CSCE 221 HW 2

Alexander Akomer

February 29 2021

1 Question 3.20a

Advantages: Reduces time complexity when deleting.
Disadvantages: Increased time complexity when traversing (still have to iterate over marked nodes) and increased storage usage.

2 Question 3.24

Two Stacks in One Array Code:

using namespace std;
include $< iostream >$
class TwoStacks
{
private:
int* arr;
size$_t capacity$;
$size_t stack1_i$;
$size_t stack2_i$;

$public:$
$//defaultconstructor$
$TwoStacks()$
{
$capacity = 10;$
$arr = newint[capacity];$
$size_t stack1_i = 0;$
$size_t stack2_i = 1;$
}

void resize()
{
int* temp = new int[capacity];
for (int i = 0; i ¡ capacity; i++)

temp[i] = arr[i];

capacity *= 2;
delete arr;
arr = temp;
}

void shiftRight()
{
for (int i = capacity - 1; i ¿ stack1$_i; i - -)$
{
$arr[i - 1] = arr[i];$
}
$stack2_i + +;$
}

$voidpush1(intnum)$
{
$if(stack2_i == capacity - 1)$
{
$resize();$
}
$shiftRight();$
$arr[stack1_i] = num;$
$stack1_i + +;$
}

$voidpush2(intnum)$
{
$if(stack2_i == capacity - 1)$
$resize();$
$arr[stack2_i] = num;$
$stack2_i + +;$
}

$voiddisplay()$
{
$for(inti = 0; i < capacity; i + +)$
$cout << arr[i] << ""$;
$cout << endl;$
}
};

3 Question 3.25a

The push and pop operations of a stack can be implemented alongside a findMin function all in O(1) time using an array implementation. To achieve this, the smallest value is constantly tracked so that it can be compared to new ”pushed” values immediately, and therefore requires no traversal to find. This is easily achieved by starting the stack at index 1 and leaving index 0 as a copy of the lowest value in the array.

4 Question D

```
char* stack;
int start = -1;

void push(char ele)
{
stack[++start] = ele;
}

char pop()
{
return stack[start–];
}

int isPalindrome(char str[])
{
int length = strlen(str);

stack = (char*)malloc(length * sizeof(char));

int i = length / 2;

for (i = 0; i ¡ length / 2; i++) {
push(str[i]);
}

if (length
i++;
}

while (str[i] != ”) {
char ele = pop();

if (ele != str[i])
return 0;
i++;
}

return 1;
}
```

5 Question E

No, it is not possible for the preorder traversal to match the postorder traversal because the preorder traversal visits the root node, then the left node, then the right node whereas the postorder visits the left, then right, then root. The only case in which these traversals could match is a tree of one node. Example: In a tree where 8 is the root, 4 is the left child, and 12 is the right child, a preorder traversal is 8 4 12 and the postorder traversal is 4 12 8.
Yes, it is possible for the preorder traversal of a tree to visit nodes in the reverse order of the postorder traversal of a tree. This only works in the case of a tree with two nodes. For instance, if the tree contains 8 as a root and 4 for the left child, the preorder traversal will be 8 4 while the postorder traversal will be 4 8, the reverse of the preorder.

6 Question 4.5

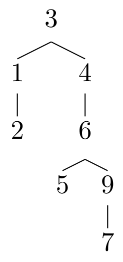
The maximum number of nodes in a binary tree of height h is 2 to the power of (h+1)-1.
Base case: When h = 0, the maximum number of nodes becomes 1 (which is correct).
Inductive hypothesis: h = k + 1
So for height of k + 1:
2 to the power of (k + 1) - 1 + 2 to the power o f(k + 1) - 1 + 1
= 2 to the power of ((k + 1) + 1) - 1
Which holds true for h = k + 1.

7 Question 4.8

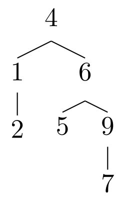
The prefix expression for the tree is -**ab+cde
The infix expression for the tree is (a * b) * (c + d) - e
The postfix expression for the tree is ab * cd + * e -

8 Question 4.9ab

Original tree:



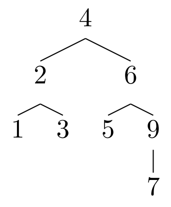
After deletion:



9 Question 4.18b

The number of nodes in an AVL tree is: S(h) = S(h-1) + S(h-2) + 1
For an AVL tree of height 15, there are 2,583 nodes (at minimum). This was determined by coding the function and testing the output.

10 Question 4.19



11 Question 4.25

(a) In an N-node AVL tree the depth is O(log(N)). To represent an integer range of M we need O(log(M)) bits. Therefore the result is O(log(log(N)). As an example, suppose we have an AVL tree with 256 nodes. The depth is around log(256) = 8
Therefore we need around log(8) = 3 bits.
(b) Assuming unsigned data type we can represent 2 to the power of 8 = 256 depth values. Then an AVL tree with more than 2 to the power of 256 nodes, or equivalently that has a height bigger than 256, will overflow the counter.