```
# Suppose we want to find 2 * 3 * 5 * 7 * … up to 100
```

```
def                 pass  # Some implementation
```

```
# Extract all the primes
```

```
                    for                     in            2  100
```

```
if
```

```
# primes == [2, 3, 5, …]
```

```
print              *primes    # equiv. to product(2, 3, 5, …)
```

Unpacking Variadic Positional Arguments

```
# Suppose we want to find 2 * 3 * 5 * 7 * … up to 100
def is_prime(n):  pass  # Some implementation

# Extract all the primes
primes = [number for number in range(2, 100)
          if is_prime(number)]

# primes == [2, 3, 5, …]
print(product(*primes))  # equiv. to product(2, 3, 5, …)
```

The syntax `*seq` unpacks a sequence into its constituent components

Variadic Keyword Arguments