```
def filter_int(f):
    def process(x):
        if isinstance(x,int):
            return f(x)
        else:
            return 'Invalid datatype'
    return process

def square(i):
    result = i ** 2
    return result
```

```python
def filter_int(f):
    def process(x):
        if isinstance(x,int):
            return f(x)
        else:
            return 'Invalid datatype'
    return process

def square(i):
    result = i ** 2
    return result
```

filter_int and square definition

filter_int →

```python
def process(x):
    if isinstance(x,int):
        return f(x)
    else:
        return 'Invalid datatype'
return process
```

square →

```python
result = i ** 2
return result
```

```python
def filter_int(f):
    def process(x):
        if isinstance(x,int):
            return f(x)
        else:
            return 'Invalid datatype'
    return process

def square(i):
    result = i ** 2
    return result

res = filter_int(square)
print(res('Hello'))
```

filter_int function call

filter_int →

```python
def process(x):
    if isinstance(x,int):
        return f(x)
    else:
        return 'Invalid datatype'
return process
```

square →

```python
result = i ** 2
return result
```

f →

Process

res →

```python
if isinstance(x,int):
    return f(x)
else:
    return 'Invalid datatype'
```

Process

```
def filter_string(f):
    def process(x):
        if isinstance(x,str):
            return f(x) # call MakeUpper function, if x is str
        else:
            return 'Invalid data type' # for other than str
    return process   # return process function from filter_string
def MakeUpper(s):
    return s.upper()
```

```
MakeUpper = filter_string(MakeUpper)
```

```python
def filter_string(f):
    def process(x):
        if isinstance(x,str):
            return f(x) # call MakeUpper function, if x is str
        else:
            return 'Invalid data type' # for other than str
    return process   # return process function from filter_string
def MakeUpper(s):
    return s.upper()
```

```python
MakeUpper = filter_string(MakeUpper)
```

filter_string(f) →

```python
def process(x):
    if isinstance(x,str):
        return f(x) # call MakeUpp
    else:
        return 'Invalid data type'
return process   # return process
```

```
def filter_string(f):
    def process(x):
        if isinstance(x,str):
            return f(x) # call MakeUpper function, if x is str
        else:
            return 'Invalid data type' # for other than str
    return process   # return process function from filter_string
def MakeUpper(s):
    return s.upper()
```

```
MakeUpper = filter_string(MakeUpper)
```

```
def process(x):
    if isinstance(x,str):
        return f(x) # call MakeUpp
    else:
        return 'Invalid data type'
return process   # return process
```

```
return s.upper()
```

MakeUpper(s)

```python
def filter_string(f):
    def process(x):
        if isinstance(x,str):
            return f(x) # call MakeUpper function, if x is str
        else:
            return 'Invalid data type' # for other than str
    return process   # return process function from filter_string
def MakeUpper(s):
    return s.upper()
```

```python
MakeUpper = filter_string(MakeUpper)
```

```python
def process(x):
    if isinstance(x,str):
        return f(x) # call MakeUpp
    else:
        return 'Invalid data type'
return process   # return process
```

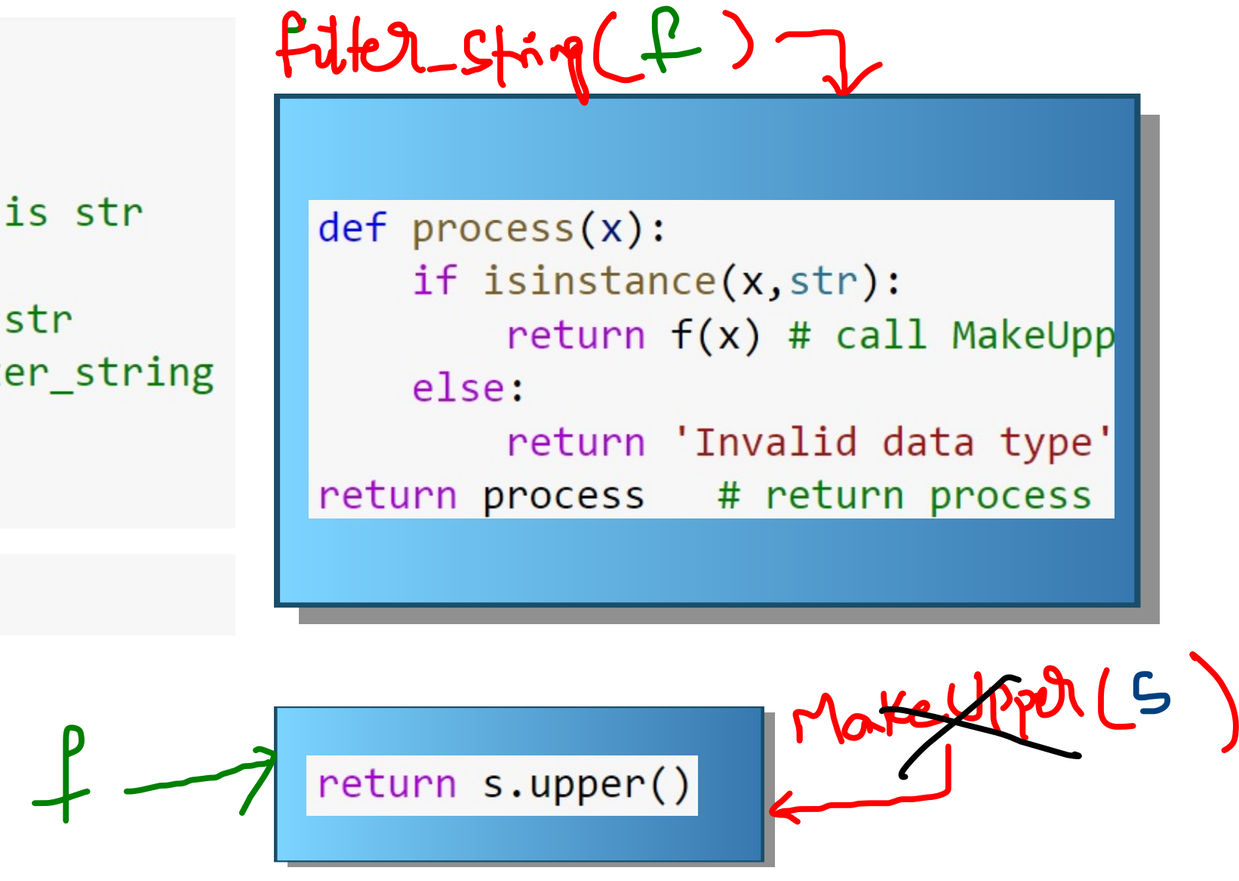
```python
return s.upper()
```

Process(x)

```python
if isinstance(x,str):
    return f(x) # call MakeUppe
else:
    return 'Invalid data type'
```

```python
def filter_string(f):
    def process(x):
        if isinstance(x,str):
            return f(x) # call MakeUpper function, if x is str
        else:
            return 'Invalid data type' # for other than str
    return process   # return process function from filter_string
def MakeUpper(s):
    return s.upper()
```

```python
MakeUpper = filter_string(MakeUpper)
```



```
def filter_string(f):
    def process(x):
        if isinstance(x,str):
            return f(x) # call MakeUpper function, if x is str
        else:
            return 'Invalid data type' # for other than str
    return process   # return process function from filter_string
def MakeUpper(s):
    return s.upper()
```

```
MakeUpper = filter_string(MakeUpper)
```

Note : MakeUpper → Process

filter_string(f)

```
def process(x):
    if isinstance(x,str):
        return f(x) # call MakeUpp
    else:
        return 'Invalid data type'
return process   # return process
```

f →

```
return s.upper()
```

MakeUpper(s)

Process(x)

MakeUpper(x)

```
if isinstance(x,str):
    return f(x) # call MakeUppe
else:
    return 'Invalid data type'
```