Assignment 05 – Concept (18 marks, 1 mark for each question)

Answer below multiple choice questions:

- **Submit a pdf file that record your answer for each question. Must be pdf or the file will not be marked!**
- **Make sure you submit the correct file, otherwise the file will not be marked!**
- **Name your file with your ID, otherwise the file will not be marked!**

The file content should look something like this:

1. A
2. C
3. D
4. A
5 D

1. Given a 11-slot double-hashing hash table. Let *hash(x) = x%TableSize* and $hash_2(x) = 5-(x\%5)$. Let 3 and 7 already be in the table. What will the table look like after we add 11, 14, 20, 18 into the table respectively?

A.

11		18	3		14		7		20	

B.

11			3	14			7	18		20

C.

11			3		14		7	18	20	

D.

11		18	3	14			7		20	

2. Given a 11-slot quadratic probing hash table. Let *hash(x) = x%TableSize.* Let 3 and 7 already be in the table. What will the table look like after we add 11, 14, 19, 18 into the table respectively?

A.

11			3	14			7	19	18	

B.

11			3	14	18		7	19		

C.

11			3	14			7	18	19	

D.

11			3	14	18	19	7			

3. Given a 11-slot double-hashing hash table. Let *hash(x) = x%TableSize* and $hash_2(x) = 5-(x\%5)$. Let 4 and 8 already be in the table. What will the table look like after we add 15, 26,19 respectively?

A.

19				4				8	15	26

B.

	26	19		4	15			8	26	19

C.

	26			4				8	15	19

D.

	19			4	15	26		8		

4. Given a 11-slot quadratic probing hash table. Let *hash(x) = x%TableSize*. Let 3 and 7 already be in the table. What will the table look like after we add 18, 19, 14, 25 respectively?

A.

18	14	25	3				7	19		

B.

			3	14	25		7	18	19	

C.

	25		3	14			7	18	19	

D.

	19	14	3			25	7	18		

5. Given a 13-slot double-hashing hash table. Let *hash(x) = x%TableSize* and $hash_2(x) = 7-(x\%7)$. What will the table look like after we add 1, 5, 18, 8 respectively?

A.

	1	8			5	18						

B.

	1				5	18		8				

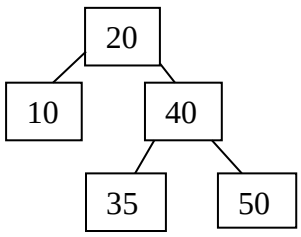
C.

	1				5		8	18				

D.

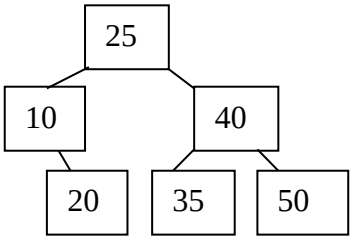
	1				5	8		18				

6. An AVL tree that stores integers looks like:

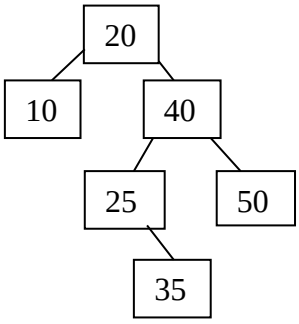


What will the tree look like after we add 25?

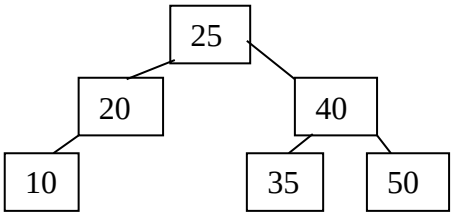
A.



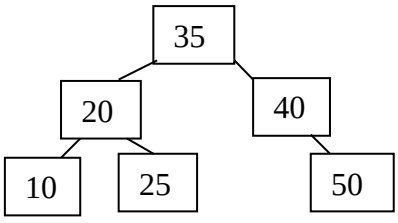
B.



C.

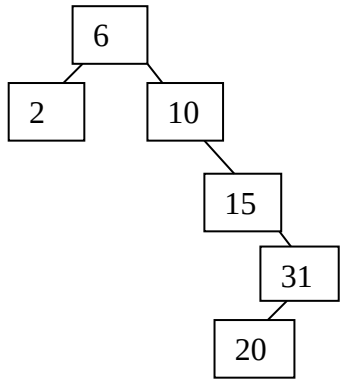


D.

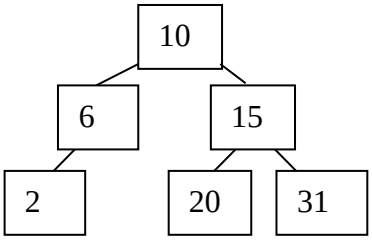


7. If we have an empty AVL tree for storing integers, what will the tree look like after we add 6, 10, 15, 2, 31, 20 respectively?

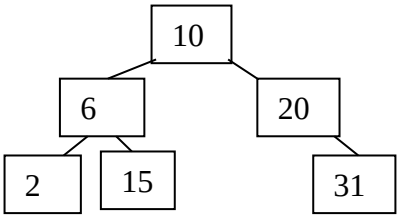
A.



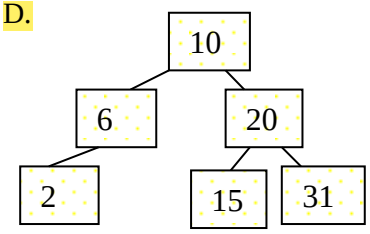
B.



C.

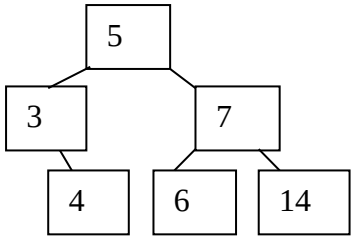


D.

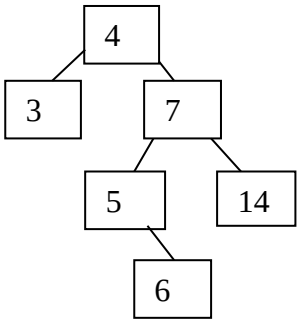


8. If we have an empty AVL tree for storing integers, what will the tree look like after we add 14, 7, 3, 5, 4, 6 respectively?

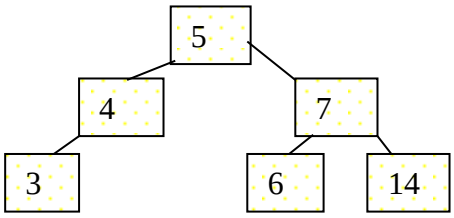
A.



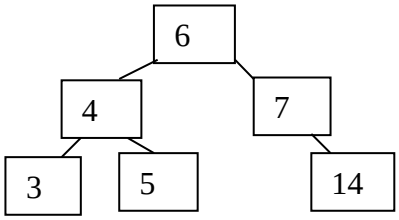
B.



C.

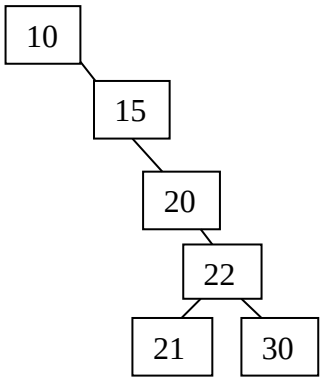


D.

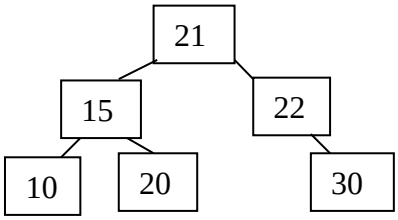


9. If we have an empty AVL tree for storing integers, what will the tree look like after we add 10, 15, 20, 22, 21, 30 respectively?

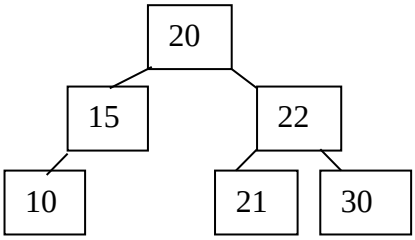
A.



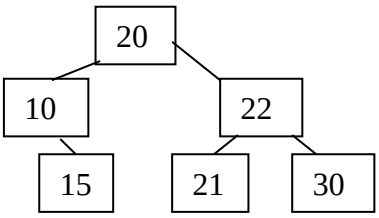
B.



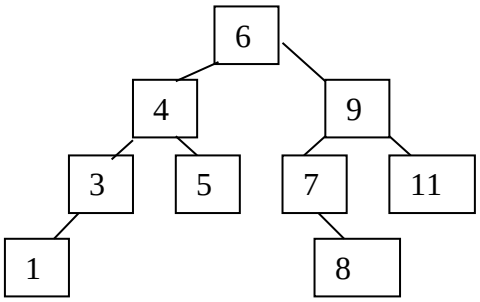
C.



D.

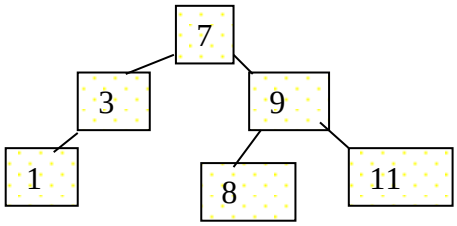


10. We have the following AVL tree:

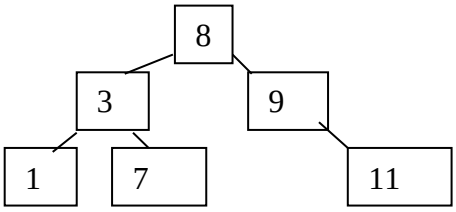


To remove a value, we replace the removed value with the maximum value of its left subtree. What will the tree look like after we remove 6, 5 and 4 respectively?

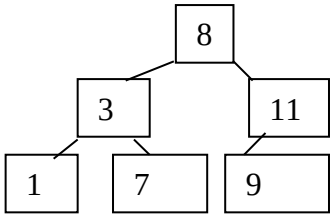
A.



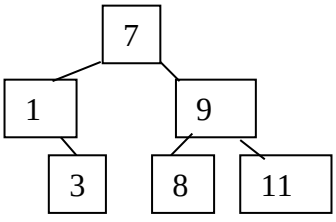
B.



C.



D.



11. What is the Huffman code of each alphabet, given that the frequency of a is 300, the frequency of b is 250, the frequency of c is 500, and the frequency of d is 100? We use min heap when creating a Huffman tree.

A. a = 10, b = 111, c = 0, d = 110
B. a = 0, b = 111, c = 10, d = 110
C. a = 10, b = 110, c = 11, d = 111
D. a = 10, b = 11, c = 0, d = 111

12. What is the Huffman code of each alphabet, given that the frequency of a is 400, the frequency of b is 550, the frequency of c is 200, and the frequency of d is 340? We use min heap when creating a Huffman tree.

A. a = 0, b = 1, c = 10, d = 11
B. a = 10, b = 0, c = 110, d = 111
C. a = 0, b = 100, c = 11, d = 101
D. a = 000, b = 001, c = 01, d = 1

13. A singly linked list is created from connected LinkedNode. Each node contains:
    - element: the data (object) stored in the node
    - next: a reference pointing to another LinkedNode.

    Assume a node has a 2-parameter constructor that sets the above variables.

    The linked list itself has 2 variables:
    - header: the first LinkedNode. This node does not store any data.
    - size: the number of objects currently stored in the list.

    Method add(int i, Object e) inserts e into the list at the i^{th} node (from left to right). A node next to header is node number 0. What is the correct code for this method?

A.
```
if (i>=0){
        LinkedNode p = header;
        for (int j = -1; j < i; j++) p = p.next;
        p.next = new LinkedNode(e, p.next);
        ++size;
}
```

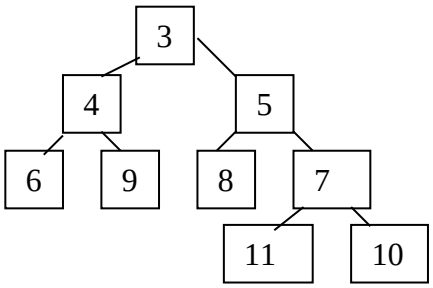
B.
```
if (i>=0){
        LinkedNode p = header;
        for (int j = -1; j < i-1; j++) p = p.next;
        p.next = new LinkedNode(e, p.next);
        ++size;
}
```

C.
```
if (i>=0){
        LinkedNode p = header;
        for (int j = -1; j < i; j++) p = p.next;
        if(p!=null){
                p.next = new LinkedNode(e, p.next);
        }
        ++size;
}
```

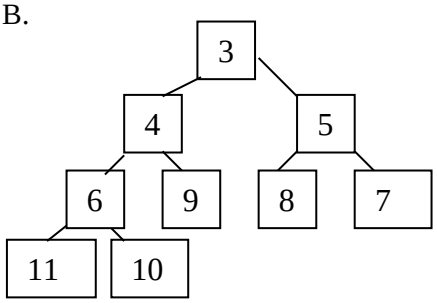
D. None of the above.

14. If we create a min heap by taking each value from an array (from left to right), the heap will organize itself during these additions (doing percolate up). What will the heap look like if we use data from array {7, 3, 8, 4, 9, 5, 11, 10, 6}?

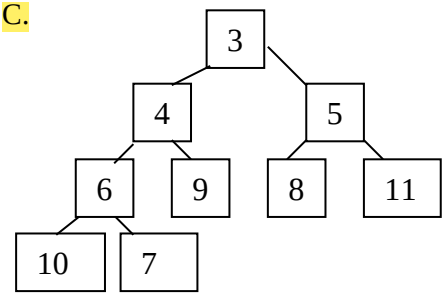
A.



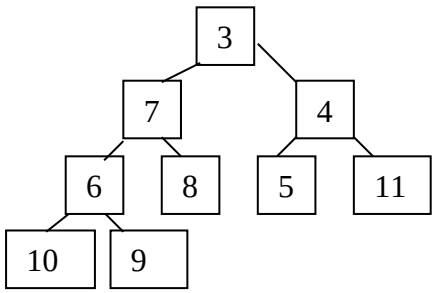
B.



C.

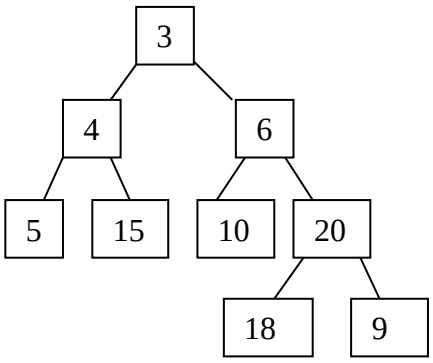


D.

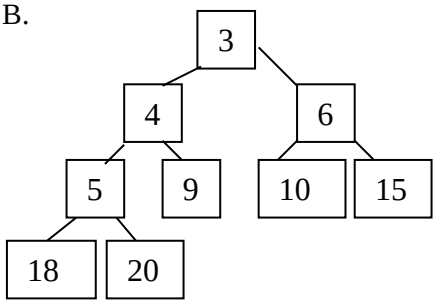


15. If we create a min heap by taking each value from an array (from left to right), the heap will organize itself during these additions (doing percolate up). What will the heap look like if we use data from array {10, 18, 3, 5, 15, 6, 20, 4, 9}?

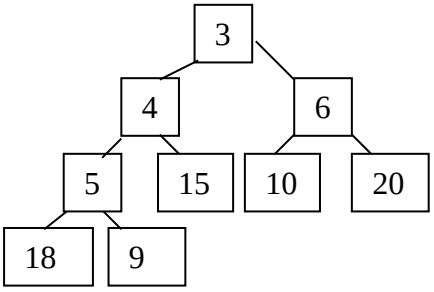
A.



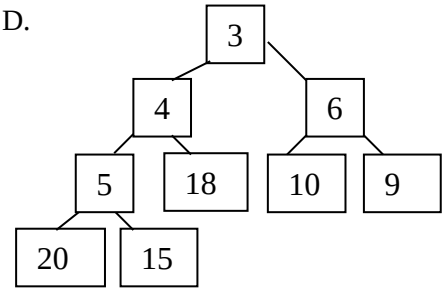
B.



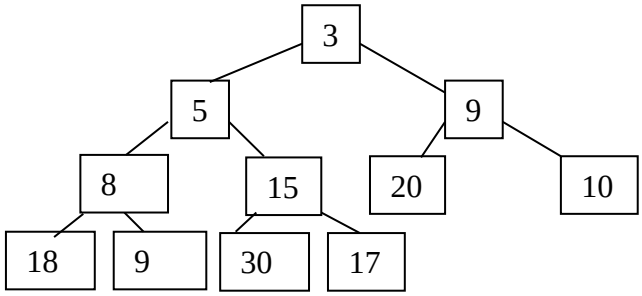
C.



D.

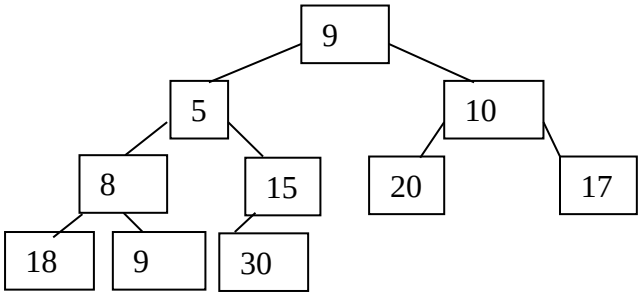


16. A min heap is shown below:

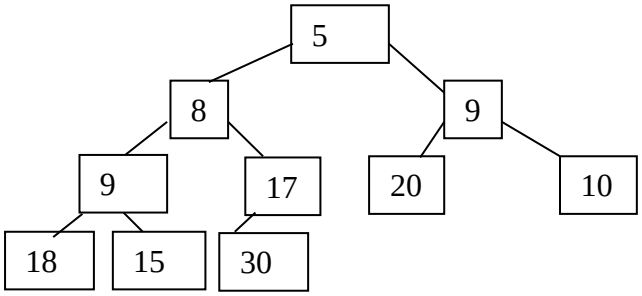


What will the heap look like after we remove 3?

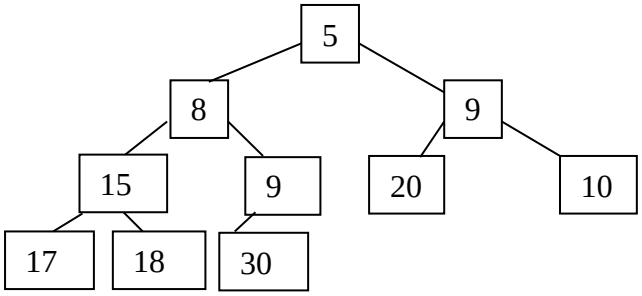
A.



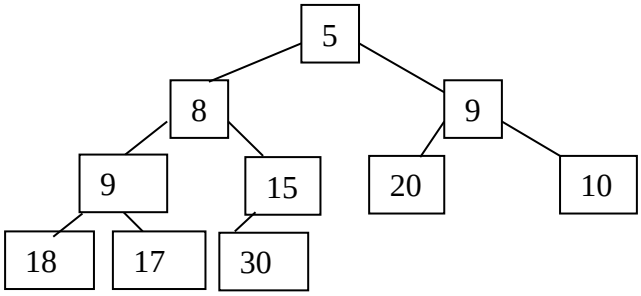
B.



C.



D.



17. A min heap for storing integer has the following code (we can call the given methods but we do not know how the methods were written):

```
public class Heap{
        int[] thearray; //this is used just like in our class.
        int size; //number of elements in the heap, always correctly updated

        public boolean isEmpty(){
                // returns true if the heap is empty, false otherwise.
        }

        public int removeMin(){
                // removes an integer with the lowest value from the heap.
                // It returns the removed integer.
        }

        public void add(int x){
                //adds new integer to heap. The integer is put in correctly according to
                //its value.
        }
}
```

We want to code a method called *public int remove(int v)* of class Heap. This method removes value, v, from the heap. The method returns the removed value (or -1 if the value is not in the heap). After the call, heap must remain heap.

If we are not allowed to create any pointer variables (any objects), can we really write the code for this remove method?

A. We cannot write the method because we need some object variables.]
B. We can, but we must call exactly one of the given methods.
C. We can, but we must call exactly two of the given methods.
D. We can, without having to call any of the given methods.

18. A min heap created using an array has the following functions similar to the ones in class:

percolateUp(int i): Move data at position i of the array up towards the root of the heap.
percolateDown(int i): Move data at position i of the array down towards the leaf of the heap.

What does this code do?

```
void  f(int i){
        if (i <= -1) return;
        array[i] = array[size-1];
        size--;
        percolateUp(i);
        percolateDown(i);
}
```

A. Remove maximum value from the heap.
B. Move data from position (size-1) to above and below position i in the heap.
C. Remove data at position i^{th} from the heap.
D. None of the above answers is correct.