Data Structures
CS 246 - 040
Department of Physics and Computer Science
Medgar Evers College
Exam 4

Instructions:

- **The exam requires completing tasks in a cpp file within 120 minutes.**
- **Accompanying this file is a template cpp file that you must modify. You cannot add additional libraries to or remove any libraries from the file. All other modifications are allowed.**
- **Your submissions must be submitted to the Exams directory of your github repository and/or as attachments on Google classroom under the Exam04 assessment. The file must have its accurate extension.**
- **Cheating of any kind is prohibited and will not be tolerated.**
- **Violating and/or failing to follow any of the rules will result in an automatic zero (0) for the exam.**

TO ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS ABOVE, AT THE BEGINNING OF YOUR SUBMISSION(S), ADD A COMMENT THAT CONSISTS OF YOUR NAME AND THE DATE

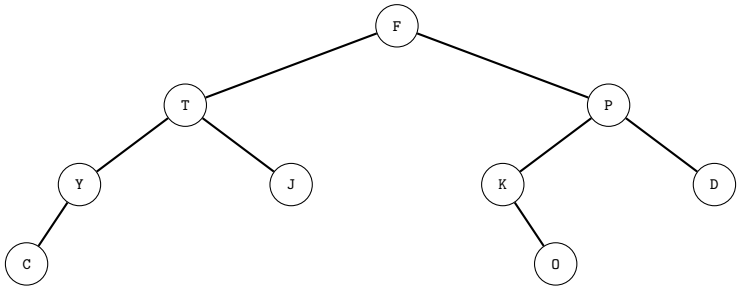
Grading:

Section	Maximum Points	Points Earned
Fundamentals	10	
Problem Solving	10	
Total	20	

Fundamentals

1. **Write ONLY what is requested in the commented section titled Fundamentals**

   a. Write the inorder, postorder and preorder traversal of the following binary tree. The values must be written in a line such that they separated with a comma. Furthermore, each traversal must be preceded by its name.



   b. Use the master theorem to determine the big-O runtimes of the following recursive runtime functions. You must show work [prove the case] to receive full credit.

      1. $T(n) = 7T(n/7) + 8$
      2. $T(n) = T(n/5) + \log(n)$
      3. $T(n) = \frac{2}{3}T(3n/2) + n$
      4. $T(n) = 9T(n/3) + n^2$
      5. $T(n) = 2T(n/4) + \lg(n)$

Problem Solving

2. **After the commented section titled Problem Solving, write the definition of the following functions and any helper functions you may need.**

   a. a bool function named `HasSum()` whose header is

```
bool HasSum(Array<int>& a,Array<int>& b,int n)
```

   Given that both *Array* parameters only consist of numbers between 1 and 100 inclusively, the function returns true if n is a positive number and there exists an element from a and an element from b whose values sum to n; otherwise, it returns false. Furthermore, the function must have a worst-case scenario linear big-O runtime.

   b. a generic *Node* pointer function named `Convert()` whose header is

```
template <typename T>
dn::Node<T>* Convert(TNode<T>* root)
```

   It returns a copy of the elements of *root* as a doubly linked list such that the copy is in inorder.

   Hint: delegate work to another function.