

Member Functions

- Are functions that are members of a class.
- Can be defined inside or outside the class
- Are automatically `inline` if defined inside the class.

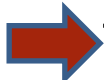
```
struct Account{  
    double getBalance() const;  
    void withdraw(double amt);  
    void deposit(double amt) { balance += amt; }  
private:  
    double balance;  
};
```

```
double Account::getBalance() const { return balance; }  
inline void Account::withdraw(double amt){ balance -= amt; }
```

static Member Functions

- `static`
 - `static` member functions can be used with or without an instance.
 - By using the scope resolution operator (`::`), you can access the attributes of the class

```
class Account{
public:
    static int getDeposits() const{ return deposits; }
private:
    static int deposits;
};
....
int total = Account::getDeposits();
```

`static` member functions have no `this` pointer.  They can only access static attributes and member functions.

`methodStatic.cpp`

const Member Functions

- `const`
 - Constant member functions can not change an instance variable if the instance variable is not declared `mutable`.
 - Constant objects can only call `const` or `constexpr` member functions.

```
class Account{  
public:  
    double getBalance() const { return balance; }  
private:  
    double balance;  
};
```

```
const Account acc;  
double account = acc.getBalance();
```

`methodConst.cpp`

constexpr Member Functions

- `constexpr`
 - Can run at compile time.
 - Must run at run time when given non-`constexpr` arguments.
 - Are implicit `const`. (C++11)
 - Have the potential to run at compile time.
 - Can only use `constexpr` functions, member functions, or global variables.

The `this` Pointer

- `this` pointer
 - All non-static members have an implicit `this` pointer.
 - Can be used explicitly
 - Refers to the instance of the class
 - Is necessary to access the hidden attributes of the class.

```
struct Base{  
    int a{1998};  
    void newA(){  
        int a{2011};  
        std::cout << this->a;    // 1998  
        std::cout << a;        // 2011  
    }  
};
```

`methodThisPointer.cpp`