COMP10020
Introduction to Programming II
Python Functions

Dr. Brian Mac Namee
brian.macnamee@ucd.ie
School of Computer Science
University College Dublin

PYTHON FUNCTIONS

Function Format

def *function_name*(*arguments*)**:**

    python statements

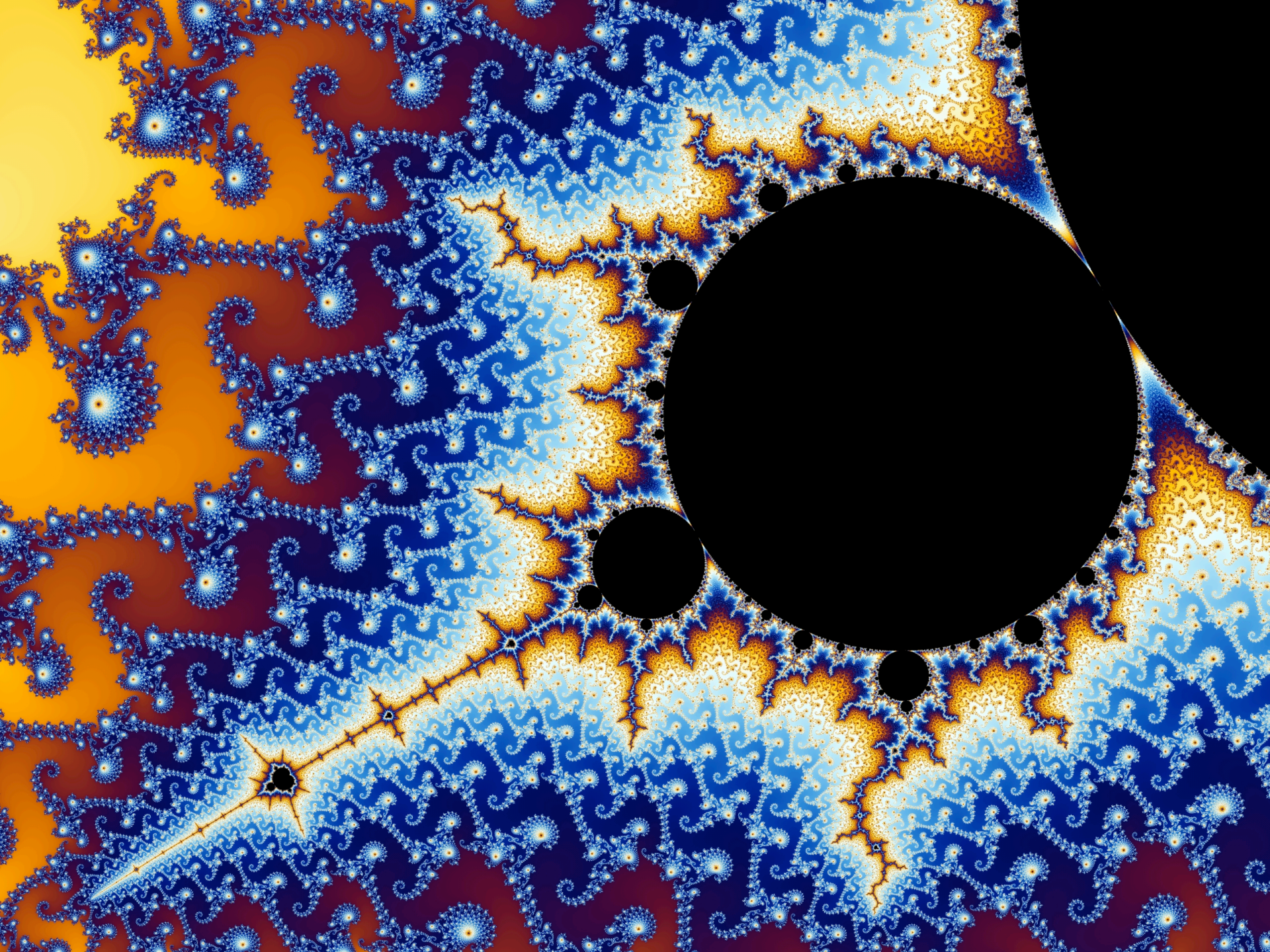
Function Format

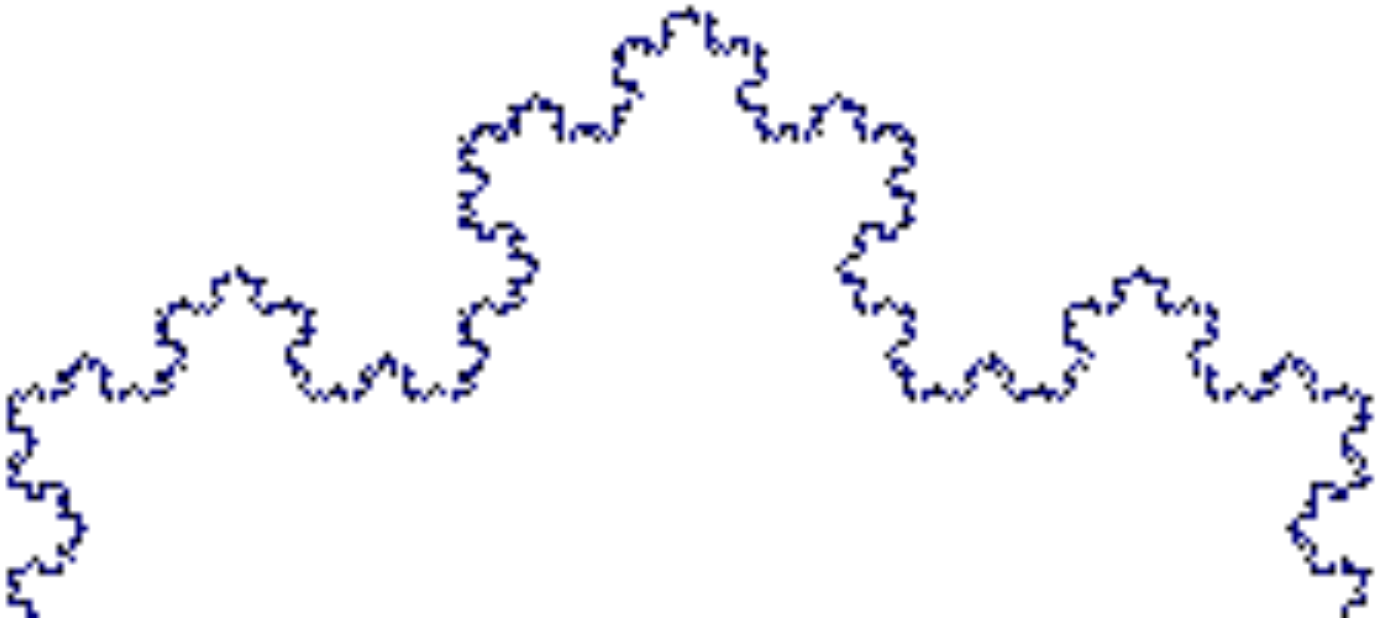
def *function_name*(*arguments*)**:**

    python statements

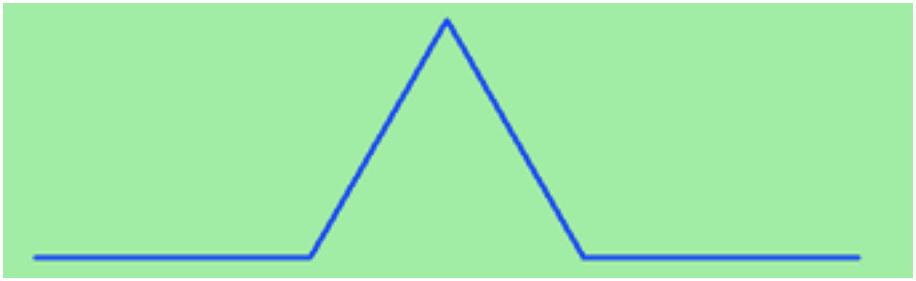
    return *variable*

RECURSION

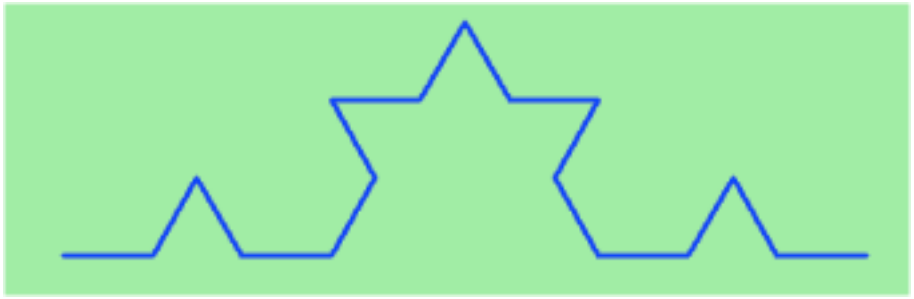




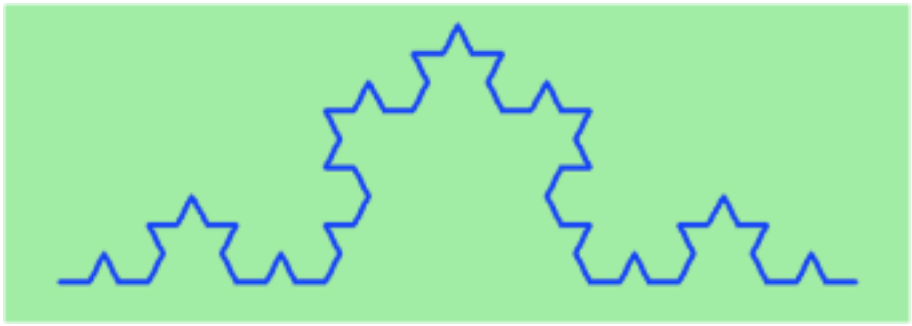
http://openbookproject.net/thinkcs/python/english3e/recursion.html



http://openbookproject.net/thinkcs/python/english3e/recursion.html



http://openbookproject.net/thinkcs/python/english3e/recursion.html



http://openbookproject.net/thinkcs/python/english3e/recursion.html

Recursive Algorithms

All recursive algorithms must have the following:

- **Base Case** - when to stop
- **Recursive Call** - call ourselves

Recursive Algorithms

Advantages of recursion

- Recursive functions lead to elegant code
- A complex task can be broken down into simpler sub-problems using recursion
- Sequence generation is typically easier with recursion than using some nested iteration

Recursive Algorithms

Disadvantages of recursion

- Sometimes the logic behind recursion can be hard to follow.
- Recursive calls are expensive (inefficient) as they take up a lot of memory and time.
- Recursive functions are hard to debug.

Example: Factorial

Use recursion to calculate the factorial of a number n:

n! = n*n-1*n-2*....*2*1

Function definition

- Base case:
- Recursive case:

Example: Factorial

Use recursion to calculate the factorial of a number n:

$$n! = n*n\text{-}1*n\text{-}2*....*2*1$$

Function definition

- Base case: factorial(1) = 1
- Recursive case:

Example: Factorial

Use recursion to calculate the factorial of a number n:

$$n! = n*n-1*n-2*....*2*1$$

Function definition

- Base case: factorial(1) = 1
- Recursive case: factorial(n) = n * factorial(n - 1)

Example: Palindrome

Use recursion to test if a word is a palindrome (the same forwards and backwards)

abba

motor

rotor

rater

redivider

Example: Palindrome

r e d i v i d e r

r e k i n d l e r

Example: Palindrome

r e d i v i d e r

r e k i n d l e r

Function definition

- Base case:

- Recursive case:

Example: Palindrome

r e d i v i d e r

r e k i n d l e r

Function definition

- Base case: a single letter **IS** a palindrome
  an empty string **IS** a palindrome
  if first and last letter do not match a word **IS NOT** a palindrome
- Recursive case:

Example: Palindrome

r e d i v i d e r

r e k i n d l e r

Function definition

- Base case: a single letter **IS** a palindrome
  an empty string **IS** a palindrome
  if first and last letter do not match a word **IS NOT** a palindrome
- Recursive case: if the first and last letter match cut them off and check if what remains is a palindrome

SUMMARY

Summary

Functions are key to breaking down complex problems into accessible pieces

Python functions are easy to define

Recursion often offers elegant solutions to problem

Use with care, however, as problems can easily arise!