Student ID: Student Name:

Course: Data Structures (CSE CS203A)

Assignment V: Tree

Due date: 2025.12.30 23:59:59

Important Notice – Use of AI Tools

In this assignment, you must use at least one AI assistant (e.g. ChatGPT, Gemini, Claude, Grok, M365 Copilot) as a learning tool to help you:

- review definitions,
- compare tree variants, and
- organize your report.

You are not allowed to let the AI directly produce your final diagrams or final report content without your own understanding and rewriting.

You must log all AI prompts and services used (see "AI Usage Log" section below).

1. Goal of This Assignment

In the lectures, we introduced the concept of the tree as a data structure, starting from the general tree and then moving to more specialized forms.

In this assignment, you will:

a. Understand and clearly define:
   1. General tree
   2. Binary tree
   3. Complete binary tree
   4. Binary search tree (BST)
   5. AVL tree
   6. Red-Black tree
   7. Max heap
   8. Min heap

b. Build a hierarchy and transformation path from the general tree to these variants, and explain how each variant adds more structure or constraints.

c. Use a fixed list of integers to construct multiple tree variants and visualize them.

d. Choose one real-world application for each tree type and explain why that data structure fits the application.

e. Practice using AI tools as study companions and keep a simple Q&A log.

2. Given Data

Use the following 20 integers as the input data for all your tree constructions:

37, 142, 5, 89, 63, 117, 24, 176, 58, 133, 92, 11, 151, 72, 39, 184, 7, 101, 54, 160

Student ID: Student Name:

You will reuse this same sequence for every tree type (binary tree, complete binary tree, BST, AVL, Red-Black, max heap, min heap).

3. Deliverables

Please complete your work in the Student Worksheet Companion and upload it to the YZU Portal System.

Your report should include the following parts:

a. Definitions (Concept Review)

Provide clear, concise definitions for each of the following:

1. General tree
2. Binary tree
3. Complete binary tree
4. Binary search tree (BST)
5. AVL tree
6. Red-Black tree
7. Max heap
8. Min heap

You are encouraged to use AI tools to help you understand these concepts, but you must rewrite the definitions in your own words.

b. Hierarchy and Transformation of Tree Variants

Based on the definitions above, build a "tree family hierarchy" that shows how these structures are related. For example:

- General tree → Binary tree
- Binary tree → Complete binary tree / Binary search tree
- BST → AVL tree / Red-Black tree
- Binary tree → Max heap / Min heap

Tasks:

- Draw a diagram or flow chart that shows the transformation or specialization path: general tree → binary tree → complete binary tree → BST → AVL/Red-Black, etc.
- For each arrow (transformation), briefly explain:
  - What new constraint or property is added?
  - e.g., "Binary tree = tree with at most 2 children per node",
    "BST = binary tree with left < root < right",
    "AVL = BST with strict height-balance rule", etc.

c. Tree Construction with the Given Integers

Using the given 20 integers, construct the following tree variants:

1. Binary tree
2. Complete binary tree
3. Binary search tree (BST)

4. AVL tree
5. Red-Black tree
6. Max heap
7. Min heap

Important Hint / Restriction:

- For these trees, you must use tree visualization tools (e.g., online visualizers or software) to build and display the tree.
- You may not ask AI tools to directly generate the final tree pictures for you.
- Instead:
    - Use AI only to help you understand algorithms,
    - Then apply those algorithms in a visualizer (or your own implementation).

What to submit for this part:

For each tree type:

- A snapshot (image) of the constructed tree.
- The URL / name of the visualization tool you used.
- A short note on how you inserted the integers (e.g., "insert in the given order as BST", "build max heap using heapify", etc.).

d. Application Example for Each Tree

For each of the following:

1. Binary tree
2. Complete binary tree
3. Binary search tree (BST)
4. AVL tree
5. Red-Black tree
6. Max heap
7. Min heap

Choose one application (real-world or system-level) and explain:

1. Application description
    - e.g., priority scheduling, dictionary lookup, memory allocation, database indexing, etc.
2. Why this tree structure fits
    - What property of this data structure makes it suitable?
    - Example:
        - Max heap → good for priority queue because the largest element is always at the root, so extracting max is efficient.
        - Red-Black tree → good for standard library maps/sets because it guarantees O(log n) operations even under many insertions/deletions.

Your explanation should show that you understand the link between the data structure and its use case.

e. Report Layout and Organization

You are free to design the layout of your report, but it should:

- Be well-structured (use sections, headings, tables, and diagrams).
- Have a clear flow from:
  - definitions →
  - hierarchy/transformation →
  - constructed trees →
  - applications →
  - AI usage log.
- Be easy for another student to read and learn from.

Feel free to use AI to suggest a good outline, but you must decide and finalize the layout yourself.

f. AI Usage Log (Q&A Table)

Every time you use an AI copilot service for this assignment, record:

- Index (1, 2, 3, …)
- Prompt (what you asked)
- Service (e.g., ChatGPT, Gemini, Copilot, …)

Example log table:

Index	Prompt	Service
1	Assist me to have the definition of general tree, binary tree, complete binary tree, binary search tree, AVL tree, red-black tree, max heap and min heap for self-learning.	ChatGPT
2	Explain the difference between AVL tree and Red-Black tree in terms of balancing strategy and use cases.	Gemini
…	…	…

Place this table at the end of your report.

4. Evaluation (100 pts)

A possible breakdown (you can adjust if needed):

a. Concept definitions (20 pts)

- Correctness and clarity of all 8 tree type definitions.

b. Hierarchy & transformation explanation (20 pts)

- Clear diagram / explanation of how each tree variant evolves from the general tree.
- Correct identification of constraints/invariants.

c. Tree constructions & visualizations (25 pts)

- Correct constructions for each tree type using the given integers.
- Proper screenshots and tool URLs.

- Consistent insertion / heap-building strategy descriptions.

d. Applications & explanations (20 pts)
   - One application per tree type.
   - Clear explanation linking data structure properties to the application.

e. Report organization & AI usage log (15 pts)
   - Logical report structure and readability.
   - AI log completeness (all prompts listed with service names).
   - Thoughtful use of AI as a learning assistant, not as a copy-paste generator.

Student ID: Student Name:

Course: Data Structures (CSE CS203A)

Assignment V: Tree

Student Worksheet Companion

Due date: 2025.12.30 23:59:59

Academic Integrity and AI Usage Statement

In this assignment, you must use AI tools (such as ChatGPT, Gemini, Claude, Grok, M365 Copilot, etc.) as learning assistants, but you must also take full responsibility for understanding and organizing your own work.

1. Permitted Use of AI Tools

   You may use AI to:

   - Review or clarify definitions and concepts.
   - Compare different tree data structures.
   - Get suggestions for report layout or examples.
   - Ask for explanations of algorithms (e.g., BST insertion, AVL rotation, heapify process).

   You should read, think about, and rewrite the content in your own words.

2. Not Permitted
   - Do not copy/paste AI-generated content directly as your final answer.
   - Do not ask AI to draw the final diagrams or directly produce the final tree screenshots.
   - Do not ask AI to complete the whole assignment report for you.
3. Your Responsibility
   - You are responsible for understanding the definitions and algorithms.
   - You are responsible for verifying whether AI answers are correct or not.
   - You must produce your own original explanations and diagrams.
4. AI Usage Log
   - You must record all AI queries related to this assignment.
   - At the end of your report, include an AI Usage Log table with: Index, Prompt, AI service name.

   By submitting this assignment, you acknowledge that you have used AI tools only as study aids, and that the final content of this assignment represents your own understanding and work.

Section 1. Definitions of Tree Variants

Task: Write your own definitions for each tree type. You may use AI for learning, but rewrite in your own words.

Student ID: Student Name:

1. General Tree
   Definition: 最基本的樹結構，節點可以擁有任意數量的子節點，沒有任何形狀或是排序的規則限制。
2. Binary Tree
   Definition: 每個節點最多只有兩個子節點。

3. Complete Binary Tree
   Definition: 除了最後一層外，其餘各層皆為完全填滿（每個內部節點有兩個子節點），最後一層由左至右依序填入。

4. Binary Search Tree (BST)
   Definition: 一種 binary tree，且節點數值的大小關係滿足: 左子樹的所有節點 < 根節點 < 右子樹的所有節點

5. AVL Tree
   Definition: 一種自平衡的 binary tree，任一節點的左右子樹高度差不超過 1

6. Red-Black Tree
   Definition: 一種自平衡的 binary tree，透過節點顏色(紅/黑)與五大規則讓樹的高度大致平衡。

7. Max Heap
   Definition: 以 complete binary tree 為基礎，其父節點的數值永遠大於或等於子節點的數值，而根節點為最大值。

8. Min Heap
   Definition: 以 complete binary tree 為基礎，其父節點的數值永遠小於或等於子節點的數值，而根節點為最小值。

Section 2. Tree Family Hierarchy and Transformations

Task: Show how these structures are related (general → specialized). Use a simple diagram and explanations of what constraints are added at each step.

2.1 Tree Family Diagram

You may draw this by hand and paste a photo, or use drawing tools.

Suggested chain example (you may extend or adjust):

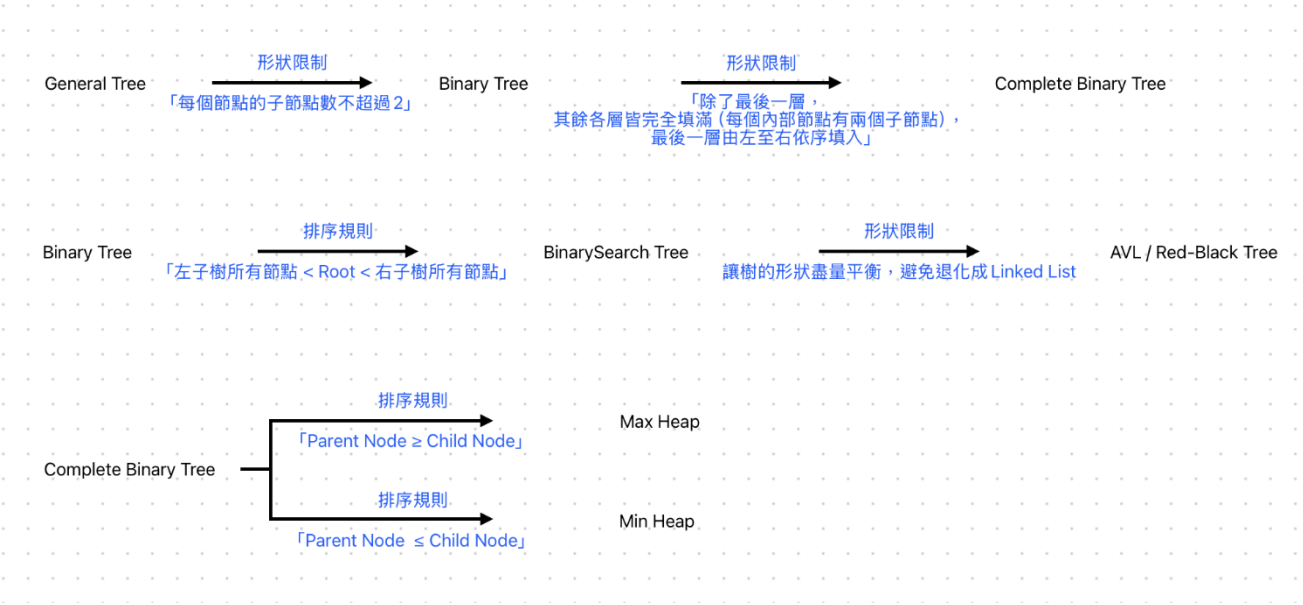
General Tree → Binary Tree → Complete Binary Tree

Binary Tree → Binary Search Tree → AVL / Red-Black

Student ID: Student Name:

Binary Tree → Max Heap / Min Heap

Your Diagram:



2.2 Explanation of Transformations

Fill in what new property or constraint is added at each step.

From	To	New property / constraint added
General Tree	Binary Tree	限制每個節點的子節點數最多為 **2**
Binary Tree	Complete Binary Tree	增加填入順序的限制，除了最後一層外，每一層的子節點皆為兩個，且最後一層由左至右填入
Binary Tree	Binary Search Tree	增加排序限制，大小關係滿足: 左子樹的所有節點 < 根節點 < 右子樹的所有節點
BST	AVL Tree	增加結構平衡的條件，任一節點的左右子樹高度差不超過 1
BST	Red-Black Tree	增加結構平衡的條件，讓樹的結構盡量平衡
Binary Tree	Max Heap	增加排序規則，父節點的數值永遠大於或等於子節點
Binary Tree	Min Heap	增加排序規則，父節點的數值永遠小於或等於子節點

Section 3. Tree Constructions Using Given Integers

Given integers (fixed for all parts):

37, 142, 5, 89, 63, 117, 24, 176, 58, 133, 92, 11, 151, 72, 39, 184, 7, 101, 54, 160

Student ID: Student Name:

Task: For each tree type below, construct the tree using these integers, take a screenshot of the tree from your chosen tool, record the tool name/URL, and describe the insertion / heap-building procedure.

3.1 Binary Tree

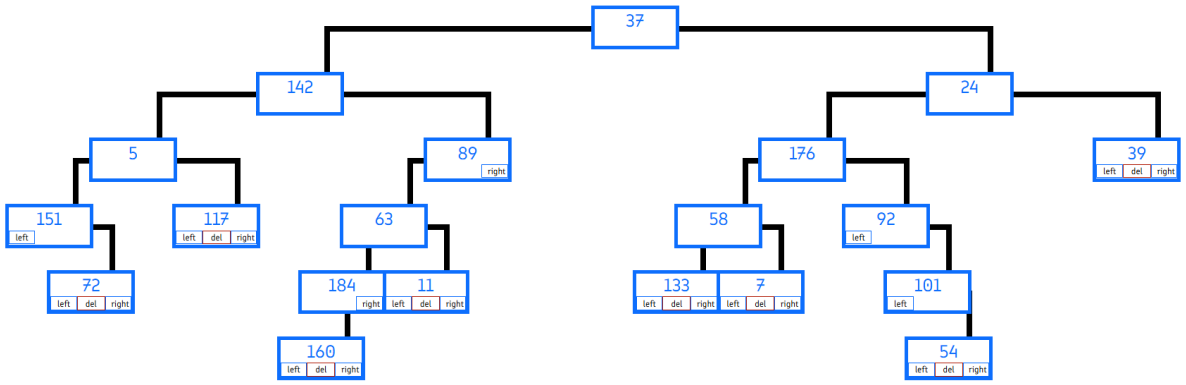
Tool name / URL:

LHS Binary Tree Builder / https://lhsbinarytree.vercel.app/

Construction / insertion description:

依照給定順序，手動插入節點，不考慮任何排序或平衡

Screenshot of Binary Tree (paste below):



3.2 Complete Binary Tree

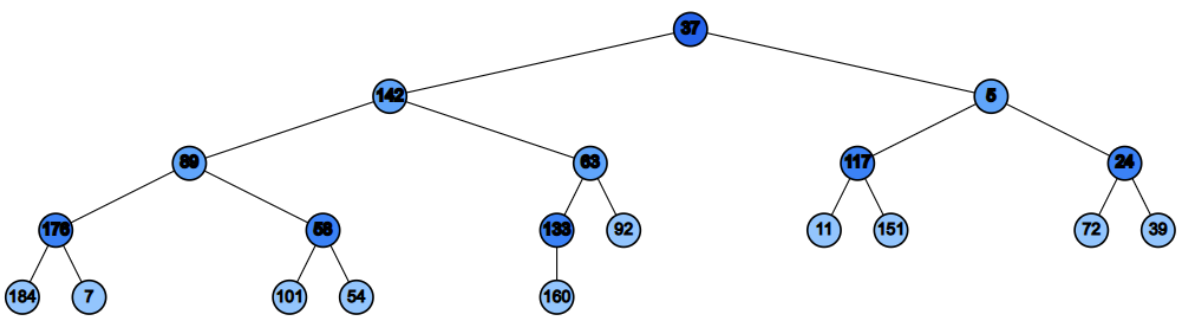
Tool name / URL:

TreeConverter / https://treeconverter.com/

Construction / insertion description:

依照給定順序輸入

Screenshot of Complete Binary Tree (paste below):



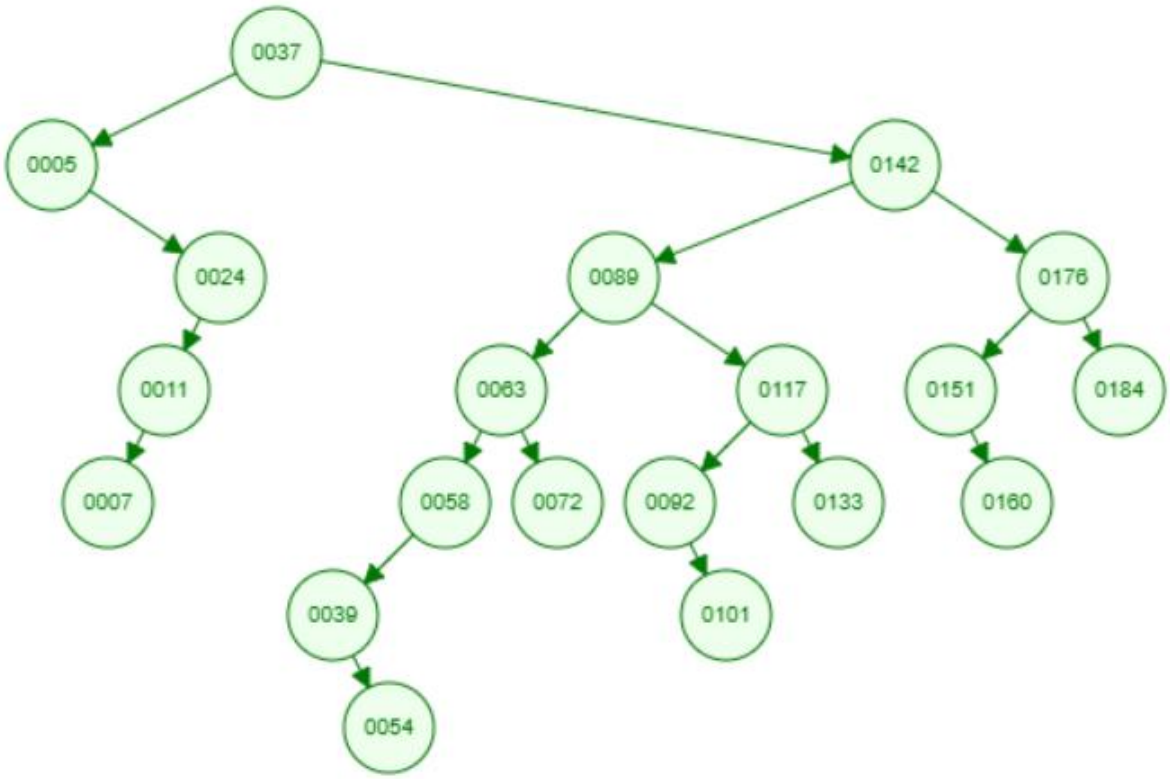
Student ID: Student Name:

3.3 Binary Search Tree (BST)
Tool name / URL:

USFCA Visualization / https://www.cs.usfca.edu/~galles/visualization/BST.html

Insertion rule (e.g., “insert in given order using BST rules”):
依照給定順序一個一個 Insert

Screenshot of BST (paste below):

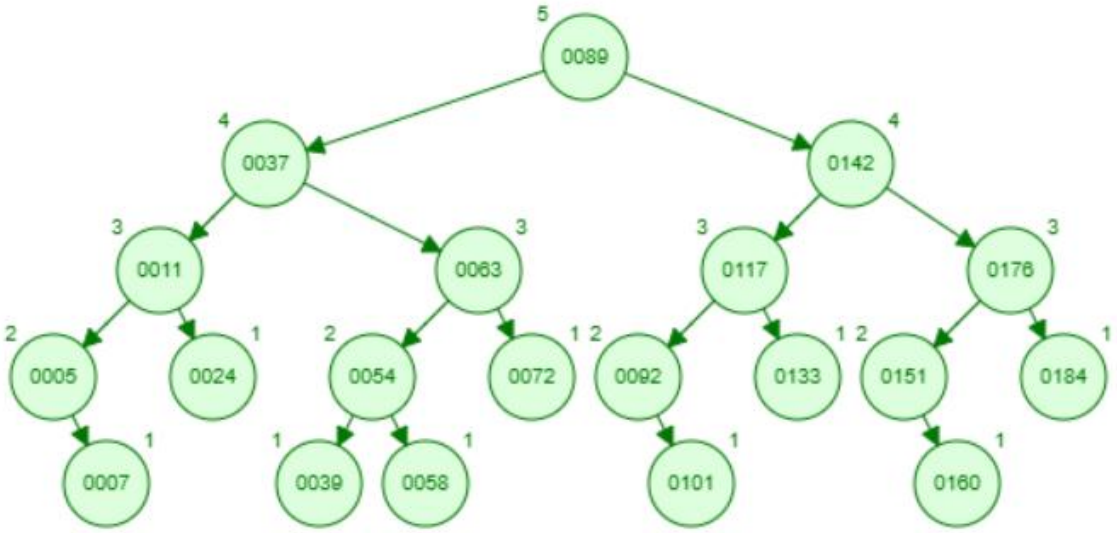


3.4 AVL Tree
Tool name / URL:
USFCA Visualization / https://www.cs.usfca.edu/~galles/visualization/AVLtree.html

Insertion & balancing description:
依照給定順序一個一個 Insert

Screenshot of AVL Tree (paste below):

Student ID: Student Name:



3.5 Red-Black Tree

Tool name / URL:

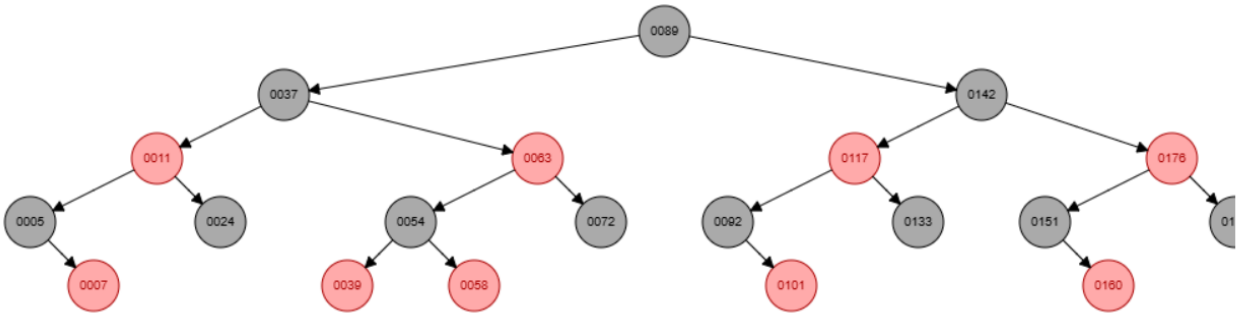
USFCA Visualization / https://www.cs.usfca.edu/~galles/visualization/RedBlack.html

Insertion & balancing description:

依照給定順序一個一個 Insert

Screenshot of Red-Black Tree (paste below):

最右邊被裁切掉的點為 黑色節點，數值 184



3.6 Max Heap

Tool name / URL:

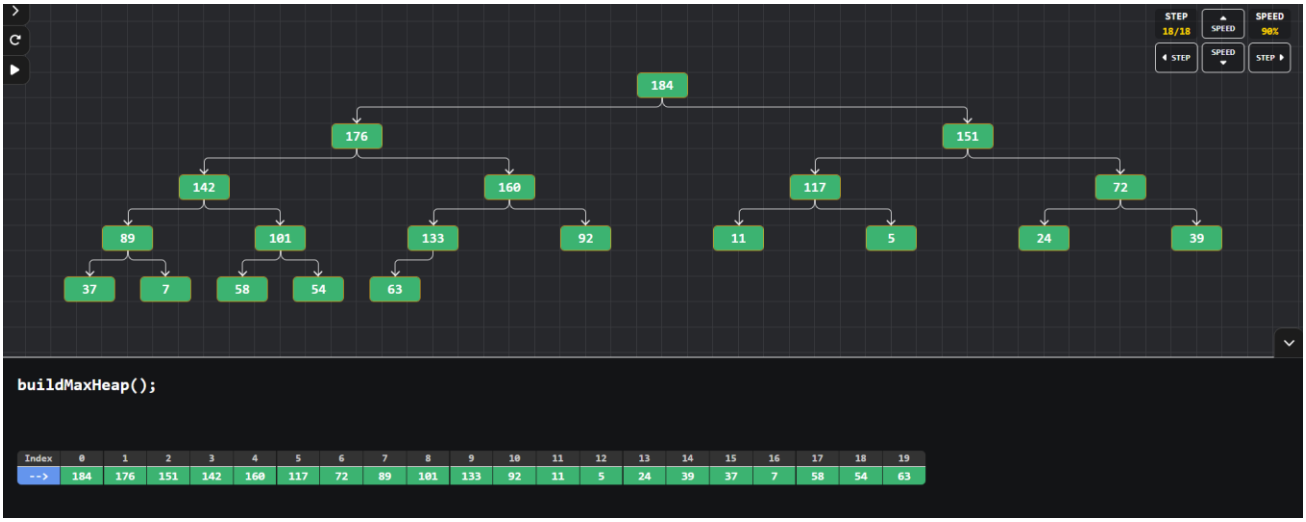
HeapSort Visualizer / https://heapsortvisualizer.web.app/

Construction / heap-building description (e.g. heapify, insert-and-sift-up):

Student ID: Student Name:

依照給定順序輸入後再點擊 Build Max Heap

Screenshot of Max Heap (paste below):



3.7 Min Heap

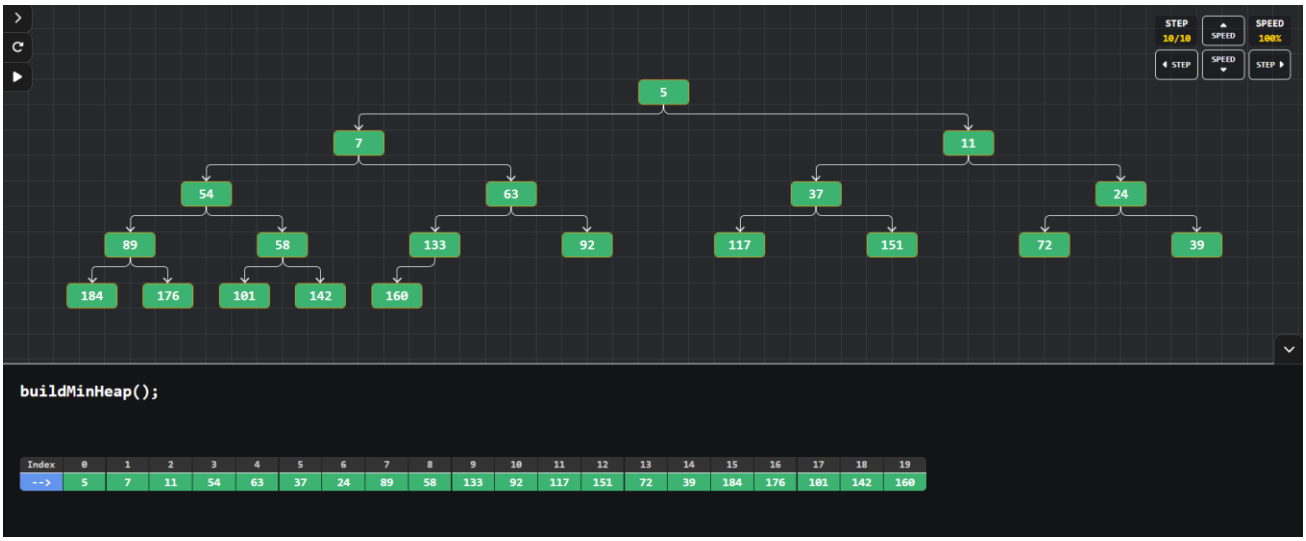
Tool name / URL:

HeapSort Visualizer / https://heapsortvisualizer.web.app/

Construction / heap-building description:

依照給定順序輸入後再點擊 Build Min Heap

Screenshot of Min Heap (paste below):



Section 4. Application Examples

Task: For each tree type, choose one application and explain why this tree is suitable.

Student ID: Student Name:

Tree Type	Application Example (name / context)	Why this tree fits (properties that matter)
Binary Tree	語法樹（Syntax Tree / Parse Tree）	程式語言的運算式（如 a + b * c）具有層次化的結構。二元樹能精確表達運算子與運算元之間的優先權關係，父節點代表指令，子節點代表參數，方便編譯器進行遞迴解析
Complete Binary Tree	排程系統、優先佇列底層	結構非常對稱且緊湊，幾乎不會產生中間空節點，因此能有效映射到連續的陣列（Array）中，使父子節點的存取效率極高
Binary Search Tree	簡易檔案系統索引	利用「左小右大」的順序，在樹結構維持相對平衡時，每一層都能排除約一半的搜尋範圍
AVL Tree	資料庫索引	資料庫系統通常「讀多寫少」。AVL 樹嚴格限制左右子樹高度差，使搜尋時間能在最壞情況下仍保證為 O(log n)，雖然插入時的平衡代價較高，但能達到極致的查詢速度
Red-Black Tree	Linux Kernel 排程與記憶體管理	Linux 核心需要頻繁地插入與刪除任務。Red-Black Tree 相對於 AVL 樹平衡時旋轉次數較少，在維持搜尋效率與處理頻繁變動之間取得了最佳的平衡點，性能表現穩定
Max Heap	優先權調度 (Priority Scheduling)	因 Max Heap 的特性是父節點一定比子節點大，這讓電腦能在 O(1) 時間取得最高優先權的任務，並在 O(log n) 時間內完成調整
Min Heap	網路封包傳輸中的最短路徑演算法 (如 Dijkstra 演算法)	尋找最短路徑時，演算法需要不斷從候選清單中取出「路徑權重最小」的節點。Min Heap 能確保最小值始終在最上方

Section 5. Reflection on Tree Family and Performance (Optional but recommended)

Among BST, AVL, and Red-Black trees, which one would you pick for:

Mostly search (few updates)? Why?

Student ID: Student Name:

AVL Tree，AVL 樹的缺點在於插入或刪除時可能引發多次旋轉達到平衡，但在 few updates 的前提下這個代價可以忽略，而搜尋的時間複雜度取決於樹的高度，AVL 樹的平均高度與最大高度相較於其他樹是最接近 $\log_2 n$，故能提供最穩定的最快查詢時間。

Frequent insertions and deletions? Why?
Red-Black Tree，它只要求「大致平衡」(最長路徑不超過最短路徑的兩倍)，故在插入節點時，最多只需要兩次旋轉，刪除時最多三次，最後再調整顏色達到平衡。在需要頻繁變動資料的情況下，Red-Black Tree 在 rebalance 與 search 之間取得最佳的 Trade-off

If you must store these 20 integers for static search only (no updates), which structure or representation would you prefer (sorted array + binary search, BST, AVL, etc.)? Why?

Sorted Array + Binary Search，在靜態資料上，排序陣列的二元搜尋與完全平衡的二元搜尋樹在時間複雜度上同樣是 O(log n)，相較於樹的結構需要另外儲存 pointer 來指到左右節點，Array 比較節省空間，且 Array 在記憶體裡為連續存放，讀取效率佳

Section 6. AI Usage Log (Required)

Task: Record every time you ask an AI assistant about this assignment.

Index	Date / Time	AI Service (ChatGPT, Gemini, etc.)	Your Full Prompt / Question
1	12/28	Gemini	有其他網站可以讓我視覺化建構樹嗎? 目前 USFCA Visualization 上沒有可以建構 Binary Tree, Complete Binary Tree, Max Heap, Min Heap
2	12/28	Gemini	告訴我這幾個樹種類的實際應用案例 Binary Tree Complete Binary Tree Binary Search Tree AVL Tree Red-Black Tree

Student ID: Student Name:

			Max Heap Min Heap
3	12/28	ChatGPT	告訴我這幾個樹種類的實際應用案例 Binary Tree Complete Binary Tree Binary Search Tree AVL Tree Red-Black Tree Max Heap Min Heap
4	12/28	Gemini	我想寫的應用例子為 Binary tree: Syntax tree Complete Binary Tree: 排程系統 BST: 檔案系統索引 AVL: 資料庫索引 (高度優化的搜尋系統) Red-Black Tree: Linux Kernel 的排程與記憶體管 幫我簡單講解各自適合的原因
5	12/28	ChatGPT	幫我看各項回答有沒有哪裡錯誤或有疏漏

You may extend this table as needed.