#15: Trees
October 2, 2017

Queue Iterator:

QueueIter.h
```
template <class QE>
class Queue {
  public:
    class QueueIterator :
    public std::iterator<std::bidirectional_iterator_tag, T> {
      public:
        QueueIterator(unsigned index);
        QueueIterator& operator++();
        bool operator==(const QueueIterator &other);
        bool operator!=(const QueueIterator &other);
        QE& operator*();
        QE* operator->();
      private:
        int location_;

    };


  /* ... */

  private:
    QE* arr_; unsigned capacity_, count_, entry_, exit_;
};
```

Does an instance of a **`QueueIterator`** have access to the **`Queue arr_`**?

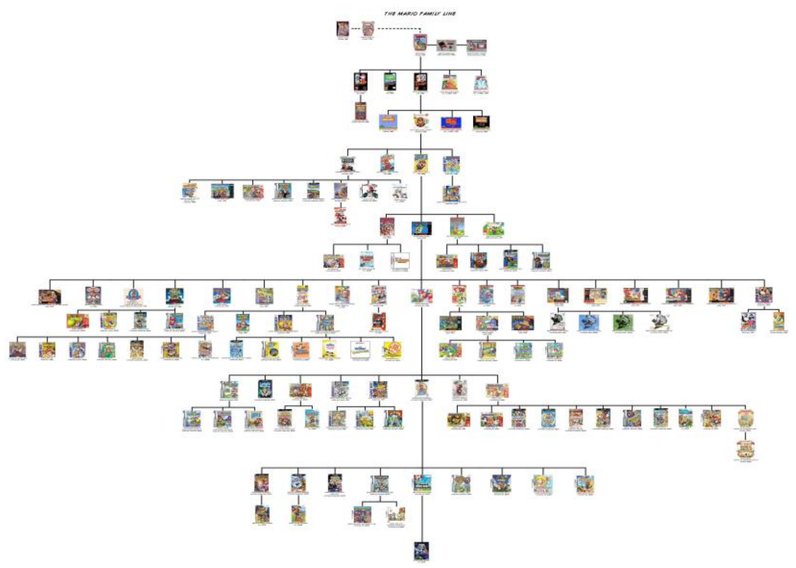
Two big takeaways:

1.

2.

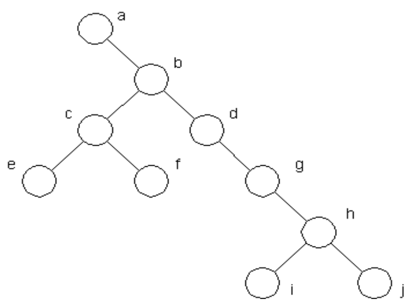
Trees!
"The most important non-linear data structure in computer science."
- David Knuth, The Art of Programming, Vol. 1

A tree is:



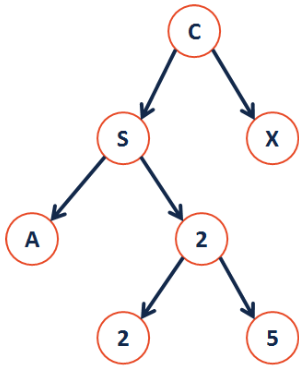
We are going to start with a specific type of tree:

- What's the longest "word" you can make using the **vertex** labels in the tree (repeats allowed)?
- Find an **edge** that is not on the longest **path** in the tree. Give that edge a reasonable name.
- One of the vertices is called the **root** of the tree. Which one?
- Make a "word" containing the names of the vertices that have a **parent** but no **sibling**.
- How many parents does each vertex have?
- Which vertex has the fewest **children**?
- Which vertex has the most **ancestors**?
- Which vertex has the most **descendants**?
- List all the vertices is b's left **subtree**.
- List all the **leaves** in the tree.

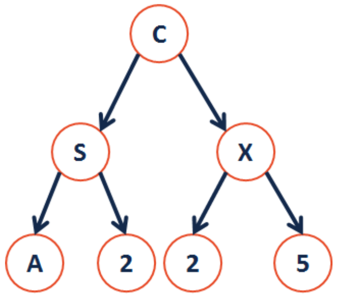


Definition: Binary Tree

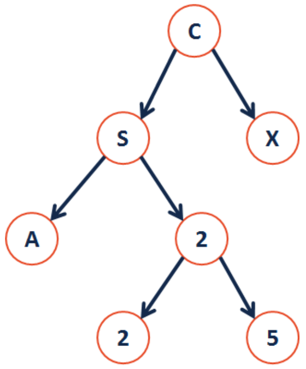
A *binary tree* **T** is either:



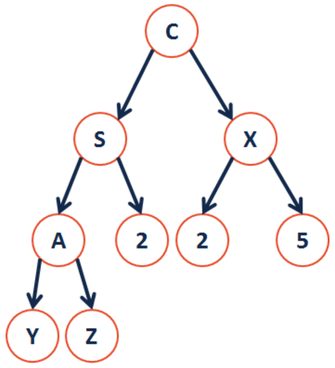
Tree Property: Perfect



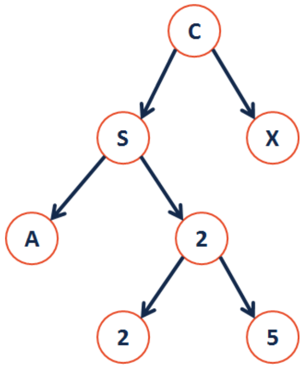
Tree Property: Tree Height



Tree Property: Complete



Tree Property: Full



CS 225 – Things To Be Doing:

1. Exam #4 currently ongoing ("Programming Exam", MP2)
2. MP3 is starting Week 2; up to +7 for submitting by Oct. 2 (11:59pm)
3. lab_trees released on Wednesday
4. Daily POTDs