CSCI 220 -- Homework 2

Nero Li

Chapter 5

R-5.3

S = 25 – (10 – 3) = 18

R-5.5

Output: 3 8 2 1 6 7 4 9

R-5.9

Output: 5 3 2 8 9 1 7 6

R-5.11

Algorithm func(deque D, queue Q):

    For i from 1 to 4:

        Q.enqueue(D.front())

        D.eraseFront()

    End for

    D.insertRear(D.front())

    D.eraseFront()

    For i from 1 to 3:

        Q.enqueue(D.front())

D.eraseFront()

End for

For i from 1 to 3:

D.insertRear(Q.front())

Q.dequeue()

End for

D.insertRear(D.front())

D.eraseFront()

For i from 1 to 4:

D.insertRear(Q.front())

Q.dequeue()

End for

C-5.2

First of all, do a for loop n times:

We check the front element in the queue to see it is a certain element x or not.

Push the front element into S and dequeue the Q.

Then, do a for loop n times again:

Enqueue the top element into Q from S.

Pop the element from S.

After that, do a for loop n times again:

Push the front element into S and dequeue the Q.

Finally, do a for loop n times again:

Enqueue the top element into Q from S.

Pop the element from S.

C-5.5

For push function:

Enqueue the element into queue A.

Running time for push function will be O(1).

For pop function:

Dequeue all the elements from queue A and send them into queue B until there are only one element exist in queue A.

Dequeue that last one element in queue and send it as what we popped.

Sent queue B as queue A.

Running time for pop function will be O(n) and n is the size of queue A.

Chapter 6

R-6.3 – C++ or Java

```
void func(int arr[], int size, int d)
{
    for (int i = 0; i < d; ++i)
    {
        int temp = arr[size - 1];
        for (int j = size - 1; j > 0; --j)
        {
            arr[j] = arr[j - 1];
        }
        arr[0] = temp;
    }

    for (int i = 0; i < size; ++i)
    {
        cout << arr[i] << ' ';
    }
}
```

R-6.9

If k = 10, to enlarge array to 20, the total cyber-dollars we should store in the array should be 30 dollars, which means each part will have three dollars. Because of that, 4 dollars will be the least charge for the array to expand. If k = 20, to enlarge array to 40, the total cyber-dollars we should store in the array should be 60 dollars, which means each part will have three dollars. Because of that, 4 dollars will still be the least charge for the array to expand.

Hence, if we have an array with size 10, when we want to enlarge that, we will need at least 4 dollars for each part to make sure the array have enough dollar to charge.

R-6.16

Minimum: 1

Maximum: n -1

R-6.18

```
bool findNum(NodeSequence S, int n, int k)
{
    --n;
    if (n == 0 && S.indexOf(n) != k)
    {
        return false;
    }
    else if (S.indexOf(n) == k)
    {
        return true;
    }
    else
    {
        return findNum(S, n, k);
    }
}
```

The maximum space my function would use will be n * findNum() when we found the number at the front or didn't find the number.

C-6.10

a. i = 0: 0

   i = 1: 0 1

   i = 2: 0 1 2

   i = 3: 0 1 3 2

   i = 4: 0 1 3 2 4

   i = 5: 0 1 3 2 5 4

   i = 6: 0 1 3 6 2 5 4

   i = 7: 0 1 3 7 6 2 5 4

   i = 8: 0 1 3 7 6 2 5 4 8

   i = 9: 0 1 3 7 6 2 5 4 9 8

   i = 10: 0 1 3 7 6 2 5 10 4 9 8

b. if n comes from power of 2

   then the sequence will be 0 1 3 … n/2 n

   else

   the sequence will be 0 1 3 … n/2 n … k/2 k

   where k is a number that comes from power of 2 and smaller than n

C-6.13

```
vector<int> vec;
void randomDeck()
{
    for (int i = 0; i < n; ++i)
    {
        int j{randomInteger(n)};
        int t;
        t = vec[i];
        vec[i] = vec[j];
        vec[j] = t;
    }
}
```

The running time for this function is O(n) where n is the size of the vector.

Chapter 7

R-7.2

a. /user/rt/courses/

b. /user/rt/courses/; cs016/; cs252/; homeworks/; programs/; projects/; papers/; demos/

c. 9

d. 1

e. grades; programs/

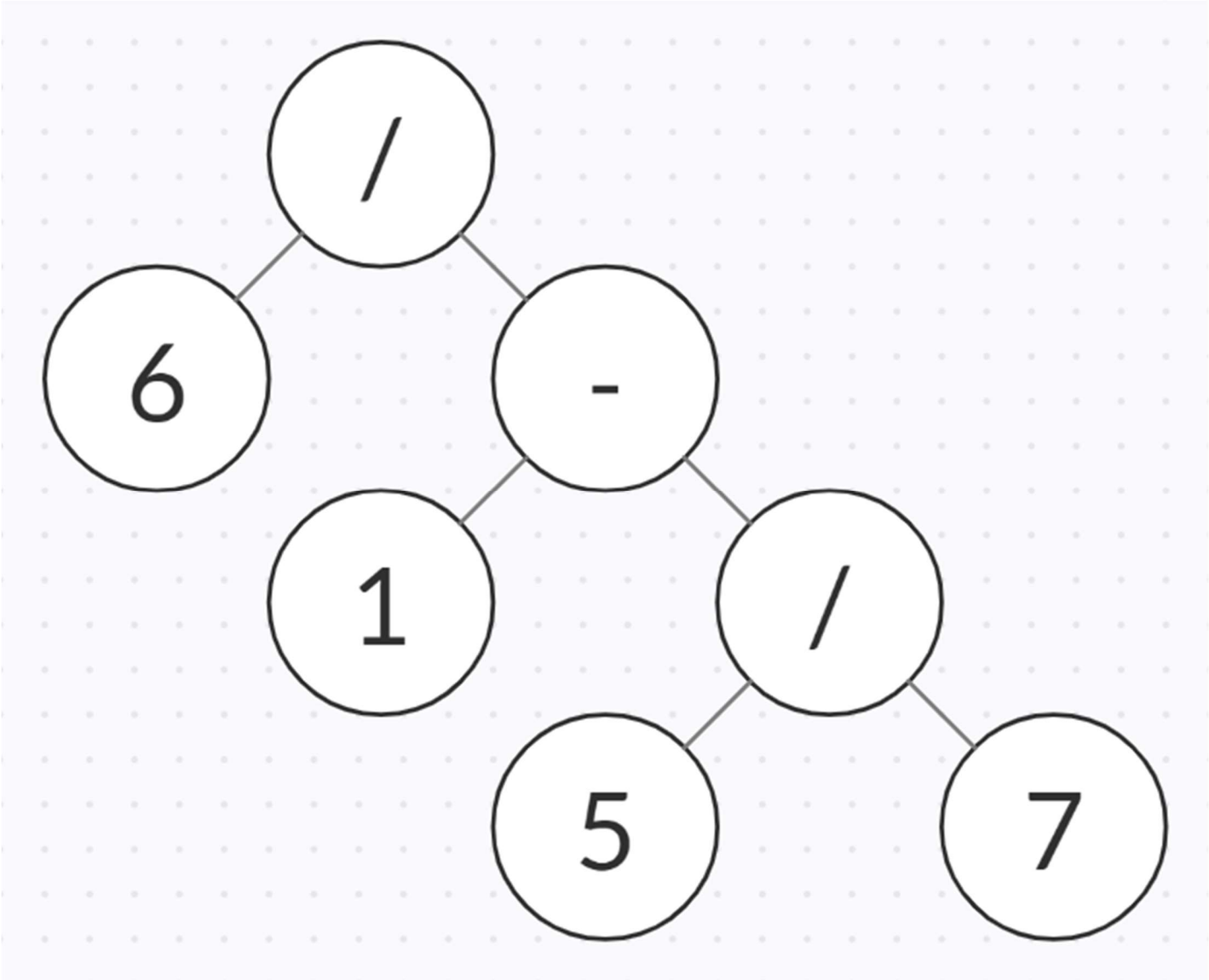
f. papers/; buylow; sellhigh; demos/; market

g. 3

h. 4

R-7.11

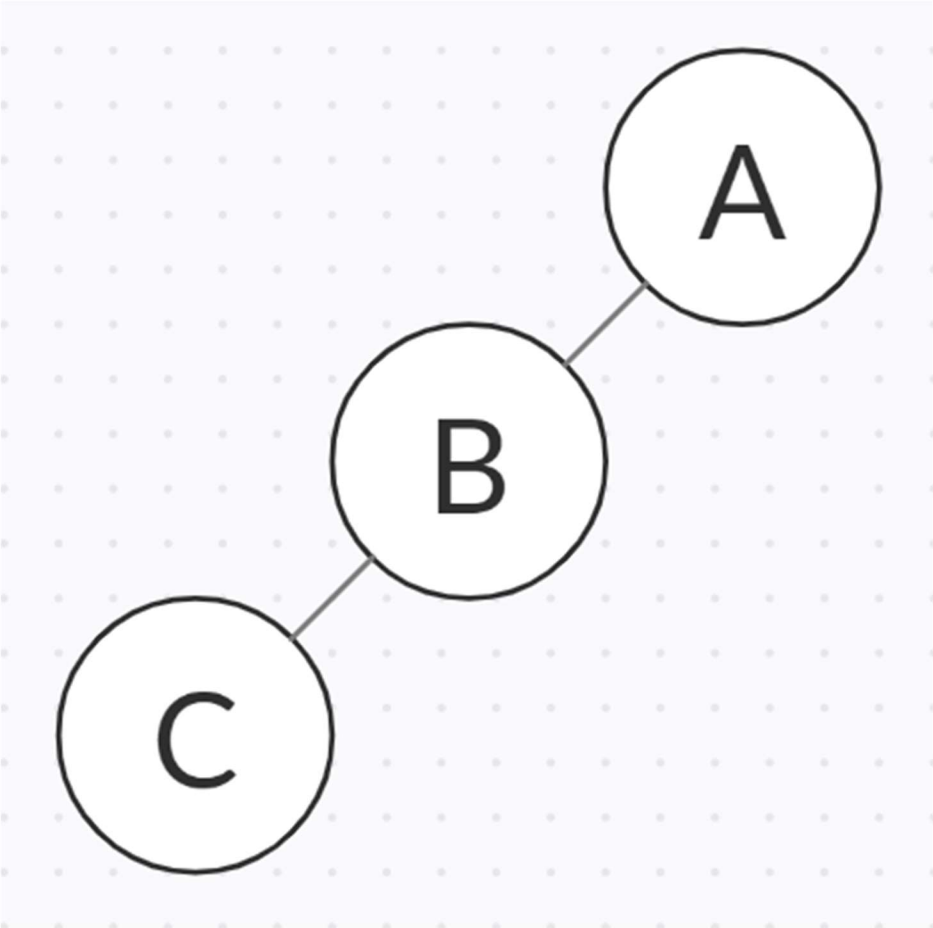
$$\frac{6}{1-\frac{5}{7}} = \frac{6}{\frac{7-5}{7}} = \frac{6*7}{2} = 3*7 = 21$$



R-7.12

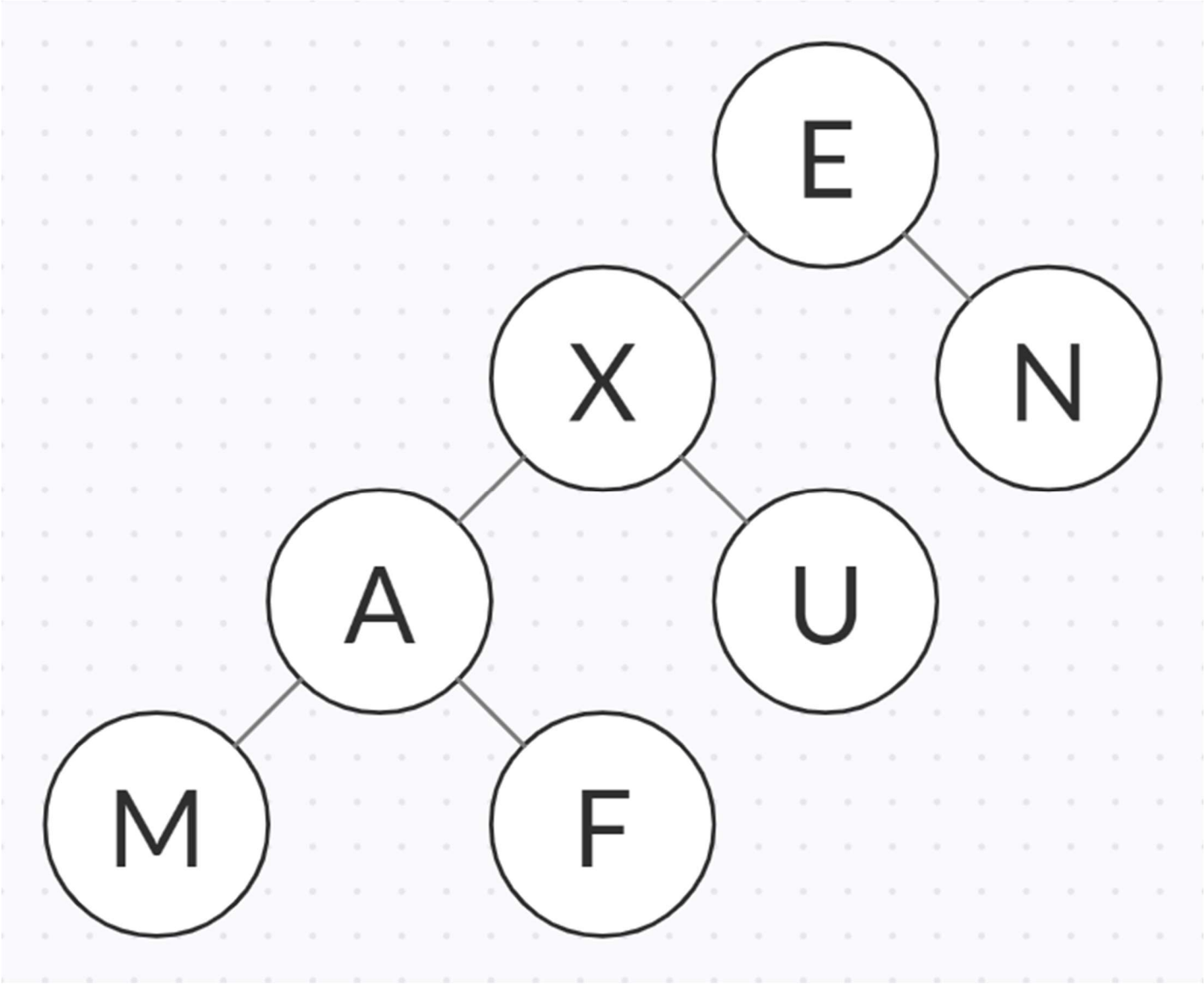
No, preorder will always travel the root first but post-order will travel the root at the end. Unless the tree has only one node, it is impossible for a tree to have the same preorder and post-order traversal path.

Yes, and here is the example:



For this example, preorder will be ABC and post-order will be CBA.

R-7.15



C-7.7

```
void indentedParentheticRepresentation(Tree *root, int t)
{
    Tree *cur = root;
    list<Tree*> curChild = cur->child;

    for (int i = 1; i < t; ++i)
    {
        cout << '\t';
    }
    cout << cur->elem;

    if (curChild.empty())
    {
        cout << endl;
        return;
    }

    cout << " (" << endl;

    while (!curChild.empty())
    {
        indentedParentheticRepresentation(curChild.front(), t + 1);
        curChild.pop_front();
    }

    for (int i = 1; i < t; ++i)
    {
        cout << '\t';
    }
    cout << ')' << endl;
}
```

C-7.34*

```
void levelOrder(Tree *tree[], int n)
{
    queue<int> que;
    int cur;

    que.push(0);
    while (!que.empty())
    {
        cur = que.front();
        que.pop();
        cout << tree[cur] << ' ';
        if (tree[2 * cur])
        {
            que.push(2 * cur);
        }
        if (tree[2 * cur + 1])
        {
            que.push(2 * cur + 1);
        }
    }
    cout << endl;
}
```