CSE 208 Offline 5

Assignment on Balanced BST

Deadline: 22 Jan, 2024 11:55pm

Problem Specification:

You have heard [or used] the STL 'map' in C++. This is an ordered container that uses **Red-Black Tree** in its source code. In this problem, you have to implement the STL 'map' using the Red-Black Tree. Here, each element in the map has an integer key and a string value. Since this container is sorted, iteration through the elements are always in a sorted order instead of insertion order.

The 'map' has the following methods:

1. Insert: It inserts a key-value element in the map. Remember a map cannot have the same key multiple times. In the event of inserting a new element with an existing key, the value of the element gets replaced. [See I/O for further details]
2. Erase: It deletes the key-value element in the tree if it exists.
3. Clear: Removes all elements from the map container.
4. Find: Searches whether the integer key is present in our tree.
5. Empty: It checks if the map is empty.
6. Size: It counts the number of elements in the map container.
7. Iteration: It iterates the map in an ascending order based on the integer key.

Input:

Each line in the input will specify one of the following operations:

1. Insert(I) followed by an integer and a string denoting the key and value of the element to be inserted
2. Erase(E) followed by an integer denoting the key of the element to be deleted
3. Clear(Clr) erases all the elements from the map container.
4. Find (F) followed by an integer denoting the key of the element to be searched for
5. Empty(Em)
6. Size(S)
7. Iteration(Itr)

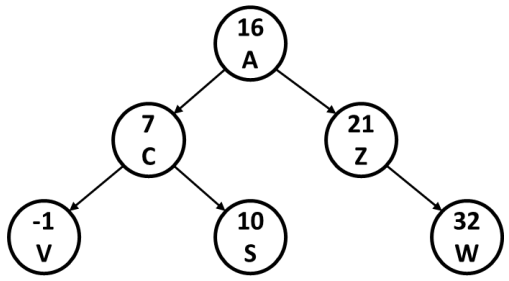
Output:

1. Insert: the current state of the tree after insertion will be printed in the nested parentheses format. For example, the nested parentheses format for the tree shown will be written as:

CSE 208 Offline 5

Assignment on Balanced BST

Deadline: 22 Jan, 2024 11:55pm



16_A(7_C(-1_V,10_S),21_Z(,32_W))

You have to print the elements in white or red according to your tree.

2. Erase: same as insert
3. Clear: print “successful” or “unsuccessful”
3. Find: print “found” or “not found”
4. Empty: print “yes” or “no”
5. Size: print number of elements
6. Iteration: print the inorder traversal of the elements based on keys. An in-order traversal of the shown tree would be:

-1 ⇒ V
7 ⇒ C
10 ⇒ S
16 ⇒ A
21 ⇒ Z
32 ⇒ W

You have to print the elements in white or red according to your tree.

Sample I/O:

Use file I/O for this problem. You will take input from input.txt and your output file should be ‘output.txt’. You also need to print on the terminal conserving the colour of the nodes in the tree in the following way.

The sample file i/o is here.

Input	**Output [On Console]**
Clr Em	unsuccessful yes

CSE 208 Offline 5

Assignment on Balanced BST

Deadline: 22 Jan, 2024 11:55pm

I 10 Thors	10_Thors
I 34 Canute	10_Thors(,34_Canute)
I 43 Olaf	34_Canute(10_Thors,43_Olaf)
I 15 Einer	34_Canute(10_Thors(,15_Einer),43_Olaf)
I 40 Olmar	34_Canute(10_Thors(,15_Einer),43_Olaf(40_Olmar,))
F 45 Estrid	45 not found
I 53 Floki	34_Canute(10_Thors(,15_Einer),43_Olaf(40_Olmar,53_Floki))
S	6
I 90 Thorfinn	34_Canute(10_Thors(,15_Einer),43_Olaf(40_Olmar,53_Floki(,90_Thorfinn)))
I 12 Snake	34_Canute(12_Snake(10_Thors,15_Einer),43_Olaf(40_Olmar,53_Floki(,90_Thorfinn)))
I 78 Askeladd	34_Canute(12_Snake(10_Thors,15_Einer),43_Olaf(40_Olmar,78_Askeladd(53_Floki,90_Thorfinn)))
F 40	40 found
E 40	34_Canute(12_Snake(10_Thors,15_Einer),78_Askeladd(43_Olaf(,53_Floki),90_Thorfinn))
F 40	40 not found
E 78	34_Canute(12_Snake(10_Thors,15_Einer),53_Floki(43_Olaf,90_Thorfinn))
S	7
E 12	34_Canute(10_Thors(,15_Einer),53_Floki(43_Olaf,90_Thorfinn))
E 43	34_Canute(10_Thors(,15_Einer),53_Floki(,90_Thorfinn))
Em	no
E 56	56 not found
I 15 Ymir	34_Canute(10_Thors(,15_Ymir),53_Floki(,90_Thorfinn))
Itr	10 ⇒ Thors 15 ⇒ Ymir 34 ⇒ Canute 53 ⇒ Floki 90 ⇒ Thorfinn
Clr	successful
S	0
E 90	90 not found

Hint:

1. You may keep the red-black tree implementation in a header file/another source file for reusability purposes and include it.
2. Modularize your code.
3. Using OOP is recommended for the better organising of your code.

CSE 208 Offline 5

Assignment on Balanced BST

Deadline: 22 Jan, 2024 11:55pm

4. You may use in total 3 classes in problem 2. Node class for each element [key-value], a class for red-black tree and another for the map container. [It is suggested, not a must do]
5. For coloured printing: you may follow this repo. [mainly installation and getting started]

Instructions:

(1) Please DO NOT COPY solutions from anywhere (your friends, seniors, internet etc.). Any form of plagiarism (irrespective of source or destination), will result in getting -100% marks in the online/offline.
(2) Rename all the problem solutions according to your student ID. If your ID is 2105XXX, then create a folder named 2105XXX. Afterward, rename problem 1 as 2105XXX_problem1.cpp, and similarly, rename the others. Next, move all the solutions inside the folder. Create a zip file of that folder. Lastly, submit the zip file.
(3) You get 10 marks for each right answer. A viva will also be conducted. If the teacher finds if you don't know how to implement it, you'll get a score of 0.