Defining a function

```
print("Peel off the bananas")
print("Add some milk to it")
print("Add some sugar")
print("Add some dry fruits")
print("Top up with ice cream")
```

```
Peel off the bananas
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```
print("Peel off the bananas")
print("Add some milk to it")
print("Add some sugar")
print("Add some dry fruits")
print("Top up with ice cream")
```

```
Peel off the bananas
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```
print("Peel off the bananas")
print("Add some milk to it")
print("Add some sugar")
print("Add some dry fruits")
print("Top up with ice cream")
```

```
Peel off the bananas
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```
def banana_shake():
    print("Peel off the bananas")
    print("Add some milk to it")
    print("Add some sugar")
    print("Add some dry fruits")
    print("Top up with ice cream")
banana_shake()
```

```
Peel off the bananas
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```
banana_shake()
```

```
Peel off the bananas
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```
banana_shake()
```

```
Peel off the bananas
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```
def tea():
    print("Make it yourself")
```

```
tea()
tea()
tea()
for i in range(10):
    tea()
```

```
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
Make it yourself
```

```
for i in range(10):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
```

```
mango_shake()
```

```
----------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
/var/folders/zn/hkv6562d6_d30glfs8yc76900000gn/T/ipykernel_5187/279465
8734.py in <module>
----> 1 mango_shake()

NameError: name 'mango_shake' is not defined
```

Passing a parameter to the function

- Do you want to make different functions for different fruit shakes?

```
# fruit shake

# fruit is a variable here and is known as parameter
def fruit_shake(fruit):
    print("Peel off the", fruit)
    print("Add some milk to it")
    print("Add some sugar")
    print("Add some dry fruits")
    print("Top up with ice cream")
```

```
fruit_shake("mango")
```

```
Peel off the mango
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```
fruit_shake("Orange")
```

```
Peel off the Orange
Add some milk to it
Add some sugar
Add some dry fruits
Top up with ice cream
```

```python
# intro

def intro(name):
    print("My name is", name)

intro("Rahul")
```

```
My name is Rahul
```

```python
## propose

def propose(name):
    print("Hey i love you", name)

name = input()
propose(name)
```

```
 Emma Watson

Hey i love you Emma Watson
```

```python
naam = "baby"
propose(naam)   # same as propose("baby")
```

```
Hey i love you baby
```

Multiple paramaters

- Introduce your family

```python
def family(father, mother, sibling):
    print("Name of father is", father)
    print("Name of mother is", mother)
    print("Name of sibling is", sibling)

family("papa", "mummy", "bhai")
```

```
Name of father is papa
Name of mother is mummy
Name of sibling is bhai
```

```
# Parameters are positional

family("harry", "hermoine", "james")
```

```
Name of father is harry
Name of mother is hermoine
Name of sibling is james
```

```
# Does position of parameters matter?

# family("James", "lilly")
# This will give me error as one parameter is missing
```

Docstrings

- Should I have Kept some Documentation?
- Tell me something about yourself please...

```
# add them

def add(a, b):
    """
    a: Give value to a
    b: give value to b
    Note: Don't pass int and str at a time.
    """
    print(a+b)

print(help(add))
```

```
Help on function add in module __main__:

add(a, b)
    a: Give value to a
    b: give value to b
    Note: Don't pass int and str at a time.

None
```

```python
add(5, 6)
```

```
11
```

```python
add("a", 5)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/var/folders/zn/hkv6562d6_d30glfs8yc76900000gn/T/ipykernel_5187/252735282.py in <module>
----> 1 add("a", 5)

/var/folders/zn/hkv6562d6_d30glfs8yc76900000gn/T/ipykernel_5187/200311516.py in add(a, b)
      1 def add(a, b):
----> 2     print(a+b)

TypeError: can only concatenate str (not "int") to str
```

```python
print?
```

```
Docstring:
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.
Optional keyword arguments:
file:  a file-like object (stream); defaults to the current sys.stdout.
sep:   string inserted between values, default a space.
end:   string appended after the last value, default a newline.
flush: whether to forcibly flush the stream.
Type:      builtin_function_or_method
```

```python
def multiply(a, b):
    """
    This is the docstring
    """
    print(a*b)
```

```python
print(help(multiply))
```

```
Help on function multiply in module __main__:

multiply(a, b)
    This is the docstring

None
```

```
multiply?
```

```
Signature: multiply(a, b)
Docstring: This is the docstring
File:
/var/folders/zn/hkv6562d6_d30glfs8yc76900000gn/T/ipykernel_5187/226779
5985.py
Type:      function
```

Return a function

print function shows all the values that it prints but actually it doesnt give any value

```
# lets revisit print

type(print("rahul"))
```

```
rahul

NoneType
```

```
def square(a, b):
    print(a*a, b*b)

result = square(3, 4)
```

```
9 16
```

```
print(result)
```

```
None
```

```
# Can function flow go beyond return statement

def square(x):
    return x*x
```

```
result = square(3)
```

```
print(result)
```

```
9
```

```
def square(x):
    return x*x
```

```
a = square(3)
b = square(5)
```

```
print(a + b)
```

```
34
```

```
def square(x):
    print(x*x)
```

```
a = square(3)
b = square(5)
```

```
print(a + b)
```

```
9
25
```

```
----------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
/var/folders/zn/hkv6562d6_d30glfs8yc76900000gn/T/ipykernel_5187/397880
5423.py in <module>
      5 b = square(5)
      6
----> 7 print(a + b)

TypeError: unsupported operand type(s) for +: 'NoneType' and
'NoneType'
```

```
def abc():
    print("before return")
    return 1
    print("After return")
```

```
res = abc()
```

```
before return
```

```
print(res)
```

```
1
```

Some inbuilt functions

```
# Absolute function

print(-3)
```

```
-3
```

```
print(abs(-234))
```

```
234
```

```
print(round(3.141))
```

```
3
```

```
print(round(3.14, 1))
```

```
3.1
```

```
print(round(3.14, 2))
```

```
3.14
```

```
print(round(3.141234124, 3))
```

```
3.141
```

Fahrenheit to celsius

```
# c = (5/9) * (f-32)

def fahrenheit_to_celsius(f):
    c = (5/9) * (f-32)
    return round(c, 2)
```

```
fahrenheit_to_celsius(32)
```

```
0.0
```

```
fahrenheit_to_celsius(100)
```

```
37.78
```

```python
## last quiz

def add_2_nums_with_return(n1, n2):
    return n1 + n2

y = add_2_nums_with_return(5, 6)
print(y)
```

```
11
```

```python
def abc():
    return 2, 3, 4
```

```python
res = abc()
```

```python
res
```

```
(2, 3, 4)
```

```python
def test(x, y, z):
    return x, y, z

res = test(2, 3, 4)
print(res)
```

```
(2, 3, 4)
```

Doubts