Lecture 21

Chapter 7
Topics

• Constructors
• Default constructors
• Constructor calls
• Overloaded constructors
• Const parameter modifier
• Const with member functions
• Inline functions
Constructor

• It is a type of function that is called when a new variable of this type is declared
• A constructor is like any other member function you define
  – But
• It must have the same name as a class
• It cannot return a value
class Rect
{
public:
    //Constructor prototype
    Rect(float side1Value, float side2Value);
    float getArea();
private:
    float side1;
    float side2;
};

int main()
{}
Rect::Rect(float side1Value, float side2Value) //Constructor definition
{
    side1 = side1Value;
    side2 = side2Value;
}
Constructor

- This is equivalent to the previous page

class Rect
{
  public:
    //Constructor prototype
    Rect(float side1Value, float side2Value);
    float getArea();

  private:
    float side1;
    float side2;
};

int main()
{
//Constructor definition
Rect::Rect(float side1Value, float side2Value) : side1(side1Value), side2(side2Value)
{
}
}
Default Constructor

- This is equivalent to the previous page

```cpp
class Rect {
public:
    // Constructor prototype
    Rect(float side1Value, float side2Value);
    Rect(); // Default Constructor
    float getArea();

private:
    float side1;
    float side2;
};

int main() {}
```

```
// Constructor definition
// Rect::Rect(float side1Value, float side2Value) : side1(side1Value), side2(side2Value)
{
}
```
Call a default constructor

• Rect companionRect(); // this is bad
  – There are two options
• Rect companionRect;
  – Or
• companionRect = Rect();
Many Constructors

- This is equivalent to the previous page

class Rect
{
public:
    // Constructor prototype
    Rect(float side1Value, float side2Value);
    Rect(); // Default Constructor
    Rect(float side);
    float getArea();
private:
    float side1;
    float side2;
};

int main()
{}

// Constructor definition
Rect::Rect(float side1Value, float side2Value) : side1(side1Value), side2(side2Value)
{
}
}
Const Parameter Modifier

• If you are passing by reference
  – And your function shouldn’t change the value of the parameter.
  – Pass it in as a constant parameter like so

• `printStuff(const bool &gradeSheet[] [MAXQUESTIONS])`

• This ensures that the values of gradeSheet remain unchanged.
Const with member functions

• If you have a member function that shouldn’t change the value then use const.

class Rect
{
public:
    void printArea() const;
    ...

    void Rect::printArea() const
    {
        ...
    
}
Inline functions

- If you have a really simple member function you can write it in the same line

```cpp
class Rect {
public:
    // Constructor prototype
    Rect(float side1Value, float side2Value);
    float printArea() const {return side1*side2;}; // Inline function
private:
    float side1;
    float side2;
};
int main()
{}
// Constructor definition
Rect::Rect(float side1Value, float side2Value) : side1(side1Value), side2(side2Value)
{
}
```