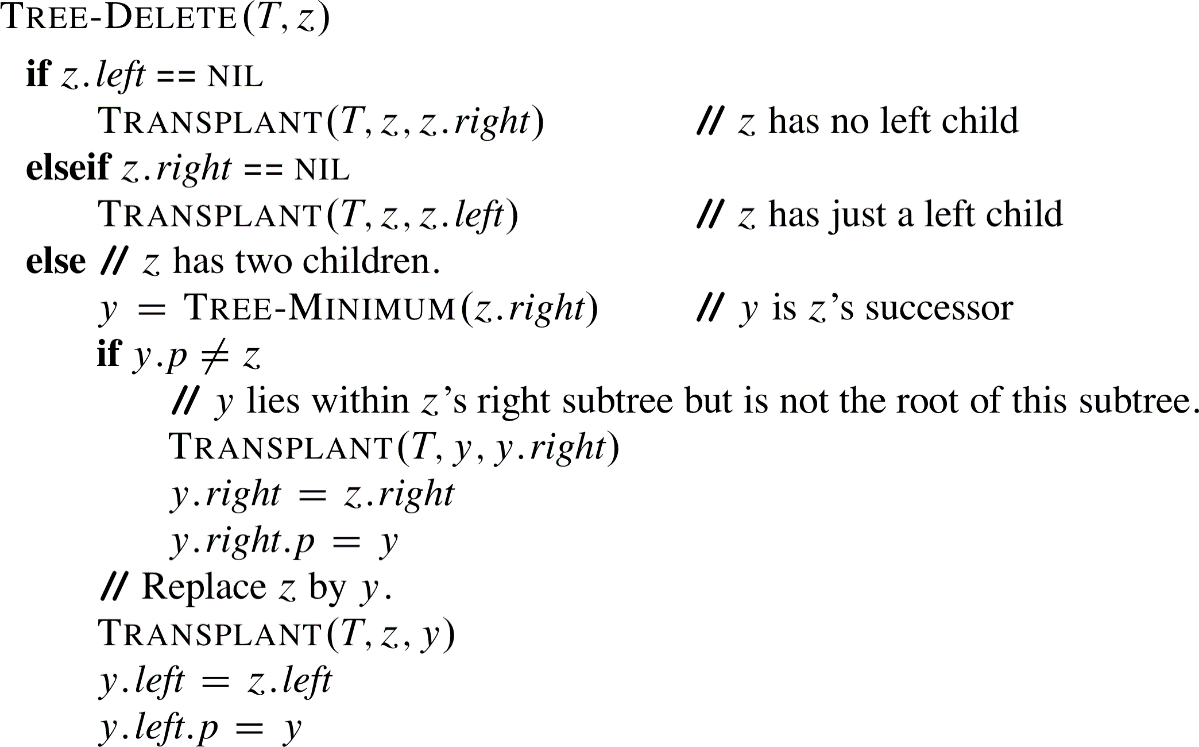
1. For each of the following program fragments, determine the time complexity using the notation. Explain your analysis next to each program fragment **[2\*3 = 6 points]**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Program Fragment | Time Complexity  Notation) | Explanation |
| 1 | for (int i=0; i<n; i++){  for (int j=0; j<n\*n; j++){  a = a + i + j;  }  } |  |  |
| 2 | for (int i = 1; i < n; i \*= 2)  {  a = a + i;  } |  |  |
| 3 | If (a==b)**{**  for (int i=0; i<3\*n; i++){  for (int j=0; j<2\*n; j++){  a = a + i + j; }  }  **}**  Else **{**  int a=0,i=n;  while(i>0){  a=a+i;  i=i/2;  }  **}** |  |  |

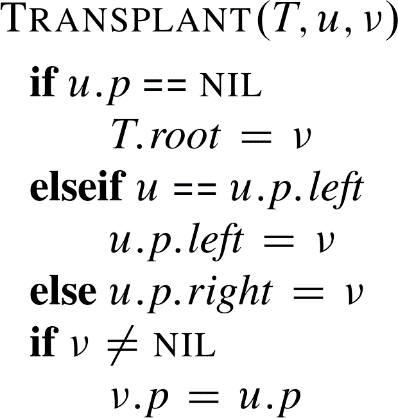
3. Given an input sequence of character keys: **[123,96,145,250,17,33,21,124,121,130]** illustrate the creation of a **binary search tree** if these keys are inserted in the above order. [**5 points**] *(You don’t have to draw each steps, you can draw the final BST if you want)*

Show what the tree would look like after the following changes are made to this tree in the following order (make sure to include all the intermediate steps and explain each step): [**6 points**]

* 1. **Delete** node with value **33**
  2. **Delete** node with value **145**
  3. **Insert** node with value **129**

*The algorithm for deleting a node in a binary search tree is given below*:

|  |
| --- |
|  |



4. Given the following character keys: **[120, 38, 165, 190, 185, 200, 250]** show the steps involved in creating a **red-black tree**. Make sure to draw the tree after each key is inserted in the given sequence. [**10 points**]

The pseudo-code for red-black tree insert is given below.

|  |  |
| --- | --- |
|  |  |
|  | |

5. What is a Binary Tree Traversal? **[2 points].** How many different traversal methods exist? Explain them **[4 points]**

6. Explain the **similarities** and **differences** between **BFS** and **DFS** in terms of *algorithms* and *data structures*. **[7 points]**