

# C++: 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 print() const;
    void set();
};

void A::print() const {
    // print(const A* const this)
    a=5; // = this->a = 5 = error
    cout << a; // OK
}

void A::set() {
    // set(A* const this)
    this->a=5; // OK
}
```

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

# Const methods

```
class A
{ int a;
public:
    void foo() const;
    void foo();
};

const int A::foo() const
{
    cout << "const foo\n";
    return a;
}

int& A::foo()
{
    cout << "foo\n";
    return a;
}
```

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