HD EDU

INFO1110
Introduction to Programming

同步辅导课6

Tutor：Hilbert

HD·EDUCATION
全球累计服务用户超十万

HD EDU



HD EDU

企业介绍

海道教育（HD EDU）成立于2018年，是一家助力国际化人才成长的教育科技企业，同时也是一个力求突破语言和文化障碍、推动国际教育与文化属性相融合的新锐品牌。

海道教育已实现全球规模化发展，旗下产品已走入英国、美国、澳大利亚、新西兰等主流留学的国家40多所院校，为超30万的中国留学生提供高品质的教学服务，成为留学后市场在线教育的领军企业。

2022年，HD EDU发布首个学术诚信合规准则，HD EDU愿始终与所有用户一起，坚守学术尊严，真正意义上打造既保护用户权益又懂用户需求的教学产品，推动海外华人留学生成长与发展。

产品介绍

成立五年来，HD EDU 作为专注于留学后市场服务的后起之秀，秉持着“提升中国留学生核心竞争力”的产品理念，深度洞察留学生教育场景叠加、需求变化，研发多阶段教育产品，为用户提供多元化、高品质的一站式留学教育服务，打造出符合留学生全生命周期的三大教育产品线：HD Tutoring、HD Career、HD Offer。

HD Tutoring

学习有海道，提分更有效

节点辅导　全学期学业辅导

HD Career

让年轻人找到喜欢的工作

技能课程　资格证书　实训项目　求职方案

HD Offer

专注藤校G5级别院校申请

选校定位　文书创作　申请服务　面试辅导


HD EDU
以用户的成长为第一

HD EDU

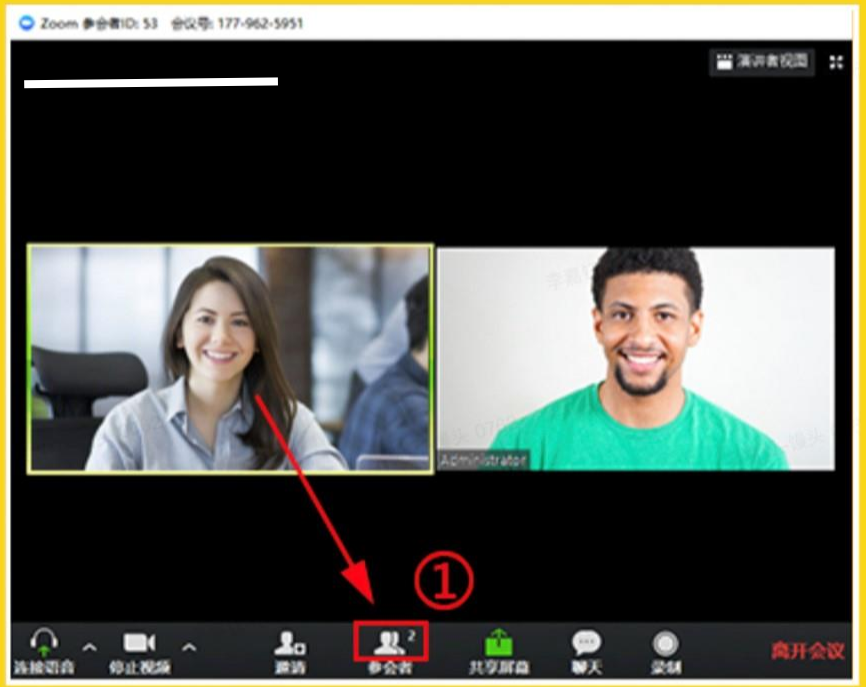


TUTOR Hilbert

Self- Introduction

- USYD Bachelor of Advanced Computing Honours First Class毕业 (Major in Software Development)
- 東京工業大学Master of Artificial Intelligence在读
- 曾任USYD INFO1110的Lab Tutor
- INFO1110总分91.0 HD
- 2018年加入HD Edu，主要教授预科及本科的IT入门课程
- HD Edu学科带头人兼面试官，曾经撰写过HD的INFO课程规范指南

同学们遇到问题怎么办？







1. 点【参会者】
2. 点【举手】即可与老师实时互动
3. 问题被解答了还可以【手放下】
4. 点【聊天】/【Chat】就可以与老师实时文字互动哦

直播平台互动方法

直播平台 ： 举手+聊天室提问



1. 点【参会者】再点【举手】，即可与老师实时互动！在此输入你想问的问题
2. 问题被解答了还可以【手放下】

本章知识点

1. Recursion
2. Array and Multi-dimensional List
3. Binary Search *
4. Tree and DFS/BFS *





知识点讲解

考点1. Recursion 递归

Definition: A function that calls itself.

- Recur (v.), Recursive (adj.)

A workable recursion must include:

- **Base case(s)**: to terminate
- **Recursive case(s)**: to expand/continue

从数学的角度来看，recursion就是在一个函数的定义里又用到了这个函数自身：

$$f(x) = f(x - 1) + x$$

考点1. Recursion – Fibonacci Series

HD EDU

In mathematics, the Fibonacci sequence is a sequence in which each number is the sum of the two preceding ones.

```
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...
```

$$
\begin{aligned}
fibonacci(n+2) &= fibonacci(n) + fibonacci(n+1) \\
fibonacci(0) &= 0 \\
fibonacci(1) &= 1
\end{aligned} \tag{1}
$$

Write a program `fibonacci.py` that receives a number `n` from the user and print the n-th term of fibonacci sequence to the screen. (Assume the user will always enter an non-negative number).

Examples:

```
$ python3 fibonacci.py
n = #0
fibonacci(0) = 0

$ python3 fibonacci.py
n = #1
fibonacci(0) = 1

$ python3 fibonacci.py
n = #6
fibonacci(0) = 8
```

考点1. Recursion – Fibonacci Series

HD EDU

Fibonacci Sequence = 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

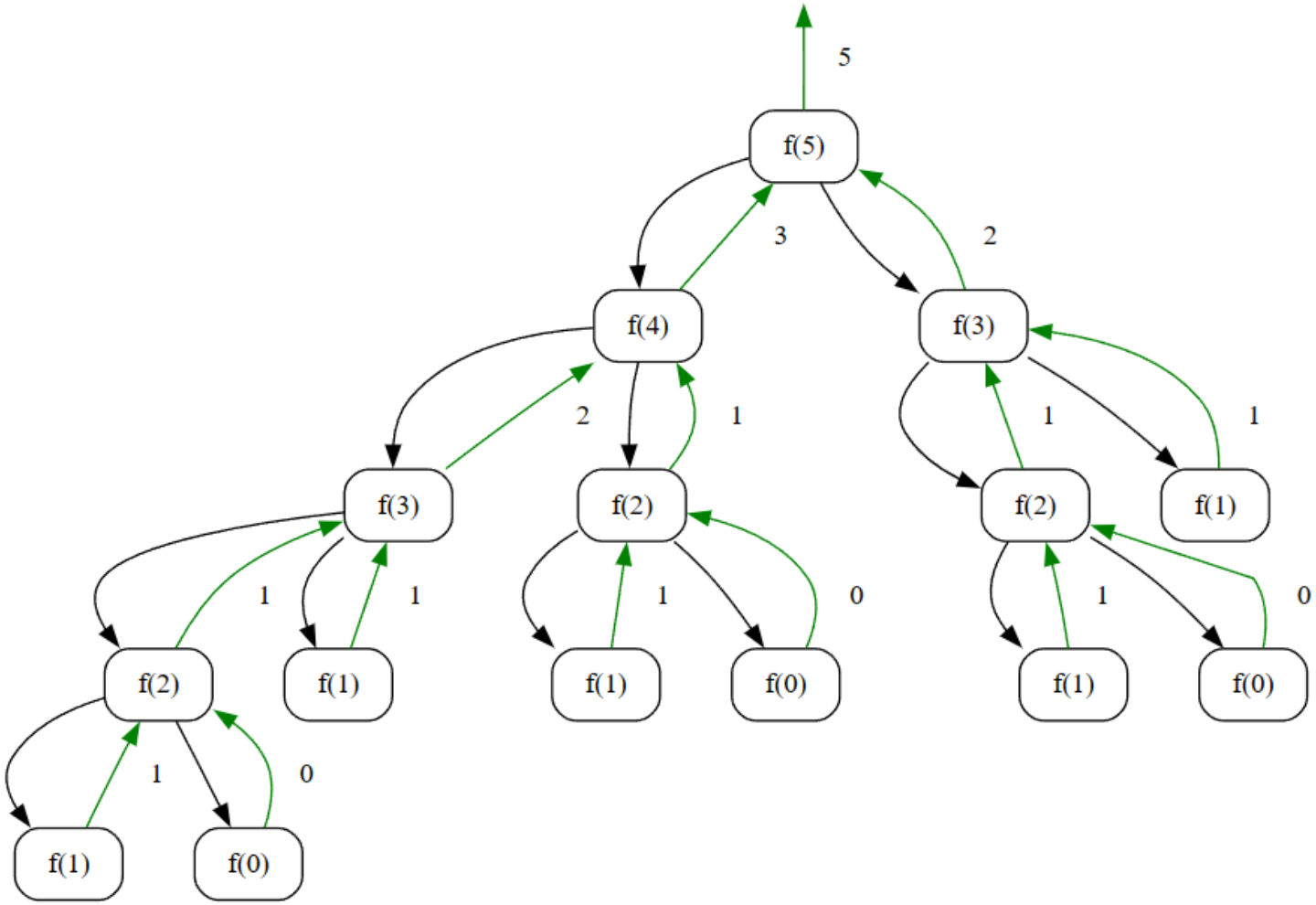
Write a recursive function fibonacci() that returns the n^{th} fibonacci number.

$$f(0) = 0 \qquad f(1) = 1 \qquad f(n) = f(n-1) + f(n-2)$$

```
def fibonacci(n):
  if n == 0:
    return 0
  if n == 1:
    return 1
  return fibonacci(n - 1) + fibonacci(n - 2)
```

```
def fibonacci_while(n):
  a = 0
  b = 1
  i = 0
  while i < n:
    a, b = b, a + b
    i += 1
return a
```

考点1. Recursion – Call Stack



考点1. Recursion – Pros and Cons

每一个能用Recursion解决的问题，都一定有Iterative（循环）的解法。

Pros:

- Elegant (Clean and more readable)
- A unique way of implementing a variable number of nested loops

Cons:

- Slower
- More memory used

考点1. Recursion – Tech 面试真题

HD EDU

```python
def f(n):
  if n < 0:
    return "n cannot be less than 0"
  elif n == 0:
    return 0
  elif n == 1:
    return 1
  else:
    return f(n - 1) + f(n - 2)



print(f(10))
```

考点1. Recursion – Tech 面试真题

HD EDU

```
def f(n):
  if n < 0:
    return "n cannot be less than 0"
  elif n == 0:
    return 0
  elif n == 1:
    return 1
  else:
    return f(n - 1) + f(n - 2)



print(f(10))
```

```
def f(n, memo={}):
    if n < 0:
        return "n cannot be less than to 0"
    elif n == 0:
        return 0
    elif n == 1:
        return 1
    elif n in memo:
        return memo[n]
    else:
        memo[n] = f2(n-1) + f2(n-2)
        return memo[n]

print(f(10))
```

考点2. Array vs. List

HD EDU

❖ 根据往届的情况，numpy和array不会成为考察的重点。← Cheatsheet

Array:

- An array is a fixed-size, contiguous memory block that stores elements of the same data type.
- Elements in an array are accessed using an index or a subscript that represents their position in the array. The index typically starts at 0.
- Arrays have a fixed size and type, meaning you need to know the number of elements you want to store in advance, and that size cannot be changed during runtime.

List:

- A list is a dynamic data structure that can grow or shrink in size during runtime.
- Also use index operator to access
- Lists are more flexible than arrays because you can easily add or remove elements of different datatypes.

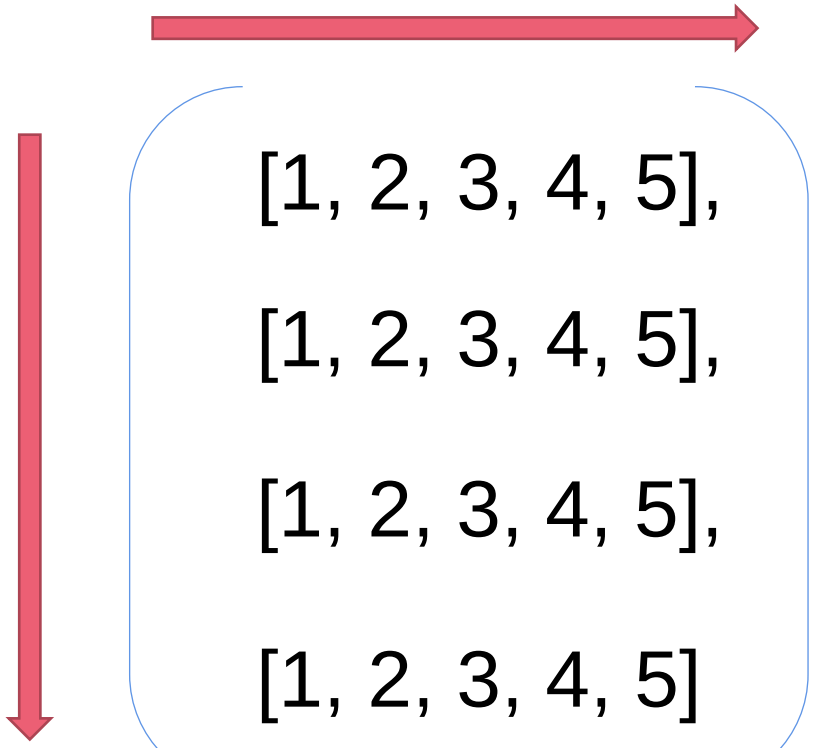
考点2. Numpy Array

HD EDU

- Array is the most primitive collection data type in most programming language (but not Python😂)
- `import numpy as np`
- `np.array()` 可以把一个collection变成array
- `np.ones()` and `np.zeros()`
- `array.shape/array.dtype/array.ndim/array.size`

```
>>> a = np.array([[1, 2], [3, 4]])
>>> a
array([[1, 2],
       [3, 4]])
>>> a.append(12)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'numpy.ndarray' object has no attribute 'append'
>>> a[0]
array([1, 2])
>>> a[0][1]
2
>>> a[0, 1]
2
>>> a[0, 1] = 3
>>> a
array([[1, 3],
       [3, 4]])
>>> a.shape
(2, 2)
>>> a.dtype
dtype('int64')
>>> a.ndim
2
>>> a.size
4
```

考点2. Multi-dimensional Traversal



matrix

- 有多少维就要有几层nested循环

```
i = 0
while i < len(matrix):
  j = 0
  while j < len(matrix):
    print(matrix[i][j])
    j += 1
  i += 1
```

考点2. Vector, Matrix and Tensor

HD EDU

❖ Vector -> 1D

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

❖ Matrix -> 2D

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

❖ Tensor -> nD

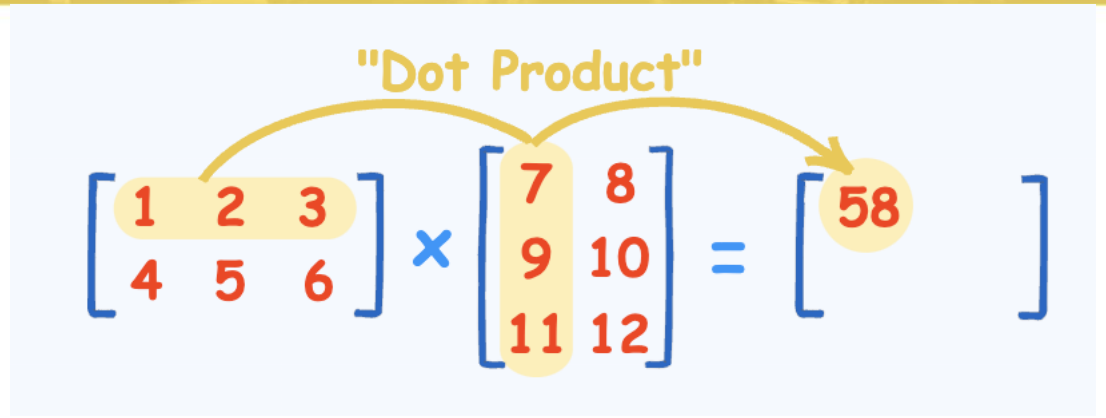
Notation: Tensors are often represented with multi-index notation. For example, a third-order tensor **T** might be written as:

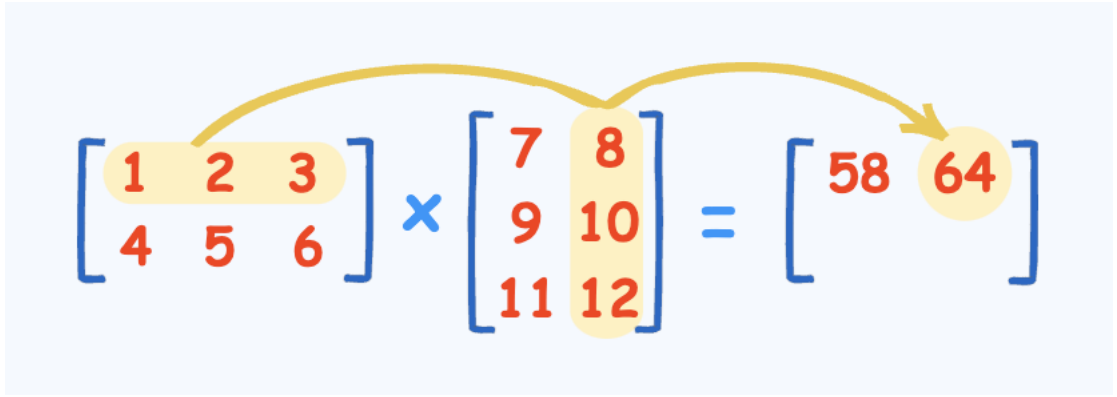
$$T_{ijk}$$

where i, j, k are dimensions of the tensor.

TensorFlow

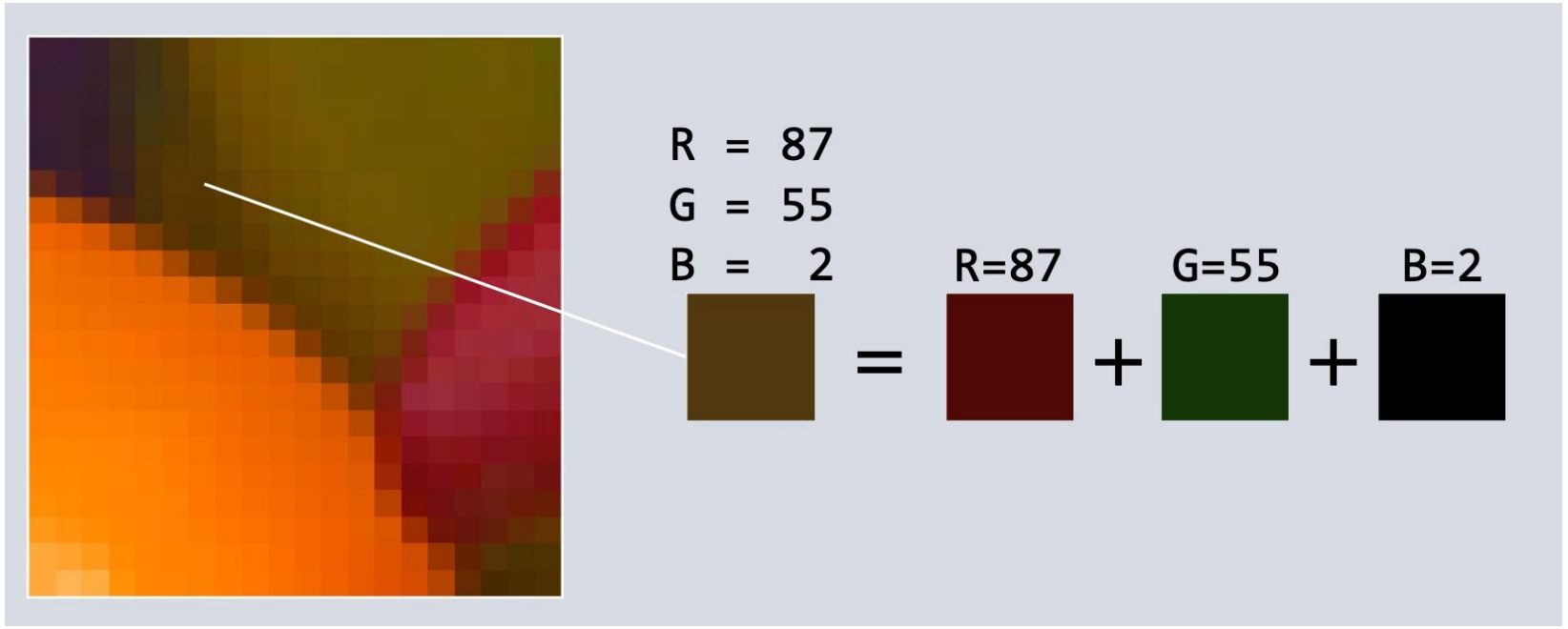
考点2. Matrix Multiplication





$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix}$$

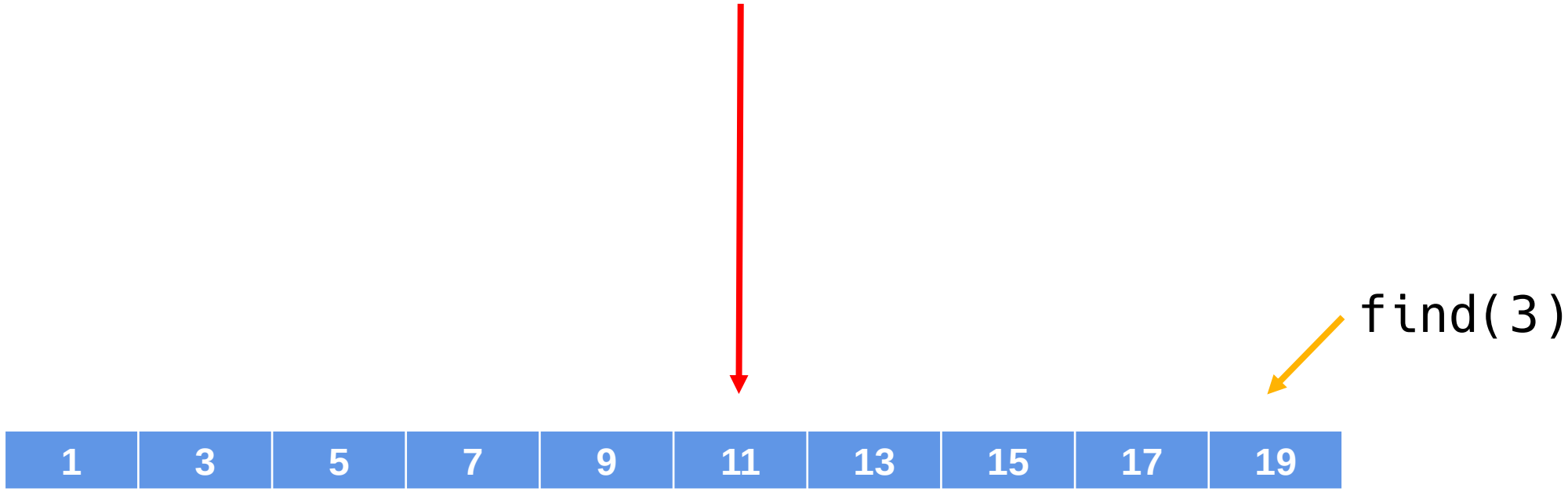
考点2. Image



- SRGB -> Standard RGB [0, 255]
- Three Channels -> Red, Green, Blue

考点3. Binary Search *

- 如果对于一个乱序的Array，没有任何办法，我们必须过n个elements
- 但如果是一个sorted的Array，情况就稍许不同了：



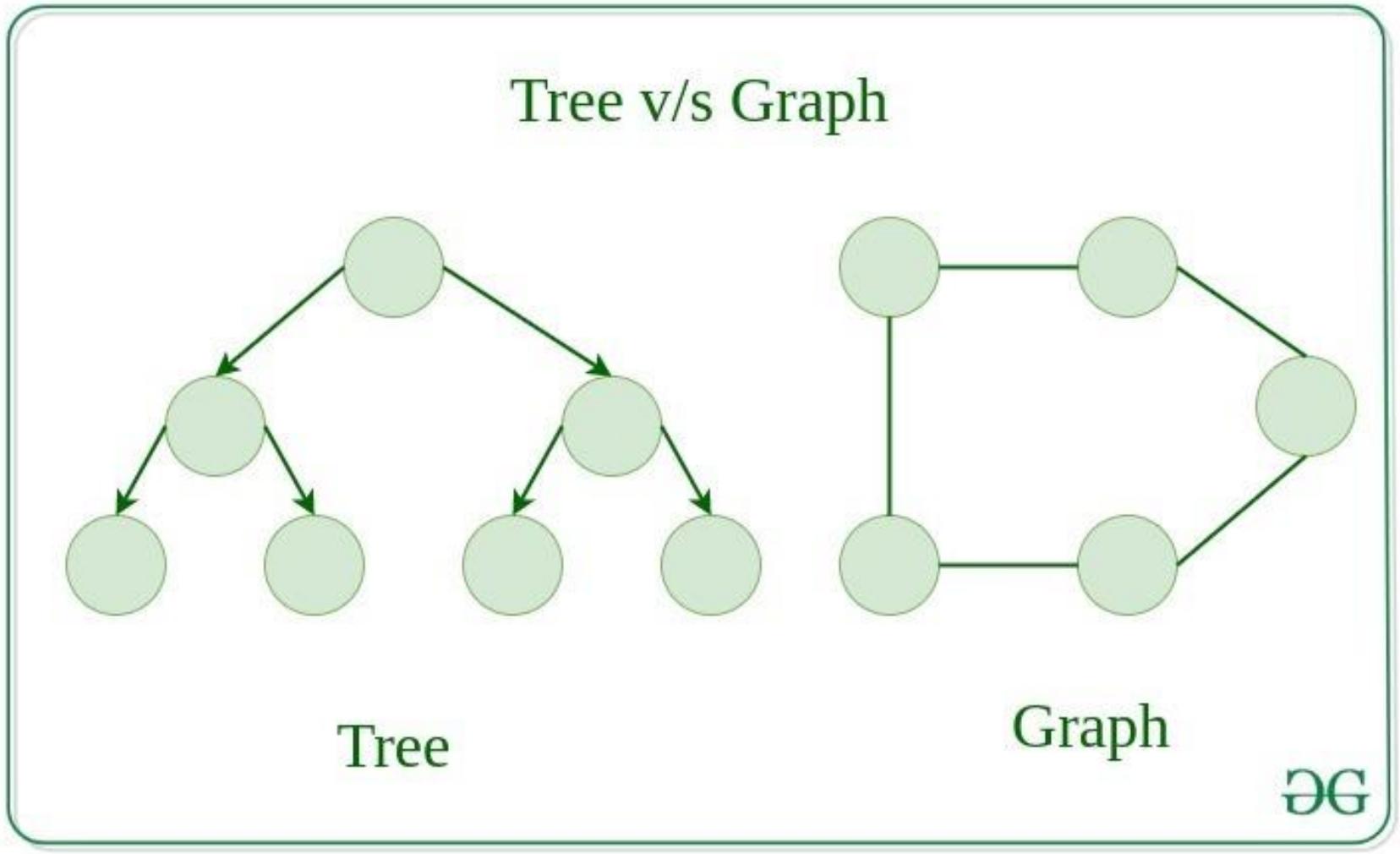
考点3. Binary Search *

HD EDU

```
def rsearch(li, n, start, end):
    index = (start + end) // 2
    if li[index] == n:
        return index
    elif li[index] > n:
        return rsearch(li, n, start, index)
    elif li[index] < n:
        return rsearch(li, n, index, end)

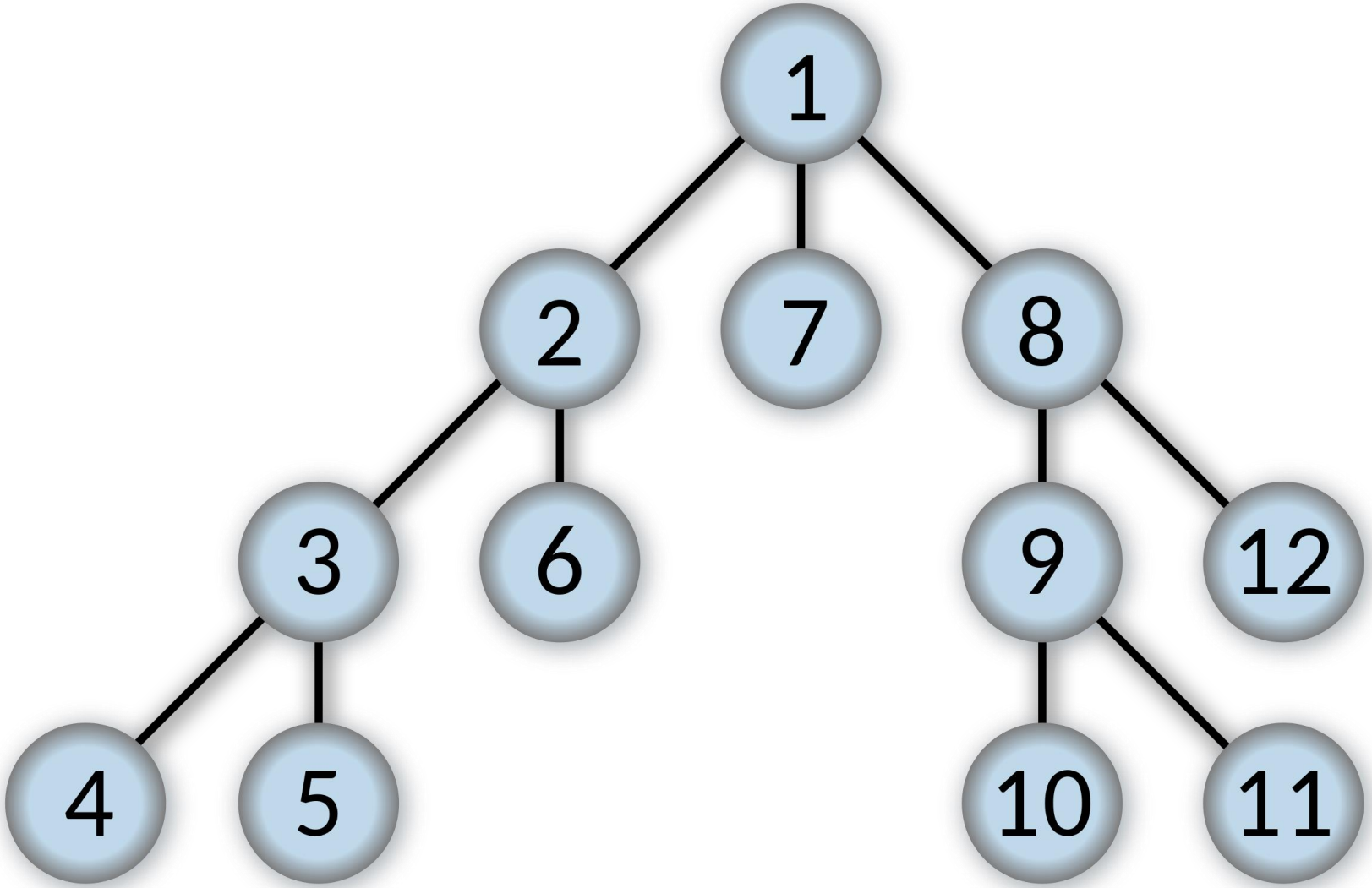
li = [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
print(rsearch(li, 13, 0, len(li)))
```

考点4. Node, Tree and Graph *

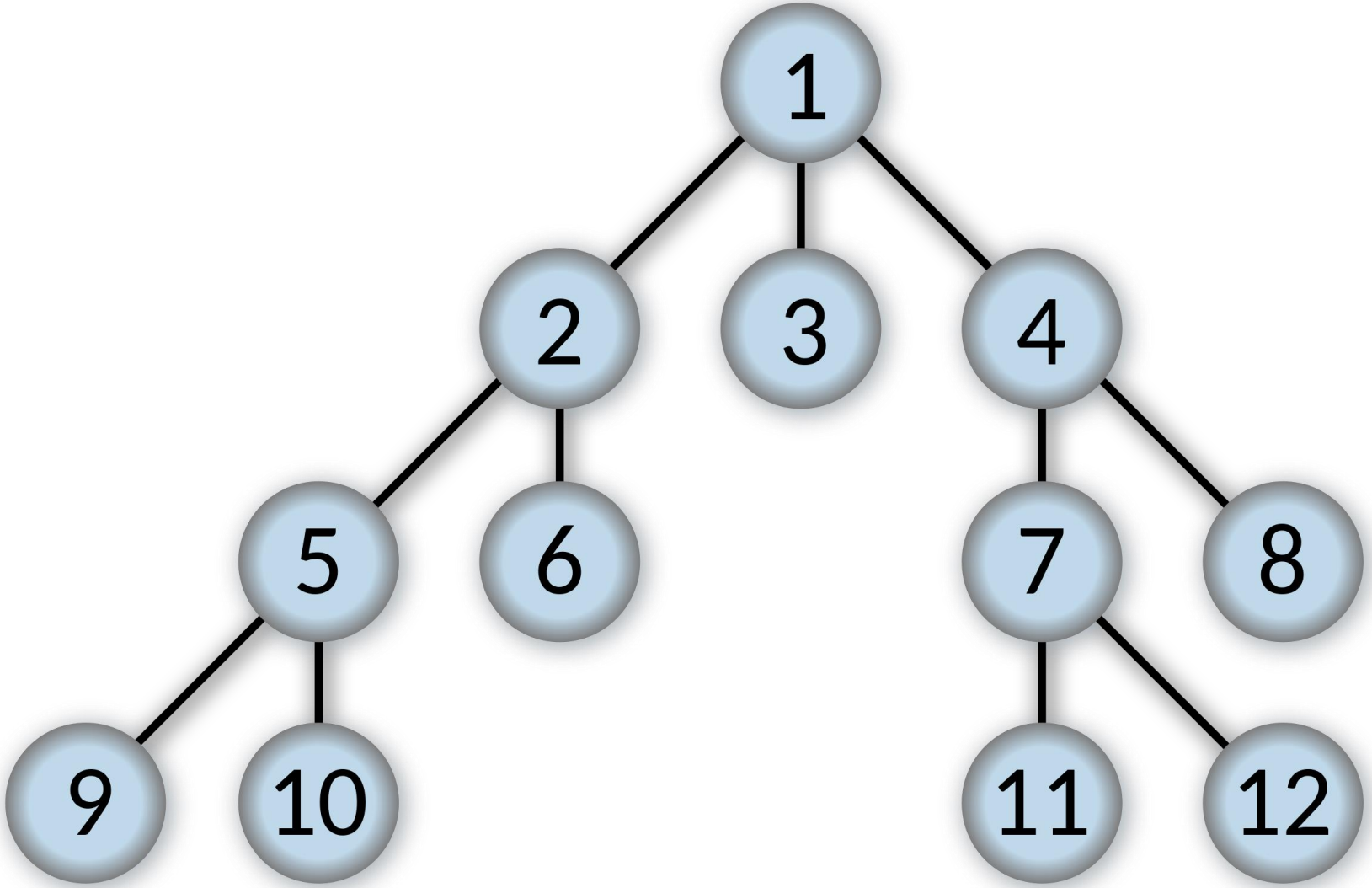


Ref. : https://www.geeksforgeeks.org/difference-between-graph-and-tree/

考点4. Depth First Search *



考点4. Breadth First Search *





经典题型放送

Find the Capital

（自编题）

HD EDU

Write a program that accepts a string and print the index of the first capital letter in the string or -1 if not exists. You are not allowed to use while and for loops.

Matrix Multiplication （自编题）

HD EDU

Write a function that accepts two matrices and perform matrix multiplication based on the definition.

Given two matrices A and B:

- A is an $m \times n$ matrix.
- B is an $n \times p$ matrix.

The product $C = A \times B$ will be an $m \times p$ matrix.

The element C_{ij} in the resulting matrix C at position (i, j) is calculated as:

$$C_{ij} = \sum_{k=1}^{n} A_{ik} \times B_{kj}$$

Where:

- A_{ik} is the element in the i-th row and k-th column of matrix A.
- B_{kj} is the element in the k-th row and j-th column of matrix B.

Find the Subsets * （自编题）

Given an array of integers, print sums of all subsets in it without using a loop. Output sums can be printed in any order.

Input: [2, 3]

Output: 0 2 3 5

Input: [2, 4, 5]

Output: 0 2 4 5 6 7 9 11

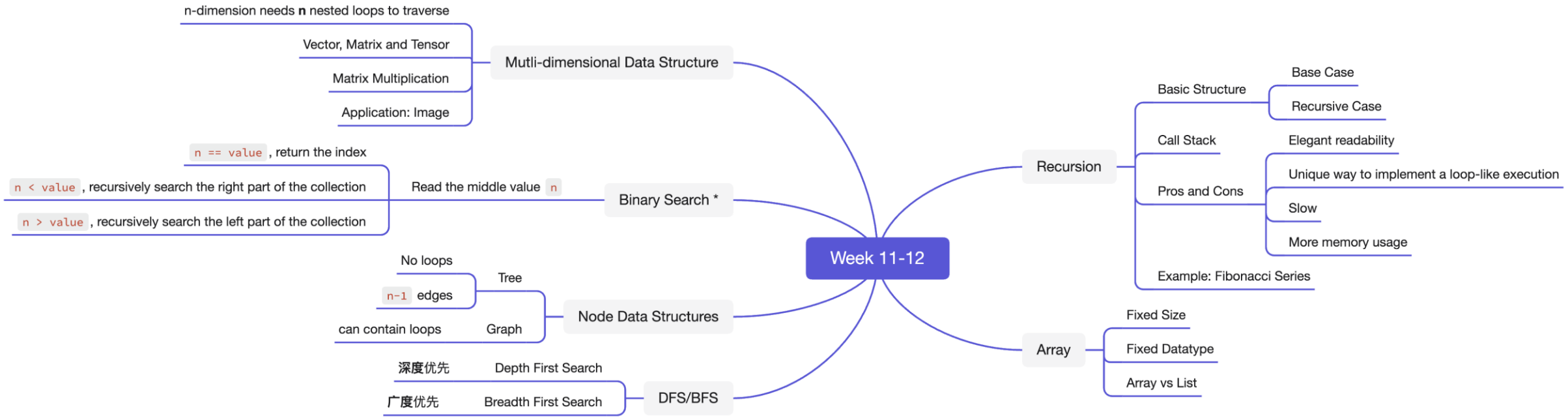
Ref. : https://www.geeksforgeeks.org/print-sums-subsets-given-set/



重难点总结

重难点总结

HD EDU


Week 11-12
Mutli-dimensional Data Structure
n-dimension needs n nested loops to traverse
Vector, Matrix and Tensor
Matrix Multiplication
Application: Image
Binary Search *
Read the middle value n
n == value, return the index
n < value, recursively search the right part of the collection
n > value, recursively search the left part of the collection
Node Data Structures
Tree
No loops
n-1 edges
Graph
can contain loops
DFS/BFS
Depth First Search
深度优先
Breadth First Search
广度优先
Recursion
Basic Structure
Base Case
Recursive Case
Call Stack
Pros and Cons
Elegant readability
Unique way to implement a loop-like execution
Slow
More memory usage
Example: Fibonacci Series
Array
Fixed Size
Fixed Datatype
Array vs List



下节课预告

下节课预告

HD EDU

- Final理论复习
  - 考试的注意事项和技巧
  - 按照Final重要考点顺序重新梳理
- Final刷题课
  - 针对自命题的考卷进行讲评



您每一份诚挚的问卷都会拯救一根濒临掉落的头发。

HD EDU



帮我开个张吧，求求你了！



课程结束后，如果您对课程或者服务的任何建议和意见

请给予我们提高和改进的机会，感谢您对 HD·EDUCATION 课程和服务的信任！

· 填写问卷操作流程 ·



第一步

关注【海道教育】服务号后
点击【购买通知】或【上课提醒】



第二步

点击【前往学习】



第三步

点击【去评价】就可以为课程进行评价



Q & A