Arguments to closures in Swift are passed in the same way as arguments to functions. To pass an argument to a closure, you simply list it inside the parentheses after the closure's opening brace. For example, the following code defines a closure that takes two arguments, `a` and `b`, and returns their sum:

```Swift
let add = {(a: Int, b: Int) -> Int in
  return a + b
}
```

To call the `add` closure, you simply pass in the arguments that you want to add:

```Swift
let sum = add(1, 2)
```

The `sum` variable will now contain the value 3, which is the sum of 1 and 2.

Arguments to closures can also be named. To name an argument to a closure, you use the `$` symbol followed by the argument's name. For example, the following code defines a closure that takes two arguments, `$a` and `$b`, and returns their sum:

```Swift
let add = {($a: Int, $b: Int) -> Int in
  return $a + $b
}
```

To call the `add` closure with named arguments, you simply list the argument names followed by the argument values:

```Swift
let sum = add(a: 1, b: 2)
```

The `sum` variable will still contain the value 3, which is the sum of 1 and 2.

Named arguments can be useful for making your code more readable and maintainable. For example, if you have a closure with many arguments, it can be helpful to name the arguments so that it is clear what each argument is for.

Here are some additional tips for passing arguments to closures:

- You can pass any type of value to a closure, including arrays, sets, and dictionaries.
- You can pass multiple arguments to a closure.
- You can pass named arguments to a closure.
- You can pass closures as arguments to other functions.

I hope this helps!