CSE 333 – SECTION 6

References, const and classes
HW3

• Due tonight by 11:59 pm.
• Revisit hw2 and ex04
• Questions?
This or that?

- Consider the following code:

**Pointers:**

```c
int i;
int *pi = &i;
```

**References:**

```c
int i;
int &ri = i;
```

In both cases,

The difference lies in how they are used in expressions:

```c
*pi = 4;
ri = 4;
```
Part 1
```c
int i = 0, j = 4;
int *pi = &i;
```

Part 2
```c
int &ri = i;
```

Part 3
```c
*pi = 3;
```

Part 4
```c
ri = j;
```
Pointers and References

• Once a reference is created, it cannot be later made to reference another object.
  • Compare to pointers, which are often reassigned.

• References cannot be *null*, whereas pointers can.

• References can never be uninitialized. It is also impossible to reinitialize a reference.

• Demo: experiments.cc
C++ const declaration

- As a declaration specifier, const is a type specifier that makes objects unmodifiable.
  ```
  const int m = 255;
  ```

- Reference to constant integer:
  ```
  int n = 100;
  const int &ri = n; // ri becomes read only
  ```

- Demo: const.cc
When to use?

- Function parameter types and return types and functions that declare overloaded operators.
- **Pointers**: may point to many different objects during its lifetime. Pointer arithmetic (++ or --) enables moving from one address to another. (Arrays, for e.g.)
- **References**: can refer to only one object during its lifetime.

**Style Guide Tip:**
- use const reference parameters to pass input
- use pointers to pass output parameters
- input parameters first, then output parameters last
#ifndef __POINT_H
#define __POINT_H

class Point {
public:
    Point(const int x, const int y);
    int get_x() const { return x_; }
    int get_y() const { return y_; }
    double distance(const Point &p) const;
    void setLocation(const int x, const int y);

private:
    int x_;  
    int y_;  
};

#endif
Section Exercise

- Define a class Rectangle whose instance variables are a pair of Point objects (upper left, lower right).
- Include at least one constructor. Make sure you get const right in the right places.
- Methods:
  - `getul()`, `getlr()` - returns upper and lower points.
  - `cornerPoints()` – to obtain the corner points.
  - `area()` - returns the Rectangle's area.
  - `contains(Point &p)` - returns true or false depending on whether point p is inside the rectangle.
- The C++ Primer text and cplusplus.com contain good reference material.