Assignment

- Extend the binary search tree (from the previous practical)
  - to support indexing
    - i.e.,

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

  - to make it callable
    - returns the tree's root element
  - to be usable in conditions
    - empty tree ~ false, otherwise true

Assignment

- Create a function decorator that masks all errors in a function

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

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

Assignment

- Extend the previous decorator such that it allows one to specify 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
```

Assignment

- Create a class decorator that wraps all “public” methods (i.e. those that don’t start with _) and prints “Method entry <NAME>”, “Method exit <NAME>” upon entry/exit in the method