Const methods

- Version 1: Dr. Ofir Pele
- Version 2: Dr. Miri Ben-Nissan
- Version 3: Dr. Erel Segal-Halevi

Reminder: const variables (in C and C++)

```
// const pointer to un-const variable
int * const p1 = &i;
```

- `p1++; // compile error`
- `(*p1)++; // ok`

```
// un-const pointer to const variable
const int * p2 = &b;
```

- `p2++; // ok`
- `(*p2)++; // compile error`

```
// const pointer to a const variable
const int * const p3 = &b;
```

Const methods

```
class A
{ int a;
public:
   void foo1() const;
   void foo2();
};
void A::foo1() const {
 // foo1(const A* const this)
 a=5; // = this→a = 5 = error
 cout << a; // OK
}
void A::foo2() {
 // foo2(A* const this)
 a=5;  // OK
}
```

```
int main() {
 A a;
 const A ca;
 a.foo1(); //=foo1(&a)
 a.foo2(); //=foo2(&a)
 ca.foo1();//=foo1(&ca)
 ca.foo2();//=foo2(&ca)
- compilation error!
}
```

Const methods

```
class A
{
public:
    void foo() const;
    void foo();
};
const int A::foo() const
{
    cout << "const foo\n";
}
int& A::foo()
{
    cout << "foo\n";
}
```

```
int main()
{
    A a;
    const A ca;
    a.foo () = 5;
    ca.foo();
}
```

```
// output
foo
const foo
```

How can we have two "foo" functions?

– Overload resolution:

```
A::foo(A* const this)


A::foo(const A* const this)
```

Why do we need two "foo" functions?

See folder 3.

mutable

- `mutable` means that a variable can be changed by a const function (even if the object is const)
- QUESTION: When would you use this?

mutable: example #1

```
class X
{
public:
  ...
  X() : _fooAccessCount(0) {}

  bool foo() const
   {
      ++_fooAccessCount;
      ...
   }
   unsigned int fooAccessCount() { return _fooAccessCount; }

private:
   mutable unsigned int _fooAccessCount;
};
```

mutable: example #2

```
class Shape
{
public:
  ...
  void set...(...) { _areaNeedUpdate= true; ... }
  double area() const
   {
      if (_areaNeedUpdate) {
         _area = ...
         _areaNeedUpdate= false;
      }
      return _area;
   }
private:
   mutable bool _areaNeedUpdate= true;
   mutable double _area;
};
```