Announcements

- Math/CS Club meets at 4 in Ritter Lobby

CS2100 - Variable Types
Classes

Scope

Last Time
What is inheritance?

Creating a "child" object from parent class

Ex: FillableShape
    Circle
    Rectangle
    Square
Example: Square Class

```java
class Square : public Rectangle

    // Constructor
    Square(double side): Rectangle(side, side, Point(0, 0)) {
        // Call parent constructor
        Rectangle(width, size, parent.center);
    }

    // Overload Rectangle's constructor
    Square(double w, double h) :
        Rectangle(w, h, Point(0, 0)) {
            width = w;
            height = h;
        }

    // Overload get width function
    double get_width() const {
        return size;
    }

    // Call parent function
    void set_size(double size) {
        // Make sure to invoke parent version
        Rectangle::set_size(size);
    }
```

end of square
Other issues

 ClassNotFoundException

protected:

Class Whatever

What about define that View Can't have

A View type of Data. So far, have

Cannot use or use children can use this
In Python to access data, we can use variables.

Given variables:

- x = 5
- y = 38
- y = 3.2
- Point = (3, 4)
- a = 6
- b = Point[0] + (3, 4)
3. Plan for
2. Reference
1. Value

Variables. C++ allows for 3 different types of variables. More versatile
More efficient (for both speed & space).

<table>
<thead>
<tr>
<th>11</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

Point a (5,7).

Point b (5,7).

Amount of memory is equal to a place.

When a variable is created a place

Value Variables.
By default = deep copies.

They stay separate.

Now set $a = b$.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Name: 

address: 

1287

6986054

678
No more deep copy.

So do changes to the point last.

is run at the beginning of the function.

Essentially, the line:

a new, separate variable

the value of pt is isolated as

When someone calls:

functions: passing by value

{
    return pt->x() == 0 || pt->y() == 0;
}

bool isOrigin(point pt) {  }
Choose value at origin or point B will not make a C point.

Ex: \( c = 0 \).

is always the same as a

- more like Python, but a
- C is created as an

Synt: \( \text{point } 8 \text{ (c)} \).

Reference Variables

\( \Box \)
Ex:

1. int a;
2. a = 35;
3. int & b(a);
4. int c(7);
5. b = 63;
6. c = 11;
7. a = 50;
8. b = c;
In mean:

```
if (isOrigin(mypt))
```

For mypt:

```cpp
return pr.getx() == 0 && pr.gety() == 0;
```

bool isOrigin(Point pt)

F.x:

Instead, primary purpose is in functions:

Use in mean:

Reference variables aren't generally

Passing by reference
Why pass by reference?

3 main reasons

- Changes persist outside of function
- Saves time — no need to make a copy of (potentially) huge input
- Saves space!
Changed inside the function.

Compiler will enforce that pt isn’t

```cpp
{ return p->getX() == 0 || p->getY() == 0;
}
```

If we want to use constant variables, we don't want changes to vars to affect them by reference. But we don't want to pass a const.
Note: don't use const. Why?

Here, & is required since & means

```cpp
{ 
  return out;
  out >> p.display();
  out >> p.getVisibility();
  ostream &operator>>(ostream &out, Point &p);
  Point operator=(const Point &p);
}
```
Can't write d = 6;
But d is not an int.
\( \text{R}^* \)
\( d = 8.6 \)
\( \Rightarrow \text{NULL memory address} \)
\( \Rightarrow \text{Null memory} \)
\( \text{in } \text{int } d; \)
\( \text{Syntax: int } \ast \text{d}; \)
\( \text{Pointer variables} \)
get y(c) \\
\text{nexter object} \\
\text{or} \\
\text{use for int} \\
\text{get X(c)} \\
500 \\
2 \text{ options:} \\
2 \\
\text{F: Point } \text{ pt } p \\
\text{G: Point } \text{ pt } d \\
\text{called dereferencing} \\
\text{Pointers: getting to the data}
inside multiple functions, so can create or modify.

The pointer is gone, persists even after main. Use: The data is 12 is
3 if is destroyed - for 12 is

\( C = \text{new } \text{int}[12] \)
\( \text{int } * C \)
\( \text{The new command} \)
allows passing a NULL pointer, but

Similar to passing by reference.

```c
bool isNull(Point *pt)
{
    return pt->next == 0;
}
```
What do we need when we are

- an array.
  
- Example: array

- The private declaration of private variables to put in
  
  - Often, we don't know all the details.
  
  - Pointers are especially useful in classes.

Pointers in a class
Example Class: Vector of Floats

A vector in 2D: \( \langle 2, 5 \rangle \)

A vector in 3D: \( \langle 90, 0, 1 \rangle \)

Dynamic Size: So how to make a Class?

Pointer: int *Size;

Float *of // pointer to an array
class MyFloatVec {

private:
    int size;  // dim of space
    float* a;  // ptr to array holding #s

public:
    MyFloatVec (int s = 10) : size(s) {
        a = new float[size];
    }
};
3 return a [index].

fout operator int (int index)

bool overide the [z] so that x[i] will give
back the kth element in the vector.

Ex: overcome the [z] so that x[i] will give

the vector (so no = or ->);

variable isn't a pointer.

With an array, you just pretend the

Accessing the array:
Function to scale by int\(\text{in}\) class.