Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professiona Learning of School Leaders and Teachers

# Leaving Certificate Computer Science

# National Workshop 6

# Workshop Overview

| | |
|---|---|
| **Session 1**<br>9:30 - 11:00 | Algorithms III – Sorting |
| **Tea/Coffee**<br>11:00 – 11:30 | |
| **Session 2**<br>11:30 - 13:00 | Evaluation & testing |
| **Lunch**<br>13:00 - 14:00 | |
| **Session 3**<br>14:00 - 16:00 | Digital portfolios and CWA video |

# Key Messages

There are many ways to use the LCCS specification.

ALTs provide an opportunity to teach theoretical aspects of LCCS.

LCCS can be mediated through a constructivist pedagogical approach.

The study of Computers and Society is one of the overarching principle of LCCS

Digital technologies can be used to enhance collaboration, learning and reflection.

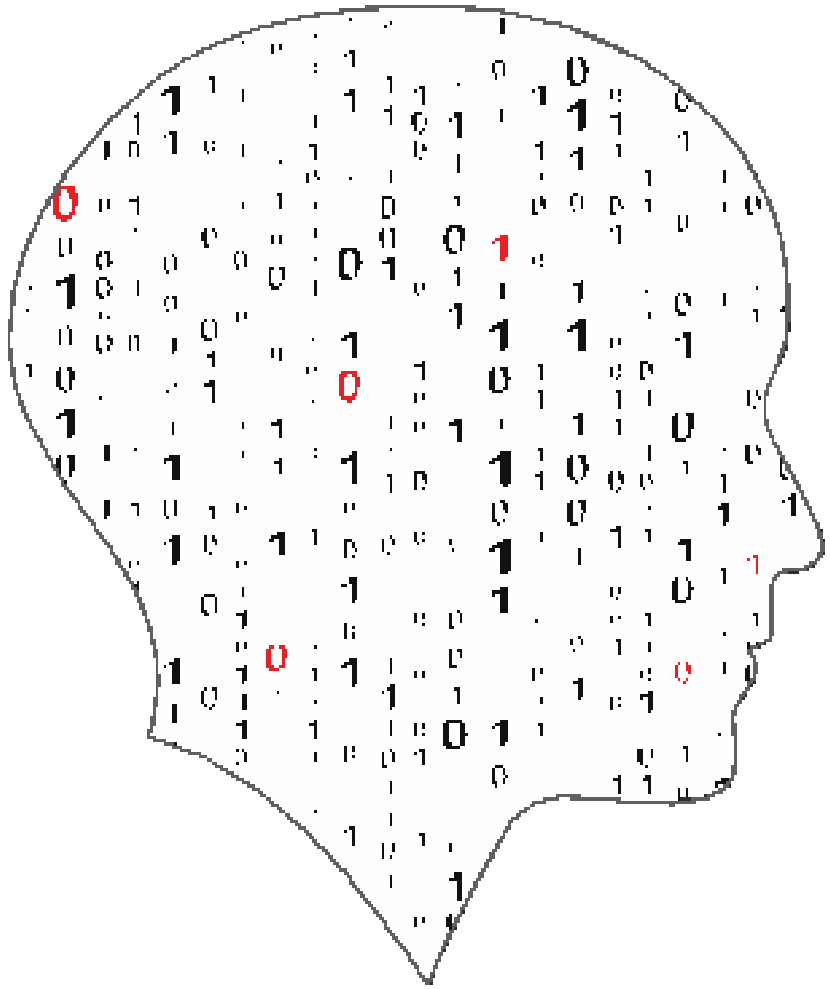
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# NW6 Session 1

## Algorithms III - Sorting



LEAVING CERTIFICATE
COMPUTER SCIENCE

# By the end of this session

Participants will have:

- reflected on the definition and characteristics of algorithms, as well as the ubiquitous nature of algorithms in today's society.
- developed their conceptual understanding of a variety of sorting algorithms.
- participated in activities to facilitate the effective learning of algorithms in their own classrooms.

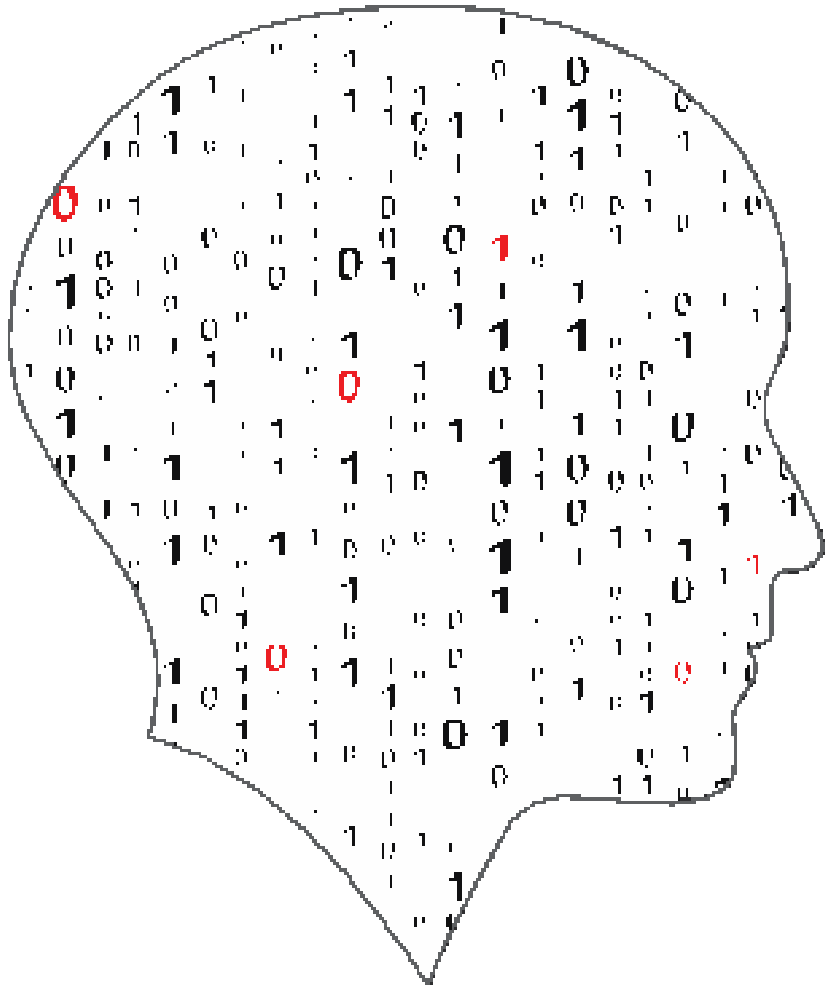
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# LCCS NW6

## Introduction to algorithms (revisited)



LEAVING CERTIFICATE COMPUTER SCIENCE

# What does the specification say?

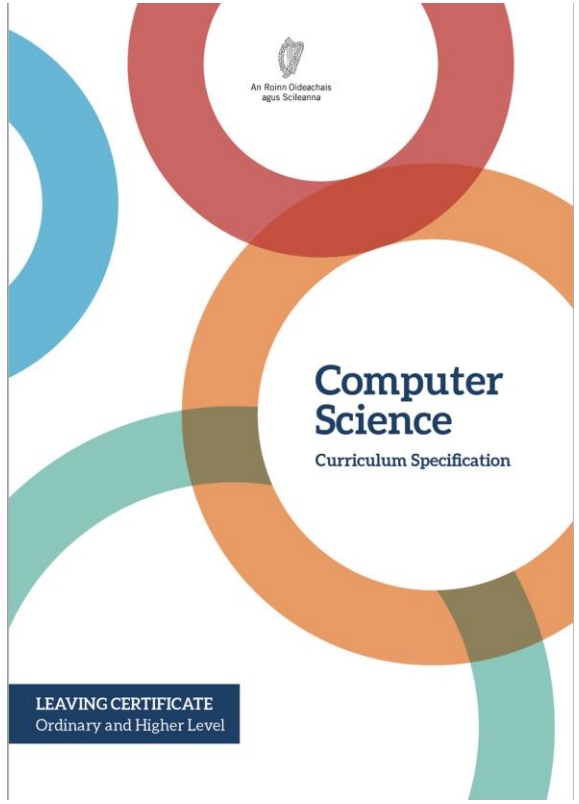
*"Computer science is the study of computers and **algorithmic processes**. Leaving Certificate Computer Science includes how programming and computational thinking can be applied to the solution of problems, and how computing technology impacts the world around us."*

LCCS Curriculum Specification, page 2

| Strand 1: Practices and principles | Strand 2: Core concepts | Strand 3: Computer science in practice |
|---|---|---|
| ▸ Computers and society<br>▸ Computational thinking<br>▸ Design and development | ▸ Abstraction<br>▸ Algorithms<br>▸ Computer systems<br>▸ Data<br>▸ Evaluation/Testing | ▸ Applied learning task 1<br>- Interactive information systems<br>▸ Applied learning task 2 - Analytics<br>▸ Applied learning task 3<br>- Modelling and simulation<br>▸ Applied learning task 4<br>- Embedded systems |

# S2: Algorithms Learning Outcomes

| Students learn about: | Students should be able to: |
|---|---|
| **S2: Algorithms** | |
| Programming concepts | 2.5 use pseudo code to outline the functionality of an algorithm |
| Sorting: Simple sort, Insert sort, Bubble sort, **Quicksort** | 2.6 construct algorithms using appropriate sequences, selections/conditionals, loops and operators to solve a range of problems, to fulfil a specific requirement |
| | 2.7 implement algorithms using a programming language to solve a range of problems |
| Search: Linear search, Binary search | 2.8 apply basic search and sorting algorithms and describe the limitations and advantages of each algorithm |
| | 2.9 assemble existing algorithms or create new ones that use functions **(including recursive)**, procedures, and modules |
| Algorithmic complexity | **2.10 explain the common measures of algorithmic efficiency using any algorithms studied** |




See also learning outcomes 1.6, 1.7 1.14, 1.22, 2.3, 3.4 and 3.7 … plus others

# What is an algorithm?

Oide

> *"An algorithm is a set of rules for getting a specific output from a specific input. Each step must be so precisely defined that it can be translated into computer language and executed by machine."*

*Source: Knuth, D The Art of Computer Programming (Vol. 1, Fundamental Algorithms, 3rd ed.)*

*Donald Knuth*

**According to Knuth, an algorithm has five important features:**

| | |
|---|---|
| **Finiteness** | An algorithm must always terminate after a finite number of steps. |
| **Definiteness** | Each step must be precisely defined. |
| **Input** | An algorithm has zero or more inputs. |
| **Output** | An algorithm has one or more outputs, which have a specified relation to the inputs. |
| **Effectiveness** | All operations to be performed must be sufficiently basic that they can in principle be done exactly and in finite length of time by someone using pencil and paper. |

# What is an algorithm?

*“ A step-by-step procedure for solving a problem or accomplishing some end especially by a computer.”*

*Merriam-Webster*

- ✓ A sequence of instructions
- ✓ A way of capturing intelligence and sharing it with others
- ✓ Provide general solutions to problems
- ✓ Some problems are so hard that they cannot be solved by algorithms (Computability)
- ✓ Can be expressed in a variety of different ways
- ✓ Common elements include input, processing, output
- ✓ Close relationship between algorithms and data structures
- ✓ Essential features are correctness and effectiveness
- ✓ Rule-based algorithms vs. Machine learning algorithms (AI)

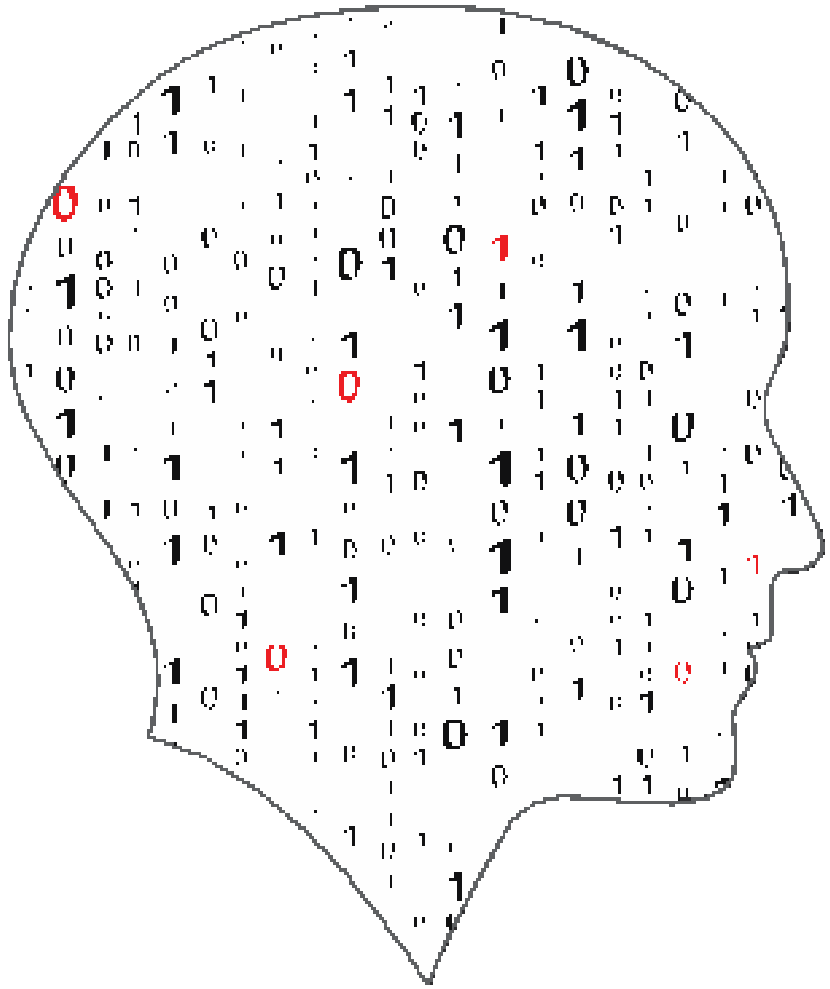
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# LCCS NW6

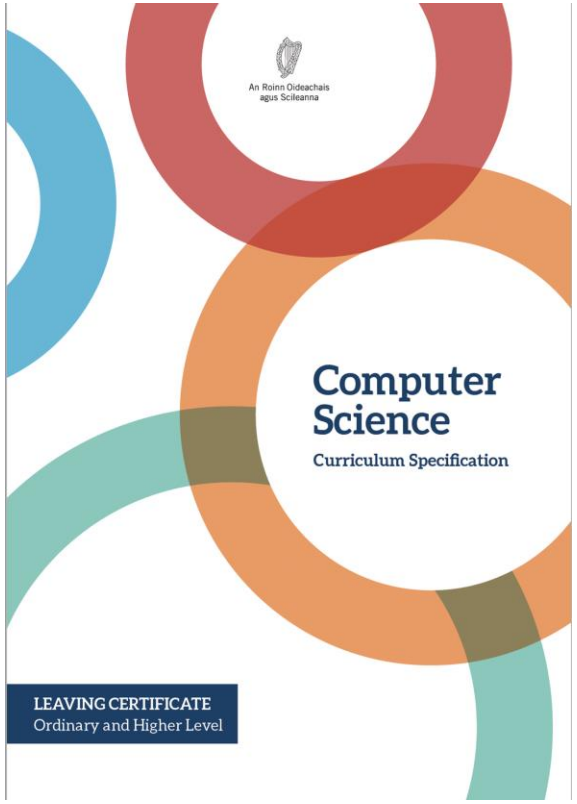
## Sorting algorithms: Selection Sort, Insertion Sort and Bubble Sort



LEAVING CERTIFICATE COMPUTER SCIENCE

# S2: Algorithms Learning Outcomes

| Students learn about: | Students should be able to: |
|---|---|
| **S2: Algorithms** | |
| Programming concepts | 2.5 use pseudo code to outline the functionality of an algorithm |
| Sorting: Simple sort, Insert sort, Bubble sort, **Quicksort** | 2.6 construct algorithms using appropriate sequences, selections/conditionals, loops and operators to solve a range of problems, to fulfil a specific requirement |
| | 2.7 implement algorithms using a programming language to solve a range of problems |
| Search: Linear search, Binary search | 2.8 apply basic search and sorting algorithms and describe the limitations and advantages of each algorithm |
| | 2.9 assemble existing algorithms or create new ones that use functions **(including recursive)**, procedures, and modules |
| Algorithmic complexity | **2.10 explain the common measures of algorithmic efficiency using any algorithms studied** |



# Sorting algorithms

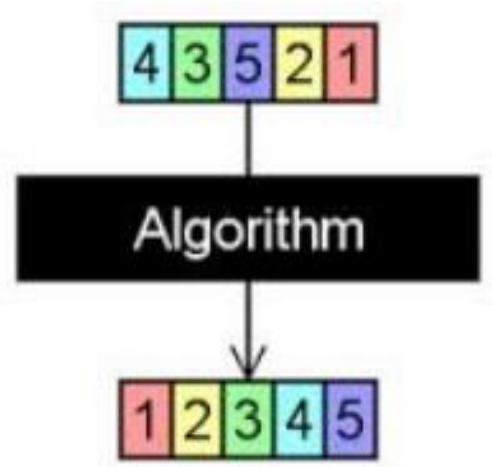
**An algorithm that maps the following input/output pair is called a sorting algorithm:**

**Input: A list (array), *L*, that contains *n* orderable elements:**

***L*[0, 1, …, *n* - *1*]**

**Output: A sorted permutation of *L*, called *S*, such that**

***S*[0] ≤ *S*[1] ≤ … ≤ *S*[*n* – 1].**




**Simple (selection) Sort**

**Insertion Sort**

**Bubble Sort**

**Quicksort**

# Simple sort demonstration

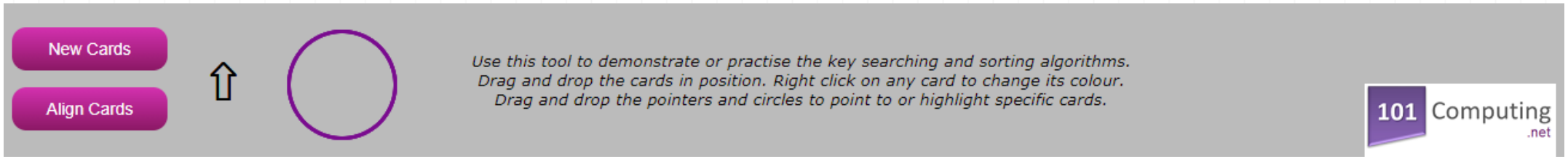
**Input:**

**Required Output:**

# Simple sort demonstration


https://www.101computing.net/card-sort/

- Search the value 69 using a linear search.
- Search the value 95 using a binary search. (Make sure to sort the list first!)
- Sort this list using an insertion sort.
- Sort this list using a bubble sort.
- Sort this list using a merge sort.
- Sort this list using a quick sort.



# Breakout Activity

Instructions:
1. Individuals read the algorithm provided and develop their own understanding (5 mins)

2. Each group then discusses and agrees a common understanding of their assigned algorithm (5 mins)

3. Groups prepare a demonstration/explanation which they will use to teach others after the breakout (5 mins)

https://www.101computing.net/card-sort/
https://deck.of.cards/

**Simple (selection) Sort**

**Insertion Sort**

**Bubble Sort**

# Breakout Activity

| | | |
|---|---|---|
| **Groups 1,4,7** | **Selection Sort** | **Pages 20-24** |
| **Groups 2,5,8** | **Insertion Sort** | **Pages 25-31** |
| **Groups 3,6,9** | **Bubble Sort** | **Pages 32-39** |

Appoint a chair, a timekeeper, a notetaker and a spokesperson

# Feedback

| | | |
|---|---|---|
| Groups 1,4,7 | Selection Sort | Pages 20-24 |
| Groups 2,5,8 | Insertion Sort | Pages 25-31 |
| Groups 3,6,9 | Bubble Sort | Pages 32-39 |

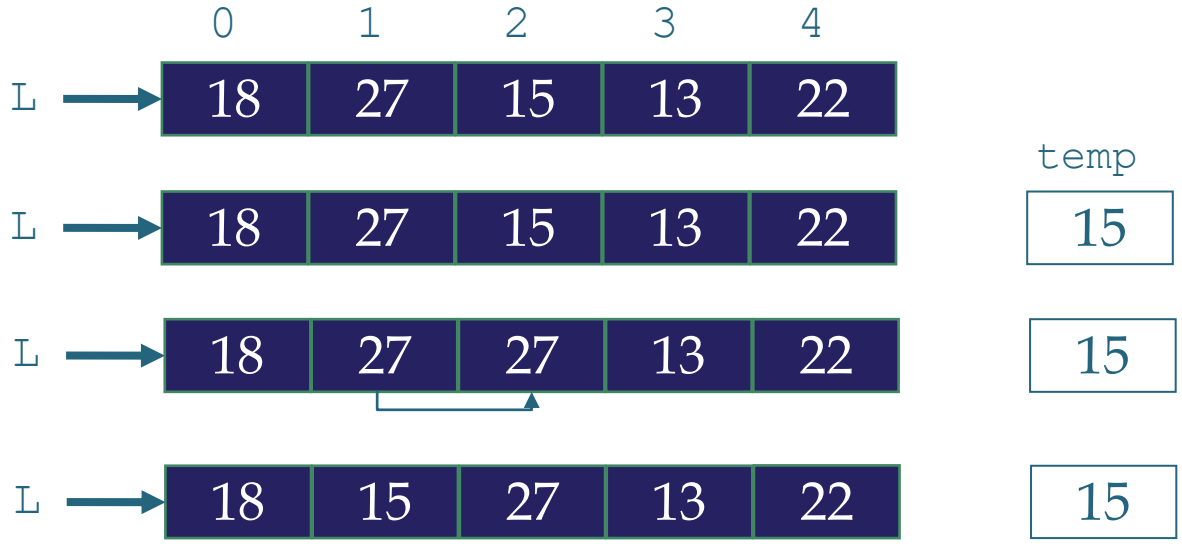
# Sorting – key skills

- ✓ **Lists/For loops**
- ✓ **Min/max (*if* statements… comparison operators)**
- ✓ **Swap operation (assignment statement)**
- ✓ **Functions**
- ✓ ***Sorted* (built-in function) Vs *sort* (list method)**

# Swap operation

**Let's say we wanted to exchange L[2] and L[1]**

```
L = [18, 27, 15, 13, 22]
```




```
temp = L[2]
```

```
L[2] = L[1]
```

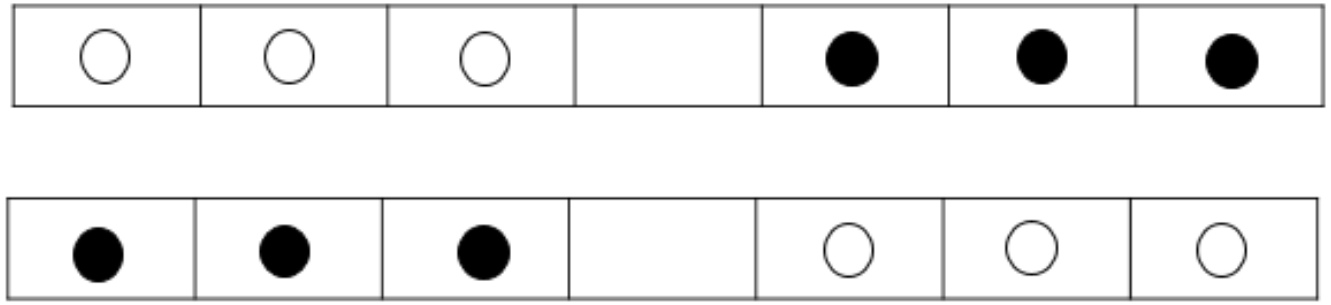
```
L[1] = temp
```

**Python supports a single statement swap:**

```
L[2], L[1] = L[1], L[2]
```

# Unplugged activity (swap)

The aim is to swap the positions of the black and white pieces.

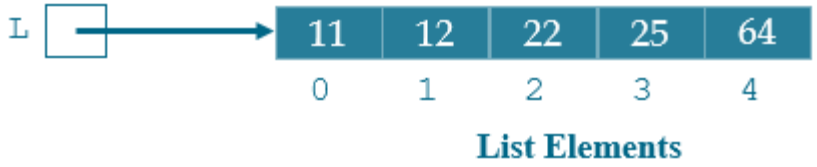


Pieces can move either by sliding into an adjacent empty square, or by jumping a single adjacent piece into the empty square immediately beyond.

# Correctness

Before implementation begins, it is a good idea to know what we want!

**Question: What does the function shown below do?**




```
1  def isSorted(L):
2      for i in range(1, len(L)):
3          if L[i] < L[i-1]:
4              return False
5
6      return True
```

# 2-minute movement break

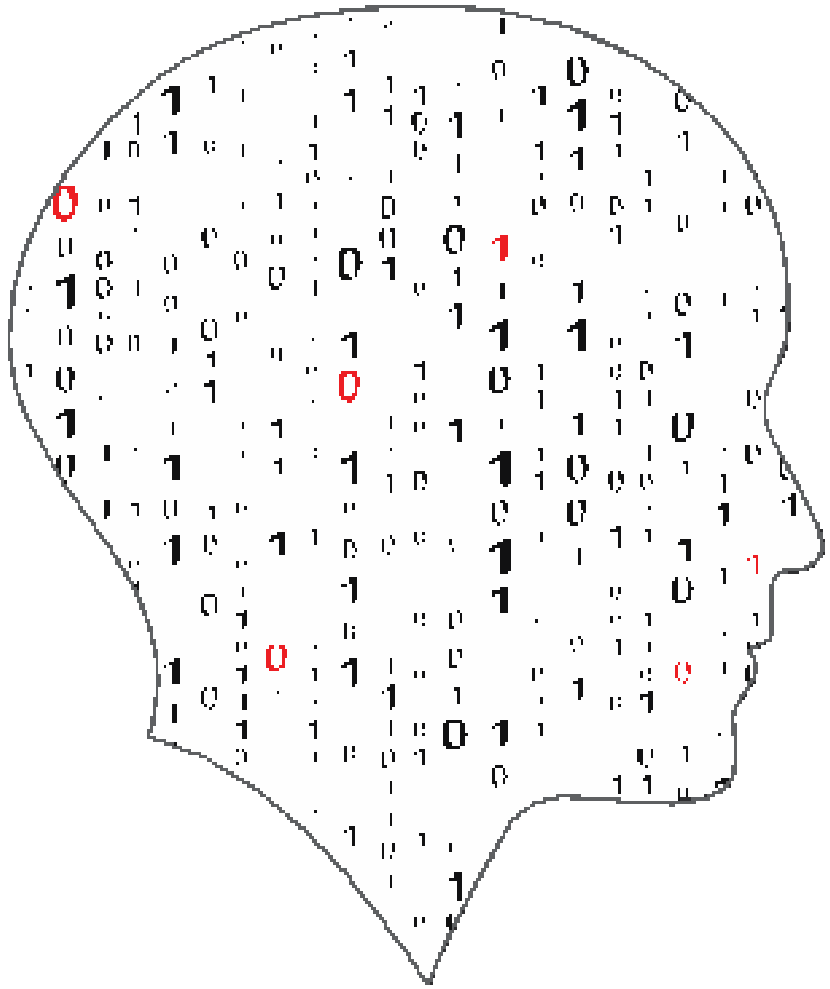
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# LCCS NW6

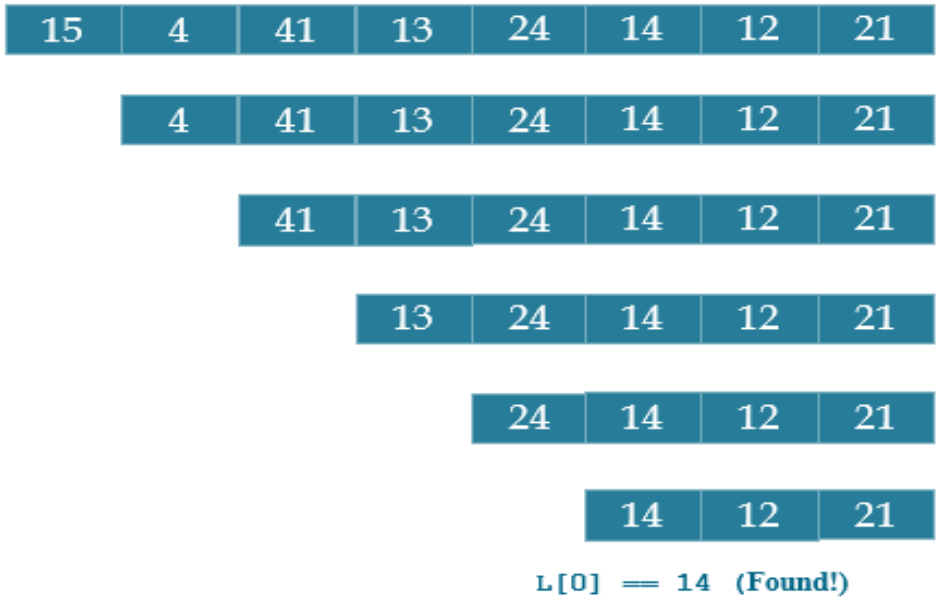
## Sorting algorithms: **Quicksort**



LEAVING CERTIFICATE
COMPUTER SCIENCE

# Recursion

```
def linear_search_v6(v, L, index=0) :
    if len(L) != 0:
        if L[0] == v:
            return index

        r =  linear_search_v6(v, L[1:], index+1)
        if r != -1:
            return r

    return -1
```



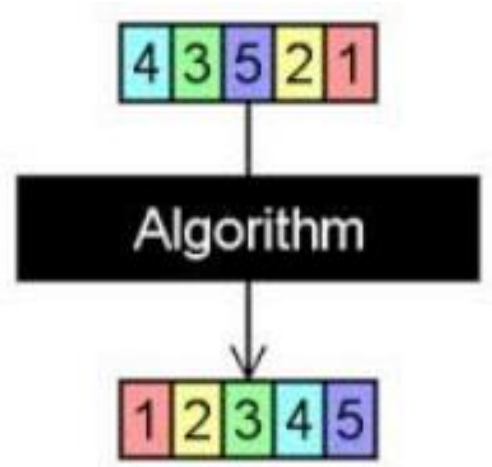
# Sorting algorithms

**An algorithm that maps the following input/output pair is called a sorting algorithm:**

**Input: A list (array), *L*, that contains *n* orderable elements: *L*[0, 1, …, *n* - *1*]**

**Output: A sorted permutation of *L*, called *S*, such that *S*[0] ≤ *S*[1] ≤ … ≤ *S*[*n* – 1].**

A general sorting algorithm devised by Tony Hoare in the late 1950s.




**Quicksort**

# Quicksort: the basic idea

**DIVIDE**

**1. Pick some number p from the list – called the pivot**

**2. Partition all the data into:**

A. The values less than the pivot (call this the left list)
B. The pivot (call this the middle list)
C. The values greater than the pivot (call this the right list)

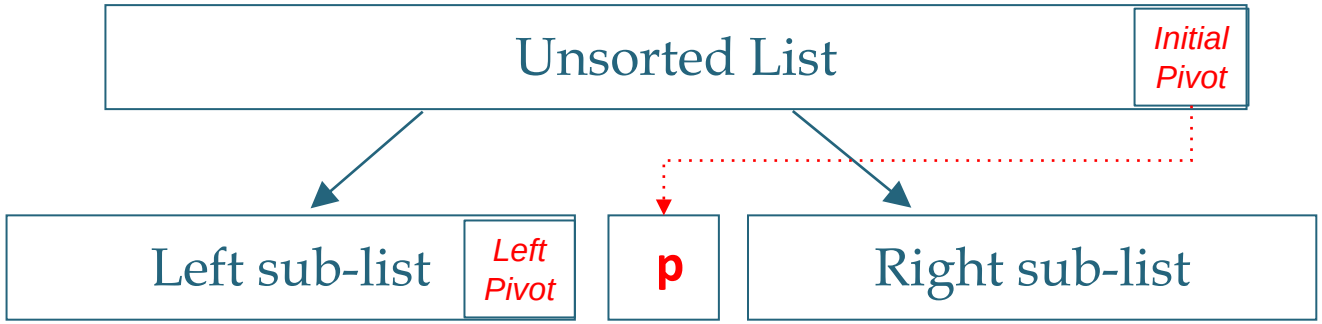
**CONQUER**

**3. Quicksort the left list (A)**

**4. Quicksort the right list (B)**

**5. The answer is left list + middle list + right list**

# Partitioning




STEP 1. Choose the rightmost element in the list as the `pivot`

STEP 2. Create three empty lists called `left_list`, `middle_list` and `right_list`

STEP 3. for each `key` (item) in the list

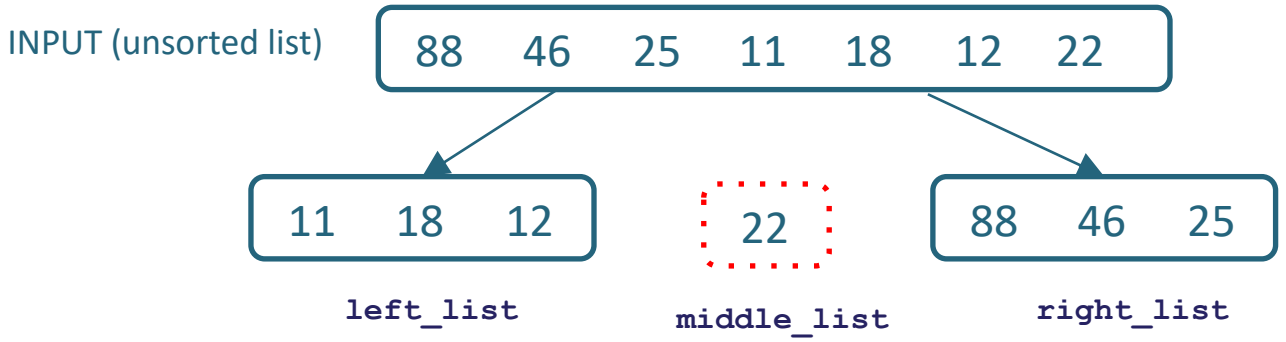
- if `key` is < `pivot` add it to `left_list`
- if `key` is == `pivot` add it to `middle_list`
- if `key` is > `pivot` add it to `right_list`

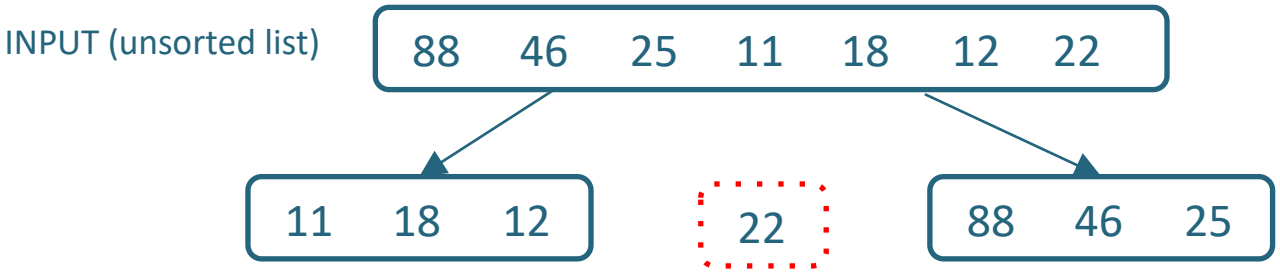
```
def quick_sort(L):
    left_list = []
    middle_list = []
    right_list = []

    # Base case
    if len(L) <=1:
        return(L)

    # Set pivot to the last element in the list
    pivot = L[len(L)-1]

    # Iterate through all elements (keys) in L
    for key in L:
        if key < pivot:
            left_list.append(key)
        elif key == pivot:
            middle_list.append(key)
        else:
            right_list.append(key)

    # Repeat the quicksort on the sub-lists and combine the results
    return quick_sort(left_list) + middle_list + quick_sort(right_list)
```

**What features of recursion can you can see in this code?**

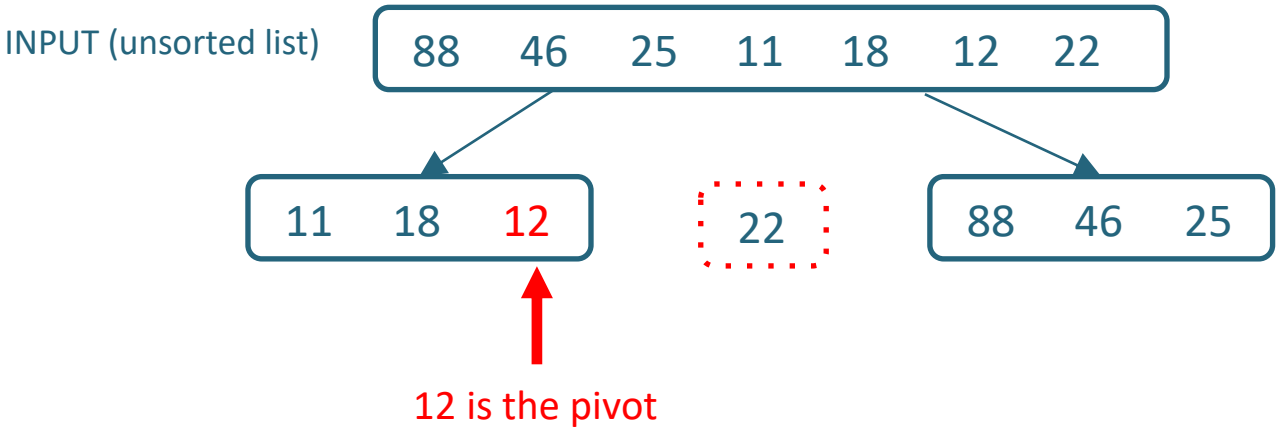
# Example

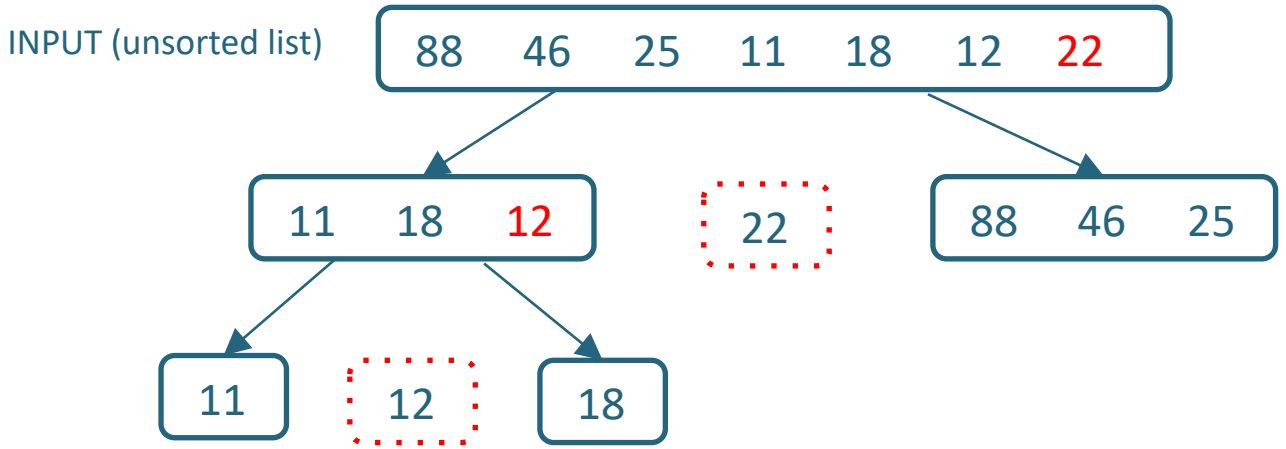

```
pivot = L[len(L)-1]
```

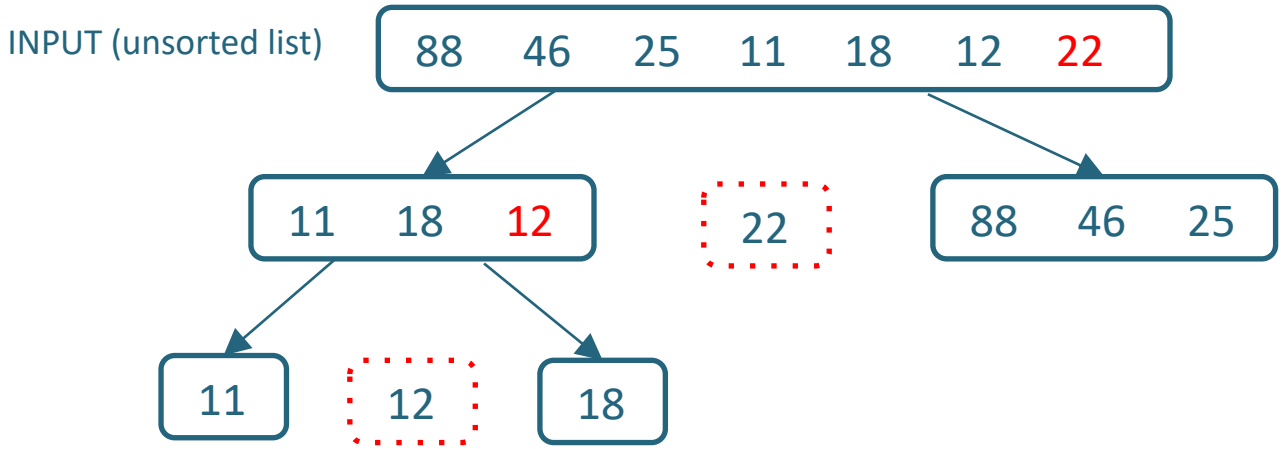

INPUT (unsorted list)
88 46 25 11 18 12 22
11 18 12
22
88 46 25
left_list
middle_list
right_list



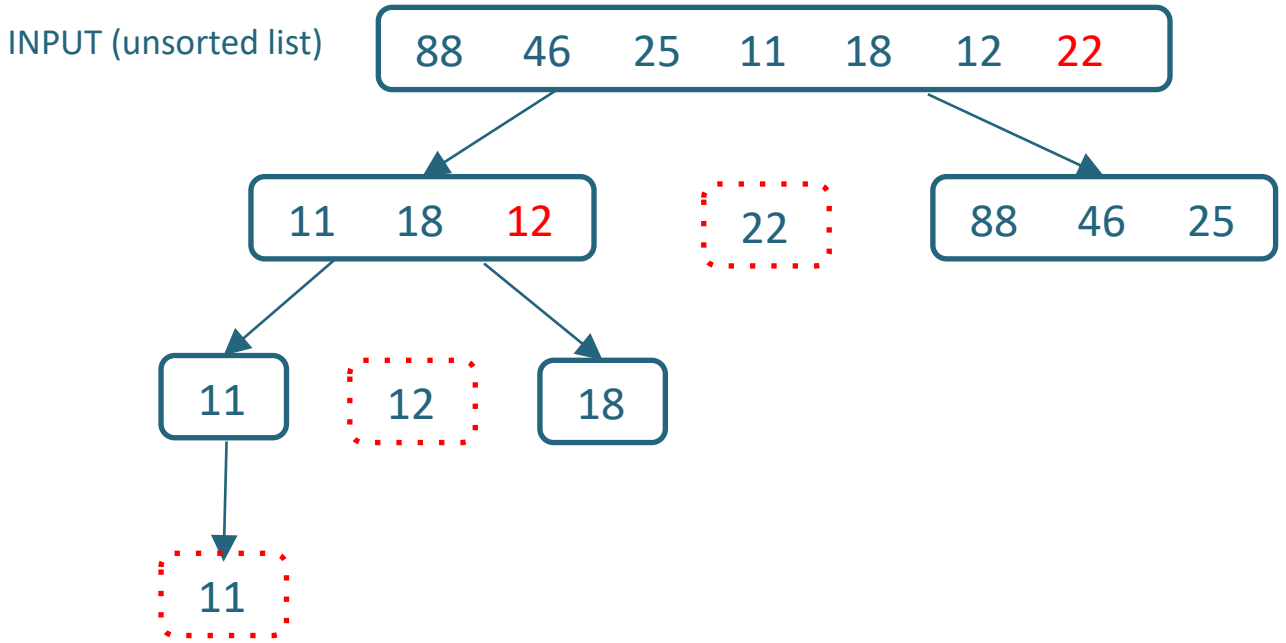

## Now quicksort `left_list`


INPUT (unsorted list)
88 46 25 11 18 12 22
11 18 12
22
88 46 25
12 is the pivot


INPUT (unsorted list)
88 46 25 11 18 12 22
11 18 12
22
88 46 25
11
12
18

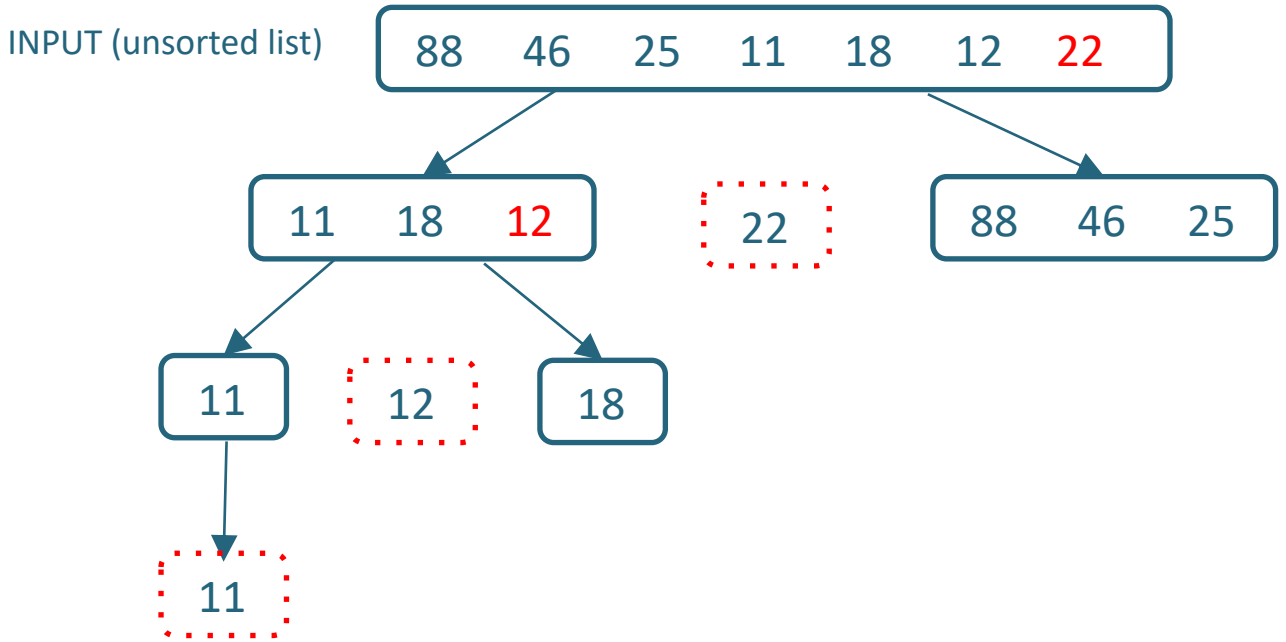



## Now quicksort `left_list`

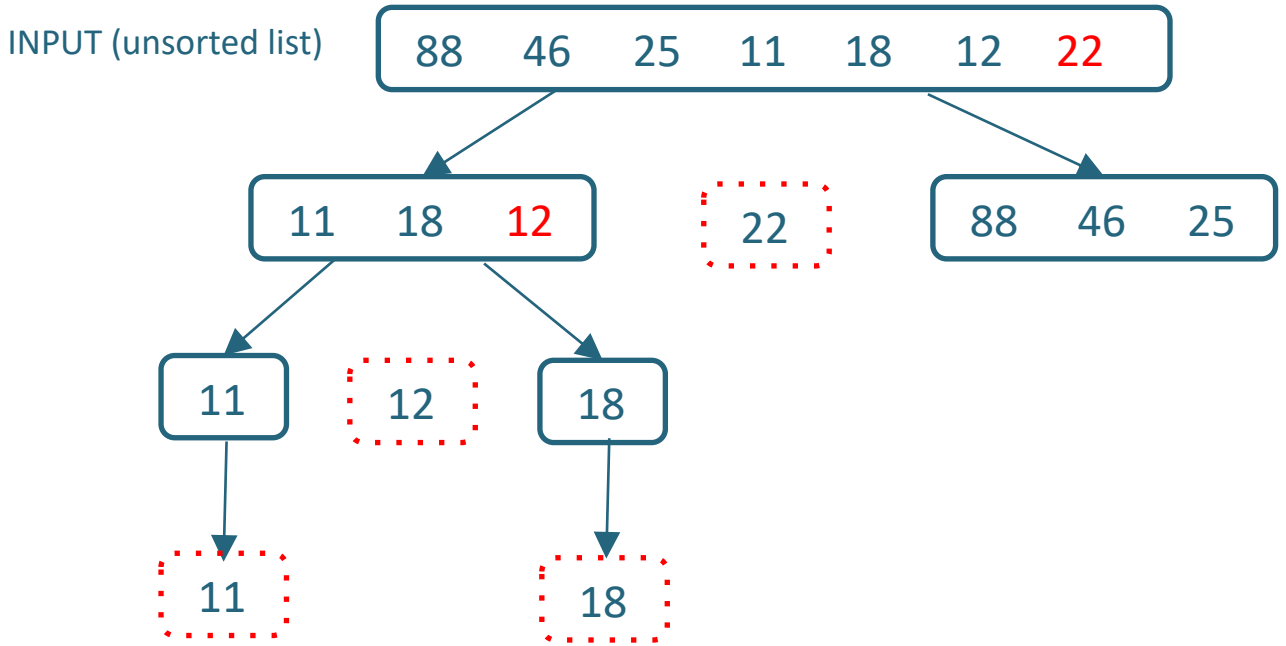



**Now quicksort `left_list`**

**Base case: len(L) <= 1 so return 11**

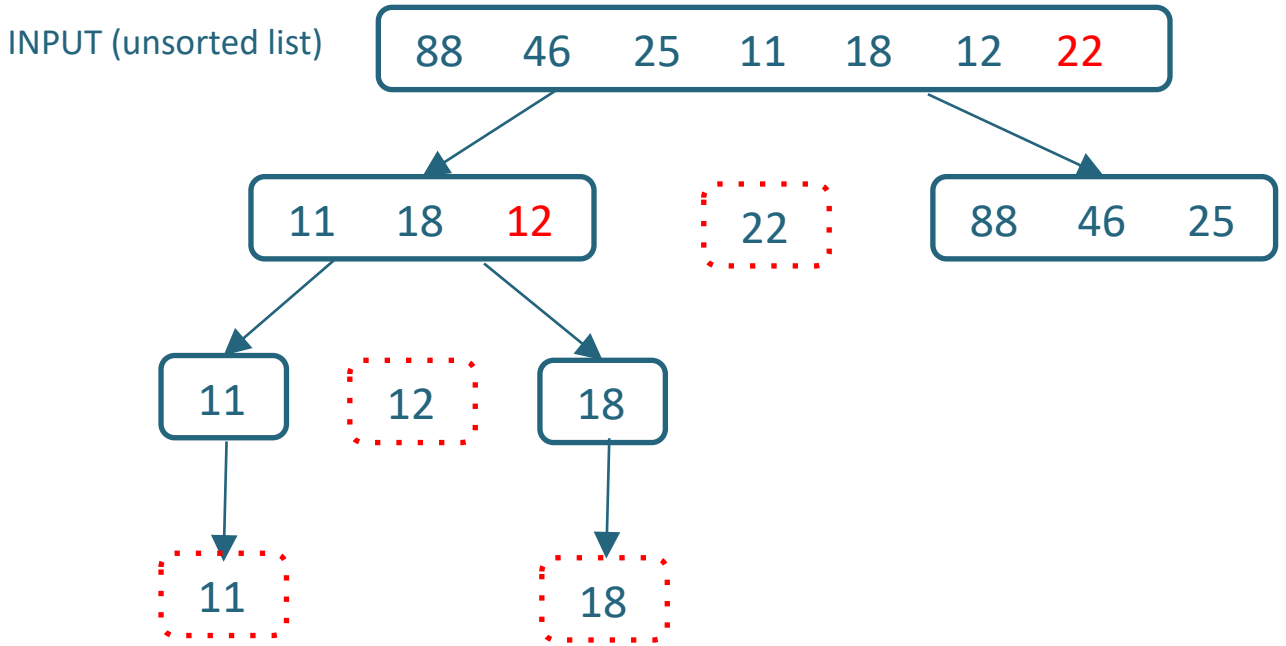



## Now quicksort `right_list`

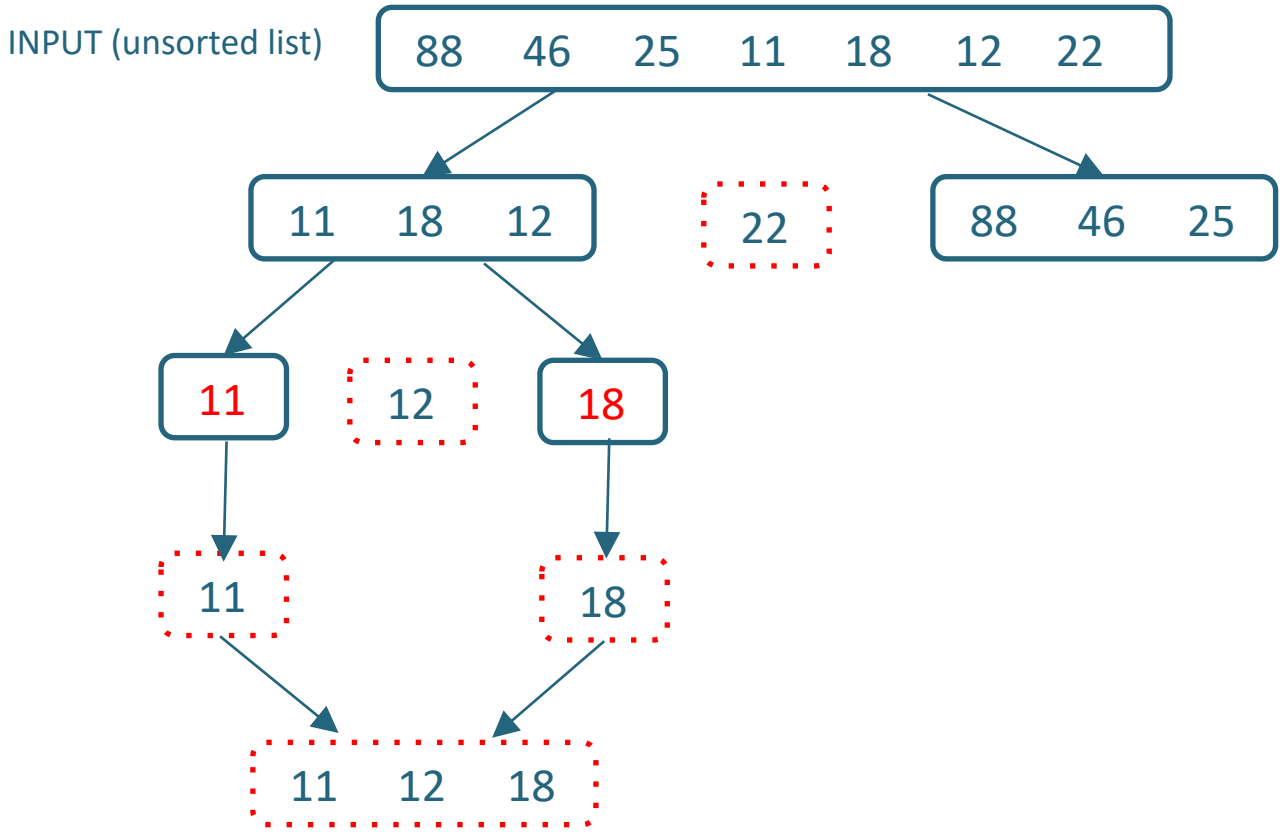


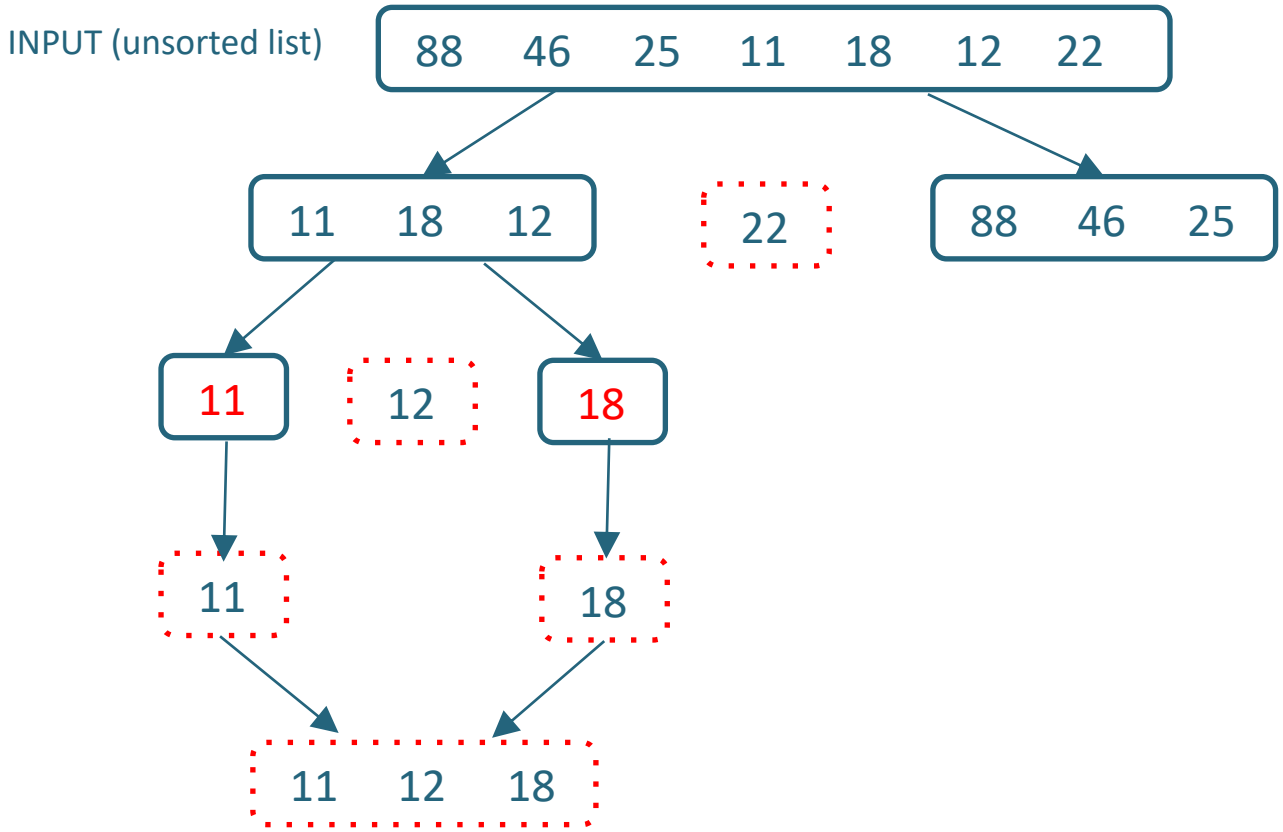

## Now quicksort `right_list`

**Base case: `len(L) <= 1` so return 18**

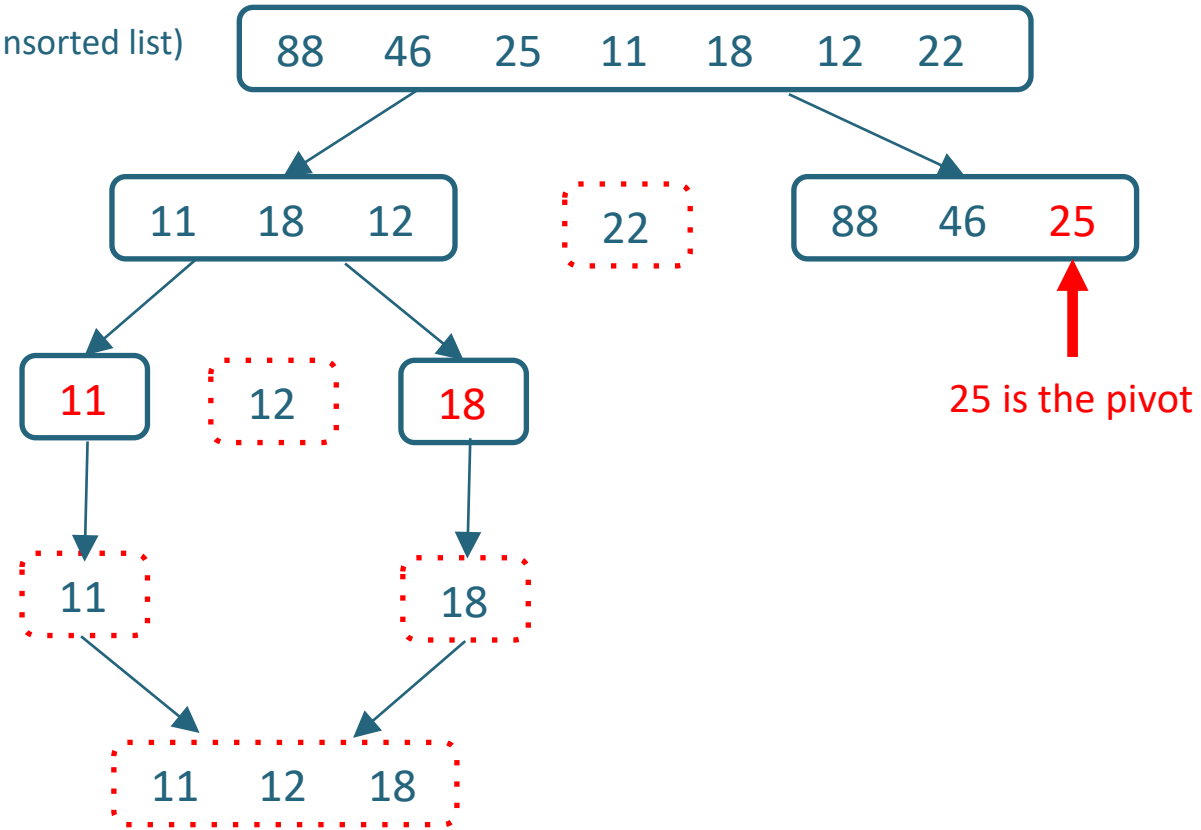



**Result is `left + middle + right` so return 11 12 18**

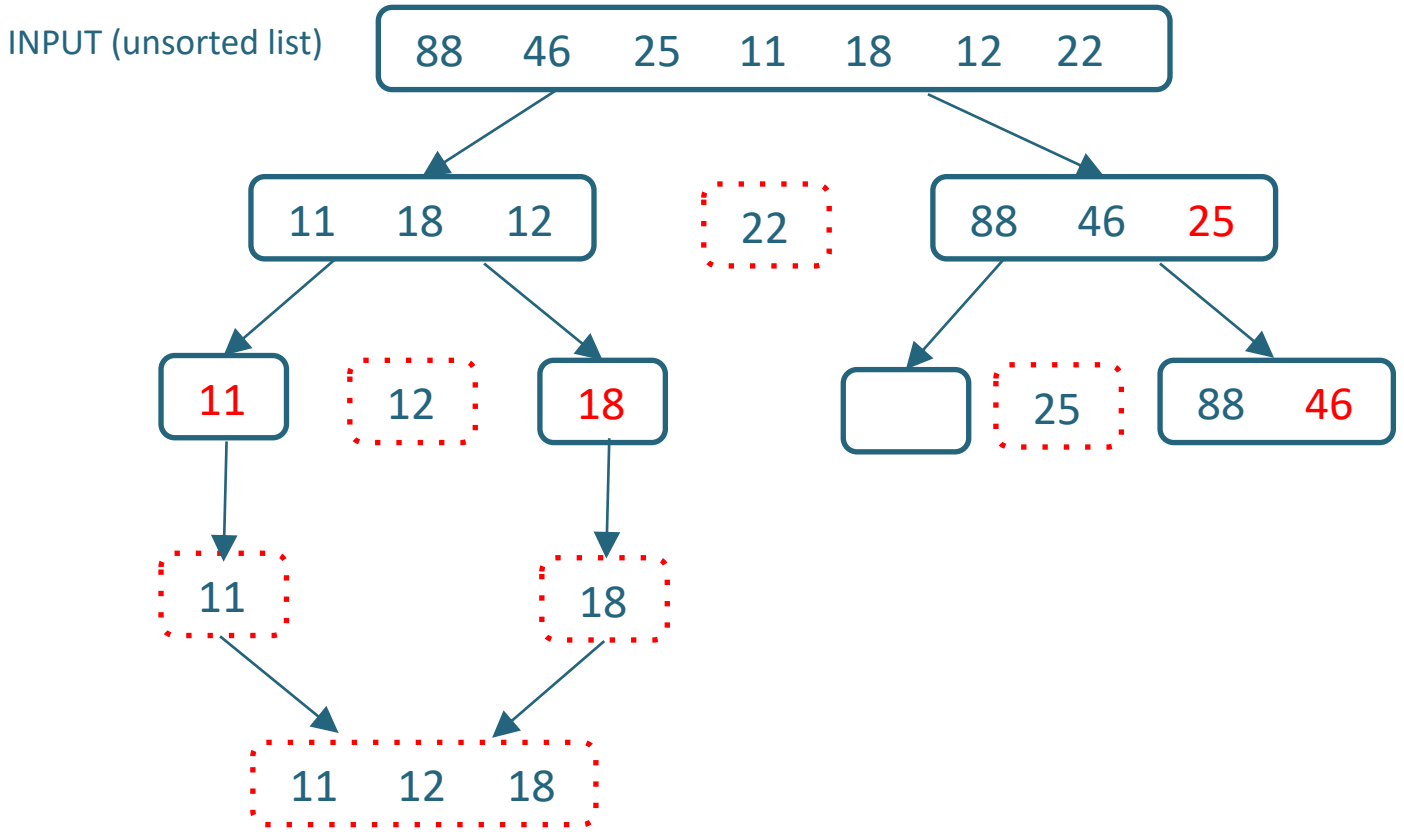

INPUT (unsorted list)
88 46 25 11 18 12 22
11 18 12
22
88 46 25
11
12
18
11
18
11 12 18



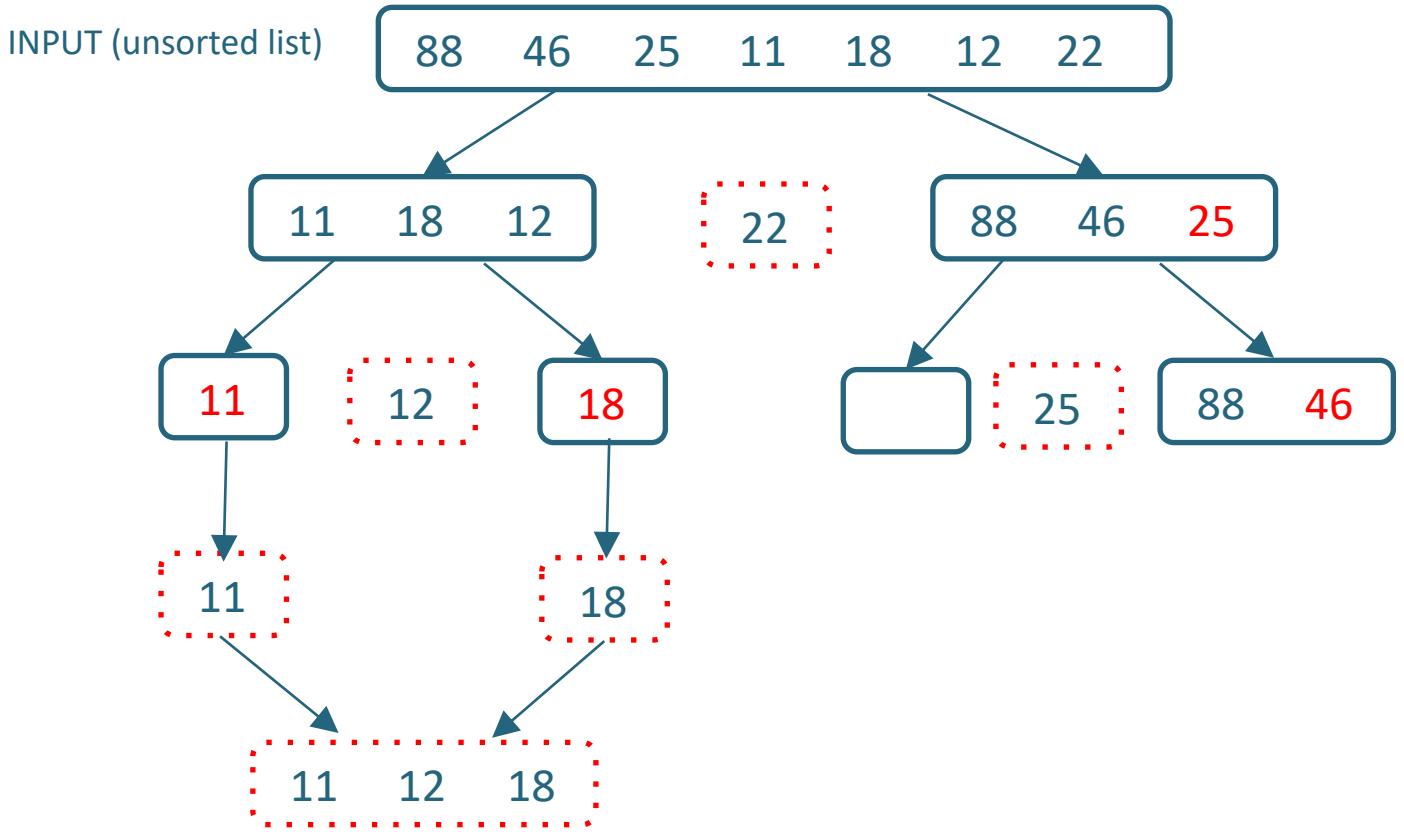

## Now quicksort `right_list`



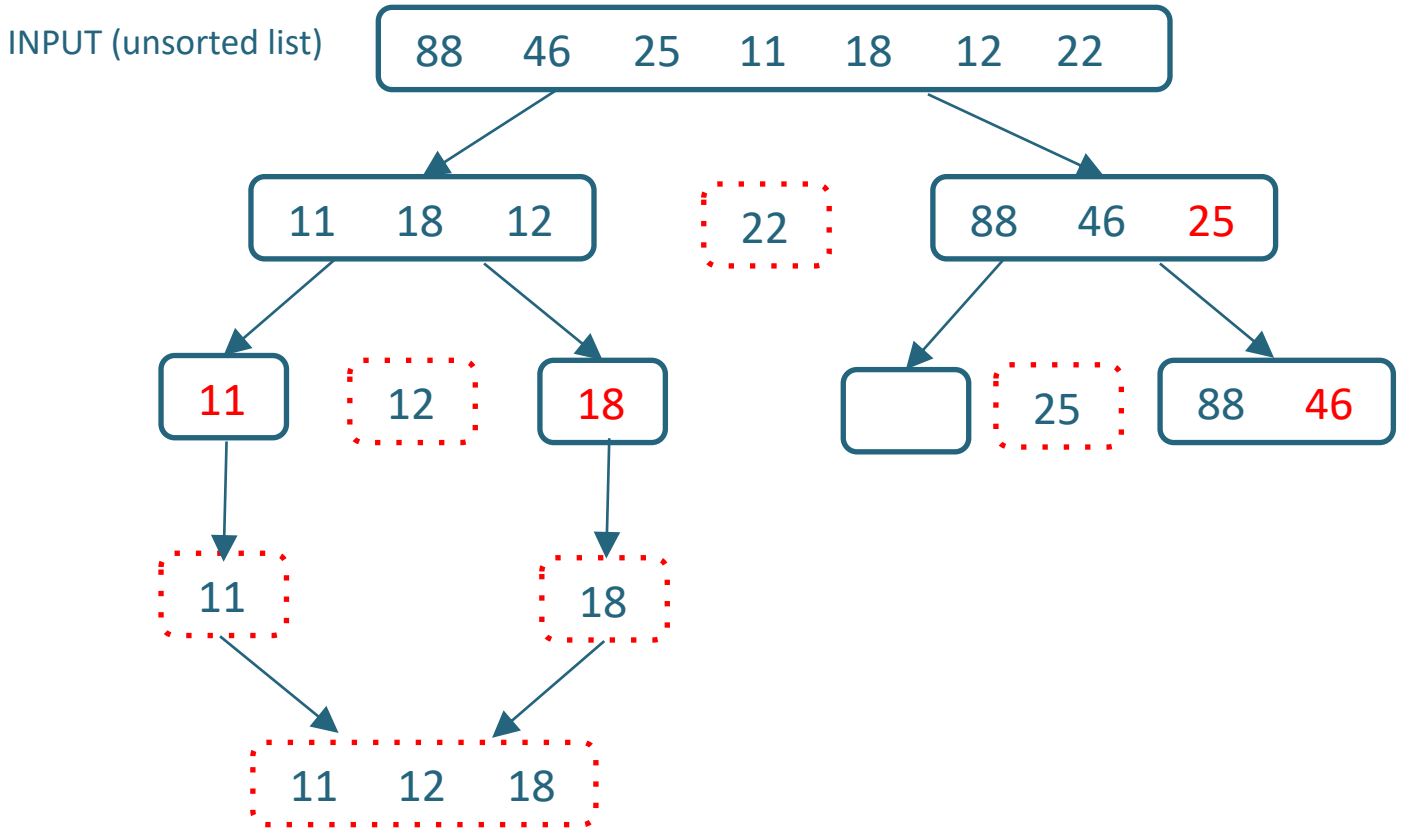

**Now quicksort `right_list`**



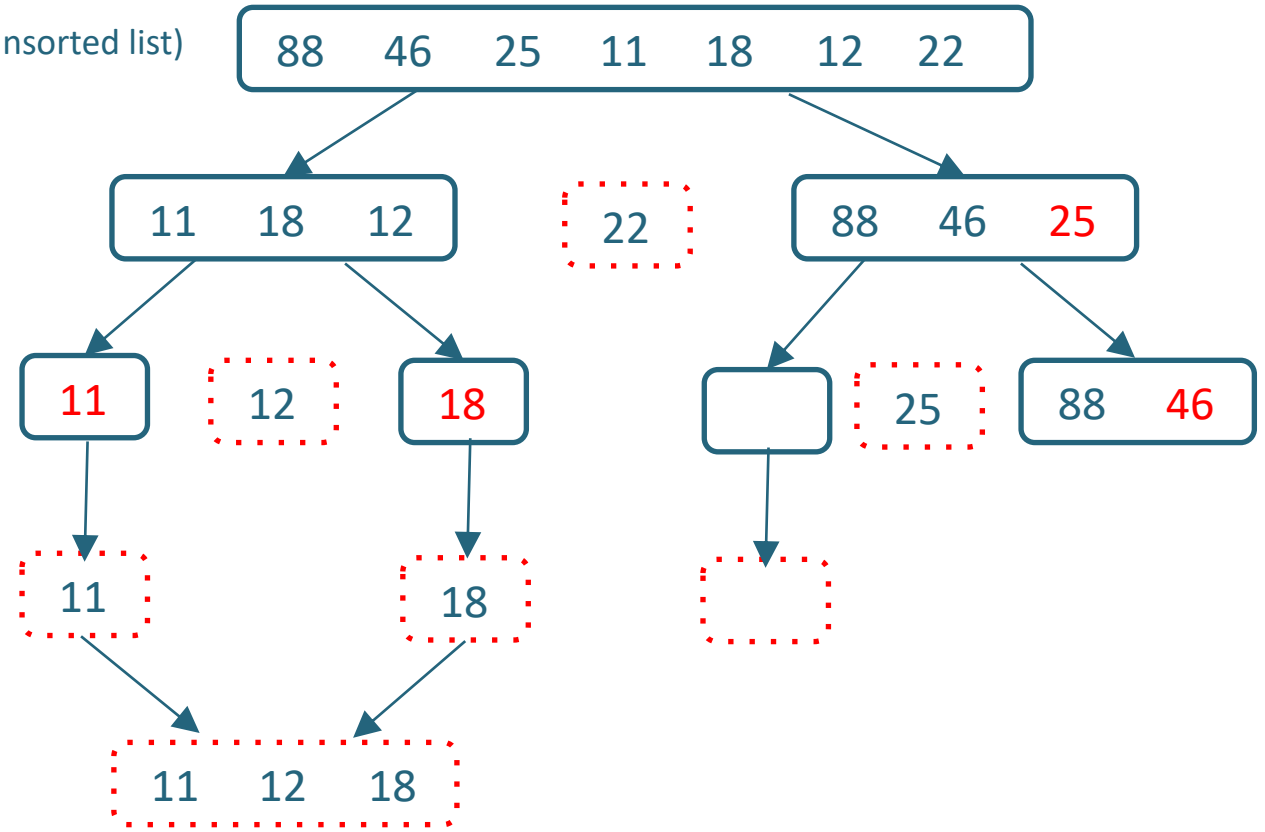

**Partition around 25**




## Now quicksort `left_list`

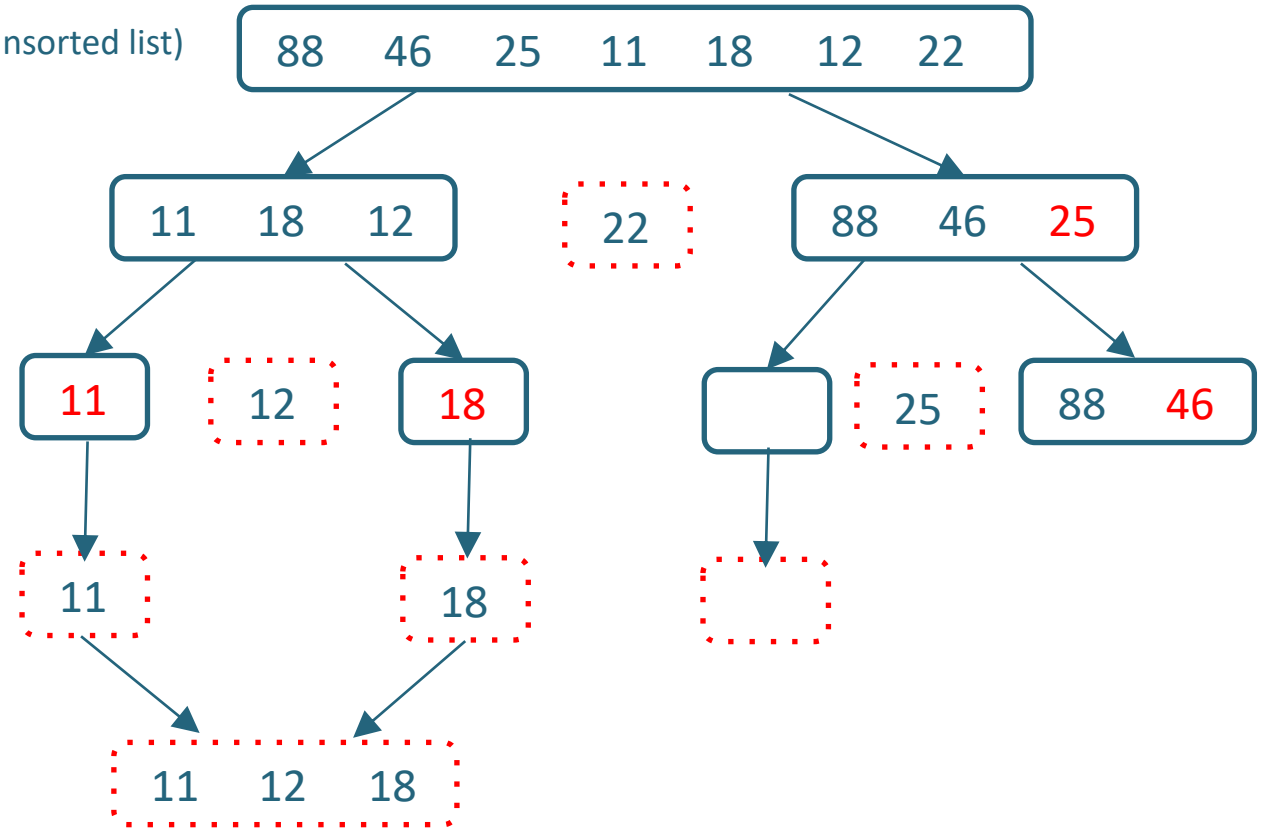



## Now quicksort `left_list`

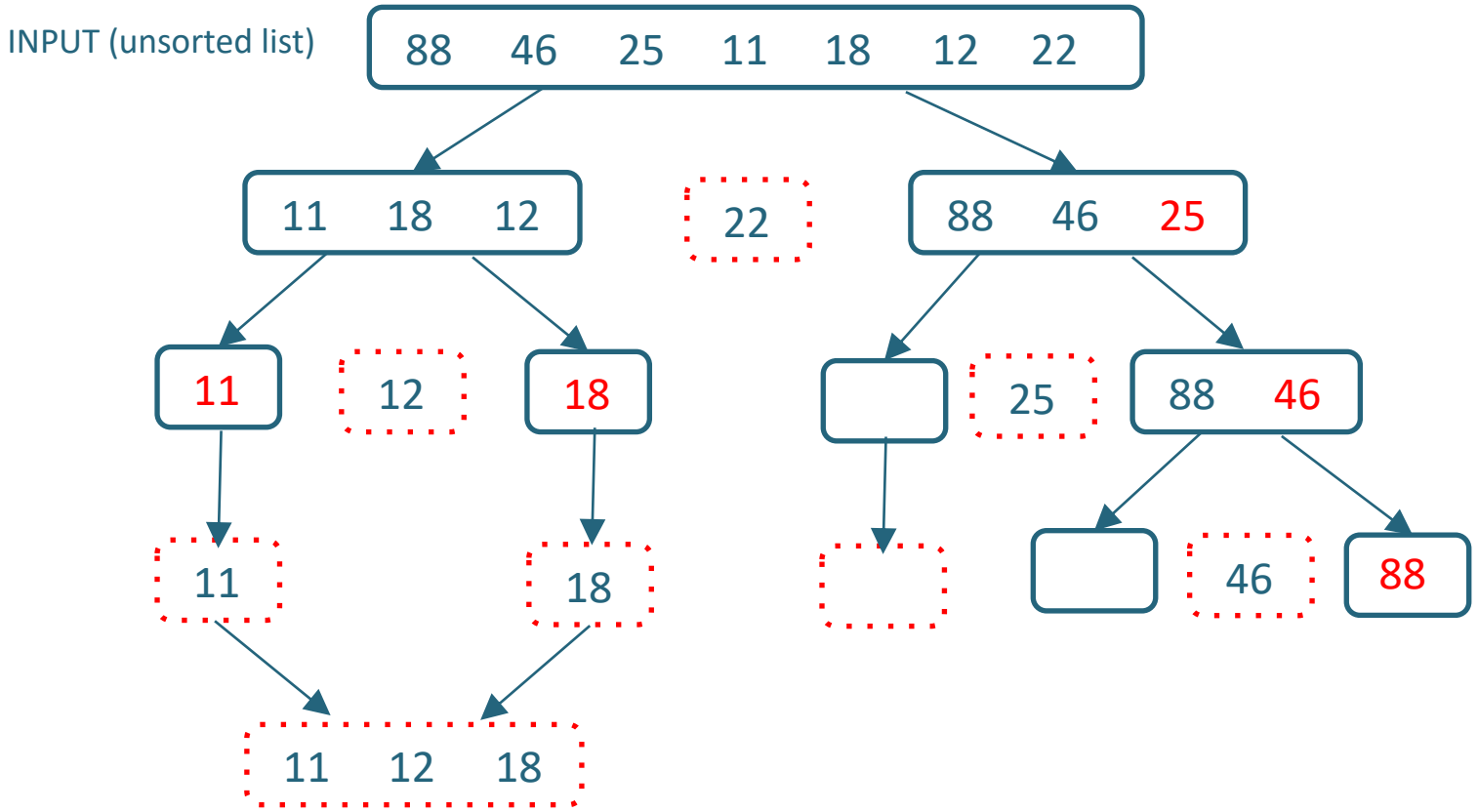



## Now quicksort `left_list`

**Base case: `len(L) <= 1` so return []**

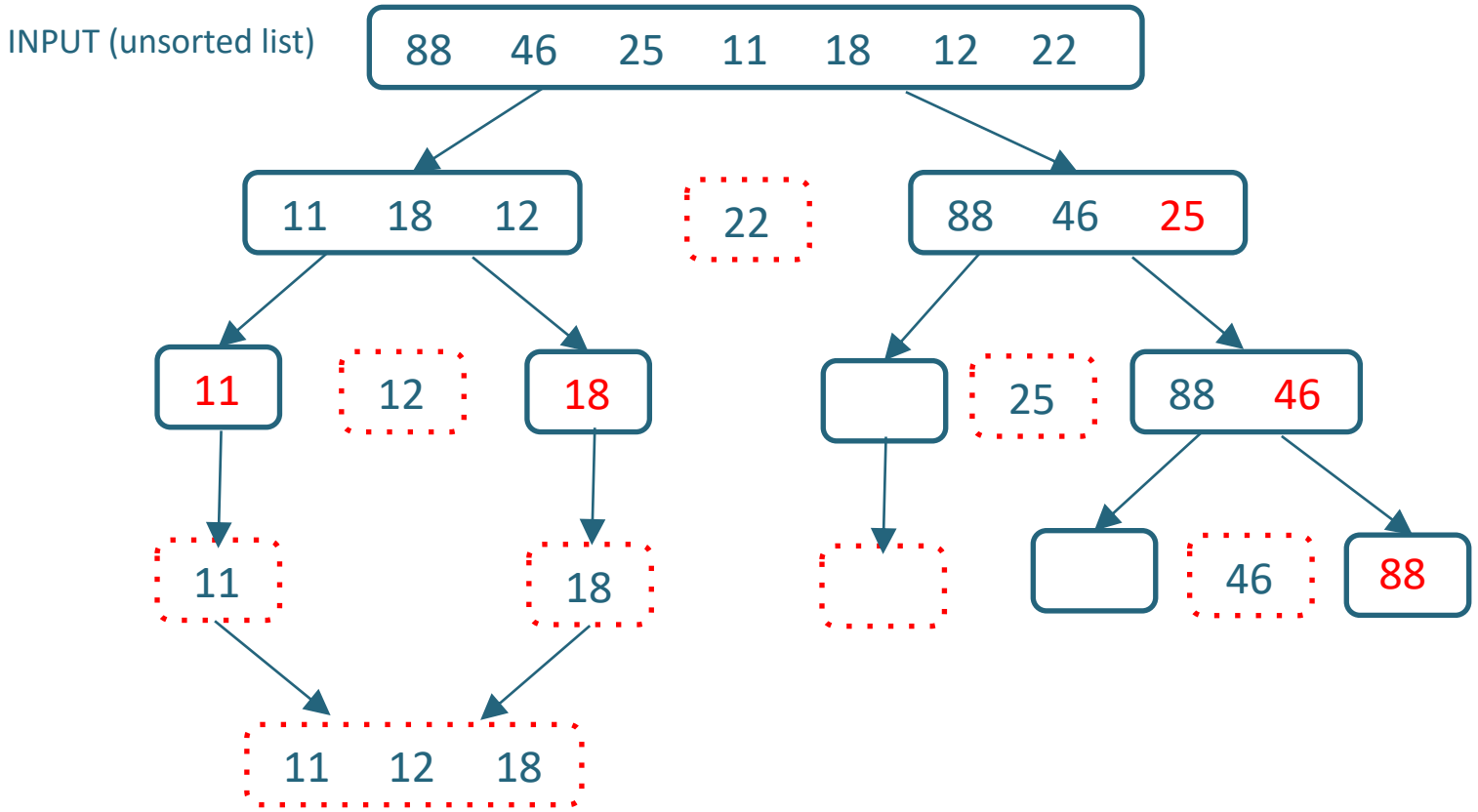



## Now quicksort `right_list`




## Now quicksort `right_list`

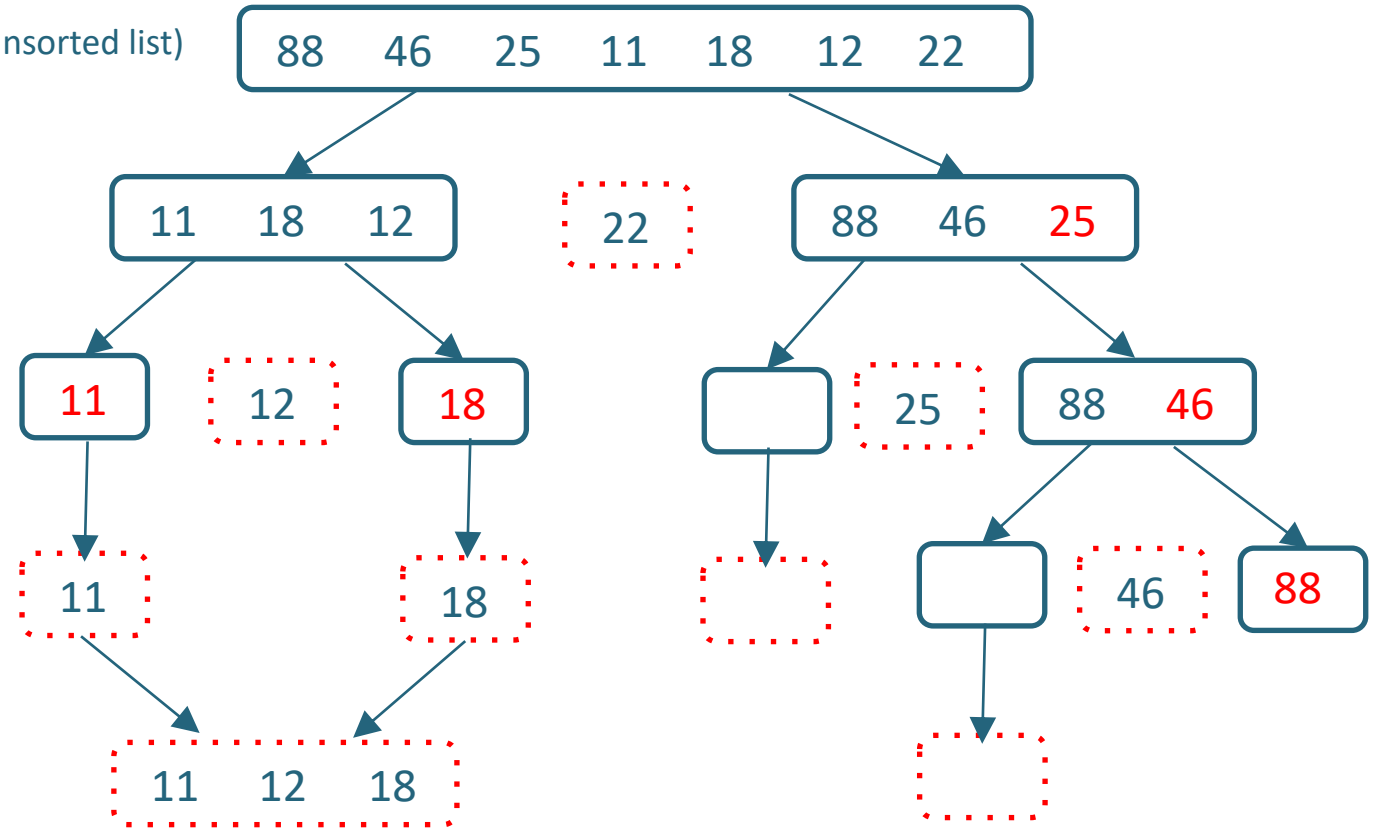
## Partition around 46




**Now quicksort `right_list`**

**Partition around 46**

**Now quicksort `left_list`**

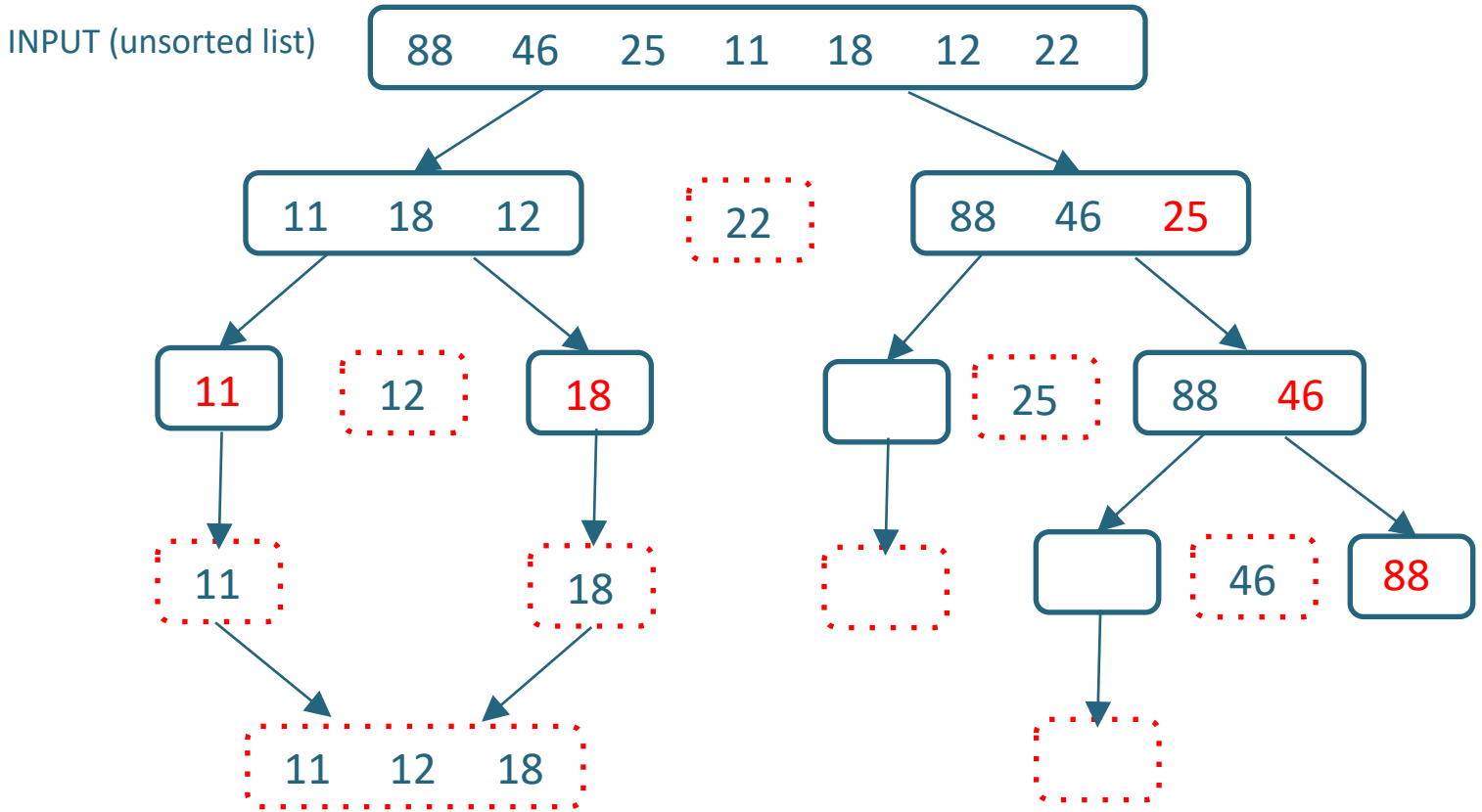



**Now quicksort `right_list`**

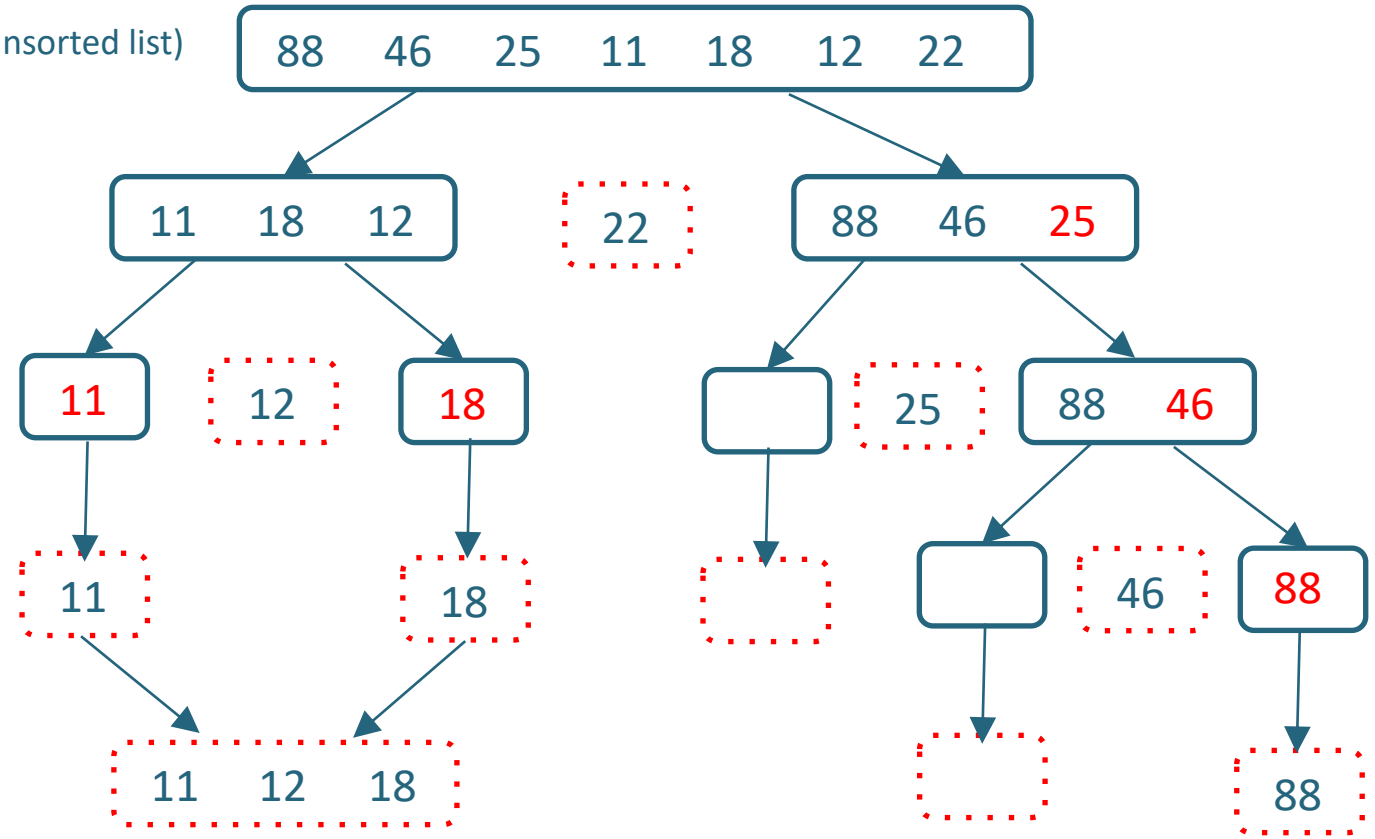
**Partition around 46**

**Now quicksort `left_list`**

**Base case: `len(L) <= 1` so return []**

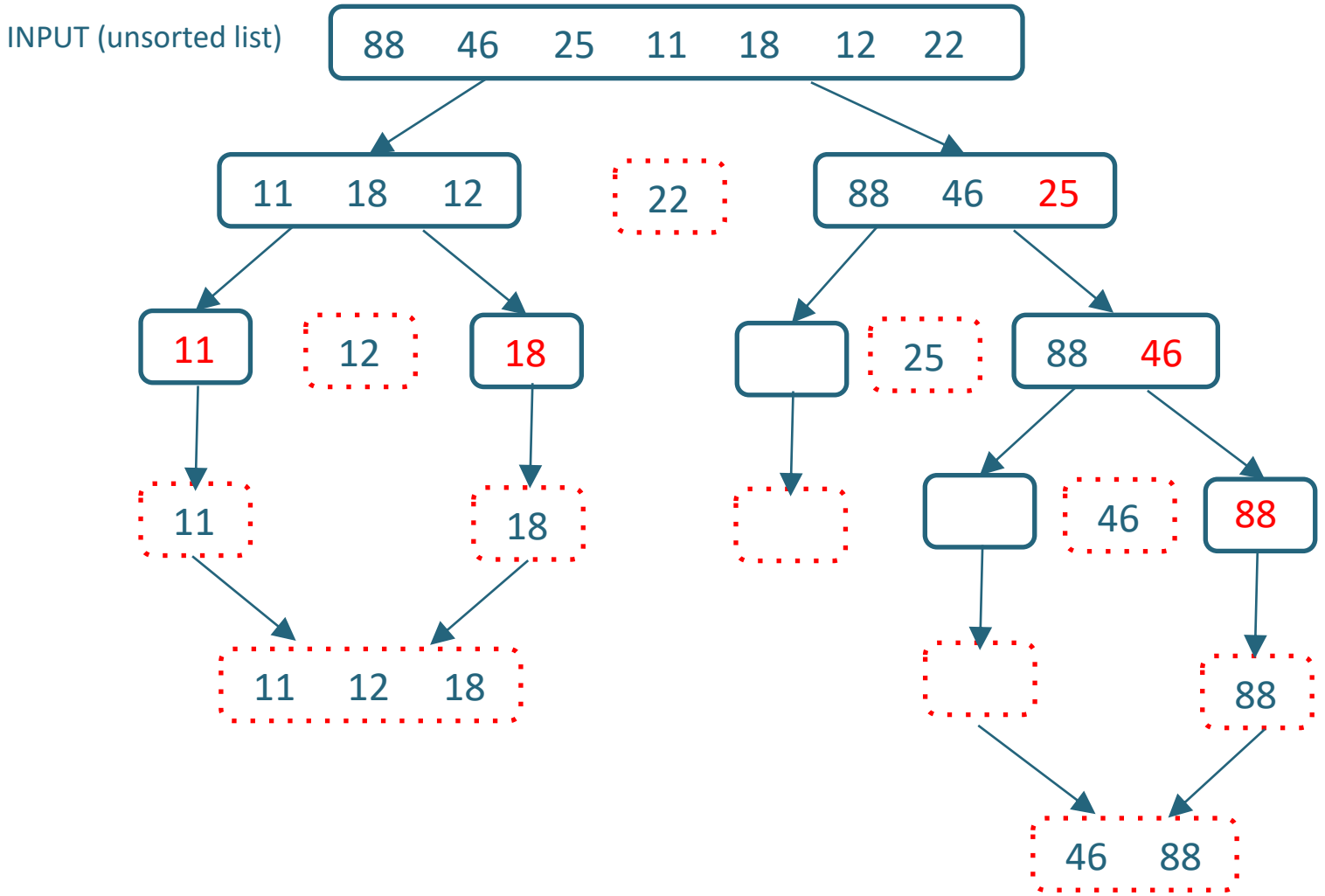



**Now quicksort `right_list`**

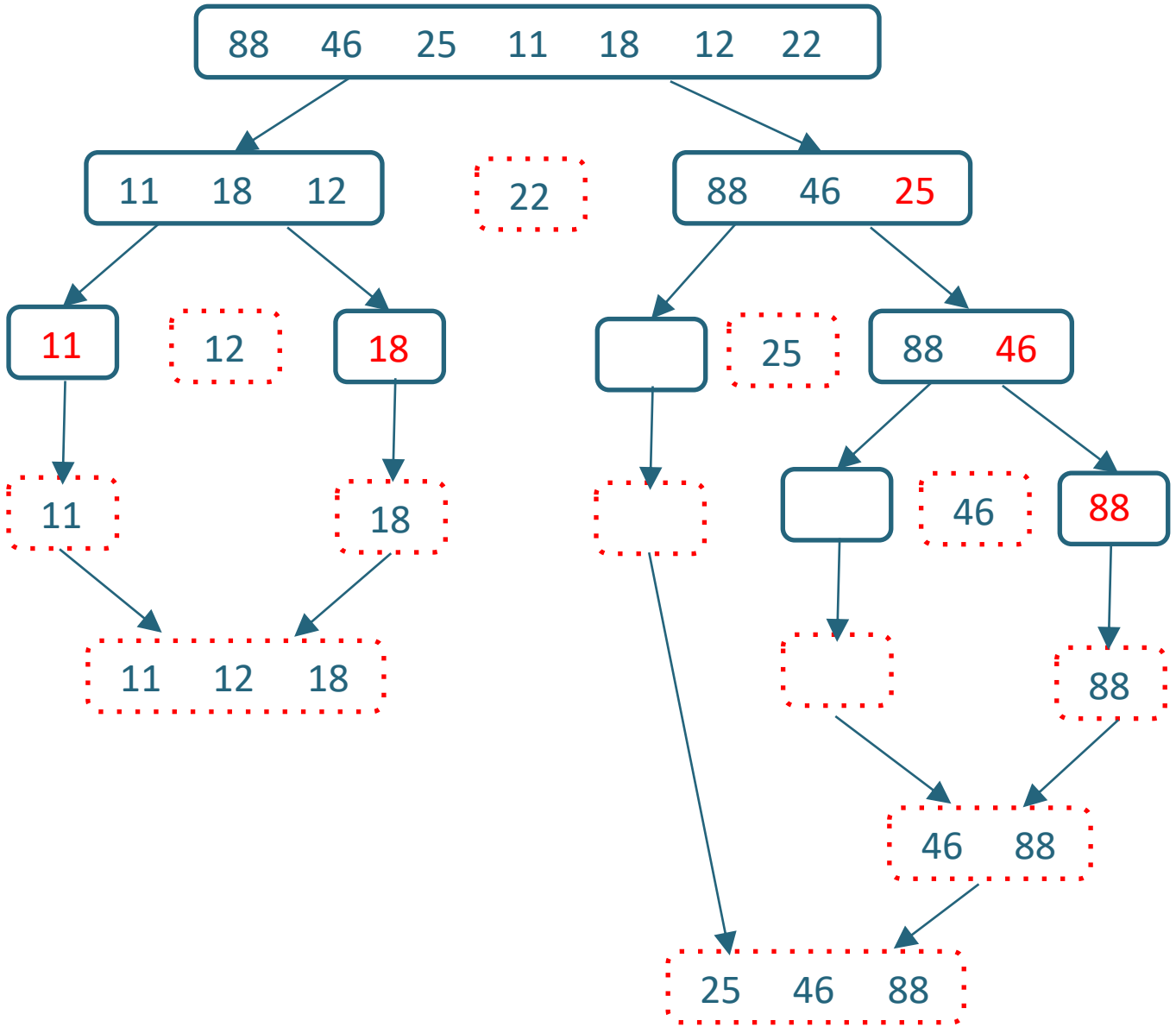



**Now quicksort `right_list`**

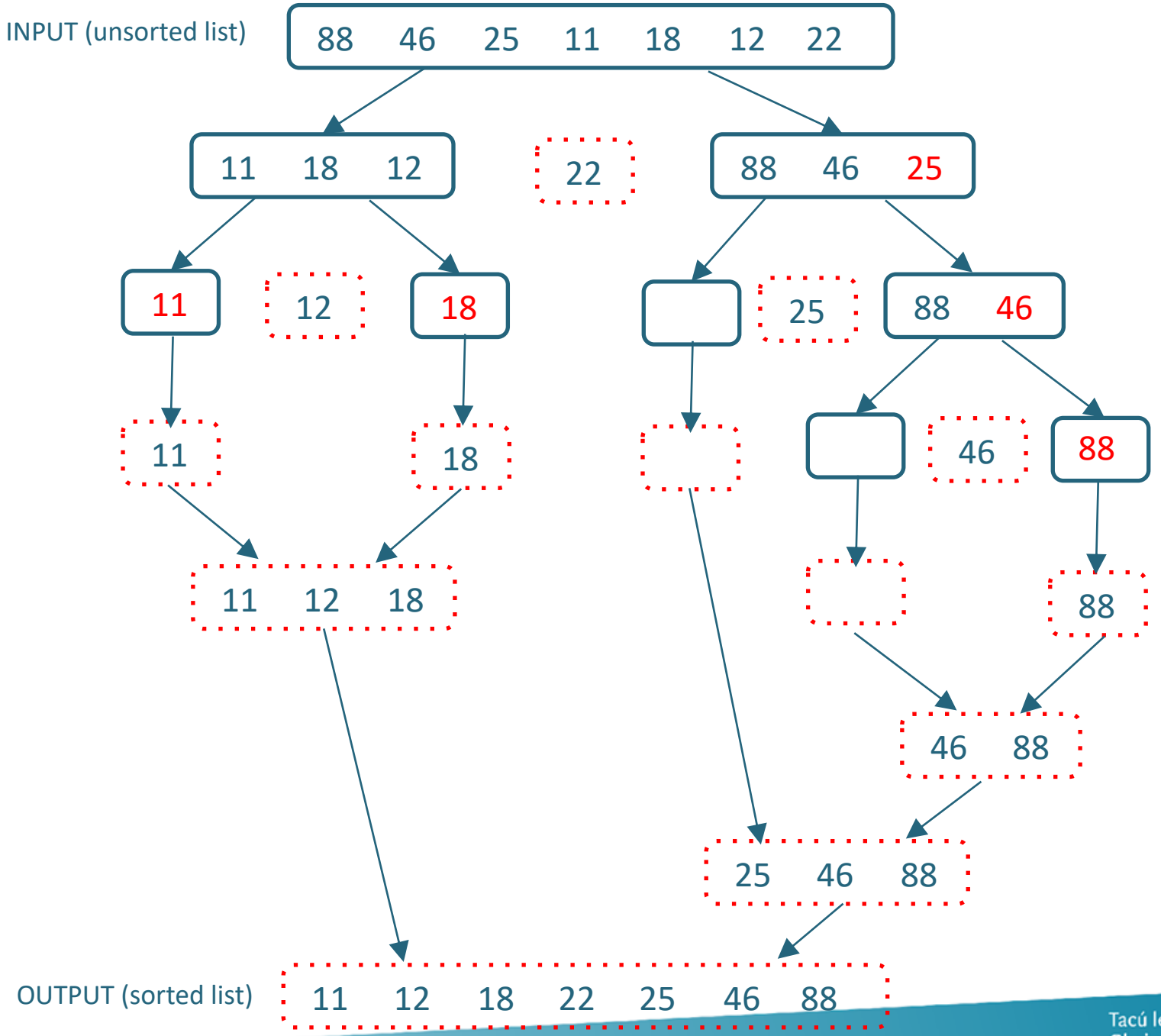
**Base case: `len(L) <= 1` so return 88**




**Result is `left + middle + right` so return 46 88**

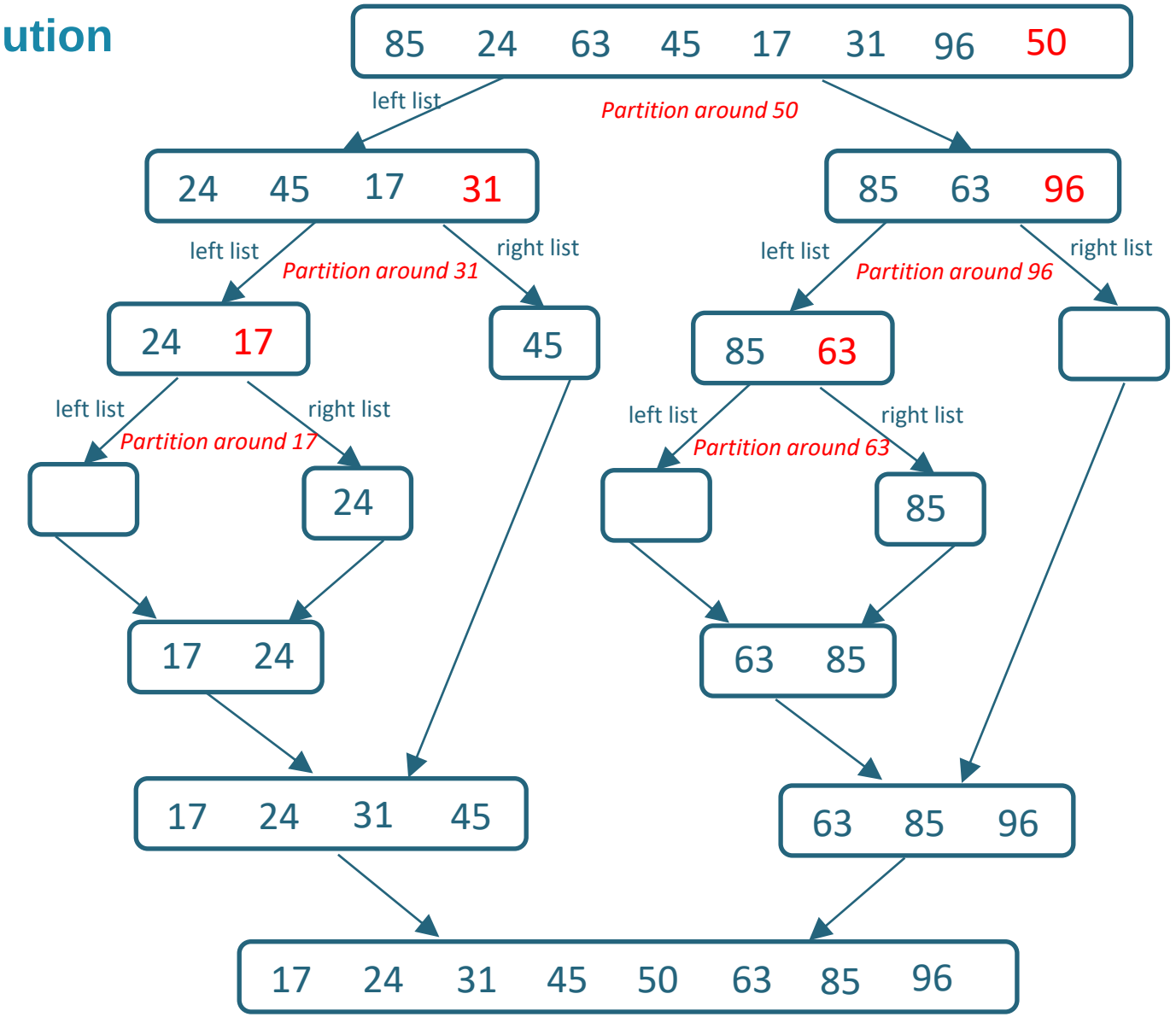



**Result is `left + middle + right` so return 25 46 88**


INPUT (unsorted list)
88 46 25 11 18 12 22
11 18 12
22
88 46 25
11
12
18
25
88 46
11
18
46
88
11 12 18
88
46 88
25 46 88
OUTPUT (sorted list)
11 12 18 22 25 46 88

# Perform a quicksort on the following:


# Sample Solution




denotes empty list

LCCS Sample Paper Q15 (d)

# Exercise

Investigate why this scenario leads to the worst-case performance for the quicksort

## Tea/Coffee

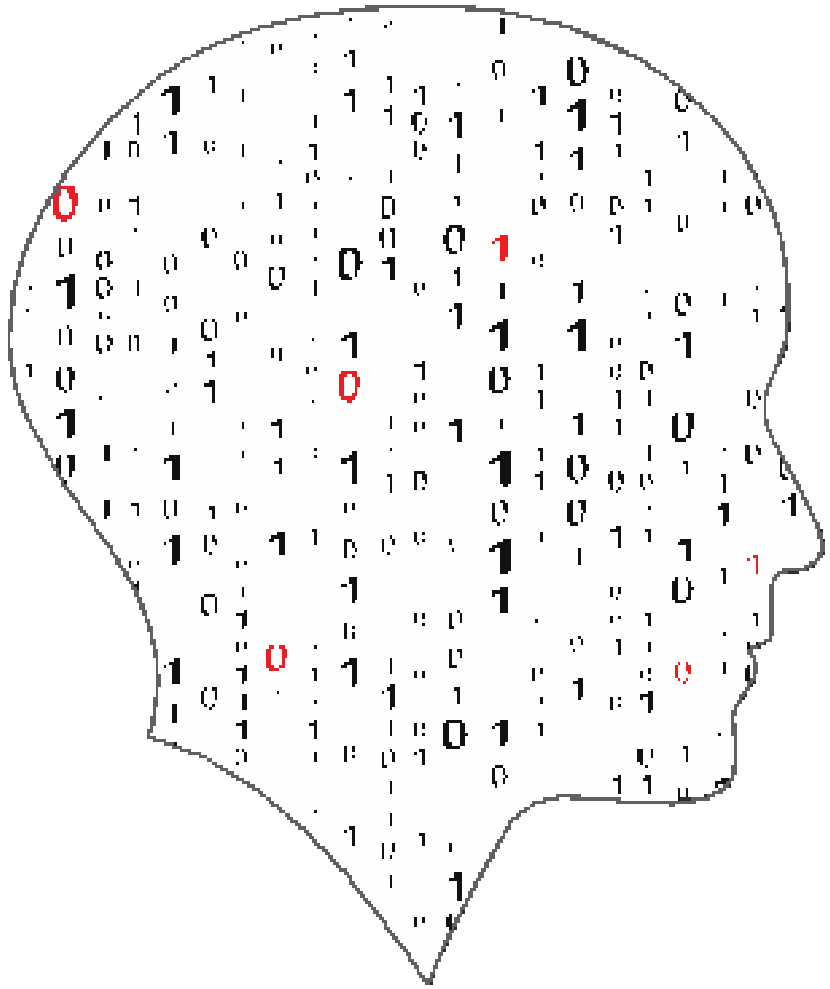
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# Session 2

# Evaluation and Testing



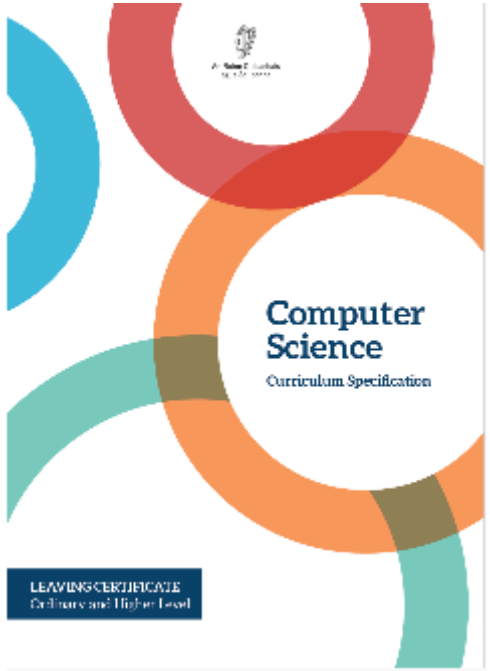
LEAVING CERTIFICATE COMPUTER SCIENCE

# By the end of this session participants will have:

- Examined the importance of testing in the software development cycle.
- Experienced real-life testing scenarios.
- Developed test plans and test cases for a variety of situations.
- Investigated and experienced test-driven development.

# LCCS Learning Outcomes

| | |
|---|---|
| **S2: Evaluation and testing**<br><br>Debugging<br><br>Testing: Unit test, **Function test, System test** | 2.19 test solutions and decisions to determine their short-term and long-term outcomes<br>2.20 identify and fix/debug warnings and errors in computer code and modify as required<br>2.21 critically reflect on and identify limitations in completed code **and suggest possible improvements**<br>2.22 explain the different stages in software testing |




But Also

Design and Developing L.O. 1.19

Computational Thinking L.O.s 1.1 - 1.9

Computers and Society L.O.s 1.11

Computers and Society L.O.s 1.22

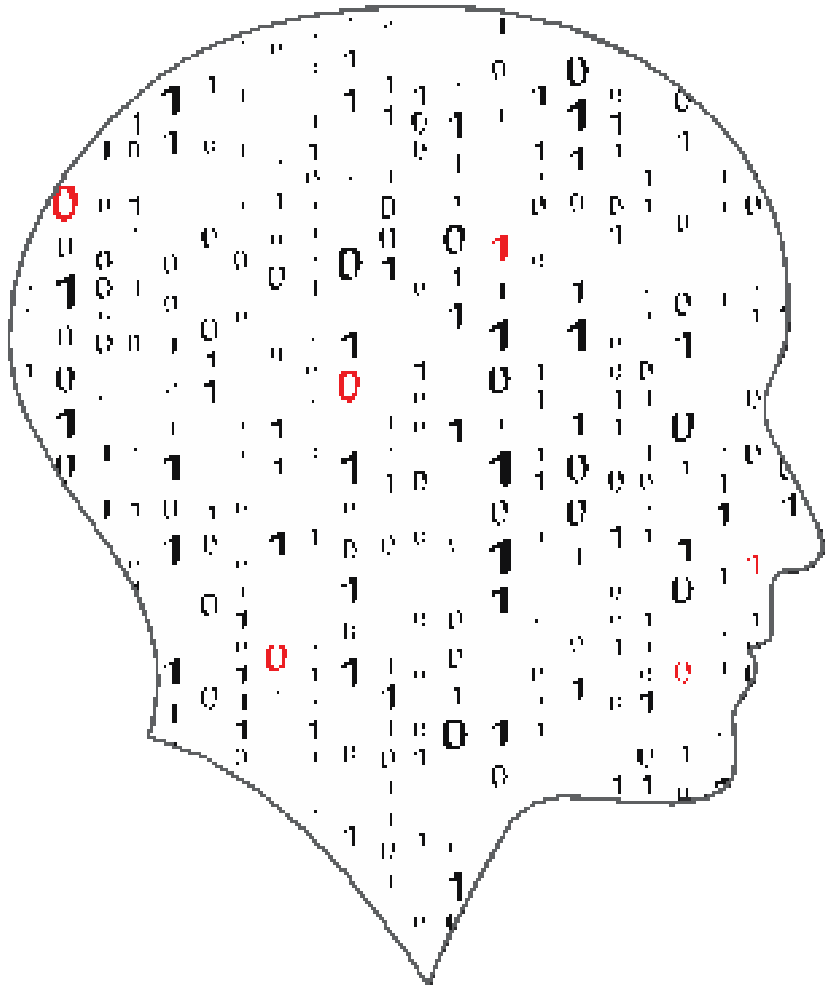
Computer Science in Practice L.O.s 3.2 – 3.14

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# 1. Introduction to Testing

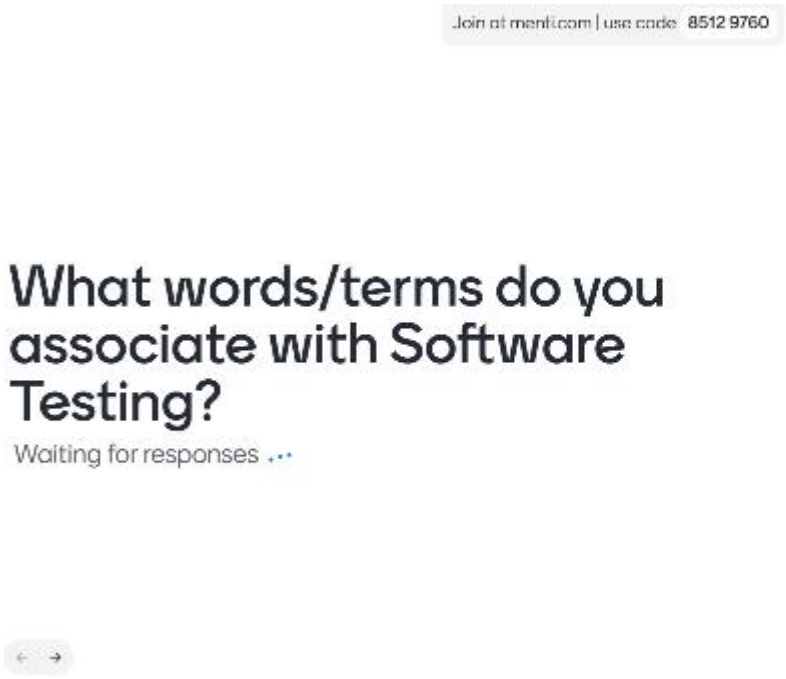


LEAVING CERTIFICATE
COMPUTER SCIENCE

# Why is Software Testing important?

Mentimeter

**8512 9760**





5 Minutes

# Why test Software?

## Some interesting scenarios

Boeing

MS Word 97 – French version – what happened?

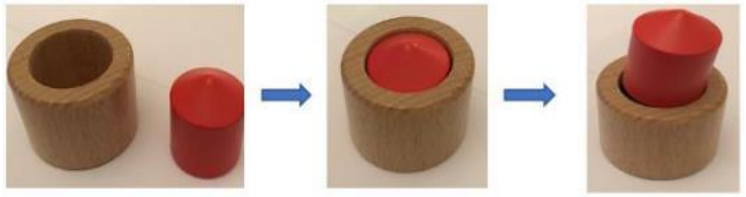
Windows XP


The  hot key E – went to ***Envoyer*** (Send) and users thought it was Edit with disastrous consequences

# Activity 1
# Reflecting on Software Testing.

P 2

## Testing the mini-rocket



Read *Evaluate testing the mini rocket* and note any thoughts you have on how you would carry out any tests.

You might consider:

- Can I test everything at the same time?
- Would I prioritise anything when testing?
- Are there any testing suggestions I would add/remove from the list?
- Is it important to have a testing strategy?

7 Minutes

Appoint a chair, a timekeeper, a notetaker and a spokesperson

# Feedback

Testing the mini-rocket

What did you include in your Test plan?

# How to Prepare an Effective Test Plan Template?

A Test Plan is like a **blueprint of how the testing activity is going to take place in a project** ensuring that a system is fully tested, and any outcomes are documented.

Test plans are made up of

- Who is testing
- The reason for the test
- The Test data
- The expected outcome
- The actual result

https://youtu.be/oD8Y2HJO7kQ?si=sEJ-bsr1NeRVnE5n

# What is a Test Case?

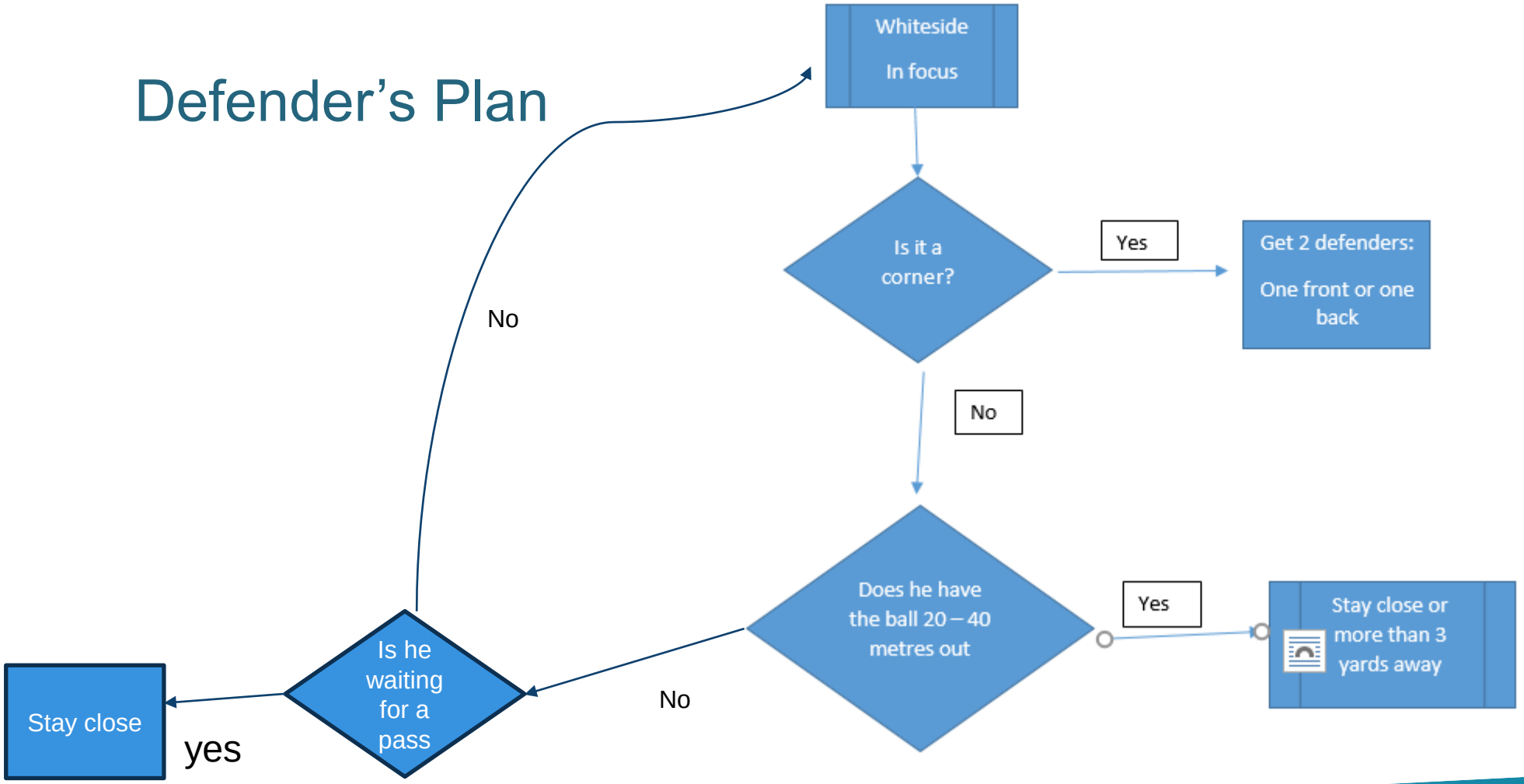
Test Cases can be simply determined as conditions that a tester will check whether the code runs perfectly or not.

https://www.geeksforgeeks.org/types-software-testing/

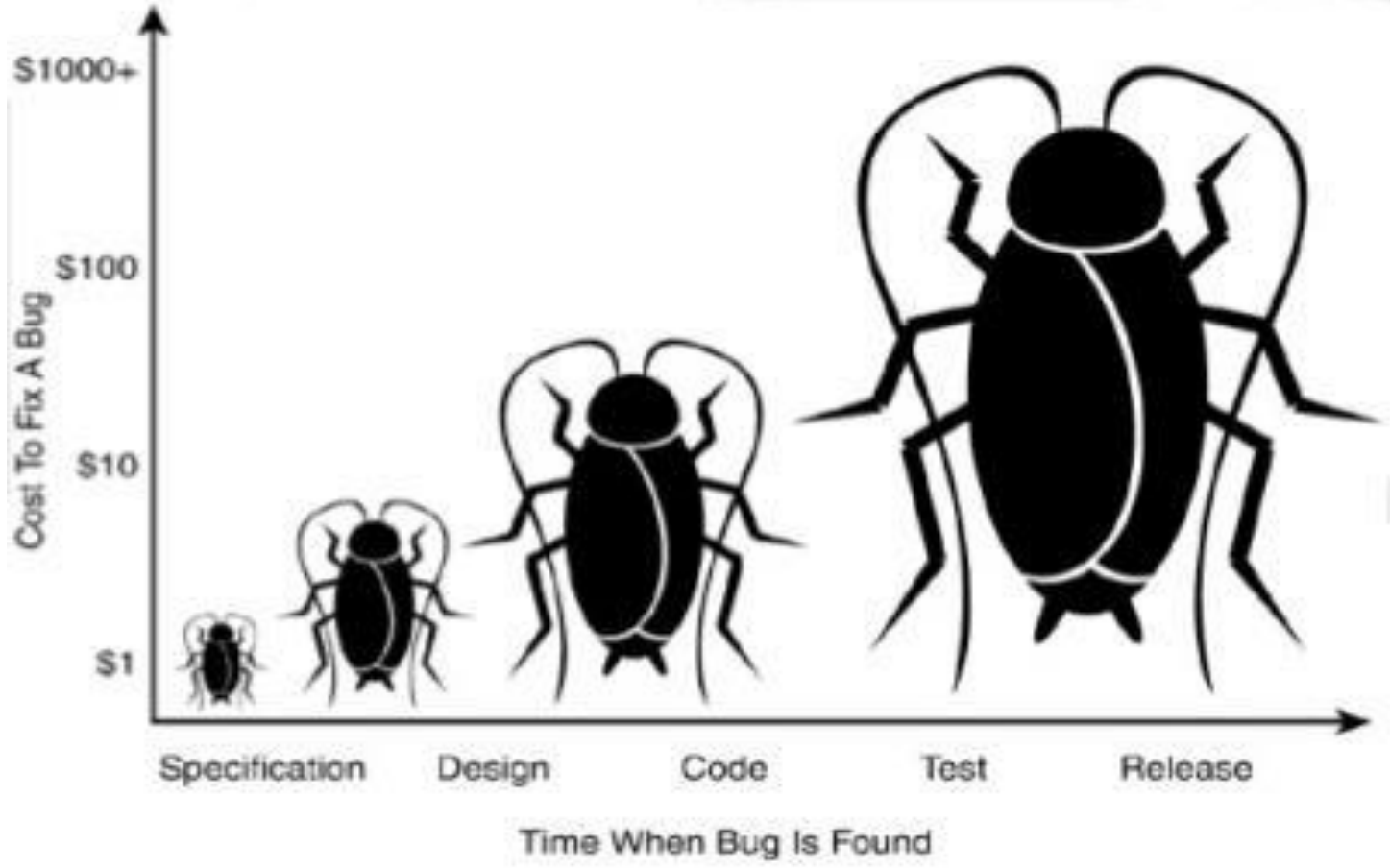
| Tester ID | Description of what is being tested | Test Data | Expected Result | Actual Result | Test Passed Y/N |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |

# Norman Whiteside’s goal in the FA Cup 1985

# Testers are similar to defenders



# Bug fixing cost




(www.systemsemantics.com)

# What is Software Testing?

“Testing shows the presence, not the absence, of bugs”… All that tests can do, after sufficient testing effort, is allow us to deem a program to be correct enough for our purposes.”

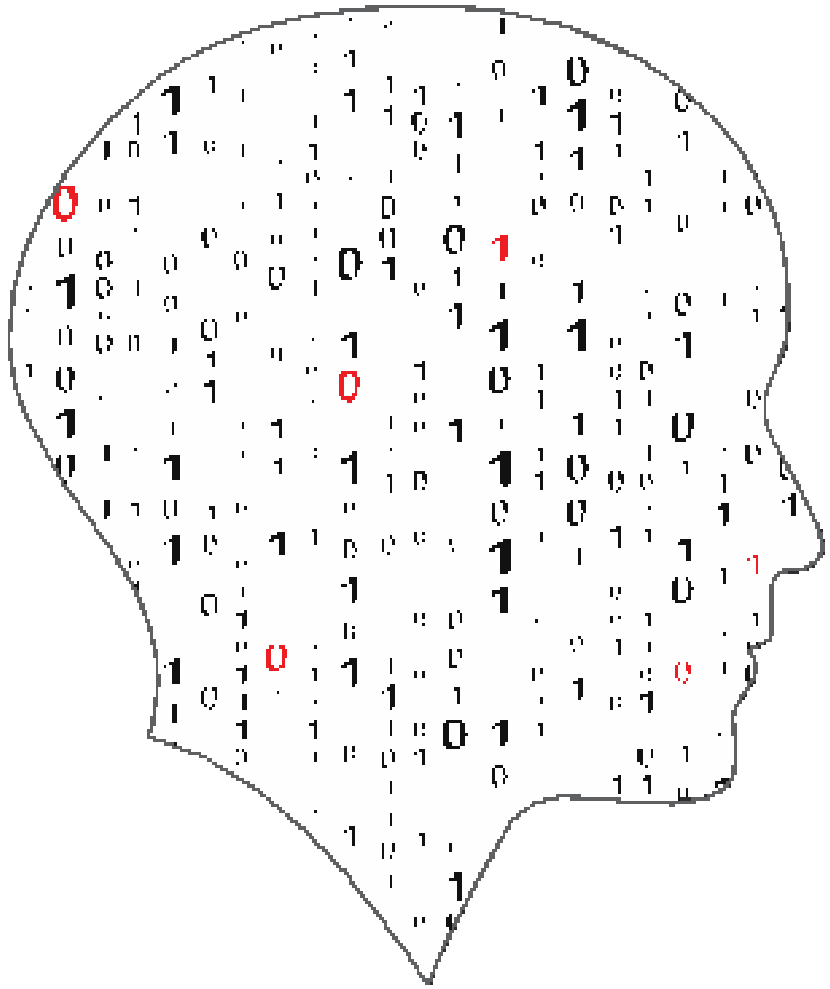

**Clean Architecture**: A Craftsman's Guide to Software Structure and Design (Robert C. Martin Series) 1st Edition

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

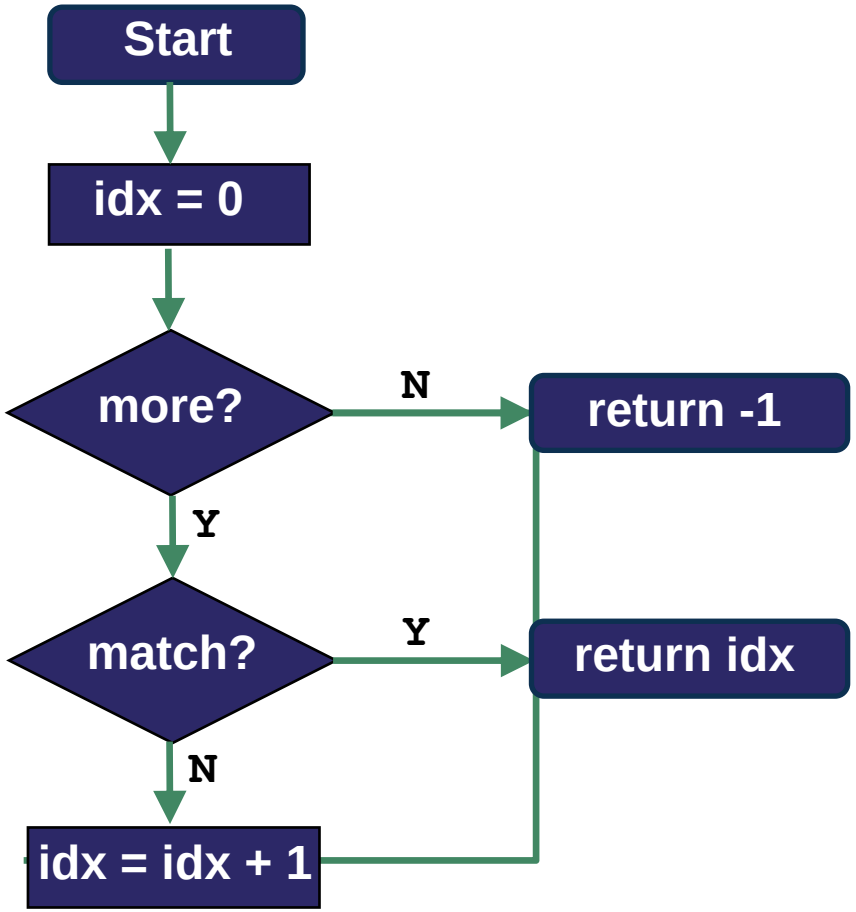
Supporting the Professional Learning of School Leaders and Teachers

# 2. Software Testing



LEAVING CERTIFICATE
COMPUTER SCIENCE

# Testing the Linear Search Algorithm




| idx | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| values | 21 | 17 | -1 | 26 | 22 | -5 | 24 |

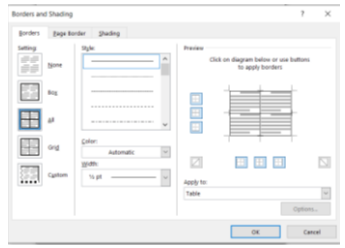
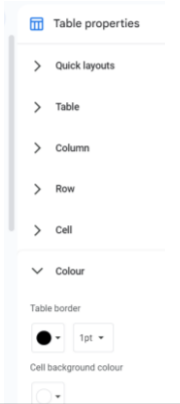
P 7

Is searching for 26 a sufficient test of this algorithm?

Take a moment to note your answer in the booklet.

# Activity 2

investigate testing the functionality of the Shading Tab in the Borders and Shading dialog box in a MS Word document/ Border Colour and Cell Background in a Google Document by testing several possible combinations of colours, border size and types.

| MS Word<br>Test the functionality of the Shading Tab in the Borders and Shading dialog box | Google Document<br>Test the functionality of the Border Colour and Cell Background using *Table Properties.* |
|---|---|
|  |  |

7 Minutes

Consider the questions:

- Can I test every possible combination of colours, border size and types?

- Do I need to prioritise what I need to test and if so, how would I do that?

P 8

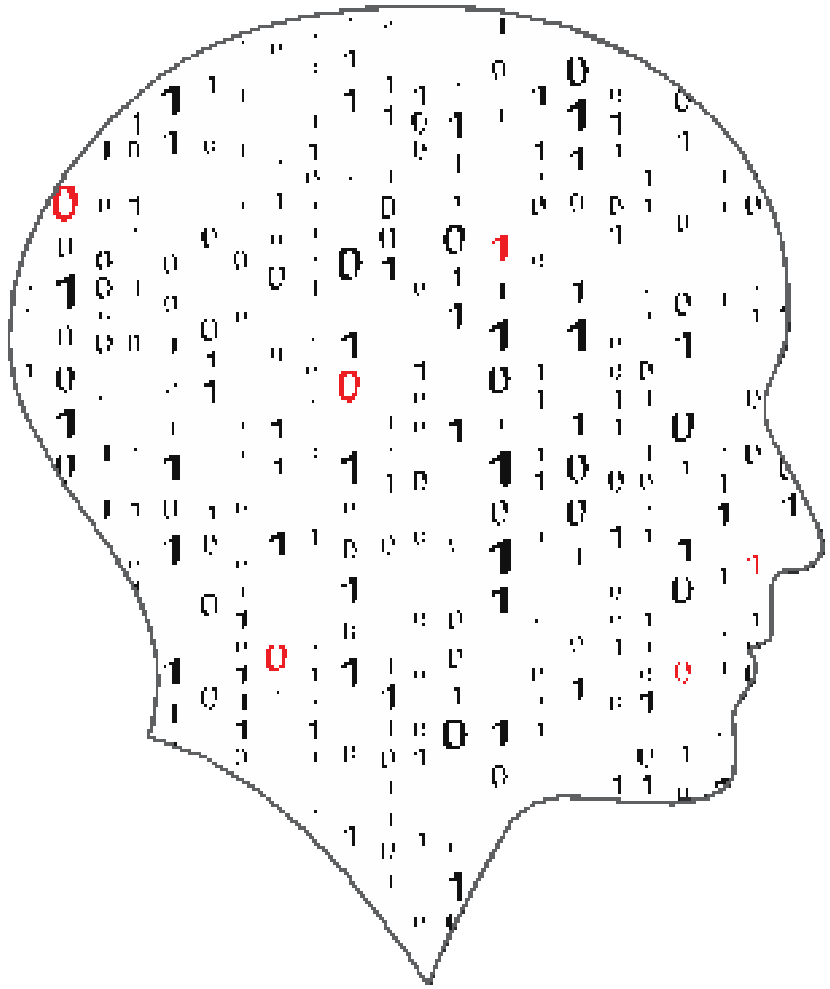
# Feedback

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# 3. Types of Testing/ Testers' roles



LEAVING CERTIFICATE COMPUTER SCIENCE

# Activity 3: Home Expert Activity

P 10

Investigating types of testing.

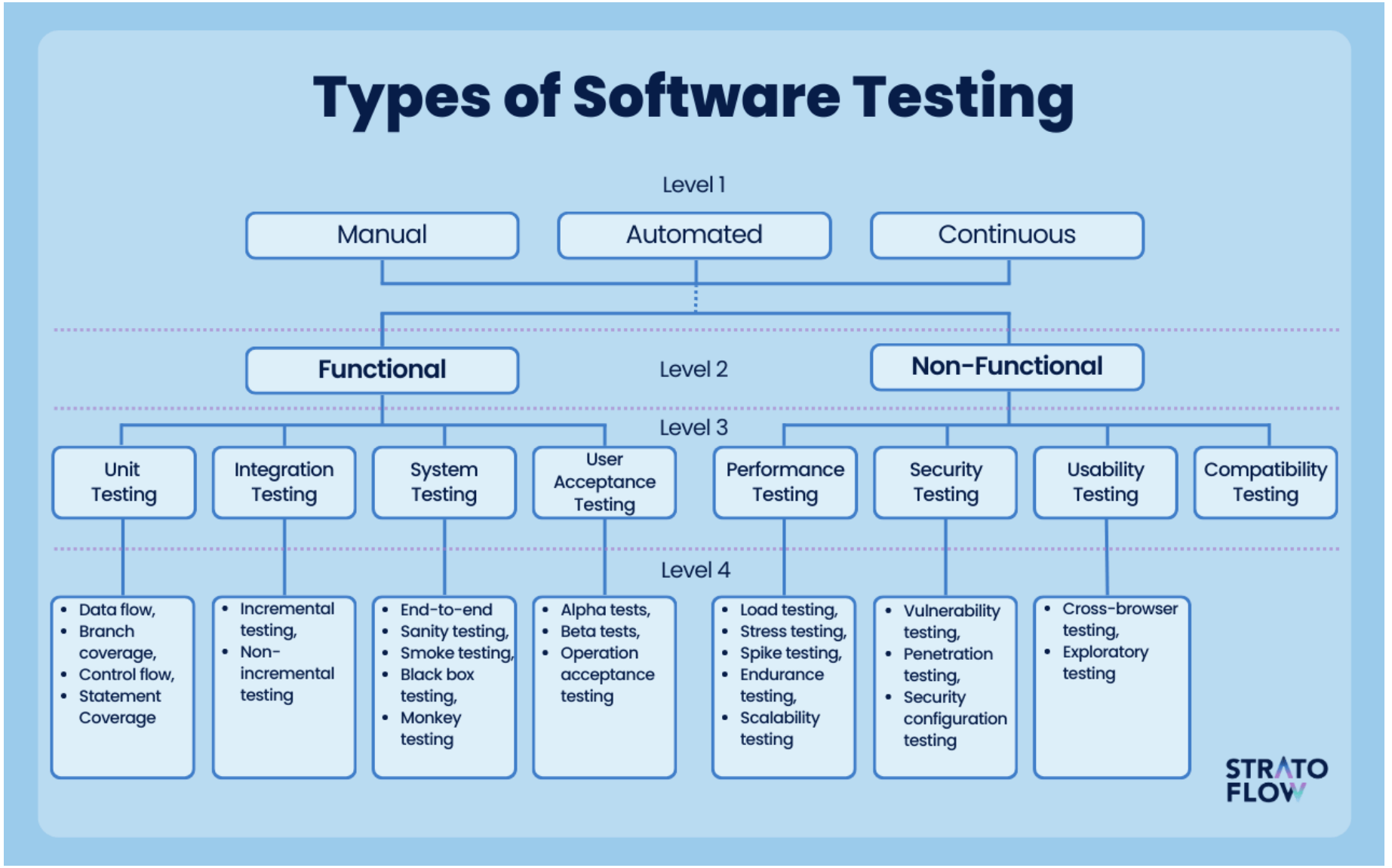
You could consider what the testing involves with an example and who carries out the test.

1. **Functional** v non-functional testing
2. Unit v Integration testing
3. **System** v User Acceptance Testing
4. White v Black box testing
5. Alpha v Beta Testing
6. Usability v Security Testing
7. Regression v Smoke Testing
8. Accessibility v Stress Testing
9. Other

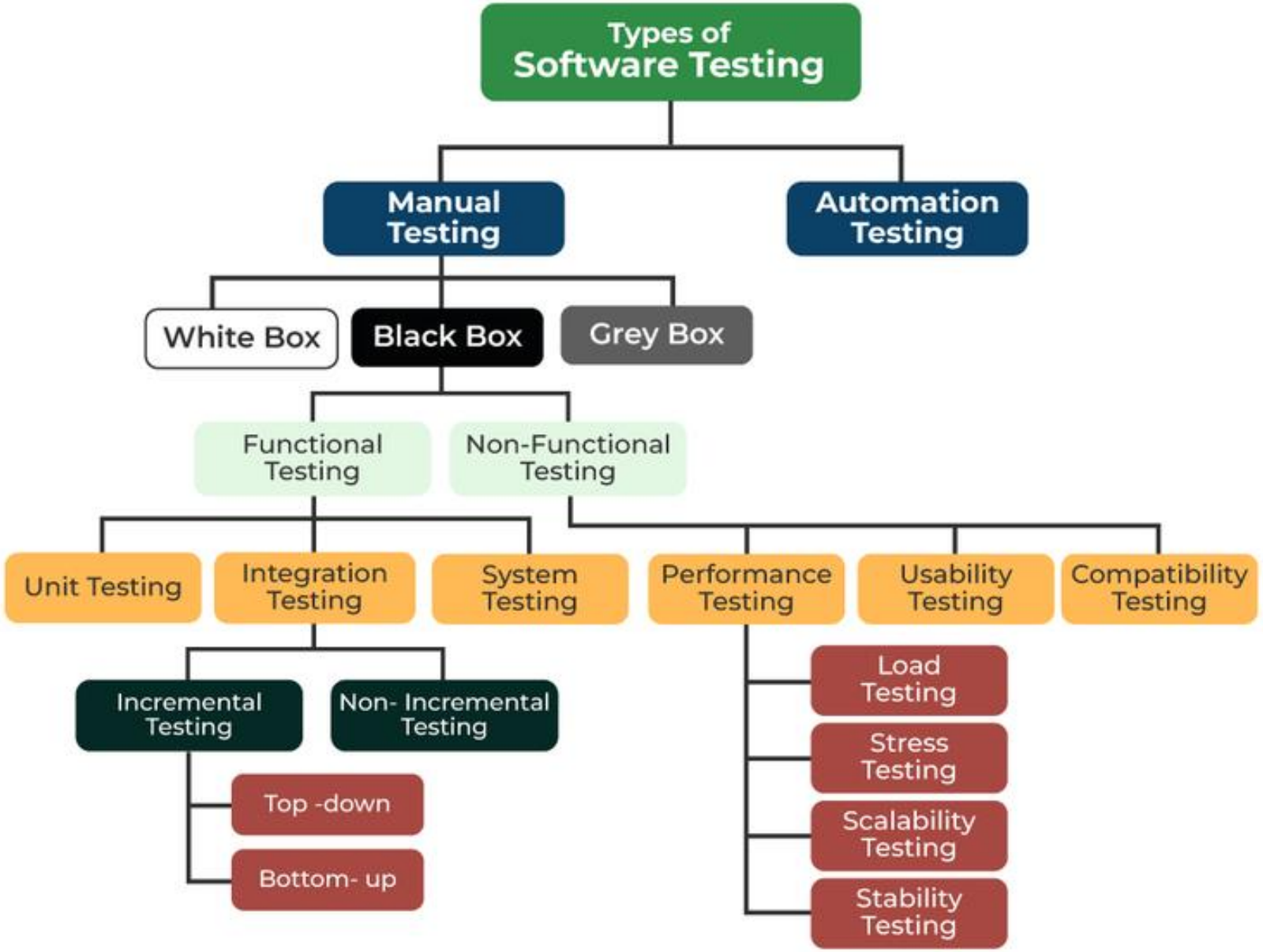
15 Minutes

Appoint a chair, a timekeeper, a notetaker and a **spokesperson**

# Feedback

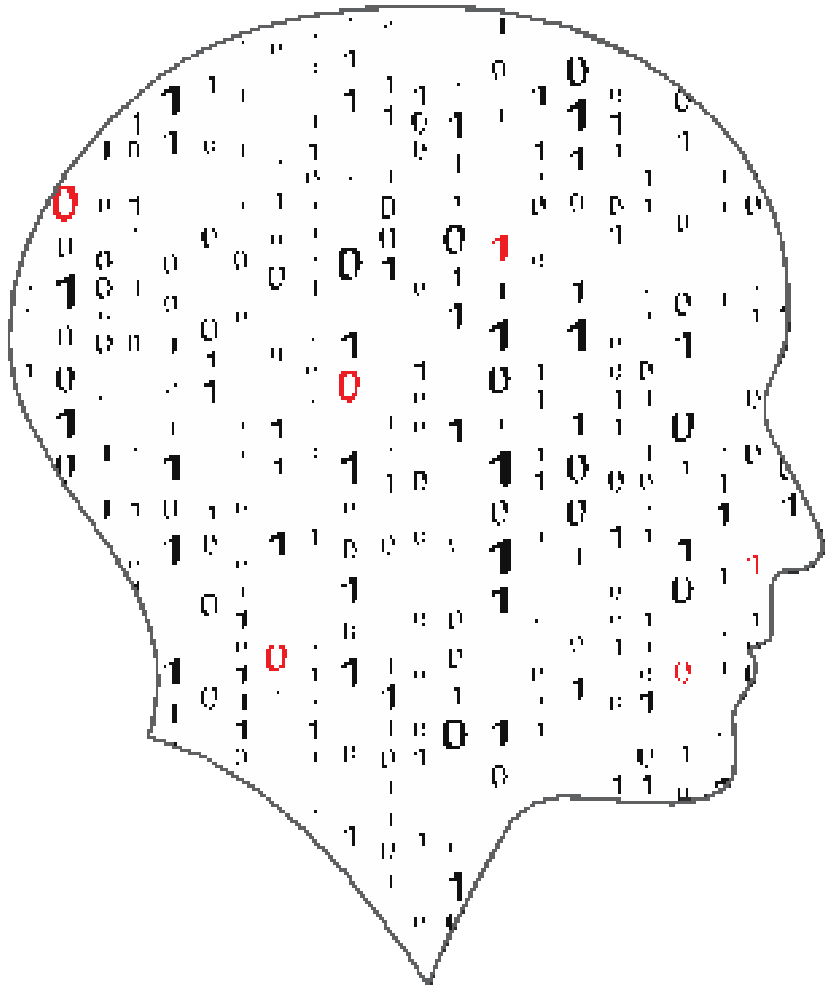



https://stratoflow.com/types-of-software-testing/




https://www.geeksforgeeks.org/types-software-testing/

# 4. Test Driven Development (TDD)

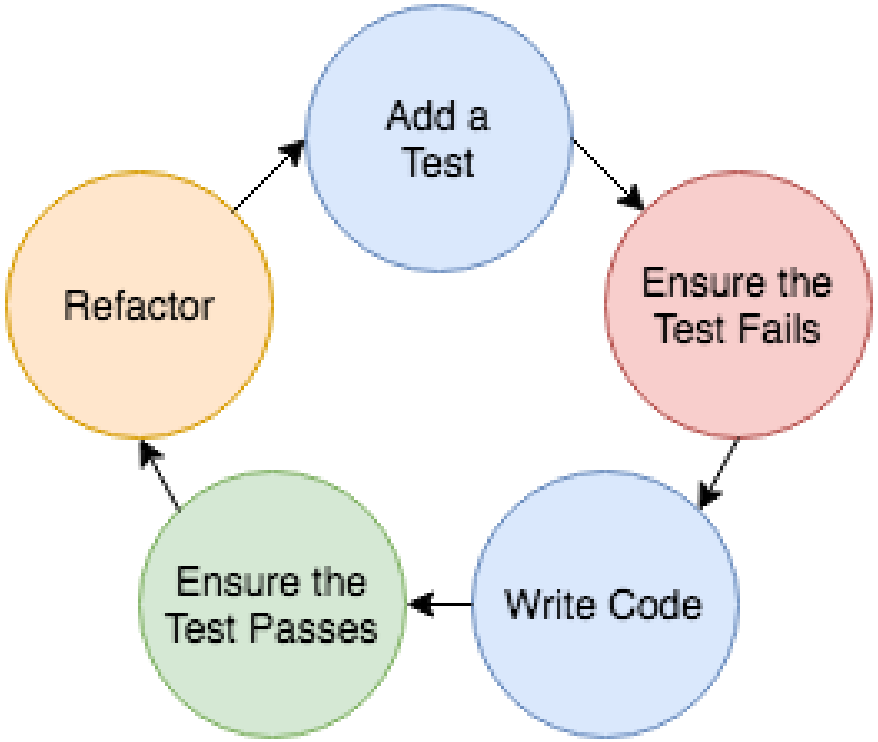


LEAVING CERTIFICATE
COMPUTER SCIENCE

# Test Driven Development (TDD)

TDD usually follows the "Red-Green-Refactor" cycle:

1. Add a test to the test suite
2. (Red) Run all the tests to ensure the new test fails
3. (Green) Write just enough code to get that single test to pass
4. Run all tests
5. (Refactor) Improve the initial code while keeping the tests green
6. Repeat




https://testdriven.io/

“Fake it till you make it” (Kent Blake): Write only code needed to pass tests.

# TDD & Software Development Models

- Does TDD fit in the Agile framework?
- Is TDD the same as Agile development?
- Is Waterfall development relevant?
- How does the role of testing change, according to these methodologies?

5 Minutes

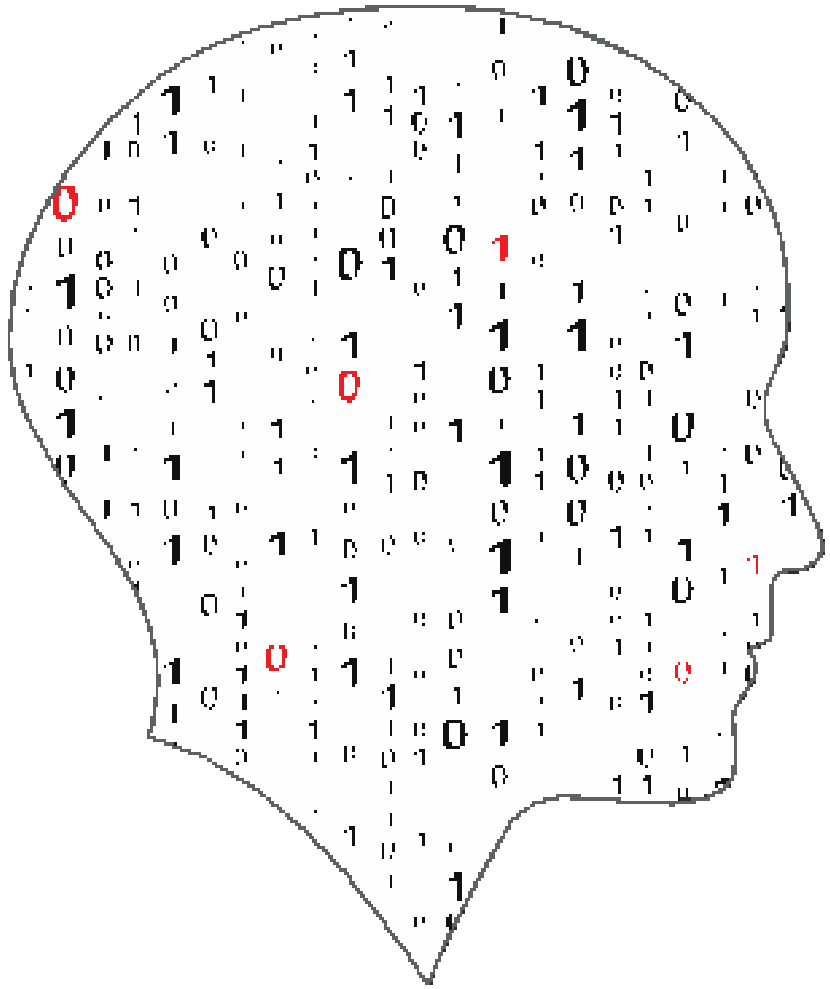
# Feedback

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# 5. Reflection



LEAVING CERTIFICATE
COMPUTER SCIENCE

# Reflection

- From what we covered today, what could you bring to your classroom?
- How could you incorporate testing into future ALTS?

P 12

Oide

# Resources

COMPSCI.IE


GeeksforGeeks
A computer science portal for geeks

Medium

Oide

Tacú leis an bhFoghlaim
Ghairmiúil i measc Ceannairí
Scoile agus Múinteoirí

# Lunch

1-2

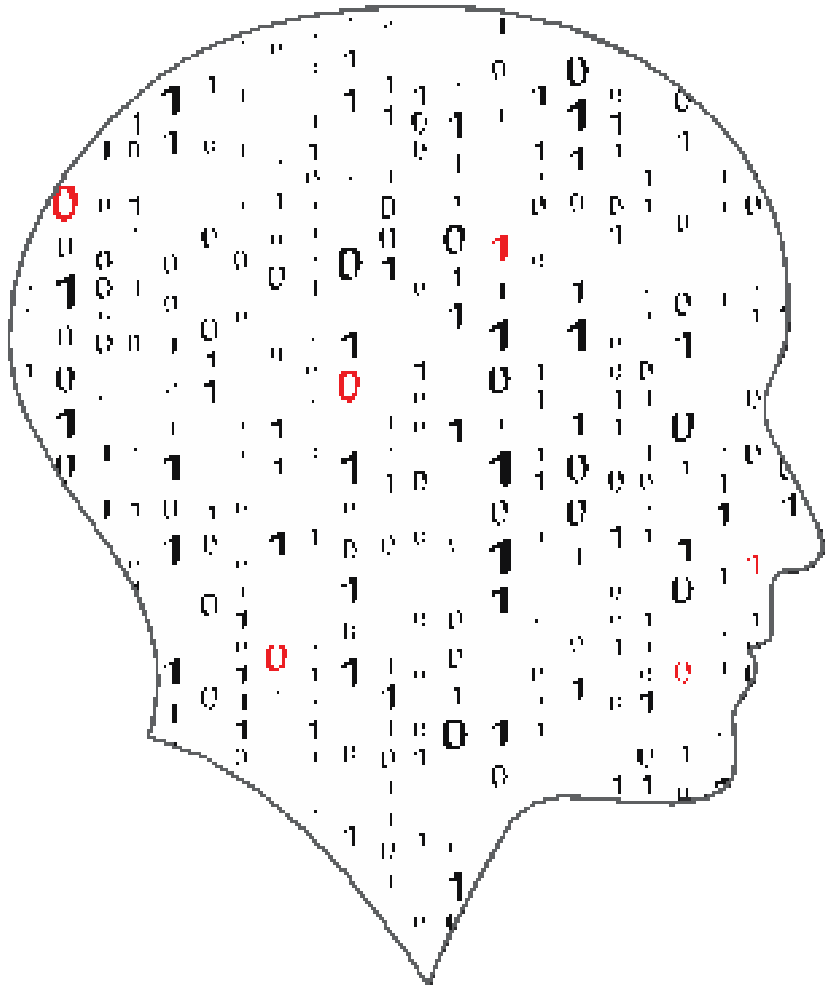
Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# NW6 Session 3

## Digital portfolios & Coursework video



LEAVING CERTIFICATE COMPUTER SCIENCE

# By the end of this session

Participants will have been enabled to:

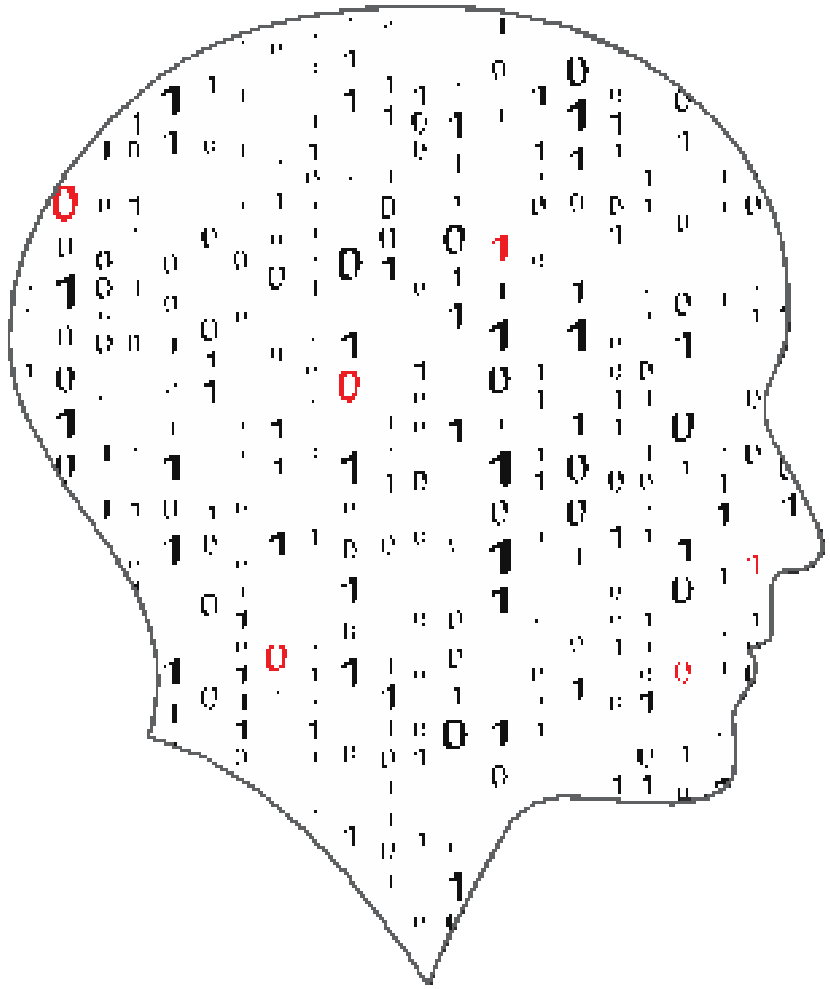
- develop an understanding of digital portfolios
- recognise how the use of Digital Portfolios is supported by educational policy
- make the link with Digital Portfolios and the LCCS subject including a ‘show and tell’ from teachers
- develop a shared understanding of the video component for the LCCS Coursework Assessment
- access a range of software relevant to the video production process

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# Digital Portfolios for ALTs



LEAVING CERTIFICATE
COMPUTER SCIENCE

# Digital Strategy for Schools to 2027

"Using digital technologies actively and working with other learners promotes and encourages active learning, problem solving, critical thinking and communication skills, all of which are vital for the world we live in."

Digital Strategy for Schools to 2027, 1.3 A Learner-Centred Approach

# Digital Strategy for Schools to 2027

"The use of digital portfolios in teaching and learning has also grown, providing a platform for student centred learning, particularly in junior cycle and Transition Year."

Digital Strategy for Schools to 2027, 1.13 Assessment

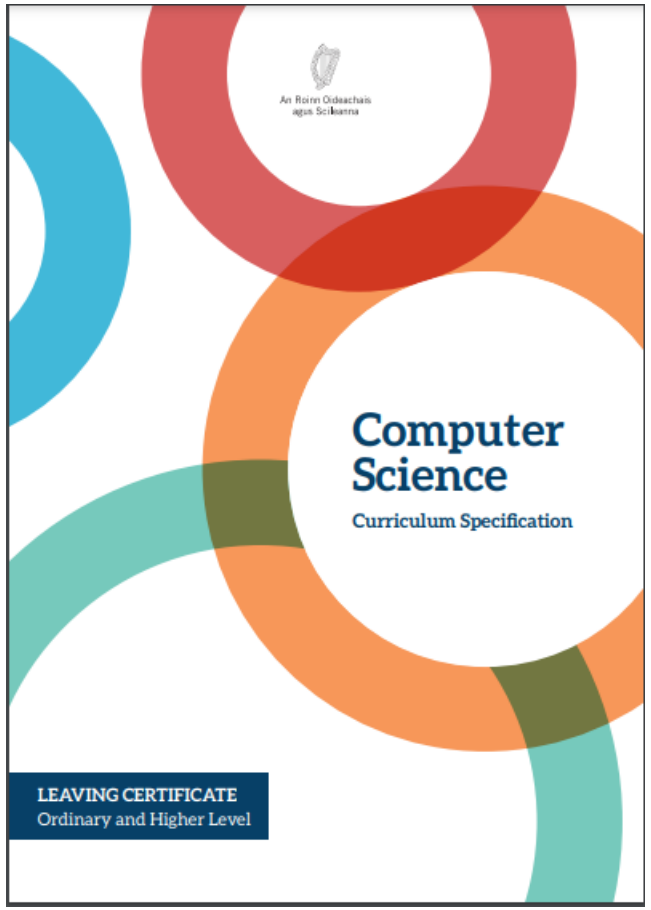
# Leaving Certificate Computer Science

*"The output from each task is a computational artefact and a concise individual report outlining its development."*

*"In the report, students outline where and how the core concepts were employed. The structure of the reports should reflect the design process."*

*"As students progress, reports should become detailed and individual."*

*"Reports are collected in a **digital portfolio** along with the computational artefact."*



# Developing Key Skills


How can the Key Skills of Senior Cycle be developed through the use of digital portfolios?

# Digital Portfolios: Warm-up Activity

How can the Key Skills of Senior Cycle be developed through the use of digital portfolios?

What is your current understanding of digital portfolios?

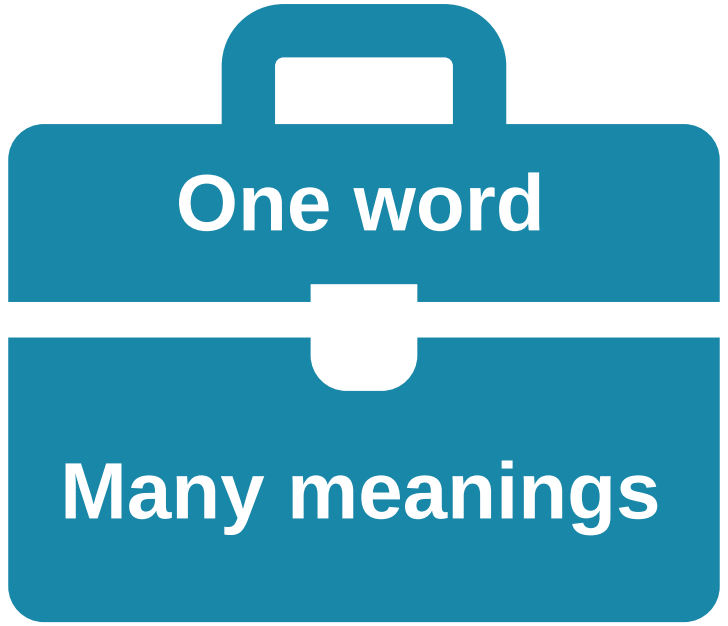

# What is a Portfolio?

*"A portfolio - paper or electronic - is a **collection of evidence** that is gathered together to show a person's **learning journey over time** and to demonstrate their abilities.*

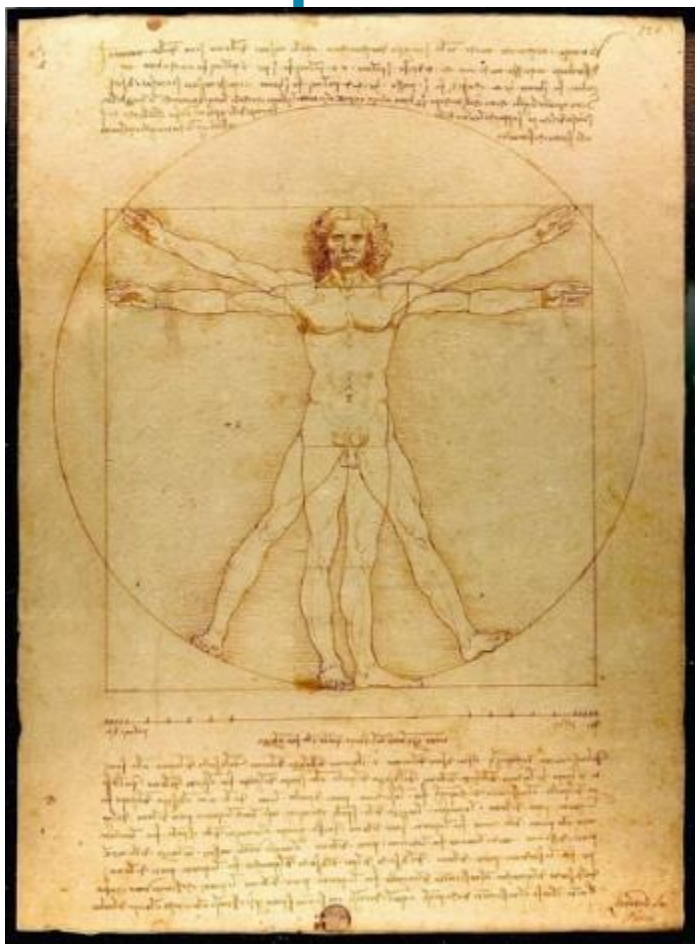
*In that way, people compiling portfolios are active participants in their own learning."*

(EUFolio, 2015, p9)

https://eufolioresources.files.wordpress.com/2015/03/eportfolio-implementation-guide_en.pdf



# Who was the first famous folio keeper?



# Defining Digital Portfolios

*“Digital portfolios are student-owned dynamic digital workspaces whereby students can capture their learning, their ideas, access their collections of work, reflect on their learning, share it, set goals, seek feedback and showcase their learning and achievements.”*

*NCCA, 2013*

# Benefits of Digital Portfolios

- Evidence of Learning
- Skill Development
- Reflection
- Portability and Sharing
- Assessment
- Artefacts
- Maintenance
- Feedback
- Access

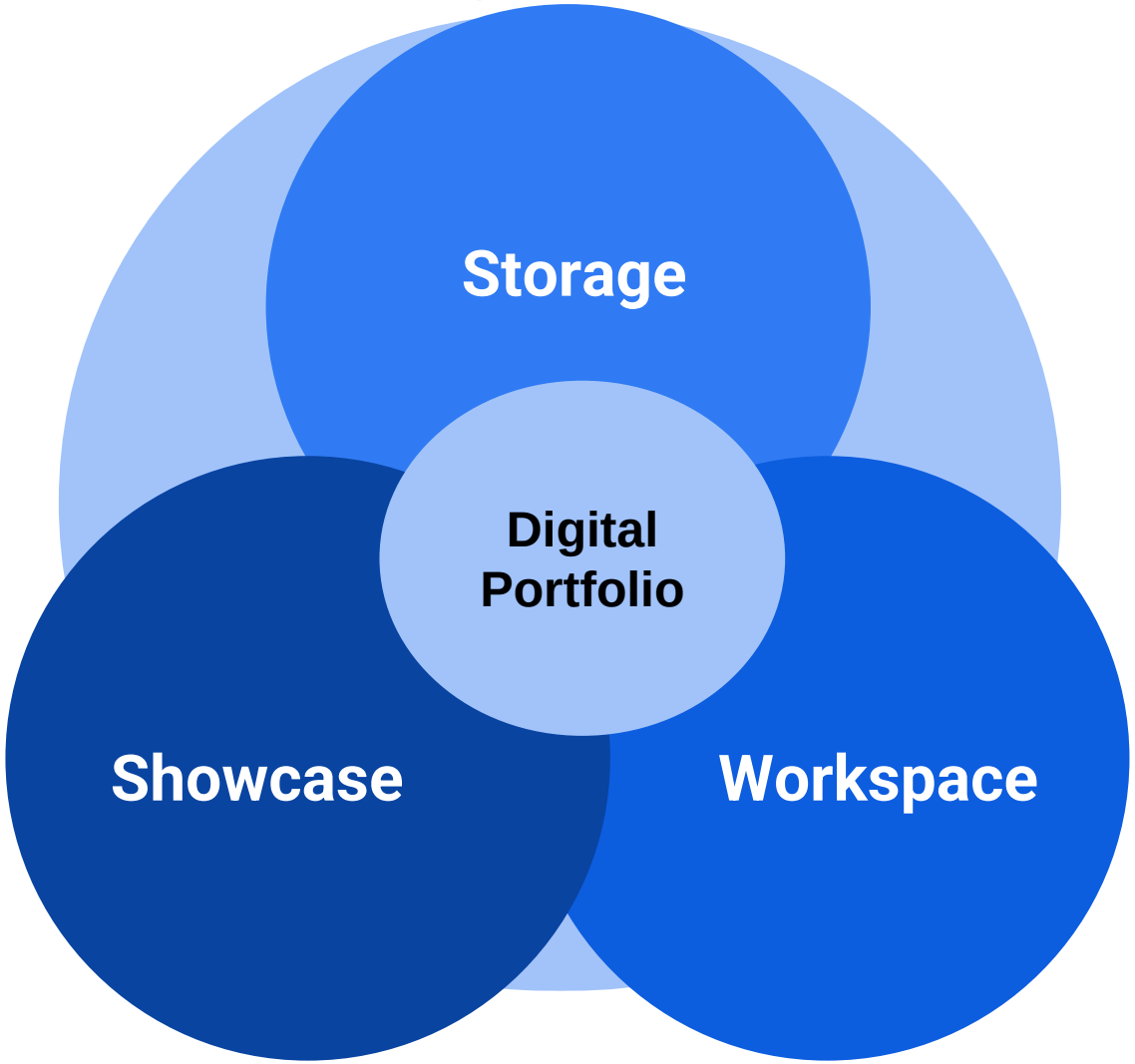
# Three Levels of Digital Portfolio

Oide

OneDrive

OneNote

Microsoft Teams

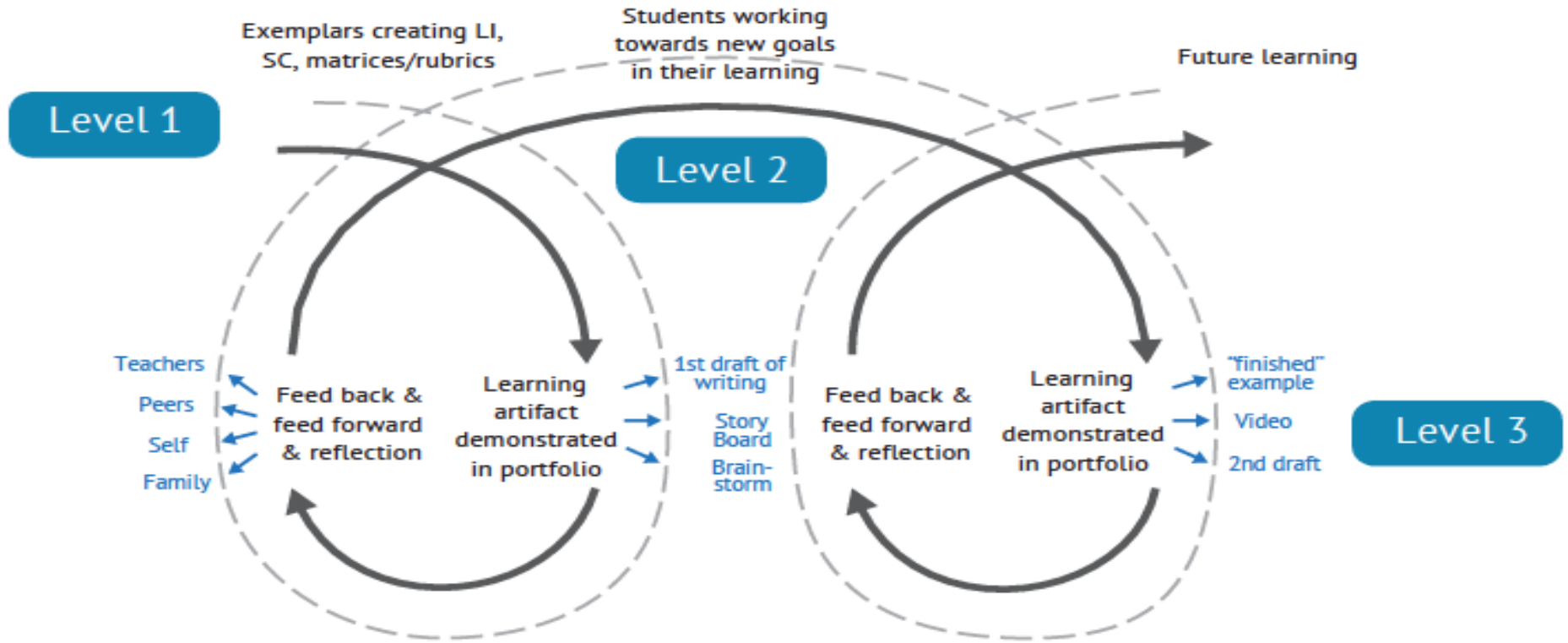



Google Drive

Google Classroom

Google Sites

# Using Digital Portfolios to support Formative Assessment



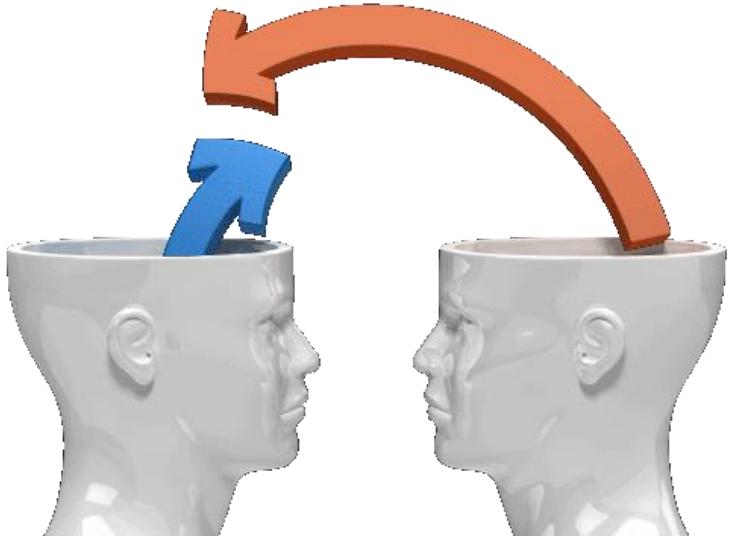

EUFolio (2015)

*Adapted from: Assessment for Learning and ePortfolios (2012)*

# Key Message

"Reflections and relationships are at the heart and soul of digital portfolios…. not the technology."

*Dr. Helen Barrett*



# Breakout Task

In your breakout group, discuss and plan a strategy for incorporating digital portfolios into the learning plan for your LCCS classroom.

# Feedback

In your breakout group, discuss and plan a strategy for incorporating digital portfolios into the learning plan for your LCCS classroom.

# Final Tips for LCCS Digital Portfolio

- Use the existing platform that is in your school already - you can combine it with other tools or platforms for code if needed.
- Keep it simple and start small first.
- Discuss with students what they are doing & WHY!
- Show a finished one to help students visualise it (current 6th years can be shown to 5th years).

# Final Tips for LCCS Digital Portfolio

- Make links with Transition Year if they have completed a portfolio already.
- Encourage regular updating and reflection on each section/task/ etc.
- Share students' digital portfolios so they can see each other's work.
- Share with the wider school community and at open days etc to showcase LCCS at your school.

Oide

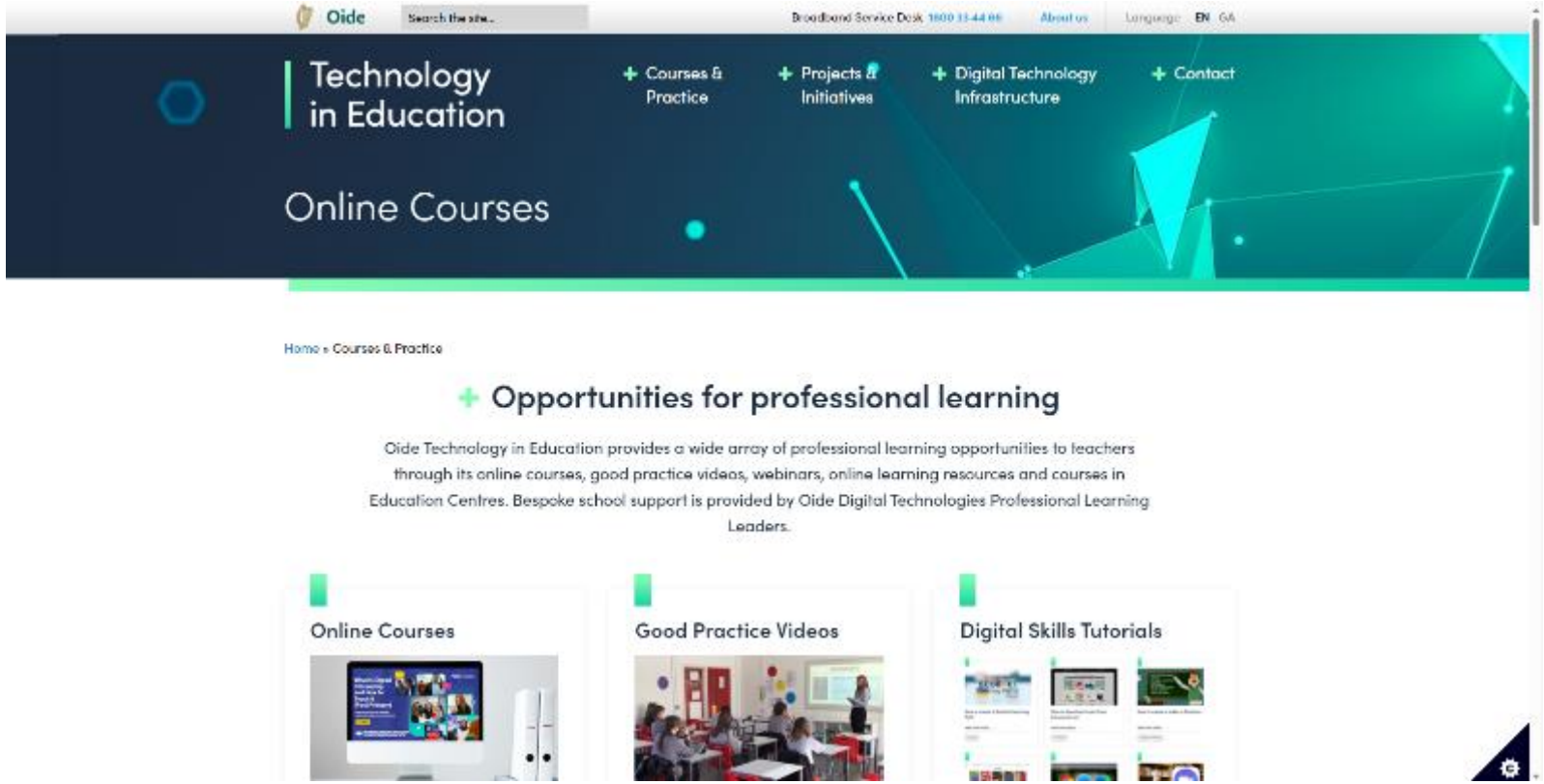
# Resources – Digital Portfolios

EportfolioIreland

Connect
Ignite
Elevate

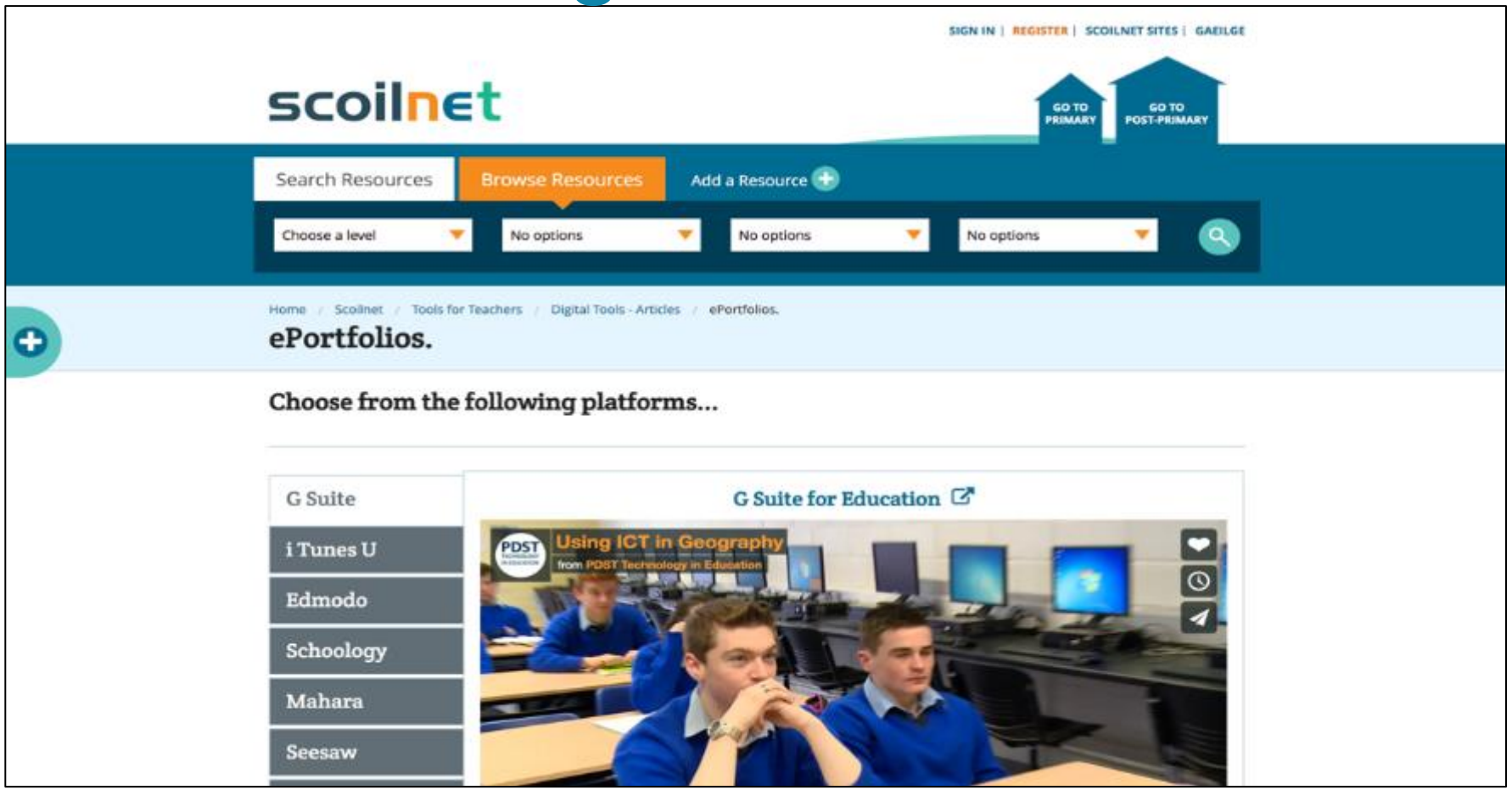
COMPSCI.IE

# Resources – Digital Portfolios




https://www.oidetechnologyineducation.ie/courses-practice/

# Resources – Digital Portfolios




https://www.scoilnet.ie/tools-for-teachers/articles/eportfolio/

Oide

Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# LCCS Coursework Video

LEAVING CERTIFICATE COMPUTER SCIENCE

*You must embed a video presentation showing the artefact in operation.*

The video must not be more than 5 minutes in duration. In deciding the content of your video, you should refer to the description of the task.

The video should show all the features of the artefact that you want the examiner to be aware of, as **this is the main piece of evidence on which the examiner will judge the quality of the artefact.**

All content of the report (images, video or other) must comply with the school's Acceptable Usage Policy and with General Data Protection Regulation (GDPR).

The video should demonstrate the quality of the user interface and the full functionality of the artefact.

The video should be no more than 1GB in size. This readily can be achieved by using standard definition (720 x 480) at 25 frames per second and a suitable commonly used format.

# What would you use to record and edit video with your students?

Oide

SCREENCAST MATIC

Clips for iOS

Camtasia®

kdenlive

iMovie

Pr

VideoPad Video Editor

Adobe Spark

OBS Studio

Screencastify

# Deciding the Coursework Video Tool

- There is no set tool to use - it is your choice.

- It will depend on:
  - Context of your own classroom.
  - Technology already available in your school (equipment and process for CBAs).
  - Your own experience with tools.
  - Student's view or previous experience.

# Planning Tips for the Video

- Planning is essential.
- Use a storyboard.
- Use the native video capture on mobile or tablet device.
- Tripod is good for some shooting.
- Consider music / voiceover / subtitles carefully.
- Consider light when shooting.
- Chat to other teachers in your school for assistance.
- Give guidance but allow students to take ownership of video.

# Planning Tips for the Video

- Record video throughout the process
- Save video files/images/etc. regularly
- Gather all your assets in one place before editing
- Exporting edited video can take a long time
- Allow plenty of time for editing and finishing touches
- Videos may not always work - use VLC (videolan.org) to check if the video is still there and usable
- Allow plenty of time for editing and finishing touches
- If your video is too big, use HandBrake to shrink it
- Allow plenty of time for editing and finishing touches

Oide

# Recording & Editing Video

PowerPoint Screen Record

Screencastify

OBS Open Broadcaster Software

Clipchamp

kdenlive

Filmora

# Final fixes - Handbrake

For putting the finishing touches to your video:

- Resolution
- Filetype
- Crop (top/bottom/left/right)
- Trim (start/finish)
- Reducing file size

Oide

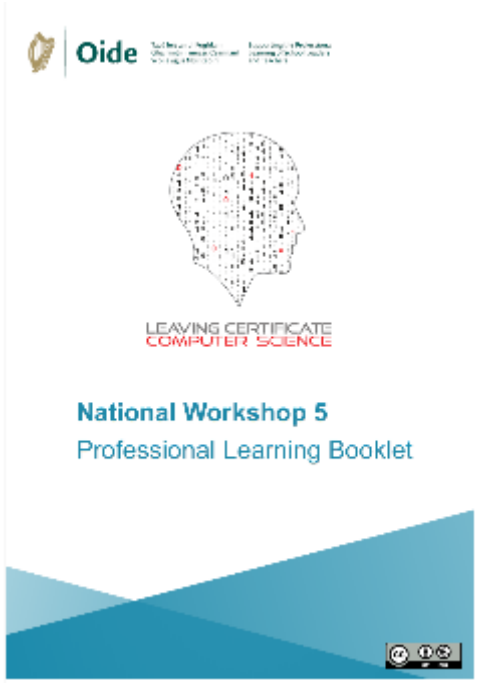
Tacú leis an bhFoghlaim Ghairmiúil i measc Ceannairí Scoile agus Múinteoirí

Supporting the Professional Learning of School Leaders and Teachers

# Coursework Video Demo

LEAVING CERTIFICATE
COMPUTER SCIENCE

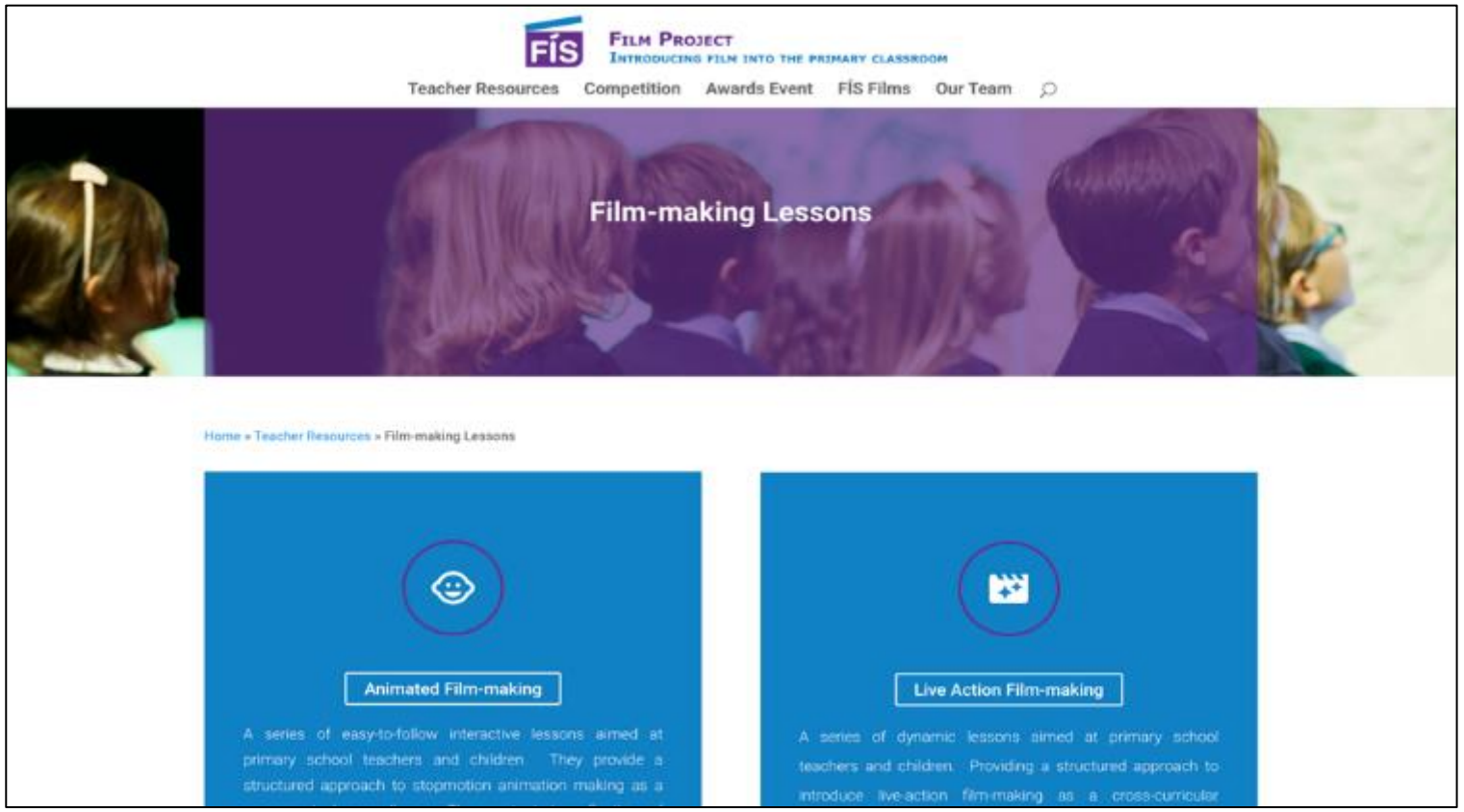
# Resources – Video for Coursework




https://fisfilmproject.ie/

COMPSCI.IE

# Resources – Video for Coursework




https://fisfilmproject.ie/teacher-resources/

Oide

An Roinn Oideachais
Department of Education