CSE 310 Assignment 4 Solutions

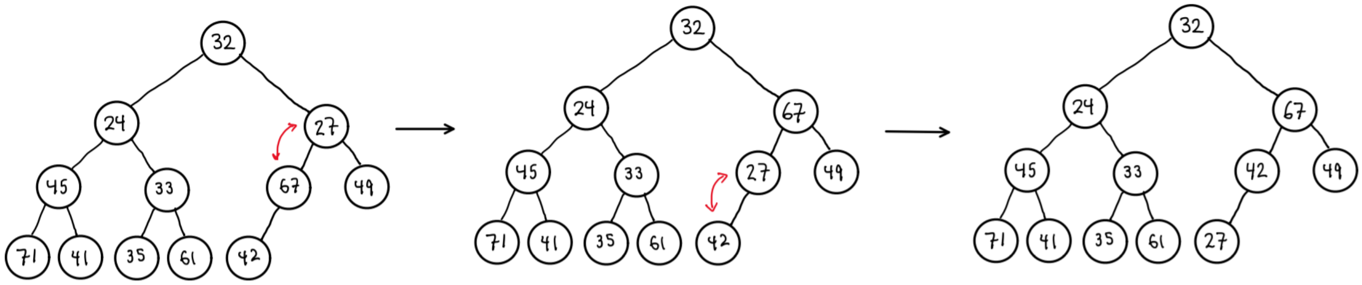
Taman Truong

February 10th, 2022

Problem 1

Illustrate the operation of MAX-HEAPIFY(A, 3) on the array $A = \{32, 24, 27, 45, 33, 67, 49, 71, 41, 35, 61, 42\}$ by re-drawing the tree for every swap.

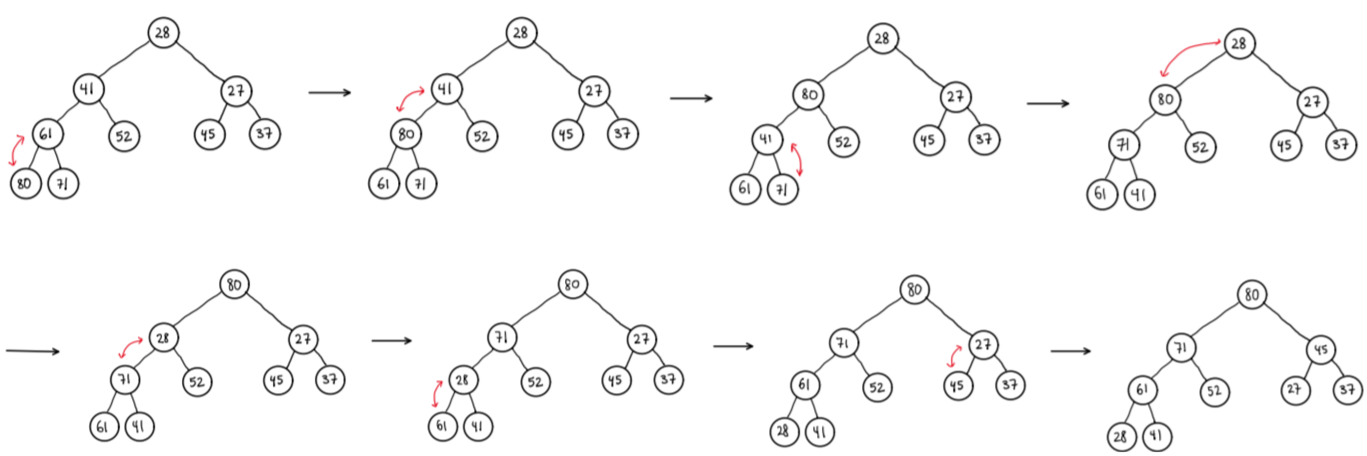
Solution



Problem 2

Illustrate the operation of BUILD-MAX-HEAP on the array $A = \{28, 41, 27, 61, 52, 45, 37, 80, 71\}$ by re-drawing the tree for every swap.

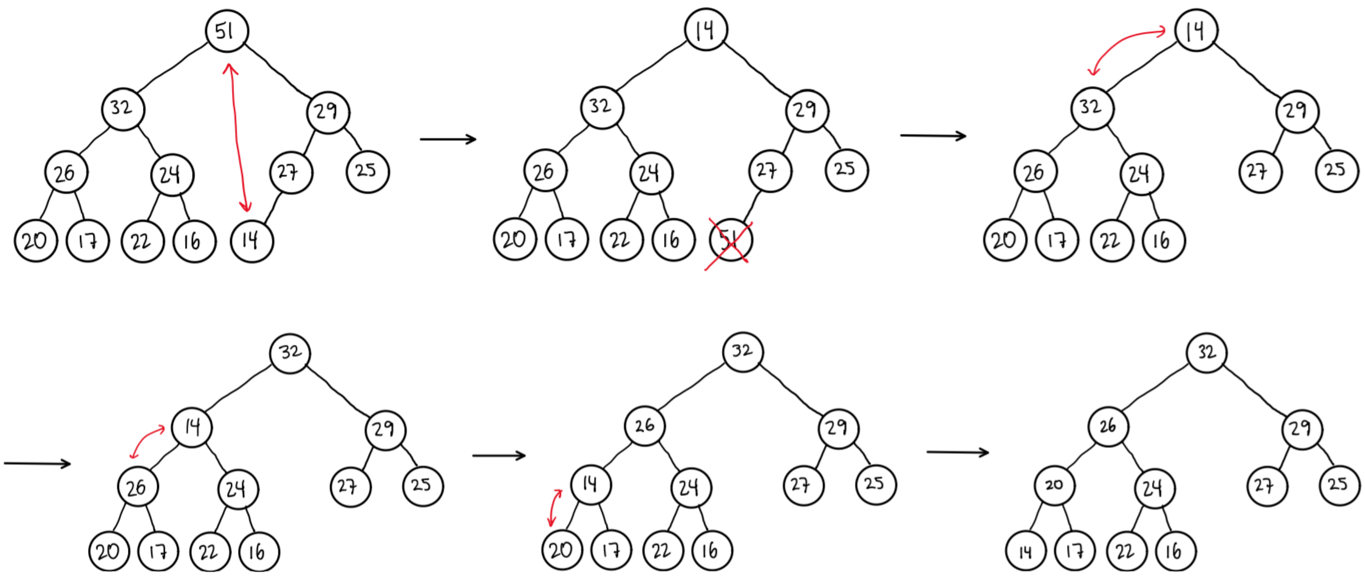
Solution



Problem 3

Illustrate the operation of HEAP-EXTRACT-MAX on the heap $A = \{51, 32, 29, 26, 24, 27, 25, 20, 17, 22, 16, 14\}$ by re-drawing the tree for every swap.

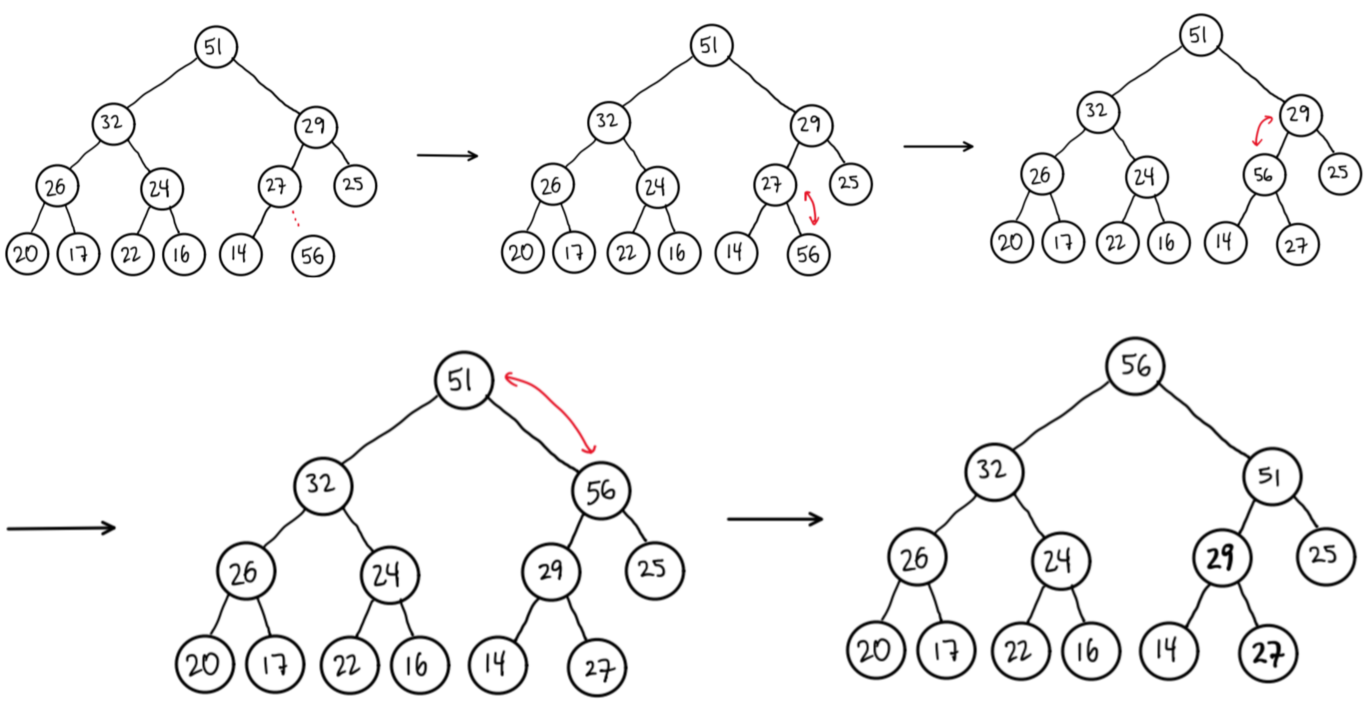
Solution



Problem 4

Illustrate the operation of MAX-HEAP-INSERT(A, 56) on the heap $A = \{51, 32, 29, 26, 24, 27, 25, 20, 17, 22, 16, 14\}$ by re-drawing the tree for every swap.

Solution



Problem 5

Consider the following algorithm given by the pseudo-code. You can assume that the input array A has at least one element.

```
FUNCTION1 (A) //A is an array of length n

n = length[A]
A[1] = 3
s = A[1]
for (i=2; i<=n; i++)
      if ((A[i] mod 3 ) == 0 )
      s = s + A[i]
return s
```

Using a loop invariant, prove that the algorithm is correct.

Solution

Proof: The loop invariant is that the algorithm always computes the sum of the numbers in the array that are divisible by 3 and returns this sum. We will prove that this loop invariant holds for all parts of the algorithm: initialization, maintenance, and termination.

Initialization: Before the first iteration, the first element in the array is $A[1] = 3$. Since there is only one element in this array, the sum of the elements in the array is 3. Since the sum 3 is divisible by 3, therefore, the loop invariant is preserved and the algorithm satisfies the initialization property.

Maintenance: Before each iteration i, we know that this array will contain at least one multiple of 3, since $A[1] = 3$. This is already accounted for in the running total for the sum of all of the multiples of 3 in the array of numbers A. For all elements after the first element in array A, we compare whether or not those number leave a remainder of 0 when they are divided by 3. If so, add that number to the running sum, and if not, move onto the next element. This algorithm will keep running until the algorithm has analyzed all of the numbers in the array, Therefore, the loop invariant is preserved and the algorithm satisfies the maintenance property.

Termination: Based on the loop invariant, since i is increased by 1 after every iteration of the for loop and $i > n$, all of the elements in the array have been checked, the numbers in the array that are divisible by 3 have been checked, and these numbers are summed up in the for loop and returned at the end of the algorithm. Therefore, the loop invariant is preserved and the algorithm satisfies the termination property.

Since the algorithm satisfies the three properties of initialization, maintenance, and termination, the algorithm is correct based on the loop invariant: it always computes the sum of the numbers in the array that are divisible by 3 and returns this sum. □