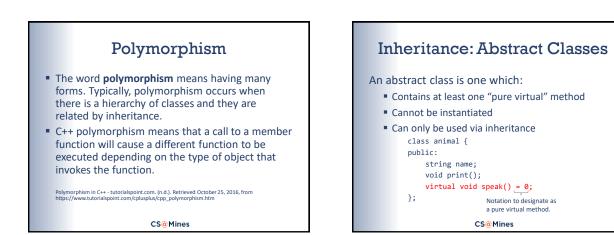
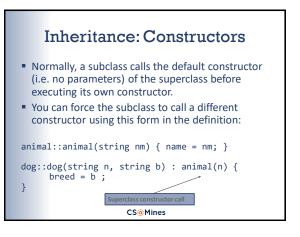


## Inheritance: Polymorphism Inheritance: Polymorphism Now using pointers, same output: Let's fix this: animal\* A[2]; class animal { A[0] = &c; A[0] = &c; for (int j = 0; j < 2; j++) A[j]->print(); public: string name; virtual void print(); Output is: My name is Fluffy. My name is Rex. I am a Dachshund. }: print\_animal(c); print\_animal(d); Note, how this is different: animal a = d; // default copy constructor called - now just an animal! a.print(); Output is: My name is Fluffy. My name is Rex. Output is: name is Rex. I am a Dachshund. CS@Mines CS@Mines



## CS⊛Mines



class	Final Example
public	
	string name;
	<pre>virtual void print(); virtual void speak() = 0;</pre>
};	the case vote speak() = 0;
class of public	dog : public animal { :
	string breed;
	<pre>void print(); void speak() { cout &lt;&lt; "Woof!"; }</pre>
};	
	cat : public animal {
public	<pre>void speak() { cout &lt;&lt; "Meow."; }</pre>
};	tora speak() ( core (c ficont ) )

