NPRG065: Progamming in Python

Practicals 12

MATEMATICKO-FYZIKÁLNÍ
FAKULTA
Univerzita Karlova

Department of
Distributed and
Dependable
Systems

Assignments

1. Create a function decorator that masks all errors in a function.

```
@ignore_errors
def divide(a, b):
    return a / b

print(divide(10, 2))  # returns 5
print(divide(10, 0))  # returns None
```

Assignments

2. Extend the previous decorator with specification of the return value if an exception is thrown.

```
@ignore_errors(return=0)
def divide(a, b):
    return a / b

print(divide(10, 2))  # 5
print(divide(10, 0))  # 0
```

ASSIGNMENTS

3. Create a class decorator that wraps all “public” methods (those that don’t start with “_”) and prints:
    - “`Method entry <NAME>`” at the method entry, and
    - “`Method exit <NAME>`” at the method exit.

Assignments

4. Extend the binary search tree (from the previous practicals):

   - To support indexing:

     ```
     tree = BST()
     ...
     for i in range(len(tree)):
         print(tree[i])
     ```

   - To make it callable – returns the root element
   - To be usable in conditions – an empty tree equals to false, otherwise to true