Announcements

MP4 available, due 10/16, 11:59p. EC due 10/9, 11:59p.

```
#include <list>
#include <iostream>
#include <string>
using namespace std;

struct animal {
   string name;
   string food;
   bool big;
   animal(string n="blob", string f="you”, bool b=true):name(n),food(f),big(b) {}
};

int main() {

   animal g("giraffe","leaves"), p("penguin","fish”,false), b("bear");
   list<animal> zoo;
   zoo.push_back(g); zoo.push_back(p); zoo.push_back(b); //STL list insertAtEnd

   for(list<animal>::iterator it = zoo.begin(); it != zoo.end(); it++)
      cout << (*it).name << "  " << (*it).food << endl;

return 0;
}
```

Generic programming: (more magic)

```
template<class Iter, class Formatter>
void print(Iter first, Iter second, Formatter printer) {
   while (!(first==second)) {
      printer(*first);
      first++;
   }
}
```

Write a short description of this function:

This is a function called _____, whose inputs are two _____ and a _____. The function appears to _____

_____.

What is printer?

Generic programming: (more magic)

```
template<class Iter, class Formatter>
void print(Iter first, Iter second, Formatter printer) {
   while (!(first==second)) {
      printer(*first);
      first++;
```

```
class printIfBig {
public:
   void operator()(animal a) {
      if (a.big) cout << a.name << endl;
   }
};
```

1. Declare an object of type `animal`:
2. Declare an object of type `printIfBig`:
3. Using your answers for 1 and 2, invoke a member function of the `printIfBig` class:

Generic programming: (more magic)

```
#include <list>
#include <ios
#include <sti
using namespa

struct anima
   string nam
   string foo
   bool big;
   animal(sti
};

int main() {

   animal g("
   list<anima

   zoo.push_

   for(list<a
      cout <<
```

```
template<class Iter, class Formatter>
void print(Iter first, Iter second, Formatter printer) {
   while (!(first==second)) {
      printer(*first);
      first++;
   }
}
```

```
class printIfBig {
public:
   void operator()(animal a) {
      if (a.big) cout << a.name << endl;
   }
};
```

```
printIfBig myFun;
print<list<animal>::iterator,printIfBig>(zoo.begin(),zoo.end(),myFun);
```

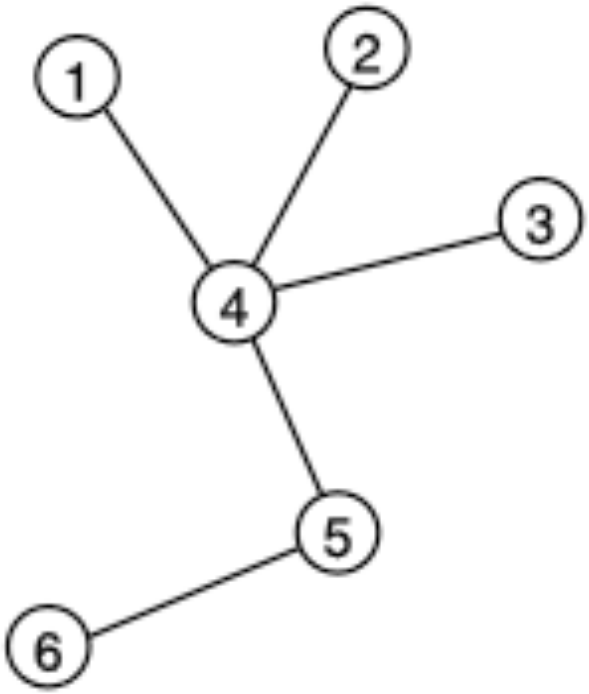
Trees:

"… most important nonlinear structure in computer science."

-- Donald Knuth, *Art of Computer Programming Vol 1*

A tree: _____

We'll study more specific trees:

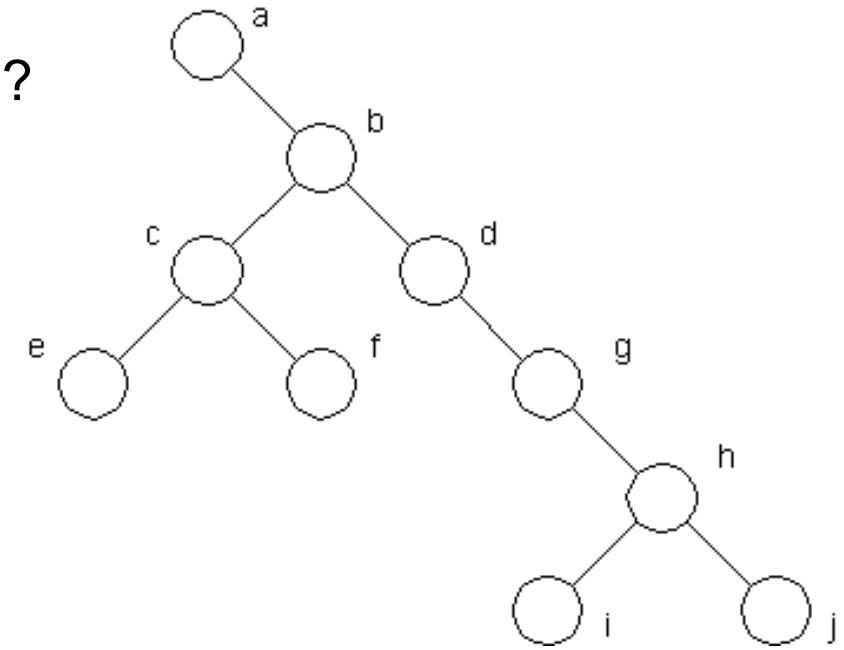


Tree terminology:

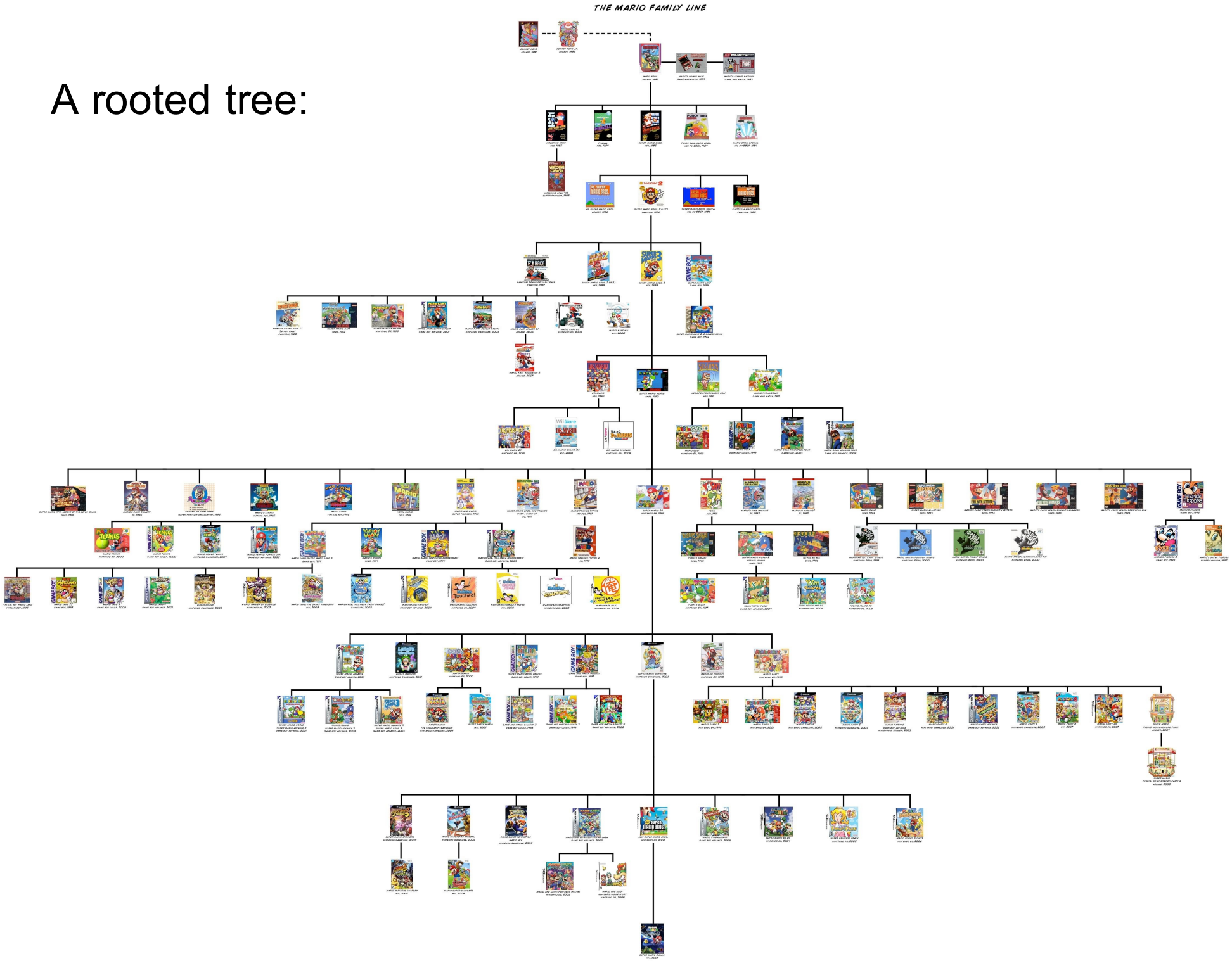
- What's the longest English word you can make using the vertex labels in the tree (repeats allowed)?
- Find an edge that is not on the longest path in the tree. Give that edge a reasonable name.

For the rest of the exercises, assume the tree is rooted.

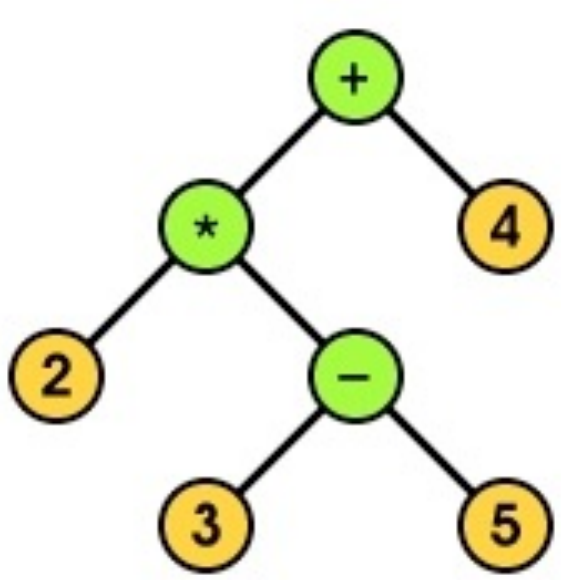
- One of the vertices is called the "root" of the tree. Guess which one it is.
- Make an English word containing the names of the vertices that have a parent but no sibling.
- How many parents does each vertex have?
- Which vertex has the fewest children?
- Which vertex has the most ancestors?
- Which vertex has the most descendants?
- List all the vertices is b's left subtree.
- List all the leaves in the tree.



A rooted tree:



Binary tree, recursive definition:



A *binary tree* T is either

-

OR

-