

Functions

Defining Functions

```
def my_function():  
    #Do this  
    #Then do this  
    #Finally do this
```



Functions with Arguments


- Arguments:
 - Information can be passed into functions as arguments.
 - Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.
-



Number of Arguments

- By default, a function must be called with the correct number of arguments.
- Meaning that if your function expects 2 arguments, you have to call the function with 2 arguments, not more, and not less.
-

```
def my_function(fname, lname):  
    print(fname + " " + lname)  
  
my_function("Emil", "Refsnes")
```



Arbitrary Arguments, *args

- If you do not know how many arguments that will be passed into your function, add a * before the parameter name in the function definition.
- This way the function will receive a tuple of arguments, and can access the items accordingly:

```
def my_function(*kids):  
    print("The youngest child is " + kids[2])  
  
my_function("Emil", "Tobias", "Linus")
```

Default Parameter Value

- The following example shows how to use a default parameter value.
- If we call the function without argument, it uses the default value

```
def my_function(country = "Norway"):  
    print("I am from " + country)
```

```
my_function("Sweden")  
my_function("India")  
my_function()  
my_function("Brazil")
```

Interactive coding task

- Write down a function which accepts a string and reverses a string

Comparing For Loop and While Loop

For

```
for item in list_of_items:  
    #Do something to each item
```

```
for number in range(a, b):  
    print(number)
```

While

```
while something_is_true:  
    #Do something repeatedly
```

Interactive Coding Task

- **Factorial of a Number:**

- Write a program to calculate the factorial of a given number using a while loop.