CS 180 - More Trees + Heaps

Announcements

- HW due tomorrow
- Program 1 is graded
- Next program will be posted tomorrow
Trees: Traversals

How to display or check information stored in a tree?

Different ways depending on what is stored in it:

- In order
- Pre order
- Post order
Preorder \((T, v)\):

- Perform "visit" at \(v\)
- For each child \(w\) of \(v\):
  - Preorder \((T, w)\)
Postorder \((T, v)\):

for each child \(w\) of \(v\):
   postorder \((T, w)\)
   perform action at \(v\)

Print:

\(d, f, g, e, b, c, a\)
Example: Size of a directory

- Need to know size of children before we can compute current directory's size.

```
15K total
  10K homework
  3K examples
  2K CS 145
```

```
2K total
  1K CS 180
  1K examples
  15K Student
  15K CreditCard
```

names - each 15K

creditcard.h creditcard.
In order: only for binary trees

do left child
myself
do right child

\[((3 + 1) \times 3) / ((9 - 5) + 2)\]
Binary Search Trees

In order

3, 4, 5, 6, 11, 15, 20, 25, 30
Binary Trees
-each internal node has exactly 2 children

Complete: all leaves have same depth

Not complete
Representations of binary trees:

**Level numbering**

- If \( v \) is the root, \( p(v) = 1 \)
- If \( v \) is left child of \( u \), \( p(v) = 2p(u) \)
- If \( v \) is right child of \( u \), \( p(v) = 2p(u) + 1 \)

![Binary tree diagram](attachment:image.png)
Ex:

\[-1 + x + x^6 + 3\]

1 2 3 4 5 6 7 8 9
Priority Queue API (Ch. 7)

Keys versus values

Sort based on these

Data stored

Ex: Standby list for a flight

Values = names of people

Key = calculated based on freq. flyer, order of request, & price
A note about keys:

Properties: need to be able to compare them

- reflexive property: if $k_i \leq k_2$ and $c_2 \leq k_3$ \implies k_1 \leq k_3$

- transitive property: if $k_i \leq k_2$ and $k_2 \leq k_1$ \implies k_1 = k_2$

- antisymmetric: if $k_i \leq k_2$ and $k_2 \leq k_1$ \implies k_1 = k_2$
P. Q. ADT:

Methods:

- \textit{insertItem}(k, e)
- \textit{maxElement}(): returns the element with the smallest key
- \textit{removeMin}(): removes element with minimum key

Last time →
Code

```cpp
template <typename ItemType>
class Heap {

private:
    ItemType* _data;
    int _size;
    int _capacity;

public:
    Heap() : _data(new ItemType[1]), _size(0), _capacity(1) {
    }
```
```cpp
void insert(const ItemType & val)
{
    if (size == capacity)
        capacity = 2 * capacity;

    ItemType * newdata = new ItemType[capacity];
    for (int i = 0; i < size; i++)
        newdata[i] = data[i];
    delete data;
    data = newdata;

    data[size] = val;
    size++;
}
```
insert(50)

```
[20, 13, 18]
0 1 2
```

each parent is larger than its children

```
18
|
3
|
13
|
12
|
5
|
2
|
1
|
4
|
9
```

max 1

```
20
```

```
parent
```

```
current
```

```
parent current
```

```
```
int current = size - 1;
int parent = (current + 1) / 2;

while (data[current] > data[parent] && current > 0) {
    Item& temp = data[current];
    data[current] = data[parent];
    data[parent] = temp;
    current = parent;
    parent = (current - 1) / 2;
}
remove Max()