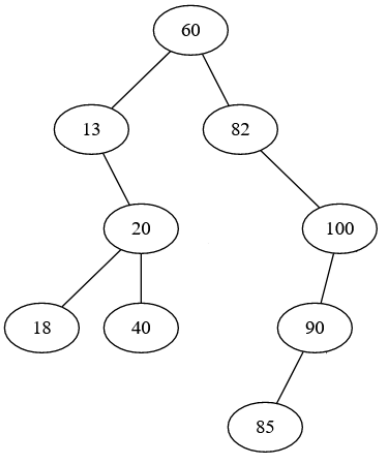
Started on	Thursday, 29 May 2025, 5:25 PM
State	Finished
Completed on	Thursday, 29 May 2025, 5:57 PM
Time taken	32 mins 29 secs
Marks	9.00/11.00
Grade	**8.18** out of 10.00 (**81.82**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Consider the tree below. Which nodes are not balanced?



- [ ] 40
- [ ] 90
- [x] 82
- [x] 13
- [ ] 60
- [ ] 20
- [ ] 18
- [x] 100
- [ ] 85

Your answer is correct.

For every node we have to compute the difference between the height of the left and right subtree. Where this difference is not 1, 0 or -1 the node is not balanced. These nodes are: 13 (-2), 82 (-3) and 100 (2)

The correct answers are:
13,

82,

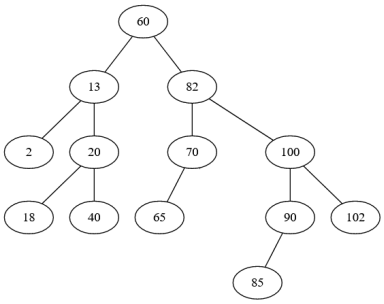
100

Question **2**

Correct

Mark 1.00 out of 1.00

Consider the AVL tree below in which we will insert a new value. For the insertion of what value is a rotation going to be necessary? Note: for every answer assume that we insert just that value, not the other answers as well.



- [ ] 80
- [x] 89
- [x] 19
- [x] 61
- [ ] 105
- [ ] 1

Your answer is correct.

You need to see where the element is going to be inserted and whether after the insertion you will have imbalanced nodes. You need a rotation if you insert 19 (13 will be imbalanced), 61 (70 will be imbalanced), and 9 (90 will be imbalanced).

The correct answers are:
19,

61,

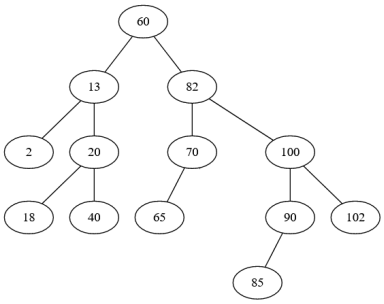
89

Question **3**

Correct

Mark 1.00 out of 1.00

Consider the AVL tree below in which we will insert a new value which will require a rotation. For the insertion of what value is a **double** rotation going to be necessary? Note: for every answer assume that we insert just that value, not the other answers as well.



- [ ] 83
- [ ] 61
- [x] 67 ✔ If you insert 67 (as the right child of 65) you will need a double right rotation on 70
- [x] 89 ✔ If you insert 89 (as the right child of 85), you will need a double right rotation on 90.
- [ ] 42
- [x] 19 ✔ If you insert 19 (as the right child of 18) you will need to do a double left rotation on node 13

Your answer is correct.

The correct answers are:
19,

67,

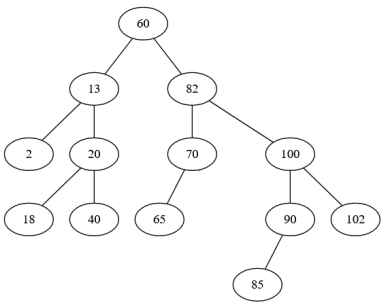
89

Question **4**

Correct

Mark 1.00 out of 1.00

Consider the AVL tree below in which we insert element 19. What is going to be the level order traversal of the resulting tree? Enumerate the values of the nodes in the traversal separated by a single space.



Answer: 60 18 82 13 20 70 100 2 19 40 65 90 102 85 ✔

You add 19 as the right child of 18, but then you need to do a double left rotation on node 13.

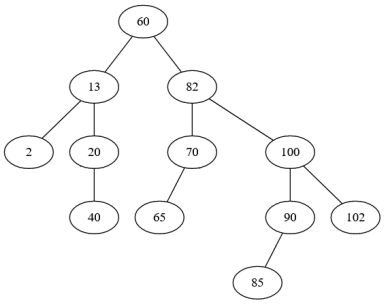
The correct answer is: 60 18 82 13 20 70 100 2 19 40 65 90 102 85

Question **5**

Correct

Mark 1.00 out of 1.00

Consider the below AVL tree. What is going to be the level order traversal of the tree after we remove element 2? Enumerate the values of the nodes in the traversal separated by a single space.



Answer: 82 60 100 20 70 90 102 13 40 65 85 ✔

When we remove 2, node 13 will become imbalanced, so we need a single left rotation on 13. But after the rotation, node 60 will become imbalanced, so we need another rotation, this time a single left rotation on node 60.

The correct answer is: 82 60 100 20 70 90 102 13 40 65 85

Question **6**

Correct

Mark 1.00 out of 1.00

Assume that we have a cuckoo hash table, with m = 7 positions and the following hash functions:

h1(elem) = elem % 7

h2(elem) = (elem / 7) % 7.

After adding some elements, the table looks like that:

14				11		62

If next we want to insert element 25 to the table, which element(s) will be moved to the second table?

- [ ] None of them
- [ ] 62
- [x] 11 ✔
- [ ] 14
- [ ] 25

Your answer is correct.

h1(25) = 4, but position 4 is occupied in the first table. Nevertheless, we put 25 on position 4 in the first table, but we need to move 11 to the second one. According to the second hash function it should go to position 1 and since that is an empty position this is where it will be added.

The correct answer is:
11

Question **7**

Correct

Mark 1.00 out of 1.00

Assume that we have a cuckoo hash table, with m = 7 positions and the following hash functions:

h1(elem) = elem % 7

h2(elem) = (elem / 7) % 7.

After adding some elements, the table looks like that:

28			31	25	19	62

	11	14				

If next we want to insert element 47 to the table, which of the already existing element(s) will be moved to a different position?

- [ ] None of them
- [ ] 62
- [ ] 11
- [x] 14
- [ ] 25
- [x] 19
- [ ] 31
- [x] 28

Your answer is correct.

h1(47) = 5, but position 5 is occupied. So we put 47 on position 5 and move 19 to the second table.

h2(19) = 2, but that position is occupied. So we put 19 on position 2 in the second table and move 14 to the first table.

h1(14) = 0, but that position is occupied. So we put 14 on position 0 and move 28 in the second table.

h2(28) = 4, finally an empty position.

The correct answers are:
14,

19,

28

Question **8**

Correct

Mark 1.00 out of 1.00

Assume that we have a cuckoo hash table, with m = 7 positions and the following hash functions:

h1(elem) = elem % 7

h2(elem) = (elem / 7) % 7.

After adding some elements, the table looks like that:

14			31	25	47	62

	11	19		28		

If next we want to insert element 39 to the table, which of the already existing element(s) will be moved to a different position?

- [ ] None of them
- [ ] 62
- [ ] 11
- [ ] 14
- [x] 25
- [ ] 19
- [ ] 31
- [ ] 28
- [ ] 47

Your answer is correct.

h1(39) = 4, but position 4 is occupied. So we put 39 on position 4 and move 25 to the second table.

h2(25) = 3 and that is an empty position, so that is where we put the element 25

The correct answer is:
25

Question **9**

Correct

Mark 1.00 out of 1.00

Which collision resolution method could be described as "a hashtable of hashtables"?

- ( ) separate chaining
- ( ) coalesced chaining
- ( ) open addressing
- ( ) cuckoo hashing
- ( ) linked hashtable
- (•) perfect hashing

Your answer is correct.

The correct answer is:
perfect hashing

Question **10**

Incorrect

Mark 0.00 out of 1.00

Assume that we are building a perfect hashing hashtable for storing the following elements: 19, 47, 81, 32, 49, 39, 12, 25, 71. For the universal hash function we choose the one from the lecture, with p = 101.

For the hash function of the primary hash table, we choose a = 2 and b = 1.

For the secondary hash table from position 7, which values for a and b could NOT be used?

- [ ] a = 4, b = 5
- [x] a = 1, b = 2 ✖
- [ ] a = 3, b = 11

Your answer is incorrect.

There are a few steps, that you need to compute yourself, to be able to answer this question.

First of all, we add 9 elements, so the size of the primary hash table will be m = 9. This also means that the hash function for the primary hash table is going to be: h(elem) = ((2*elem + 1) % 101) % 9

There are two elements which hash to position 7 in the primary hash table: 39 and 12, so that hash table will have 4 positions (square of number of elements) and the corresponding hash function will be: h(elem) = ((a * elem + b) % 101) % 4

Now we only have to check for what values of a and b, there is no collision in the table:

a = 1, b = 2 => h(39) = 1, h(12) = 2 => OK

a = 3, b = 11 => h(39) = 3, h(12) = 3 => NOT OK

a = 4, b = 5 => h(39) =0, h(12) = 1 => OK

The correct answer is:
a = 3, b = 11

Question **11**

Incorrect

Mark 0.00 out of 1.00

Assume that we are building a perfect hashing hashtable for storing the following elements: 19, 47, 81, 32, 49, 39, 12, 25, 71. For the universal hash function we choose the one from the lecture, with p = 101.

For the hash function of the primary hash table, we choose a = 2 and b = 1.

How many positions are we going to have, in total in all the primary and secondary hash tables?

- [ ] 22
- [ ] 9
- [x] 18 ✖
- [ ] 81

Your answer is incorrect.

There are a few steps, that you need to compute yourself, to be able to answer this question.

First of all, we add 9 elements, so the size of the primary hash table will be m = 9. This also means that the hash function for the primary hash table is going to be: h(elem) = ((2*elem + 1) % 101) % 9

h(19) = 3

h(47) = 5

h(81) = 5

h(32) = 2

h(49) = 0

h(39) = 7

h(12) = 7

h(25) = 6

h(71) = 6

So we have 9 positions for the primary hash table.

On position 0 we have 1 element => secondary table of size 1

On position 1 we have 0 elements => no secondary table

On position 2 we have 1 element => secondary table of size 1

On position 3 we have 1 element => secondary table of size 1

On position 4 we have 0 elements => no secondary table

On position 5 we have 1 element => secondary table of size 1

On position 6 we have 2 elements => secondary table of size 4

On position 7 we have 2 elements => secondary table of size 4

On position we have 1 elements => secondary table of size 1

Total: 9 + 1+ 1+ 1+ 1+ 4 + 4 + 1 = 22 positions

The correct answer is:
22

◄ Lecture 12 Quiz - Mathematics and Computer Science in English, Artificial Intelligence, Information Engineering

Jump to...