Write a decorator function for your taking input for you any kind of function you want to build,

example-You make a fibonacci series function,in which your input range is been defined by the decorator program input

In [1]:

```
def memoize(f):
    cache = {}
    def decorated_function(*args):
        if args in cache:
            return cache[args]
        else:
            cache[args] = f(*args)
            return cache[args]
    return decorated_function
```

In [2]:

```
@memoize
def fib(n):
    if n ==0:
        return 0
    if n <= 2:
        return 1
    return fib(n-1) + fib(n-2)
```

In [3]:

```
fib(20)
```

Out[3]:

```
6765
```

Python program to open a file in read only mode and try writing something to it and handle the subsequent errors using Exception Handling

In [10]:

```
try:
    name = open("letsupgrade.txt","r")
    fname.write("hello world")
except:
    print("Cannot write the contents to the file")
finally:
    f.close()
    print("File closed")
```

```
Cannot write the contents to the file

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-10-aac1ab812bbd> in <module>
      5     print("Cannot write the contents to the file")
      6 finally:
----> 7     f.close()
      8     print("File closed")

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

In [11]:

```
file = open("letsupgrade.txt","r")
file.write("HI")
file.close()
```

```
---------------------------------------------------------------------------
UnsupportedOperation                      Traceback (most recent call last)
<ipython-input-11-ad1669fd3109> in <module>
      1 file = open("letsupgrade.txt","r")
----> 2 file.write("HI")
      3 file.close()

UnsupportedOperation: not writable
```

In [12]:

```
try :
    file = open("letsupgrade.txt","r")
    file.write("HI")
    file.close()
    print("Success")
except Exception as e:
    print(e)
finally:
    print("HEY I WILL EXECUTE WHT so ever it may be ")
```

```
not writable
HEY I WILL EXECUTE WHT so ever it may be
```

In [13]:

```
filename = input("Enter file name: ")

try:
    f = open(filename, "r")

    for line in f:
        print(line, end="")

    f.close()

except FileNotFoundError:
    print("File not found")

except PermissionError:
    print("You don't have the permission to read the file")

except:
    print("Unexpected error while reading the file")
```

```
import unittest
import prime

class primenumber(unittest.TestCase):
    def testprime(self):
        Number = 32
        result = prime.isprime(Number)
        self.assertEquals(result, False)


    def testprimenum(self):
        Num = 199
        res = prime.isprime(Num)
        self.assertEquals(res, True)



if __name__ == "__main__":
    unittest.main()
```

In [14]:

```
filename = input("Enter file name: ")

try:
    f = open(filename, "r")

    for line in f:
        print(line, end="")

    f.close()

except FileNotFoundError:
    print("File not found")

except PermissionError:
    print("You don't have the permission to read the file")

except:
    print("Unexpected error while reading the file")
```

```
File not found
```

In [ ]: