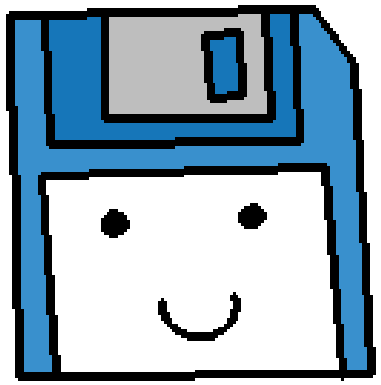
Structs

Written by Rachel J. Morris, last updated 2015-02-05

Beginning the Object Oriented Programming Adventure...



Structures

So far in our programs, we've been utilizing **variables** and **functions**.

Now that we're getting into Object Oriented Programming, we're going to begin making our own **data types**.

To do this, we define **structs** or **classes** (more on classes later).

Both of these can contain their own variables and functions internally.

Structures

Object Oriented Programming is one of the strengths of C++ (and similar languages),

It helps us create models of data, grouping like things together.

By grouping information together, it also becomes easier to read, more manageable to maintain, and more capable of extending for future features.

Example Structures

Some example **objects**:

Game Character

Text File

Student

Button

Account

Image

Instant Message

Shopping Cart

Example Structures

Some example **objects** and their **variables** and **functionality**:

Game Character

Variables
- X and Y coordinates
- Bitmap image
- Experience Points
- Maximum Speed

Functions
- Move(direction)
- TakeDamage(amount)
- Animate()

Text File

Variables
- Filename
- Last update date
- Text contents

Functions
- ReadLine()
- AddLine(text)
- Save(filename)
- Load(filename)

Example Structures

Some example **objects** and their **variables** and **functionality**:

Shopping Cart

Variables
- List of Purchasable Items
- User Account ID
- Seconds until timeout

Functions
- CalculateTotal()
- SendOrder()

Purchasable Item

Variables
- Name
- Manufacturer
- Price

Functions
- SubmitReview(info)
- GetLatestReviews()
- AddToCart()

Structures

With structures, we will group together small amounts of information, like...

CoordinatePair
X, Y

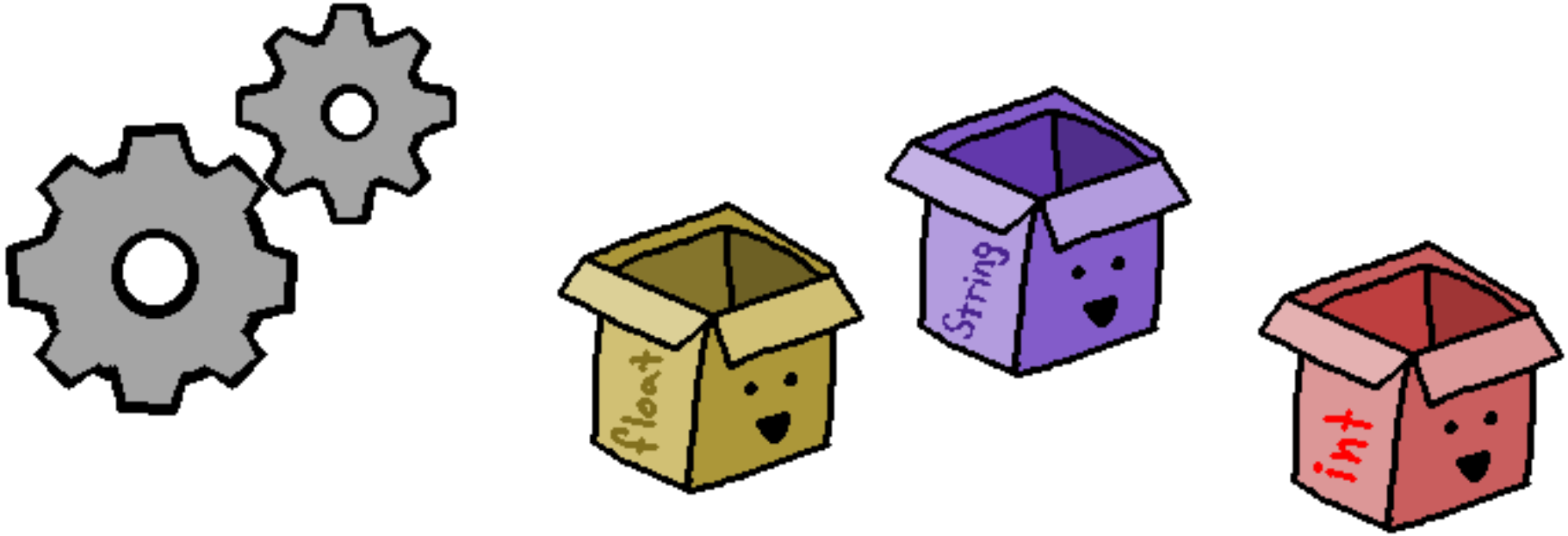
Rectangle
X, Y,
Width, Height

Fraction
Numerator,
Denominator,
Add(...)
Multiply(...)

Structures

Classes will end up being better for larger, more sophisticated objects.

Both classes and structures can store variables and functions.



Defining a Struct

We will create our **struct** objects outside of any functions (including main).

It begins with the keyword **struct**, followed by the object's name.

The members of the struct are declared/defined within curly braces.

The end of the struct definition must have a semi-colon after the closing brace.

```
struct Rectangle
{
    int x, y, width, height;
    void GetX1()
    {
        return x;
    }
    void GetX2()
    {
        return x+width;
    }
};
```

Defining a Struct

After a struct has been created, we can create a data type of that object in our program.

```
int main()
{
    Rectangle box;
    box.x = 5;
    box.y = 10;
    box.width = 2;
    box.height = 5;

    return 0;
}
```

Here, we declare a variable of type **Rectangle**, and name it **box.**

```
struct Rectangle
{
    int x, y, width, height;
    void GetX1()
    {
        return x;
    }
    void GetX2()
    {
        return x+width;
    }
};
```

Defining a Struct

After a struct has been created, we can create a data type of that object in our program.

```
int main()
{
    Rectangle box;
    box.x = 5;
    box.y = 10;
    box.width = 2;
    box.height = 5;

    return 0;
}
```

```
struct Rectangle
{
    int x, y, width, height;
    void GetX1()
    {
        return x;
    }
    void GetX2()
    {
        return x+width;
    }
};
```

To access internal members (variables, functions) of our custom object, we need to use the dot-operator .

Passing by Reference

Because a custom object can contain many variables, arrays, or generally take up a lot of memory...

```
bool IsSame( const Rectangle& box1, const Rectangle& box2 );
```

…you should make sure to get into the habit of passing any objects by **const reference** into other functions.

Unless you actually need to change that object, then just pass it as a **reference**.

```
void CreateBox( Rectangle& box );
```

Sample Programs...

Let's write some code to demonstrate how to use structs.

A. Coordinate Pairs

B. Rectangles

C. Students and Classes

Sample Programs...

Let's write some code to demonstrate how to use structs.

A. Coordinate Pairs

B. Rectangles

C. Students and Classes

Sample Programs...

Let's write some code to demonstrate how to use structs.

A. Coordinate Pairs

B. Rectangles

C. Students and Classes

Sample Programs...

Let's write some code to demonstrate how to use structs.

A. Coordinate Pairs

B. Rectangles

C. Students and Classes