UMBC

AN HONORS UNIVERSITY IN MARYLAND

# CMSC202 Computer Science II for Majors

# Lecture 08 – Overloaded Constructors

Dr. Katherine Gibson

## Student Disability Services

- If you need to take the exam at SDS:

- You need to set it up with them
- You need to make sure I know about it
- This needs to be done <u>ahead of time</u>
  - This should already have been done by now!

## Exam Rules

- The exam is closed everything:
  - No books
  - No notes
  - No cheat sheets
  - No laptops
  - No calculators
  - No phones

## Exam Rules

- Place your bookbag under your desk/chair
  - NOT on the seat next to you

- You may have on your desk:
  - Pens, pencils, erasers
  - Water bottle
  - **UMBC ID**

## Exam Rules

- DO NOT CHEAT!!!
- Cheating will be dealt with severely and immediately
  - If a TA or instructor sees you looking at another student's paper (or anything other than your own exam) they will take your test from you
- Space allowing, you will sit every other seat, so that you are not next to another student

## Exam Format

- True/False
- Multiple Choice
- Short Answer
  - Explain basic concepts in writing
- Debugging
  - Find and correct errors

## Exam Format

- Code Evaluations
  - Given code, what does it do?
- Code Completions
  - Given a partial piece of code
  - Correctly fill in blanks to complete code
- Coding Problems
  - Given a problem, write code to solve it

# Exam Tips: General

- Write down your name and circle your section
- Flip through the exam and get a feel for the length of it and the types of questions

- If a problem is unclear or you think there is an error on the exam, raise your hand

- Most questions have partial credit
  - You should at least <u>attempt</u> every problem

## Exam Tips: Coding

- When coding:
  - Read the question carefully
  - Plan out what your code needs to do
- After you are done coding the programming problems, try “running” your program with some input and making sure it works the way you think it does

## Exam Content

- Everything we’ve covered so far! Including...
  - C++ Syntax
    - Loops, data types, cin, cout, C-strings, etc.
  - Functions
    - Prototype, definition, call, return value, parameters
  - Pointers, arrays
    - Passing arrays to functions, **&**, *****, addresses, etc.
  - Classes!
    - Access modifiers, class methods, member variables, constructors, objects, dot operator
  - And more!

## Questions about Exam 1?

## Last Class We Covered

- Classes
  - Access modifiers
  - Methods
    - Mutators
    - Accessors
    - Facilitators
  - Constructors
- Livecoding: Rectangle class

## Any Questions from Last Time?

## Today’s Objectives

- To learn about overloading methods
  - “Regular” class methods
  - Overloaded constructors
- To complete our Rectangle class
- To review for Exam 1

## Overloading

## Overloading

- We can define multiple versions of the constructor – we can ***overload*** it
- Different constructors for:
  - When all values are known
  - When no values are known
  - When some subset of values are known

## All Known Values

- Have the constructor set user-supplied values

```
Date::Date (int m, int d, int y)
{
    SetMonth(m);
    SetDay(d);
    SetYear(y);
}
```

invoked when constructor is called with all arguments

UMBC
AN HONORS UNIVERSITY IN MARYLAND

## No Known Values

- Have the constructor set all default values

```
Date::Date ()
{
   SetMonth(DEFAULT_MON);
   SetDay(DEFAULT_DAY);
   SetYear(DEFAULT_YEAR);
}
```

invoked when constructor is called with no arguments

## Some Known Values

- Have the constructor set some default values

```
Date::Date (int m, int d)
{
   SetMonth(m);
   SetDay(d);
   SetYear(DEFAULT_YEAR);
}
```

invoked when constructor is called with two arguments

## Overloaded Date Constructor

- so far we have the following constructors:

```
Date::Date (int m, int d, int y);
Date::Date (int m, int d);
Date::Date ();
```

- would the following be a valid constructor?

```
Date::Date (int m, int y);
```

## Avoiding Multiple Constructors

- Defining multiple constructors for different sets of known values is a lot of unnecessary code duplication

- We can avoid this by setting ***default parameters*** in our constructors

## Default Parameters

- In the ***function prototype*** only, provide default values you want the constructor to use

```
Date (int m      , int d       ,
      int y        );
```

## Default Parameters

- In the ***function prototype*** only, provide default values you want the constructor to use

```
 Date (int m = 1, int d = 12,
       int y = 1967);
```

- (You should, of course, use constants when providing default parameters.)

# Default Parameters

- In the ***function definition*** nothing changes

```
Date::Date (int m, int d, int y) {
   SetMonth(m);
   SetDay(d);
   SetYear(y);
}
```

# Using Default Parameters

- the following are all valid declarations:

```
Date graduation(5,19,2016);
Date gritBDay;
Date halloween(10,31);
Date july(4);

// graduation: 5/19/2016
// gritBDay:   1/12/1967
// halloween:  10/31/1967
// july:       4/12/1967
```

NOTE: when you call a constructor with no arguments, you do not give it empty parentheses

## Default Constructors

- A ***default constructor*** is provided by compiler
  - Will handle declarations of **`Date`** instances

- This is how we created **`Date`** objects in the slides before we declared and defined our constructor

## Default Constructors

- **But**, if you create **any** other constructor, the compiler doesn't provide a default constructor
- So if you create a constructor, make a default constructor too, even if its body is just empty

```
Date::Date ()
{
   /* empty */
}
```

## Function Overloading

- Functions in C++ are uniquely identified by both their names and their parameters
  - **But NOT their return type!**

- We can overload any kind of function
  - We can even use default values, like with constructors

## Overloading Example

```
void PrintMessage (void) {
   cout << "Hello World!" << endl;
}

void PrintMessage (string msg) {
   cout << msg << endl;
}
```

## Time for…

LIVECODING!!!

## Livecoding Exercise

- Update our Rectangle class with
  - Overloaded Constructor
    - Implemented through default parameters
  - Create a class method to:
    - Print all of a Rectangle's information

- Update our **`main()`** function

# Announcements

- Project 1 is due tonight by 9:00 PM
  - Make sure you have correctly submitted all of your files!
- Exam 1 will be held on Thursday (the 25th) during our regular class time