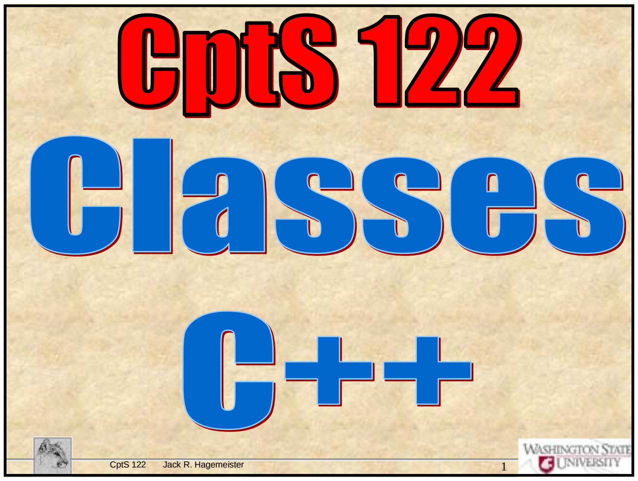
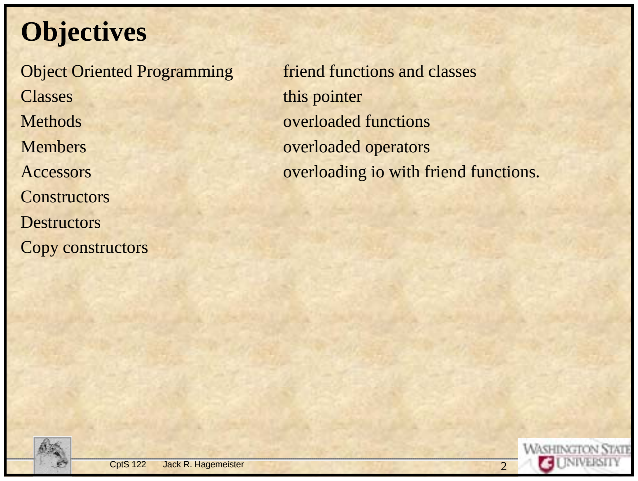


# CptS 122

# Classes

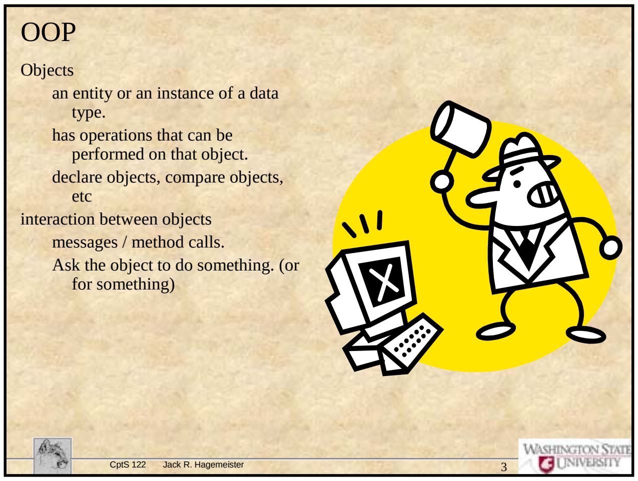
# C++



## Objectives

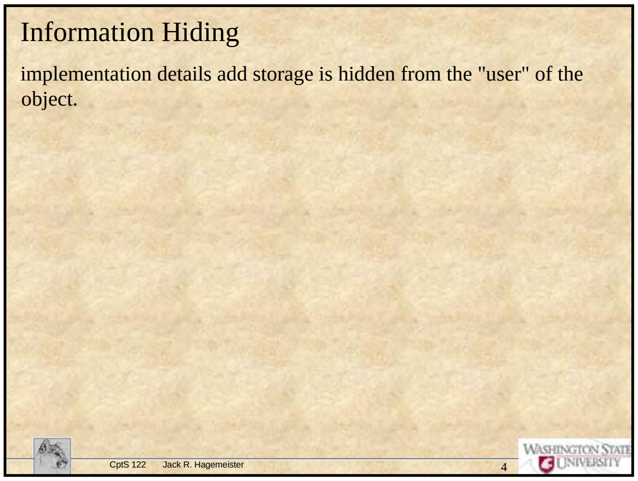
Object Oriented Programming
Classes
Methods
Members
Accessors
Constructors
Destructors
Copy constructors

friend functions and classes
this pointer
overloaded functions
overloaded operators
overloading io with friend functions.



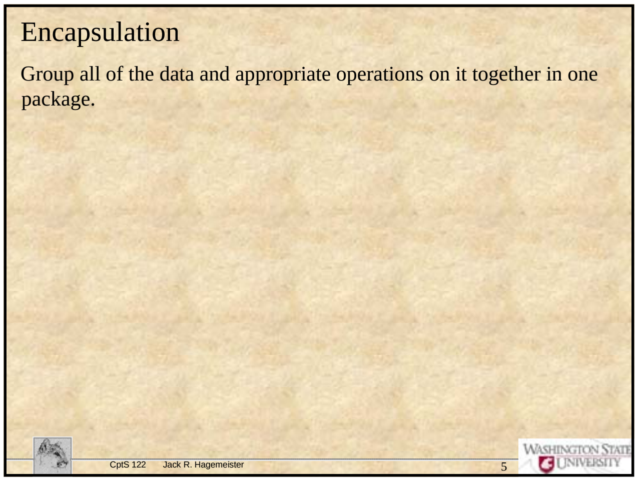
## OOP

- Objects
  - an entity or an instance of a data type.
  - has operations that can be performed on that object.
  - declare objects, compare objects, etc
- interaction between objects
  - messages / method calls.
  - Ask the object to do something. (or for something)



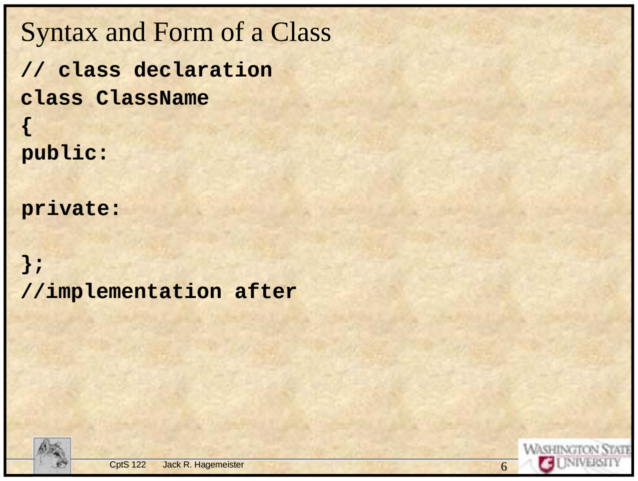
## Information Hiding

implementation details add storage is hidden from the "user" of the object.



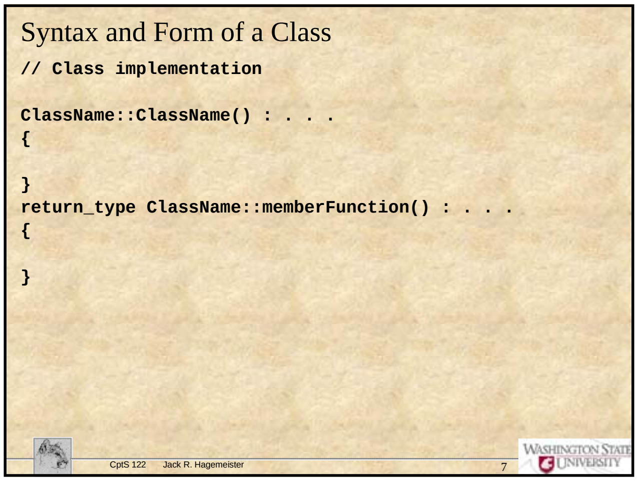
## Encapsulation

Group all of the data and appropriate operations on it together in one package.



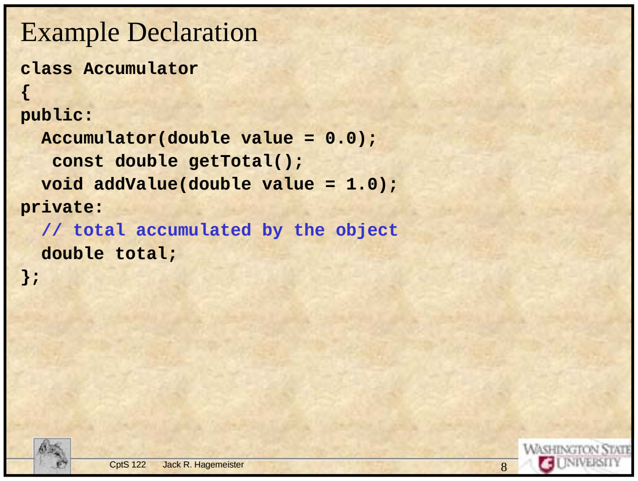
## Syntax and Form of a Class

```
// class declaration
class ClassName
{
public:

private:

};
//implementation after
```



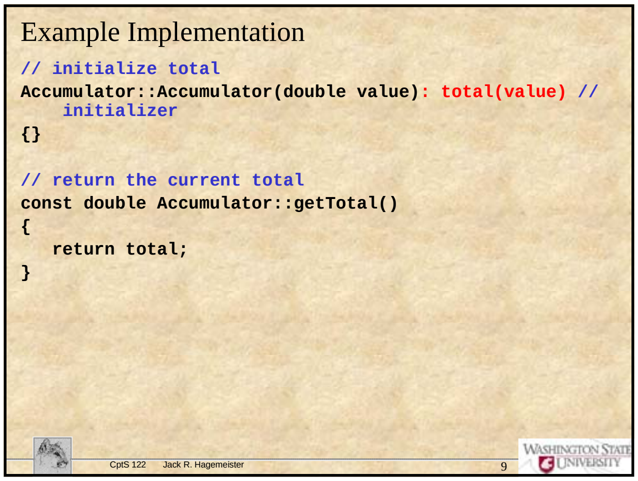
## Syntax and Form of a Class

```
// Class implementation

ClassName::ClassName() : . . .
{

}
return_type ClassName::memberFunction() : . . .
{

}
```



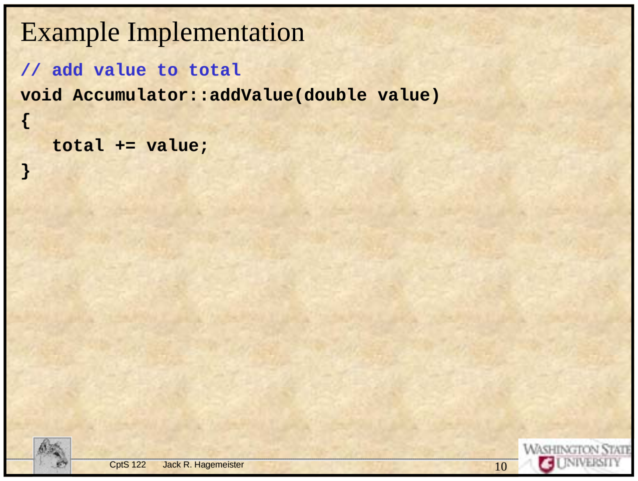
## Example Declaration

```
class Accumulator
{
public:
  Accumulator(double value = 0.0);
   const double getTotal();
  void addValue(double value = 1.0);
private:
  // total accumulated by the object
  double total;
};
```



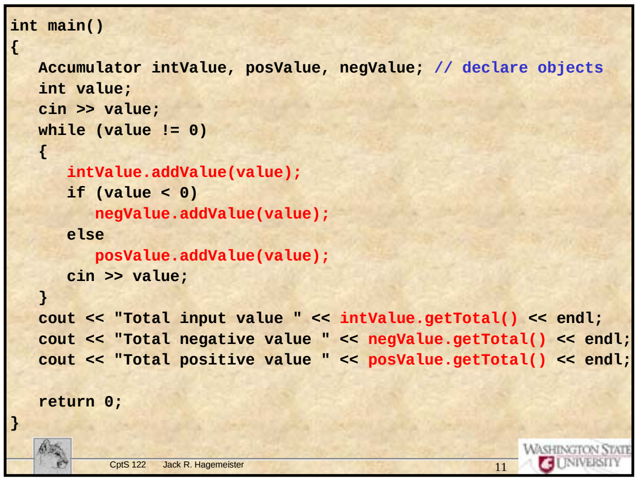
## Example Implementation

```
// initialize total
Accumulator::Accumulator(double value): total(value) //
    initializer
{}

// return the current total
const double Accumulator::getTotal()
{
   return total;
}
```

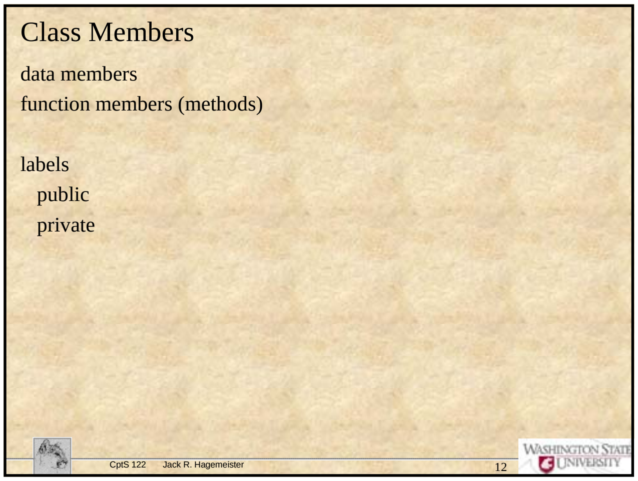


## Example Implementation

```
// add value to total
void Accumulator::addValue(double value)
{
   total += value;
}
```



```
int main()
{
   Accumulator intValue, posValue, negValue; // declare objects
   int value;
   cin >> value;
   while (value != 0)
   {
      intValue.addValue(value);
      if (value < 0)
         negValue.addValue(value);
      else
         posValue.addValue(value);
      cin >> value;
   }
   cout << "Total input value " << intValue.getTotal() << endl;
   cout << "Total negative value " << negValue.getTotal() << endl;
   cout << "Total positive value " << posValue.getTotal() << endl;

   return 0;
}
```



## Class Members

data members

function members (methods)

labels

- public
- private