Partial

February 19, 2026

1 Partial Functions in Python

A partial is a function that is created by fixing a certain number of arguments of another function. Python provides a built-in module called **functools** that includes a function called **partial** that can be used to create partial functions. The partial function takes a callable (usually another function) and a series of arguments to be pre-filled in the new partial function. This feature is similar to bind in C++.

Partial functions support both positional and keyword arguments to be used as fixed arguments.

In the functional programming world, partial application is referred to as **currying**.

1.0.1 Example 1

In this example, we use default values to implement the partial function. The default values start replacing variables from the left. In the example, we have pre-filled our function with some constant values of a, b, and c. And g() just takes a single argument i.e. the variable x.

[1]:
```
from functools import partial

# A normal function
def f(a, b, c, x):
    return 1000*a + 100*b + 10*c + x

# A partial function that calls f with
# a as 3, b as 1 and c as 4.
g = partial(f, 3, 1, 4)

# Calling g()
print(g(5))
```

```
3145
```

1.0.2 Example 2

In the example, we have used pre-defined value constant values in which we have assigned the values of c and b and add_part() takes a single argument i.e. the variable a.

[1]:
```
from functools import partial

```

```
# A normal function
def add(a, b, c):
    return 100 * a + 10 * b + c

# A partial function with b = 1 and c = 2
add_part = partial(add, c = 2, b = 1)

# Calling partial function
print(add_part(3))
```

```
312
```

1.0.3 Uses of partial functions

- Integration with Libraries: Partial functions can be used to customize the behavior of third-party functions or methods by providing partial arguments and can be used to integrate it with other libraries.
- Simplifying Callbacks: Partial functions can be used to create specialized callback handlers by fixing some callback-specific parameters and providing a cleaner interface for the rest of the code.
- Parameter Fixing: Partial functions can be very useful when we have a function with multiple parameters and we frequently want to use it with some parameters fixed. Instead of repeatedly passing those fixed parameters, we can create a partial function and call it with the remaining arguments.
- Reducing Duplication: If we are using the same arguments for a function in various places, creating a partial function with those fixed arguments can help to reduce code duplication and maintenance efforts.
- Default Arguments: Python's built-in functools.partial can be used to set default values for function arguments.
- Reusability of code: Partial functions can be used to derive specialized functions from general functions and therefore help us to reuse our code.

[ ]: