CST 370 Design and Analysis of Algorithms
Spring A 2020
Midterm-II

Name: Matthew Robertson

Four-digits Class ID: 1618

- Test time is **2 hours and 30 minutes**.
- There are **12 problems**
- This is a **closed book exam**. You **can't use a calculator** during the exam. However, as a reference during the exam, you can prepare "two pages (= total of 4 sides)" **cheat sheet**. The cheat sheet can be typed or hand-written.
- If possible, enter your answers directly into the Word file to increase the readability of your answers. However, if it is too difficult or time consuming to type in a Word file, you can write down your answer on a paper. Then, take a picture and insert the picture into the Word file.
- During the exam, you must sign into the **Zoom** session and **turn on the video**. We will record the video. However, **turn off the audio** on your computer.
- If you have **a question** during the exam, please **use "Chat"** in Zoom. I will answer.
- When you finish the exam, submit your file in PDF format (optional Word file) on the iLearn.
- Use your time wisely—make sure to answer the questions you know first.
- Read the questions carefully.

1. (5 points) Suppose we are sorting an array of eight integers using quicksort, and we have just finished the first partitioning with the array looking like this:
2, 5, 1, 7, 9, 12, 11, 10

Which statement is correct? Why?

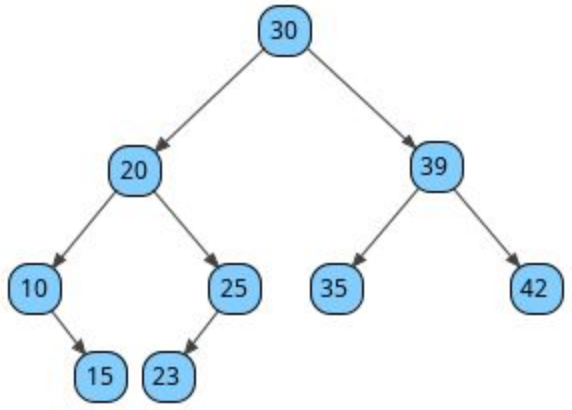
(A) The pivot could be either the 7 or the 9.
(B) The pivot could be the 7, but it is not the 9
(C) The pivot is not the 7, but it could be the 9
(D) Neither the 7 nor the 9 is the pivot.

(A) The pivot could be either the 7 or the 9.
Since:
every value to the left of the 7 is less than 7
every value to the right of the 7 is greater than 7
every value to the left of the 9 is less than 9
every value to the right of the 9 is greater than 9

2. (5 points) Draw a binary search tree with 9 nodes labeled 10, 15, 20, 23, 25, 30, 35, 39, 42 in such a way that the **preorder traversals** of the tree yield the following lists:

30, 20, 10, 15, 25, 23, 39, 35, 42

If you can't completely draw a binary search tree with the given information, explain why.



3. (10 points) Consider the following master theorem as we covered in the class

$T(n) = aT(n/b) + f(n)$ where $f(n) \in \Theta(n^d)$, $d \geq 0$

Master Theorem: If $a < b^d$, $T(n) \in \Theta(n^d)$
If $a = b^d$, $T(n) \in \Theta(n^d \log n)$

If $a > b^d$, $T(n) \in \Theta(n^{\log_b a})$

Based on the above theorem, determine the time efficiency of the following formula T(n).

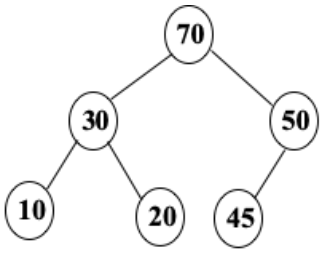
(a) $T(n) = 8 * T(n/2) + 2n^2 + 2n + 1$

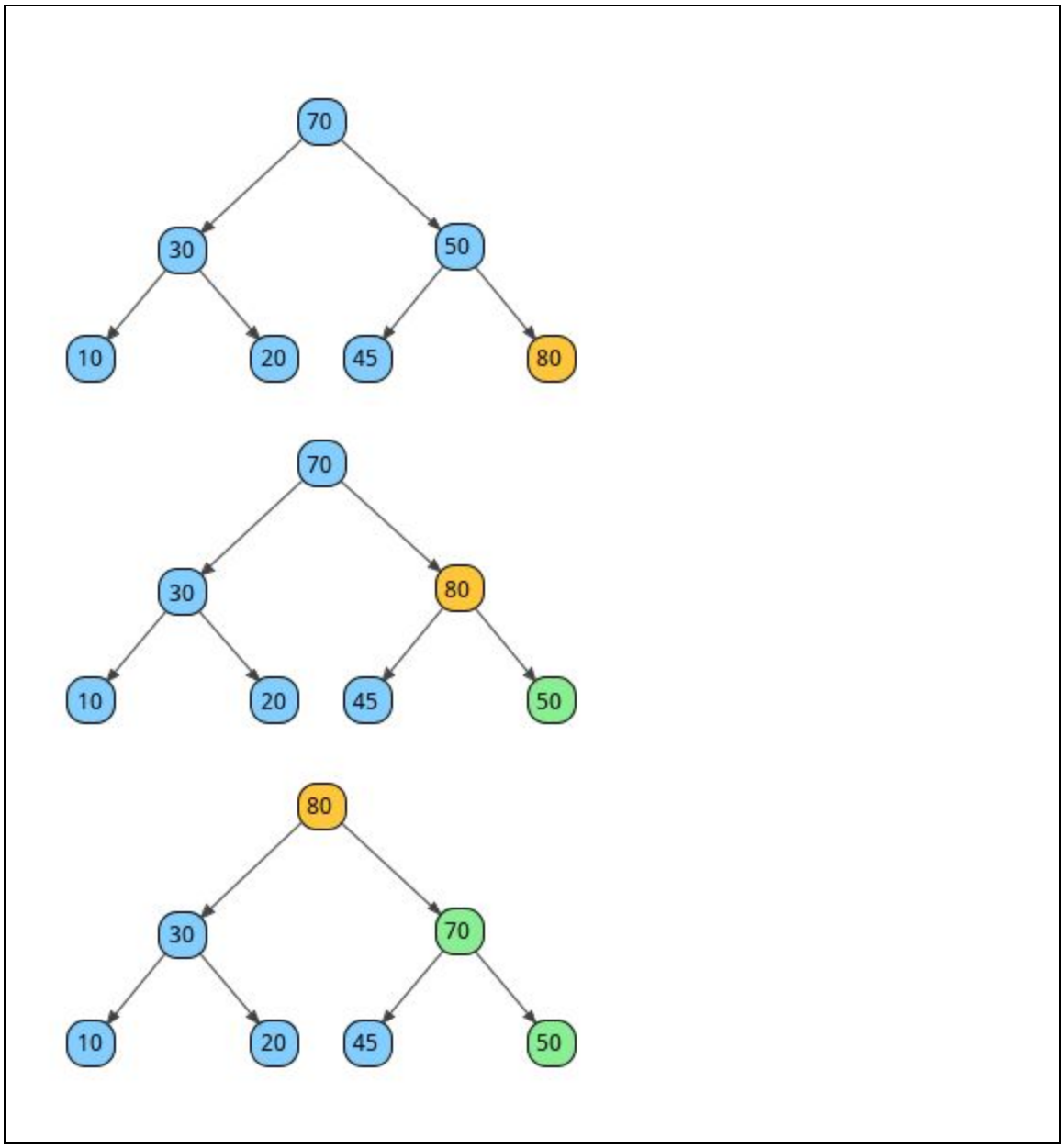
a = 8, b = 2, d = 2
a ? b^d = 8 ? 2^2 = 8 > 4
case three applies
T(n) $\in \Theta(n \log_2 8)$

(b) $T(n) = 3 * T(n/2) + 6n$

a= 3, b = 2, d = 1
a ? b^d = 3 ? 2^1 = 3 > 2
case three applies
T(n) $\in \Theta(n \log_3 2)$

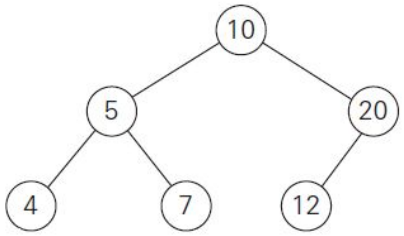
4. (5 points) Add 80 to the following heap. After adding the node, show intermediate diagrams (if any) that make it a heap again





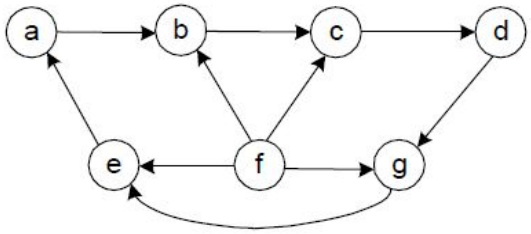
5. (15 points)

(a) Is this an AVL tree? (Yes/ No)



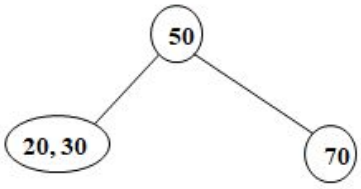
Yes

(b) Is the following graph a DAG (= directed acyclic graph)? (Yes/No)



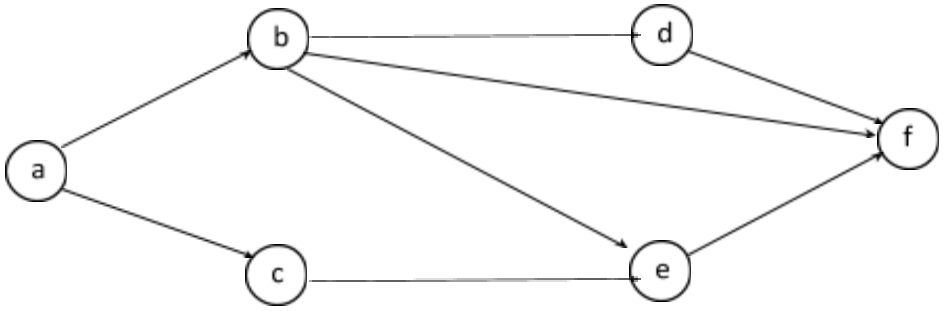
No, it has a cycle, a→ b → c → d → g → e → a

(c) Is this a 2-3 tree? (Yes/ No)



Yes

6. (10 points) Consider the following **directed graph**.



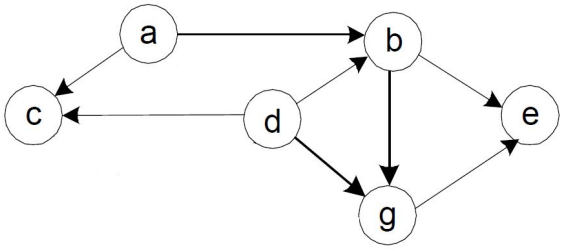
Starting at vertex ***a***, traverse the graph using the **breadth-first search** algorithm. You should present the **mark array** and **BFS tree** with only tree edges as we covered in the class. For the algorithm, you have to use our convention of the class (= ascending order for the alphabetical characters).

[a,b,c,d,e,f]

7. (5 points) Let T be a complete binary tree with n nodes. Finding a path from the root of T to a given vertex v in T using breadth-first search takes O(lg n) time. Is this true or not? Explain your answer.

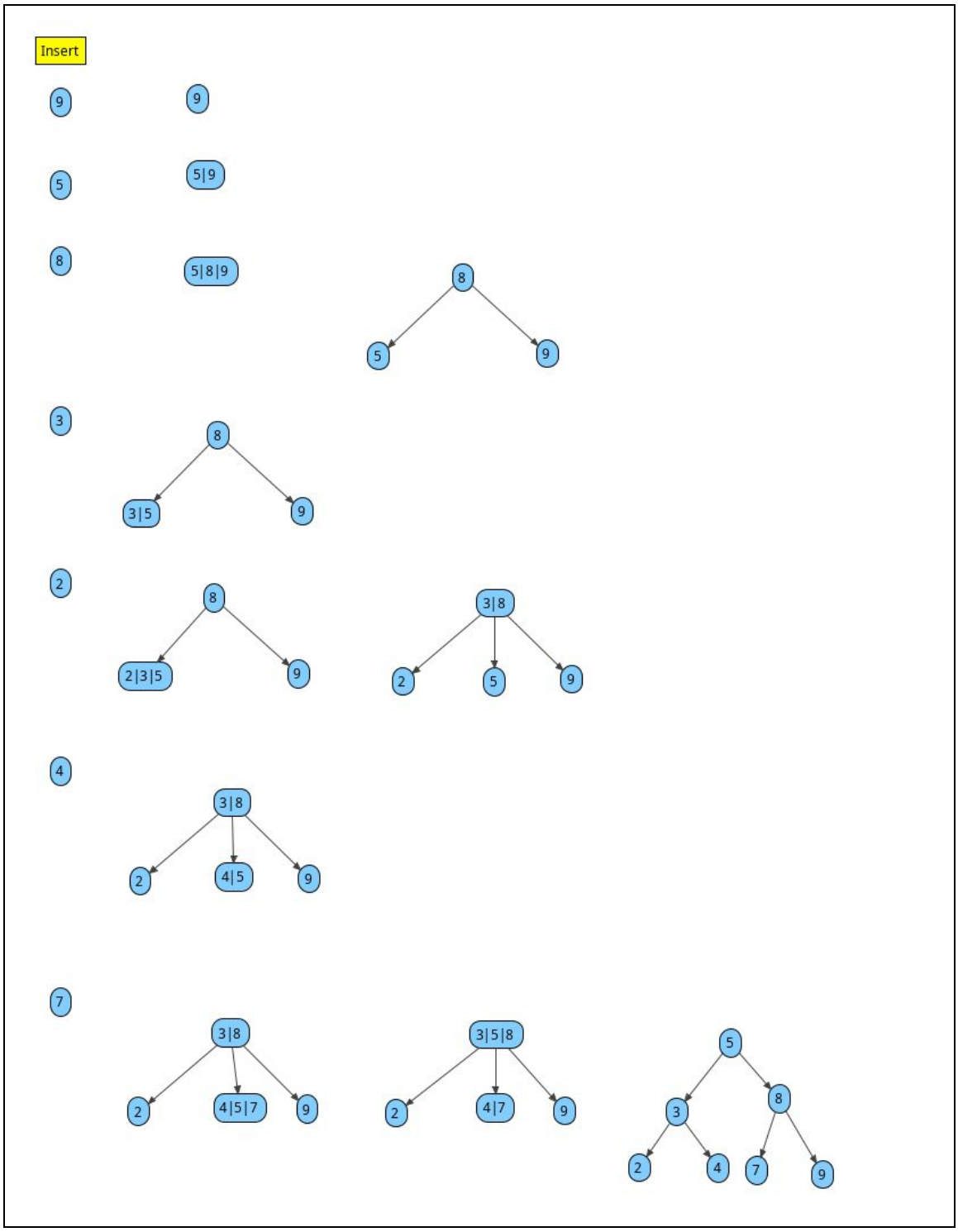
Not true. Breadth-first search traverses the tree from top to bottom, and left to right—it effectively visits every node in the tree, yielding *O(n)* time.

8. (10 points) For the following graph, present the source vertices (or starting vertices) when you use the **source-removal algorithm** to get the topological order. And then, present the result (= **topological order**) of the algorithm. For the problem, you have to follow our convention of alphabetical order removing.



Nodes *a* and *d* have no in-edges, those would be sources vertices.
After applying the source removal algorithm, the topological order is:
a → b → d → c → g → e

9. (5 points) Construct a 2-3 tree with the list of **9**, **5**, **8, 3, 2, 4**, and **7**.

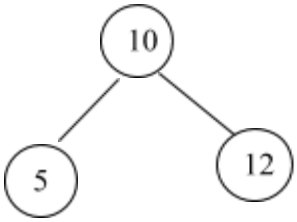


10. (10 points) Assume that you have an array A[0.. n–1] which contains n integers from 0 to n in the increasing order. Because the size of the array is n, one integer in the range from 0 to n is missing. For the problem, you have to design an efficient algorithm to find the missing integer. For example, let's assume that the array A has eight elements such as 0, 1, 2, 3, 4, 5, 7, and 8. For the array A, your algorithm should return 6 as the missing integer number.

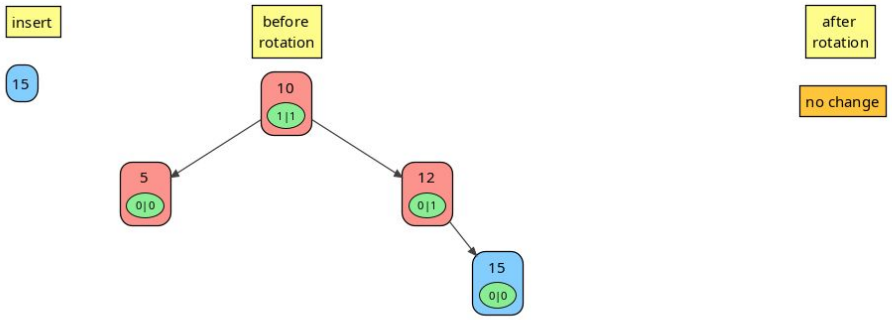
Assume that your algorithm receives the array A and the array size n as input arguments. Describe your algorithm in English. Determine the complexity of your algorithm in Big O notation.

Go to the index at halfway point, does value equal index? Yes, look at value at halfway point between the current index and the end. No, look at the halfway point between the current index and start. Continue until an index does not equal and then does equal, at that time, the missing value is the previously visited index. If all values are visited, with all being equal, the missing one is the last. If all values are visited with all being unequal the value is the first.
This is a binary search, so its time complexity is *O(log n)*.

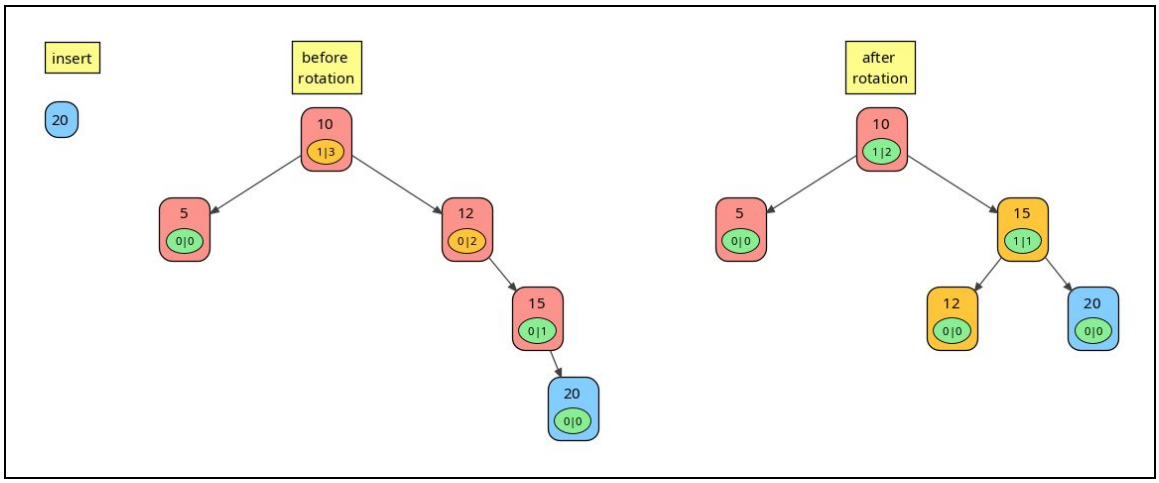
11. (15 points) Consider an AVL tree as below.



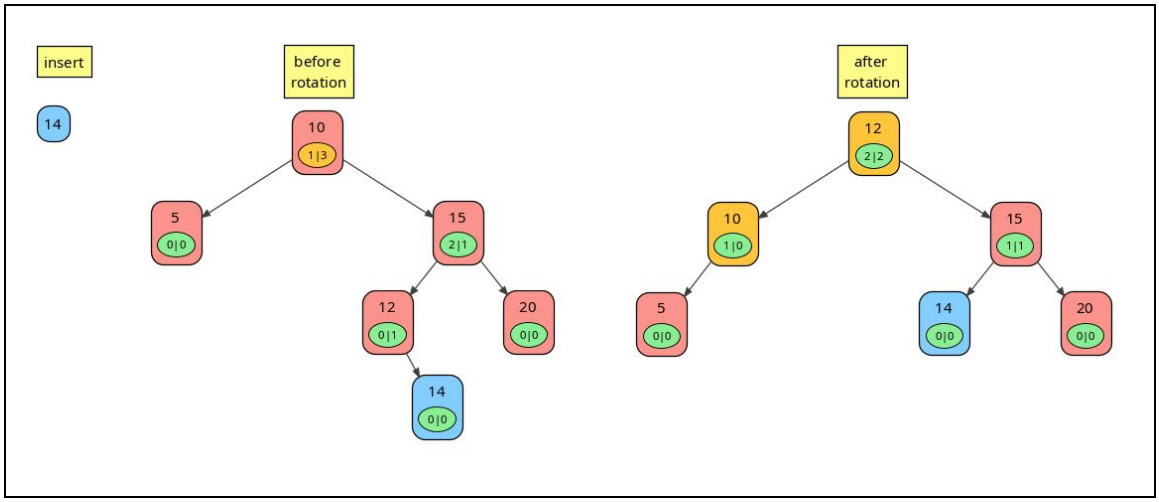
(a) Add a vertex with the value **15** to the above AVL tree. If there is a rotation, you should present it clearly.



(b) To your solution of the question (a), add a vertex with the value **20**. If there is a rotation, you should present it clearly.



(c) To your solution of the above question (b), add a vertex with the value **14**. If there is a rotation, you should present it clearly.



12. (5 points) Note that this is a **puzzle** problem. Assume that you have **100** identical-looking coins and a two-pan balance scale with no weights. One of the coins is fake, but it is not known whether it is lighter or heavier than the genuine 99 coins. Describe your idea to determine in the minimum number of weighings whether the fake coin is lighter or heavier than the others. Present the minimum number of weighings and your answer clearly.

Not that you don't need to find out which coin is fake. The question is to identify whether the fake coin is heavier or lighter than the real coins.

Since whether it is heavier is in question the coin needs to be in the weighing one time and not in the weighing one time. Divide the coins into four piles of 25 coins. Call the piles A, B, C, D for reference.

Weigh piles A & B.
If they are equal, weigh piles B & C. If B & C are different weight (while taking note of which

pile is heavier), coin is in pile C and it is lighter or heavier depending on whether its corresponding pile was lighter or heavier.

If those are equal weight piles C & D. Note which is heavier, if D is heavier the coin is as well, otherwise the coin is lighter.

If A & B are different weight (while taking note of which pile is heavier), weigh piles B & C , if those are equal the coin is in pile A and it is lighter or heavier depending on whether its corresponding pile was lighter or heavier.