

To open a binary file, you need to open the file in read, write or append mode.

```
file = open(filename, mode)
```

Substitute `mode` for the mode below

Mode	Description
rb	This mode will be open for reading only
wb	This mode will be open for writing only. If file containing that name does not exist, it will create a new one
ab	This mode will append to the previous output of that file
rb+	This mode will be open for both reading and writing

Reading from Binary Files

To read from a file use

```
file.method()
```

Substitute `.method()` for the method you want to use on the file

Method	Description
`.read()`	This method reads the entire file
`.readline()`	This method reads one line of the file
`.readlines()`	This method returns a list of lines from the file
`.close()`	Closes file

Writing to Binary Files

To write to a file use

```
file.method()
```

Substitute `.method()` for the method you want to use on the file.

Method	Description
.write(*bytearray*)	This method writes *bytearray* to a file
.writelines(*bytearra*)	This method writes *bytearray* as a list of strings to the file overwriting existing data
.append(*bytearra*)	This method appends *bytearray* to the file instead of overwriting the file.
.close()	Closes file

Substitute *`bytearray`* for the data you want to write to the file.

Note that you need to write bytes to a binary file not integers or text strings, otherwise you'll get an error when you run the program. To do this add 'b' before a string, such as:

```
b"String"
```

If you're using integers use a method called `.to_bytes()` to convert it

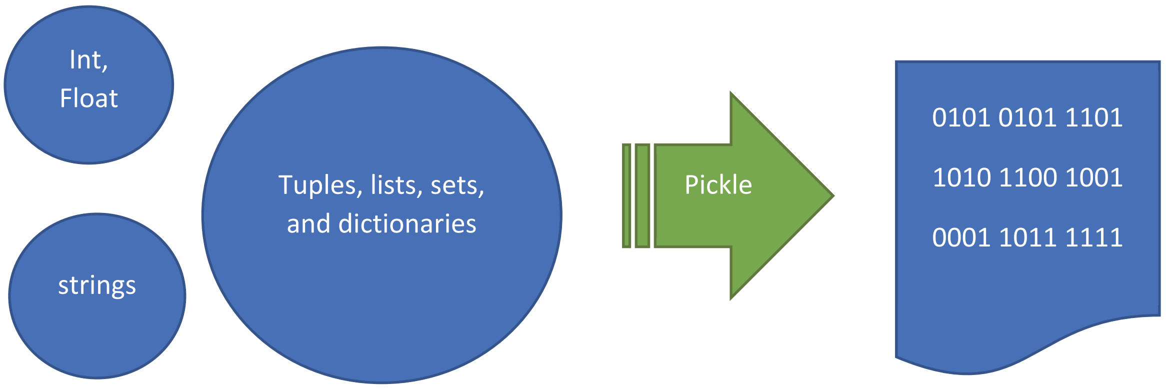
```
num=4
binarynumbertowrite = num.to_bytes(length=1, byteorder='little')
```

`Length` is the number of bytes

Now you'll be able to use the file methods in binary mode.

Using Pickle to Write Binary Files

To write your data in binary format, a better method than the one used above, is called pickling. Pickling is the process where data objects such as integers, strings, lists, and dictionaries are converted into a byte stream.



Import the following module at the top of your program.

```
import pickle
```

Open your file as normal

```
file = open(filename, mode)
```

To load a file use the `pickle.load()` method

```
pickle.load (file-to-read-from)
```

For example

```
datafromfile = pickle.load(file)
```

To write to a file, use the `pickle.dump()` method

```
pickle.dump (data-to-be-written, file-to-write-to)
```

For example

```
pickle.dump ("Data to be written", file)
```