Mürüvvet Bozkurt
150160133 14.12.2019

1.INTRODUCTION:

In this homework, I understood Minheap algorithm. Minheap is sort algorithm. We can implement tree or array structure. I used array structure. I create 3 .cpp file and 1 .h file. The code is compiled and run in SSH terminal.

Minheap Algorithm:

1. Root elements is the smallest element on the array.
2. Parent must be smaller than their child.
3. When adding elements to the heap tree, it starts to be added from the left.
4. There is no connection between the right and left child.

2.CODE FILES:

2.1 *Heap.h file :*

This is header file. Event and Minheap classes are defined this file. Event class is used to create object for information of events. Event class contains constructor, mutator functions and "event_name", "time" and "what" that a bool variable that defines whether the time is the beginning or the end. If "what" is true, time is starting time. If it is false, time is ending time. There is an unnecessary extra variable called "index" that is used it for control when code is written. In order to compare objects within the "Event" class, the "operator" function is defined. Minheap class is used to create Minheap tree. Minheap functions that are defined in header file provide to implement objects to Minheap tree.

2.2 *Event.cpp :*

Event.cpp file contains functions of Event class. "heap.h" is included to this file .Constructor functions provide to create objects. "Operator= " is created for comparing objects each other and accessor, mutator functions are defined.

2.3 *Minheap.cpp:*

Minheap.cpp file contains functions of Minheap class. If the index of the minheap array is initialized from zero and the parent is assumed to be "i", index of left child will be 2*I +1, index of right child will be 2*i +2 and if parent is desired, subtract 1 from index of child and divide by 2. If you want to add new object Minheap structure. Heap size increase by 1 and new index of last element will be equal to new object. If the last element added is called heapify function to implement the minheap algorithm. Heapify function is called recursively until reaching structure of minheap algorithm. In heapifyUp function, if the time of the inserted object is greater than the time of parent, they are swapped each other. To delete the first element in the min heap tree: Firstly, fist element of minheap array is hold variable "root". Then, first element of the min heap array is equal to last element and heap size is decreased by "1" . Thus, first element of array is deleted but new structure may be not compatible with the minheap algorithm. Therefore, "heapifyDown(0)" function is called. In HeapifyDown function, index of right child is calculated for controlling left and right child. If index of right child is greater than heap size, it returns to the functions it was called. Then, same comparison is made for left child. If there is no left child, the right child is definitely not. If there is no right child and left child is, "min" variable is equal to index of left child. If there is right child, it means to have left and right child and index of right and left child are compared to each other. "min" is equal to the index of the smaller one. Value of min and index of parent are compared each other. If

“min” values is smaller than index of parent, they are swapping each other heapifyDown(min) function is called recursively until structure is compatible to Minheap algorithm.

2.4 *main.cpp:*

Object is created for start and end time as reading file. These objects are push to vector. Vector that is created sent to Minheap. The object is added to the minheap structure up to the size of the vector. Then, we need to delete elements from the heap tree in order to do the task requested from us. Firstly, first elemant of structure is deleted and it is hold “temp” variable. If T is not equal to time of temp, there is no event and it is printed on the screen. If there are same times in heap, an extra while loop is created so that they can be syncronized with T.