DSA Coursework 1 23011124

data keys: 33, 30, 2, 52, 10, 14, 59, 80, 89, 96

Binary Heap

2. Heap Build

`

Heap Sort

AVL Tree

Hash Table (1)

Hash Table (2)

References

Yusuf, A.D. et al. (2021) ‘Collision resolution techniques in Hash table: A review’, International Journal of Advanced Computer Science and Applications, 12(9), pp. 758–758. doi:10.14569/ijacsa.2021.0120984.

|  |  |  |
| --- | --- | --- |
| Operation | Explanation | Heap (as list) |
| H1 | Create a hole at 1 (root of tree) | [ ] |
| I33 | Insert 33 at the first hole | [33] |
| H2 | Create a hole at 2 (left child of root) | [33, ] |
| X | Move hole up as next number is less than 33 |  |
| I30 | Insert 30 at the second hole (now at root) | [30, 33] |
| H3 | Create a hole at 3 (right child of root) | [30, 33, ] |
| X | Move hole up as next number is less than 30 |  |
| I2 | Insert 2 at the third hole (now at root) | [2, 30, 33] |
| H4 | Create a hole at 4 (left child of hole 2) | [2, 30, 33, ] |
| I52 | Insert 52 at the fourth hole | [2, 30, 33, 52] |
| H5 | Create a hole at 5 (right child of hole 2) | [2, 30, 33, 52, ] |
| X | Move hole up as next number less than 33 |  |
| I10 | Insert 10 at the fifth hole (now left child of root) | [2, 10, 30, 33, 52] |
| H6 | Create a hole at 6 (left child of hole 3) | [2, 10, 30, 33, 52, ] |
| X | Move hole up as next number less than 30 |  |
| I14 | Insert 14 at the sixth hole (now right child of root) | [2, 10, 14, 30, 33, 52] |
| H7 | Create a hole at 7 (right child of hole 3) | [2, 10, 14, 30, 33, 52, ] |
| I59 | Insert 59 at the seventh hole | [2, 10, 14, 30, 33, 52, 59] |
| H8 | Create a hole at 8 (left child of hole 4) | [2, 10, 14, 30, 33, 52, 59, ] |
| I80 | Insert 80 at the eight hole | [2, 10, 14, 30, 33, 52, 59, 80] |
| H9 | Create a hole at 9 (right child of hole 4) | [2, 10, 14, 30, 33, 52, 59, 80, ] |
| I89 | Insert 89 at the ninth hole | [2, 10, 14, 30, 33, 52, 59, 80, 89] |
| H10 | Create a hole at 10 (left child of hole 5) | [2, 10, 14, 30, 33, 52, 59, 80, 89, ] |
| I96 | Insert 96 at the tenth hole | [2, 10, 14, 30, 33, 52, 59, 80, 89, 96] |

|  |  |  |
| --- | --- | --- |
| Operation | Explanation | Heap (as list) |
|  | No operation yet – all values inserted | [33, 30, 2, 52, 10, 14, 59, 80, 89, 96] |
| X10 | Swap 10 with 30 as 30 above and 10 smaller | [33, 10, 2, 52, 30, 14, 59, 80, 89, 96] |
| X2 | Swap 2 with 33 as 33 above and 2 smaller | [2, 10, 33, 52, 30, 14, 59, 80, 89, 96] |
| X14 | Swap 14 with 33 as 33 above and 14 smaller | [2, 10, 14, 52, 30, 33, 59, 80, 89, 96] |

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Explanation | Heap (as list) | Removed |
| M2 | Remove 2 from the root as smallest | [ , 10, 14, 52, 30, 33, 59, 80, 89, 96] | [2] |
| L | Move Left child up (10) and hole down | [10, , 14, 52, 30, 33, 59, 80, 89, 96] |  |
| R | Move Right child up (30) and hole down | [10, 30, 14, 52, , 33, 59, 80, 89, 96] |  |
| X96 | Move 96 up into the hole as it is the next smallest number down and gets rid of the hole | [10, 30, 14, 52, 96, 33, 59, 80, 89] |  |
| M10 | Remove 10 from the root as smallest | [ , 30, 14, 52, 96, 33, 59, 80, 89] | [2, 10] |
| R | Move Right child up (14) and hole down | [14, 30, , 52, 96, 33, 59, 80, 89] |  |
| X33 | Move 33 up into the hole as it is the next smallest number down and gets rid of the hole | [14, 30, 33, 52, 96, 59, 80, 89] |  |
| M14 | Remove 14 from the root as smallest | [ , 30, 33, 52, 96, 59, 80, 89] | [2, 10, 14] |
| L | Move Left child up (30) and hole down | [30, , 33, 52, 96, 59, 80, 89] |  |
| L | Move Left child up (52) and hole down | [30, 52, 33, , 96, 59, 80, 89] |  |
| X80 | Move 80 up into the hole as it is the next smallest number down and gets rid of the hole | [30, 52, 33, 80, 96, 59, 89] |  |
| M30 | Remove 30 from the root as smallest | [ , 52, 33, 80, 96, 59, 89] | [2, 10, 14, 30] |
| R | Move Right child up (33) and hole down | [33, 52, , 80, 96, 59, 89] |  |
| X59 | Move 59 up into the hole as it is the next smallest number down and gets rid of the hole | [33, 52, 59, 80, 96, 89] |  |
| M33 | Remove 33 from the root as smallest | [ , 52, 59, 80, 96, 89] | [2, 10, 14, 30, 33] |
| L | Move Left child up (52) and hole down | [52, , 59, 80, 96, 89] |  |
| L | Move Left child up (80) and hole down | [52, 80, 59, , 96, 89] |  |
| X89 | Move 89 up into the hole as it is the next smallest number down and gets rid of the hole | [52, 80, 59, 89, 96] |  |
| M52 | Remove 52 from the root as smallest | [ , 80, 59, 89, 96] | [2, 10, 14, 30, 33, 52] |
| X59 | Move 59 up into the hole as it is the next smallest number down and gets rid of the hole | [59, 80, 89, 96] |  |
| M59 | Remove 59 from the root as smallest | [ , 80, 89, 96] | [2, 10, 14, 30, 33, 52, 59] |
| L | Move Left child up (52) and hole down | [80, , 89, 96] |  |
| X89 | Move 89 up into the hole as it is the next smallest number down and gets rid of the hole | [80, 89, 96] |  |
| M80 | Remove 80 from the root as smallest | [ , 89, 96] | [2, 10, 14, 30, 33, 52, 59, 80] |
| L | Move Left child up (89) and hole down | [89, , 96] |  |
| X96 | Move 96 up into the hole as it is the next smallest number down and gets rid of the hole | [89, 96] |  |
| M89 | Remove 89 from the root as smallest | [ , 96] | [2, 10, 14, 30, 33, 52, 59, 80, 89] |
| X96 | Move 96 up into the hole as it is the next smallest number down and gets rid of the hole | [96] |  |
| M96 | Remove 96 from the root as smallest | [ ] | [2, 10, 14, 30, 33, 52, 59, 80, 89, 96] |

|  |  |  |
| --- | --- | --- |
| Operation | Explanation | AVL Tree (as list) |
| I33 | Insert 33 as root | [33] |
| I30L33 | Insert 30 left of 33 as less than 33 | [33, 30] |
| I2L30 | Insert 2 left of 30 as less than 30 | [33, 30, 2] |
| R30 | Rotate tree about 30 so 30 becomes the root | [30, 2, 33] |
| I52R33 | Insert 52 right of 33 as bigger than 33 | [30, 2, 33, 52] |
| I10R2 | Insert 10 right of 2 as bigger than 2 but less than 30 | [30, 2, 33, 10, 52] |
| I14R10 | Insert 14 right of 10 as bigger than 10 but less than 2 | [30, 2, 33, 10, 52, 14] |
| R10 | Rotate tree about 10 so left child becomes 10 and its children are 2 on the left and 14 on the right now | [30, 10, 33, 2, 14, 52] |
| I59R52 | Insert 59 right of 52 as bigger than 52 | [30, 10, 33, 2, 14, 52, 59] |
| R52 | Rotate tree about 52 so right child becomes 52 and its children are 33 on the left and 59 on the right now | [30, 10, 52, 2, 14, 33, 59] |
| I80R59 | Insert 80 right of 59 as bigger than 59 | [30, 10, 52, 2, 14, 33, 59, 80] |
| I89R80 | Insert 89 right of 80 as bigger than 80 | [30, 10, 52, 2, 14, 33, 59, 80, 89] |
| R80 | Rotate tree about 80 so that right child of 52 is now 80 and its children are now 59 on the left and 89 on the right | [30, 10, 52, 2, 14, 33, 80, 59, 89] |
| I96R89 | Insert 96 right of 89 as bigger than 89 | [30, 10, 52, 2, 14, 33, 80, 59, 89, 96] |
| R33 | Rotate tree about 33 twice so that 33 becomes the new root | [33, 30, 52, 10, 80, 2, 14, 59, 89, 96] |
| R33 | 33 should now be the root as there are 4 elements less than it and 5 more than it, tree can be balanced this way | [33, 30, 52, 10, 80, 2, 14, 59, 89, 96] |
| R14 | Rotate tree about 14 twice so that 14 becomes the left child of 33 and its children are 10 on the left and 30 on the right now | [33, 14, 52, 10, 30, 80, 2, 59, 89, 96] |
| R59 | Rotate tree about 59 twice so that 59 becomes the right child of 33 and its children are 52 on the left and 80 on the right now | [33, 14, 59, 10, 30, 52, 80, 2, 89, 96] |
| R89 | Finally, rotate tree about 89 so that 89 becomes the right child of 59 and its children are 80 on the left and 96 on the right now | [33, 14, 59, 10, 30, 52, 89, 2, 80, 96] |

|  |  |  |
| --- | --- | --- |
| Operation | Explanation | Hash Table (as array) |
| P0 | Probe index 0 as h1(33) = 33 mod 11 = 0 | [ , , , , , , , , , , ] |
| I33@0 | Insert 33 at 0 as it is free | [33, , , , , , , , , , ] |
| P8 | Probe index 8 as h1(30) = 30 mod 11 = 8 | [33, , , , , , , , , , ] |
| I30@8 | Insert 30 at 8 as it is free | [33, , , , , , , , 30, , ] |
| P2 | Probe index 2 as h1(2) = 2 mod 11 = 2 | [33, , , , , , , , 30, , ] |
| I2@2 | Insert 2 at 2 as it is free | [33, , 2, , , , , , 30, , ] |
| P8 | Probe index 8 as h1(52) = 52 mod 11 = 8 | [33, , 2, , , , , , 30, , ] |
| P9 | Probe index 9 as index 8 isn't free and linear probing is used (+1 to previously probed index) | [33, , 2, , , , , , 30, , ] |
| P52@9 | Insert 52 at 9 as it is free | [33, , 2, , , , , , 30, 52, ] |
| P10 | Probe index 10 as h1(10) = 10 mod 11 = 10 | [33, , 2, , , , , , 30, 52, ] |
| I10@10 | Insert 10 at 10 as it is free | [33, , 2, , , , , , 30, 52, 10] |
| P3 | Probe index 3 as h1(14) = 14 mod 11 = 3 | [33, , 2, , , , , , 30, 52, 10] |
| I14@3 | Insert 14 at 3 as it is free | [33, , 2, 3, , , , , 30, 52, 10] |
| P4 | Probe index 4 as h1(59) = 59 mod 11 = 4 | [33, , 2, 3, , , , , 30, 52, 10] |
| I59@4 | Insert 59 at 4 as it is free | [33, , 2, 3, 59, , , , 30, 52, 10] |
| P3 | Probe index 3 as h1(80) = 80 mod 11 = 3 | [33, , 2, 3, 59, , , , 30, 52, 10] |
| P4 | Probe index 4 as index 3 isn't free and linear probing is used (+1 to previously probed index) | [33, , 2, 3, 59, , , , 30, 52, 10] |
| P5 | Probe index 4 as index 3 isn't free and linear probing is used (+1 to previously probed index) | [33, , 2, 3, 59, , , , 30, 52, 10] |
| I80@5 | Insert 80 at 5 as it is free | [33, , 2, 3, 59, 80, , , 30, 52, 10] |
| P1 | Probe index 1 as h1(89) = 89 mod 11 = 1 | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| I89@1 | Insert 89 at 1 as it is free | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P8 | Probe index 8 as h1(96) = 8 mod 11 = 8 | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P9 | Probe index 9 as index 8 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P10 | Probe index 10 as index 9 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P0 | Probe index 0 as index 10 isn't free and linear probing is used (+1 to previously probed index, reset to 0 as mod 11 used [max index 10]) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P1 | Probe index 1 as index 0 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P2 | Probe index 2 as index 1 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P3 | Probe index 3 as index 2 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P4 | Probe index 4 as index 3 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P5 | Probe index 5 as index 4 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| P6 | Probe index 6 as index 5 isn't free and linear probing is used (+1 to previously probed index) | [33, 89, 2, 3, 59, 80, , , 30, 52, 10] |
| I96@6 | Finally, insert 96 at 6 as it is free | [33, 89, 2, 3, 59, 80, 96, , 30, 52, 10] |

|  |  |  |
| --- | --- | --- |
| Operation | Explanation | Hash Table (as array) |
| P0 | Probe index 0 as h1(33) = 33 mod 11 = 0 | [ , , , , , , , , , , ] |
| I33@0 | Insert 33 at 0 as it is free | [33, , , , , , , , , , ] |
| P8 | Probe index 8 as h1(30) = 30 mod 11 = 8 | [33, , , , , , , , , , ] |
| I30@8 | Insert 30 at 8 as it is free | [33, , , , , , , , 30, , ] |
| P2 | Probe index 2 as h1(2) = 2 mod 11 = 2 | [33, , , , , , , , 30, , ] |
| I2@2 | Insert 2 at 2 as it is free | [33, , 2, , , , , , 30, , ] |
| P8 | Probe index 8 as h1(52) = 52 mod 11 = 8 | [33, , 2, , , , , , 30, , ] |
| P10 | Since there is a collision, use double hashing substituting into the formula: new\_bucket = (h1(x) + i\*h2(x)) mod n where n is the size of the hash table (i.e. 11 in this case) and i is incremented every time there is a collision [can be considered as number of collisions so far] and h2(x) is the secondary hashing function giving h2(52) = (52 mod 3) + 1 = 2 (Yusuf et al., Collision resolution techniques in Hash table: A review - 2021, Page 758) | [33, , 2, , , , , , 30, , ] |
| I52@10 | Insert 52 at 10 as it is free | [33, , 2, , , , , , 30, , 52] |
| P10 | Probe index 10 as h1(10) = 10 mod 11 = 10 | [33, , 2, , , , , , 30, , 52] |
| P1 | Probe index 1 as there is a collision and new\_bucket = (h1(10) + i\*h2(10)) mod 11 = (10 + 1\*2) mod 11 = 12 mod 11 = 1 | [33, , 2, , , , , , 30, , 52] |
| I10@1 | Insert 10 at 1 as it is free | [33, 10, 2, , , , , , 30, , 52] |
| P3 | Probe index 3 as h1(14) = 14 mod 11 = 3 | [33, 10, 2, , , , , , 30, , 52] |
| I14@3 | Insert 14 at 3 as it is free | [33, 10, 2, 14, , , , , 30, , 52] |
| P4 | Probe index 4 as h1(59) = 59 mod 11 = 4 | [33, 10, 2, 14, , , , , 30, , 52] |
| I59@4 | Insert 59 at 4 as it is free | [33, 10, 2, 14, 59, , , , 30, , 52] |
| P3 | Probe index 3 as h1(80) = 80 mod 11 = 3 | [33, 10, 2, 14, 59, , , , 30, , 52] |
| P6 | Probe index 6 as there is a collision and new\_bucket = (h1(80) + i\*h2(80)) mod 11 = (3 + 1\*3) mod 11 = 6 mod 11 = 6 | [33, 10, 2, 14, 59, , , , 30, , 52] |
| I80@6 | Insert 80 at 6 as it is free | [33, 10, 2, 14, 59, , 80, , 30, , 52] |
| P1 | Probe index 1 as h1(89) = 89 mod 11 = 1 | [33, 10, 2, 14, 59, , 80, , 30, , 52] |
| P4 | Probe index 4 as there is a collision and new\_bucket = (h1(89) + i\*h2(89)) mod 11 = (1 + 1\*3) mod 11 = 4 mod 11 = 4 | [33, 10, 2, 14, 59, , 80, , 30, , 52] |
| P7 | Probe index 7 as there is a collision and new\_bucket = (h1(89) + i\*h2(89)) mod 11 = (1 + 2\*3) mod 11 = 7 mod 11 = 7 | [33, 10, 2, 14, 59, , 80, , 30, , 52] |
| I89@7 | Insert 89 at 7 as it is free | [33, 10, 2, 14, 59, , 80, 89, 30, , 52] |
| P8 | Probe index 8 as h1(96) = 96 mod 11 = 8 | [33, 10, 2, 14, 59, , 80, 89, 30, , 52] |
| P9 | Probe index 9 as there is a collision and new\_bucket = (h1(96) + i\*h2(96)) mod 11 = (8 + 1\*1) mod 11 = 9 mod 11 = 9 | [33, 10, 2, 14, 59, , 80, 89, 30, , 52] |
| I96@9 | Finally, insert 96 at 9 as it is free | [33, 10, 2, 14, 59, , 80, 89, 30, 96, 52] |