Data types
Single pieces of information
• one integer – `int`
• one symbol – `char`
• one truth value – `bool`
Multiple pieces of information
• group of symbols – `string`
• group of anything – `array`
• group of multiple things – `struct, class`

Introducing: classes
• A `class` defines a new data type
• Each instance of a class is an `object`
• Each object can contain
  – Actions to perform (functions)
  – Information about the object (variables)

Class dog
Example information
• Size
• Weight
• Location
Example actions
• Eat
• Walk
• Bark

Class syntax – declaration
```cpp
class Dog
{
public:
    void Bark();
    void Eat(float foodQuantity);
    void Walk(float distance);
    float size, weight, location;
};
```

Class syntax – function definitions
```cpp
void Dog::Bark()
{
    cout << "Woof woof!\n";
}

void Dog::Eat(float foodQuantity)
{
    weight+=foodQuantity/2;
    size+=foodQuantity/10;
}
Class syntax – create and use an object

```cpp
int main()
{
    Dog fido;
    fido.weight=40.5;
    fido.size=10;
    fido.Eat(20);
    cout << fido.weight << " "
         << fido.size << endl;
    return 0;
}
```

Class syntax – declaration

```cpp
class Dog
{
public:
    float size, weight, location;
    void Bark();
    void Eat(float foodQuantity);
    void Walk(float distance);
};
```

Typical program layout

```cpp
class Dog {
    . . .
};

int main() {
    Dog fluffy;
    fluffy.Bark();
    . . .
}

void Dog::Bark() {
    . . .
}
```

The Dot Operator .
- Used for functions and data of individual objects
  - fido.Bark()

The Scope Resolution Operator ::
- Used for functions of a class
  - Dog::Bark()

Note: a function inside a class is called a "member function"

Multiple instances of a class

```cpp
int main()
{
    Dog fido, spot;
    fido.weight=40.5; fido.size=10;
    spot.weight=30; spot.size=7.5;
    fido.Eat(20); fido.Eat(2);
    cout << fido.weight << " "
         << spot.weight << endl;
    return 0;
}
```

Time to walk the dog...

```cpp
int main()
{
    Dog rufus;
    rufus.weight=35; rufus.size=7.2;
    rufus.location=5;
    rufus.Walk(3.4);
    cout << "New location for Rufus: "
         << rufus.location << endl;
    // Will output location 8.4
    return 0;
}
```
Class syntax – create and use an object

```cpp
int main()
{
    Dog fido;
    fido.weight=40.5;
    fido.Bark();
    ?? // Set fido's location to 3
    cout << ??; // output location
    ?? // Have fido change location
    return 0;
}
```

**public vs. private**

- **public**: any function can see and use
- **private**: only visible to member functions
- Good style:
  - make all member variables private
  - use public functions to access and mutate variables

Class declaration, take 2

```cpp
class Cat
{
    public:
        void set(float inWeight, float inSize, float inLoc);
        float getSize();
        float getWeight();
        float getLocation();
        . . .
        void Walk(float distance);
    private:
        float size, weight, location;
};
```

Function definitions – take 2

```cpp
void Cat::set(float inWeight, float inSize, float inLoc)
{
    weight=inWeight;
    size=inSize;
    location=inLoc;
}
```

Class usage – take 2

```cpp
int main()
{
    Cat feline1;
    feline1.set(5.5,20.1,2);
    cout << feline1.location; // Error
    cout << feline1.getLocation() << endl;
    return 0;
}
```

Bank account

**Variables**
- Name
- Current balance
- History of cash in (and out)

**Functions**
- Deposit:
  - Add entry to history
  - Update balance
class Account {
public:
    void open(string inName);
    void deposit(float money);
    float getBalance();

private:
    string name;
    float balance;
};

void Account::open(string inName){
    name=inName;
    balance=0;
}

void Account::deposit(float money) {
    if(money>=0) {
        balance = balance+money;
    } else {
        cout << "Error! " << "Negative deposit!\n";
    }
}

What does this do?

int main()
{
    Account acc1;
    acc1.open("Tina");
    cout << acc1.getBalance() << endl;
    acc1.deposit(250);
    acc1.deposit(20.25);
    cout << acc1.getBalance() << endl;
}

Withdrawal function?

• How can we write withdraw function to reduce the money in our account?
• How can we prevent over-drawing?

Account review

Member variables
• name
• balance
• history,
  num_transactions

Member functions
• open
• deposit
• getBalance
• withdraw
• printHistory

New accessor function
• string getName()