CS2810

ADVANCED PROGRAMMING LAB

REPORT FOR ASSIFNMENT 5

IMPLEMANTATION OF HEAPS

Team:

Madhavi Y	*CS09B025*
Vamsi krishna	*CS09B006*
Uday Chowhan	*CS09B034*

Files included:
1. heap_header.h
2. heap.cpp
3. rand.cpp

heap_header.h contains declarations of the functions used in of various functions used in the heap class.

heap.cpp has the definitions and the main function.

Description of various functions:
heap class has the arr[] as the private variable and the functions declared as public.

1. reader(char*) function reads a given array into the heap class.
2. rchild ---- gives the index of right child of a fiven index
3. lchild ---- gives the index of left child of a given index
4. parent --- gives the index of parent of a given index
5. size ------ gives the size of the heap
6. swap ----- swaps the elements

7. min_heapify :
- This function assumes that the children of the given index are proper heaps and it forms a heap including the given index

Pseudocode:
```
MIN_HEAPIFY(A,index)
        max=find max(rchild,lchild)
        m=find max(index,max)
        if m-max
                MIN_HEAPIFY(A,m)
```
time complexity : O(log n)

8. build_heap :
- min_heapify is done for all the non-leaves of the heap which gives a min_heap

Pseudocode:

```
BUILD_MIN_HEAP:
    for i=heap_size/2 to 2
        MIN_HEAPIFY(A,i)
```
Time complexity : O(n)

9. heap_decrease_key :
 - Decreases the value at an index to a given value which is less than the present value
 - Checks at an index if the heap is proper and it checks at the parent and goes on till it meets a parent which already follows the rule p < l , p < r

<u>*Pseudocode:*</u>
```
DECRESE_KEY(A,i,key)
    A[i]=key
    while i>1 and A[parent]>A[i]
        swap(i.,parent)
        i=parent(i);
```
Time complexity : O(log n)

10. min_heap_insert :
 - Inserts an element into the heap
 - The element is inserted at the last and decrease_key function is called

<u>*Pseudocode:*</u>
```
INSERT(A,key)
    size=size+1
    A[size]=large number
    DECREASE_KEY(A,size,key)
```
Time complexity : O(log n)

11. heap_minimum
 - Finds the minimum of all the elements that is the root

12. extraxt_min
 - Removes the least element from the heap and rearranges the heap so that the heap is a min_heap.
 - The first and last element are exchanged and last element is removed and min_heapify is called on the root

Time complexity : O(log n)

<u>*Pseudocode:*</u>
```
EXTRACT_MIN(A):
    min=A[1]
    A[1]=A[size]
    size=size-1
    MAX_HEAPIFY(A,1)
```

13. heap_copy :
- Copies the elements into an array

14. heap_sort :
- First and last element are interchanged
- Last element is deleted from the array and inserted into an other array
- min_heapify on ist element is called
- Thus the elements are sorted.

Pseudocode:
```
BUILD_MIN_HEAP(A)
        i=length to 2
        swap A[1] and A[last]
        size=size-1
        MIN_HEAPIFY(A,1)
Time complexity : O(nlogn)
```

15. heap_meld:
- An array is constructed with all the elements in file 1 and file 2
- build_heap is called on the newly constucted array

Pseudocode:
```
merge arr1 and arr2
build_heap()
Time complexity O(n1+n2)
```

rand.cpp:
This program takes the size required as the input from the user and produces random numbers in the files.
We used this program to check for various input sizes.
This is also being included in the uploaded zip file.
Following are the times taken for various input sizes with respect to meld , extract and insert operations.

Time taken to insert 50 keys into the heap for various values of n :

No of inputs	time taken in milli seconds
100	10
1000	370
5000	7700
10000	33970

Time taken to extraxt 50 keys from the heap for various values of n :

No of inputs	time taken in milli seconds
100	10
1000	900
5000	22200
10000	100570

Time taken to meld and sort two heaps:

input size 1 and 2	Time to meld(in ms)	Time to sort(in ms)
100	0	10
1000	100	460
5000	2520	14710
10000	9940	66330