ADS project Report

Rujuta Ashish Hajarnis
UF-ID : 4129-7919 rhajarnis@ufl.edu

Project Objective :

- This project uses a min heap and a red black tree to keep a track of buildings being constructed.
- The minHeap class contains an array of objects with the building number, executed time and total time.
- The min heap keeps a track of the execution time and keeps the element with minimum execution time on top. This is done using the createHeap function.
- The red black tree prints the data according to the building number.

Code Structure:

- The code begins with the building class whose constructor is used to initialise the building attributes(building number , execution time and total time.)
- Next, the risingCity class constructor initialises the size and maxsize.

Methods of risingCity class:

1. IsLeaf method: used to check if a given node is a leaf or not.
2. exchange method: used to swap two nodes.
3. Insert method : used to insert a new value into the min heap from the input file.
4. minHeap method : it is called to create the min heap. It ensures that min heap property is always satisfied by all elements of the min heap.
5. Remove method : used to remove the building whose work is done(i.e executed time equals total time).
6. heapCreation method : used to minheapify the heap. This method covers various cases of minheapifying.

Cases in heapCreation method :

- First it checks if the execution time of left and right child of any parent node is not null.
- If the condition is true it checks if the execution time of parent is greater than that of its children. Further it checks which of the children have a smaller execution time.

- Once the smallest execution time is known, swap is performed between the parent node and the child node.
- If both children have the same execution time, swap with the child having the smaller building number.
- If parent , left child and right child have the same execution time , check if the building number of parent is greater than both of its children.
- If the above condition is true, swap with the child having smaller building number.
- If one of children(say right) is null , check if the execution time of parent is greater than the left child. If yes then swap.
- If both left child and parent have the same execution time , swap according to the building number.
- Similar case works when left child is null and right child is not null.

7. Next is the main function. The functionality of this method is as follows:

- Here, first the min heap is initialised. The array has a size of 2000 as maximum number of buildings that can be active at anytime are 2000.
- The input file is read using a bufferedReader. It then compares the condition with the input file and checks if the function to be performed is insert or print.
- It then extracts the value of building number and total time.
- The global_time variable is incremented continuously. If executed time for any building becomes equal to the total time , reset the counter. If that does not happen, reset the init variable to 0 after 5 days.
- Once all insertions have been performed, reduce the global time by 5.
- Until the execution time is not equal to total time check if the remaining time is less than 5. If yes then increase the global time by the remaining time and remove the building once the executed time becomes equal to the global time.
- If the remaining time is equal to 5, increase the global time and execution time for that building by 5.
- If remaining time is greater than 5, increase global time , execution time by 5 and call the mishap function.

Red Black Tree class :

- Second part of the code contains the code for a red black tree which is used for printing.
- RedBlackTree class contains the class node whose attributes are left child, right child, parent and colour.

Methods in the redblacktree class :

- The nodeLookup method searches the element to be deleted and returns its value to the delete function.
- The insert method checks if the the node inserted is the root. If yes, then it is initialised to black colour otherwise the new node is initialised to red colour and and root becomes black in colour.
- FixT method is used to fix the red black tree whenever a new node is inserted.
- RotateLC is used to rotate the left child. It considers the cases of left left rotation , in which first the grandparent is rotated to right and colours of parent and grandparent are exchanged. Second case is left right rotation in which you first left rotate the parent and then right rotate grandparent. Finally colours of parent and grandparent are exchanged.
- RotateRC is used to rotate the right child. It considers the cases of right right rotation , in which first the grandparent is rotated to left and colours of parent and grandparent are exchanged. Second case is right left rotation in which you first right rotate the parent and then left rotate grandparent .Finally colours of parent and grandparent are exchanged.
- PrintRange method prints the building data for all the buildings whose building numbers lie between b1 and b2.
- RecurPrint method returns the building data(i.e building number, execution time and total time) to the printRange method whenever it finds something in the specified range.
- The delete method is used to delete a specific node from the red black tree.
- The deleteFixUp method is used to stabilise the tree(i.e ensure all the properties are being followed) once the deletion is done.
- The OUT method is called in the main method. It takes 2 parameters. The first parameter(can take up the values as 1 or 2), specifies if insert is performed or print is performed. The second parameter specifies the node for which the operation is being performed.