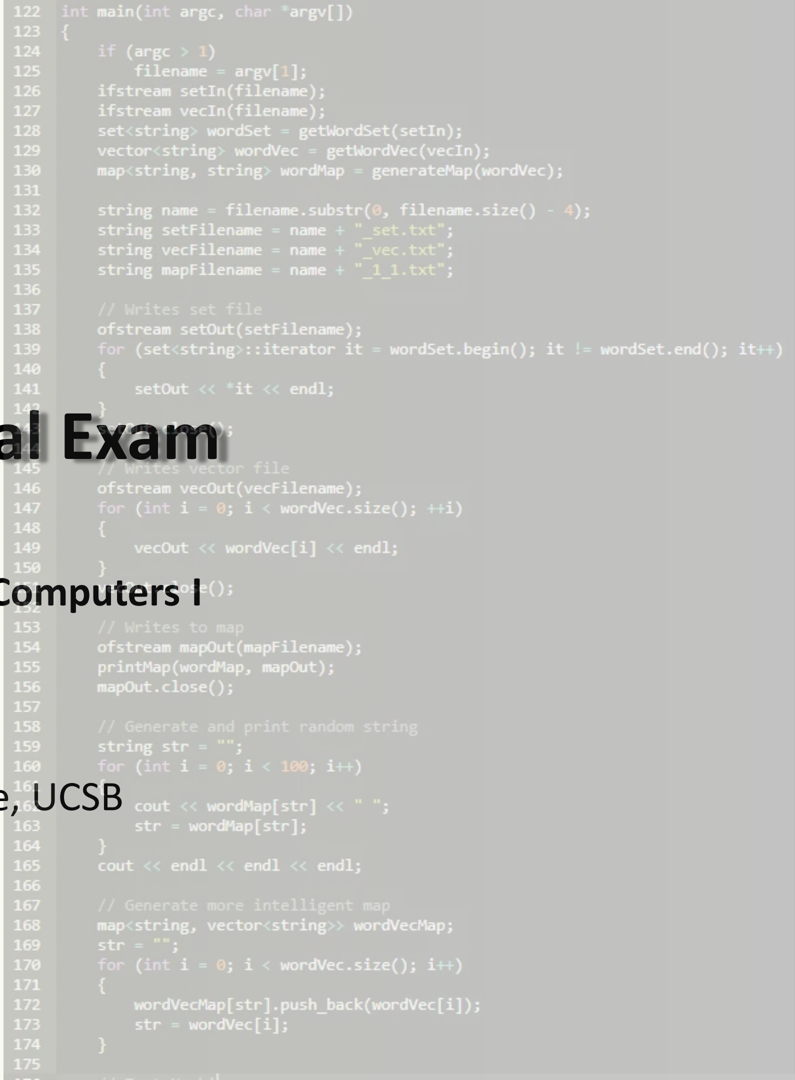
Review for the Final Exam

CS 16: Solving Problems with Computers I
Lecture #18

Ziad Matni
Dept. of Computer Science, UCSB



```
int main(int argc, char *argv[])
{
    if (argc > 1)
        filename = argv[1];
    ifstream setIn(filename);
    ifstream vecIn(filename);
    set<string> wordSet = getWordSet(setIn);
    vector<string> wordVec = getWordVec(vecIn);
    map<string, string> wordMap = generateMap(wordVec);

    string name = filename.substr(0, filename.size() - 4);
    string setFilename = name + "_set.txt";
    string vecFilename = name + "_vec.txt";
    string mapFilename = name + "_1_1.txt";

    // Writes set file
    ofstream setOut(setFilename);
    for (set<string>::iterator it = wordSet.begin(); it != wordSet.end(); it++)
    {
        setOut << *it << endl;
    }
    setOut.close();

    // Writes vector file
    ofstream vecOut(vecFilename);
    for (int i = 0; i < wordVec.size(); ++i)
    {
        vecOut << wordVec[i] << endl;
    }
    vecOut.close();

    // Writes to map
    ofstream mapOut(mapFilename);
    printMap(wordMap, mapOut);
    mapOut.close();

    // Generate and print random string
    string str = "";
    for (int i = 0; i < 100; i++)
    {
        cout << wordMap[str] << " ";
        str = wordMap[str];
    }
    cout << endl << endl << endl;

    // Generate more intelligent map
    map<string, vector<string>> wordVecMap;
    str = "";
    for (int i = 0; i < wordVec.size(); i++)
    {
        wordVecMap[str].push_back(wordVec[i]);
        str = wordVec[i];
    }

```

Administrative

- Remember to turn in your assignments on time this week!
  - Lab/Hwk8 **tomorrow (Wed)**
  - Lab/Hwk9 **Friday**
  - No late submissions please

- No quiz this week! (yay?!)
- No pre-recorded videos this week!

- We have regular office hours this week
  - I will have special hours next week (Monday) and will announce on Piazza

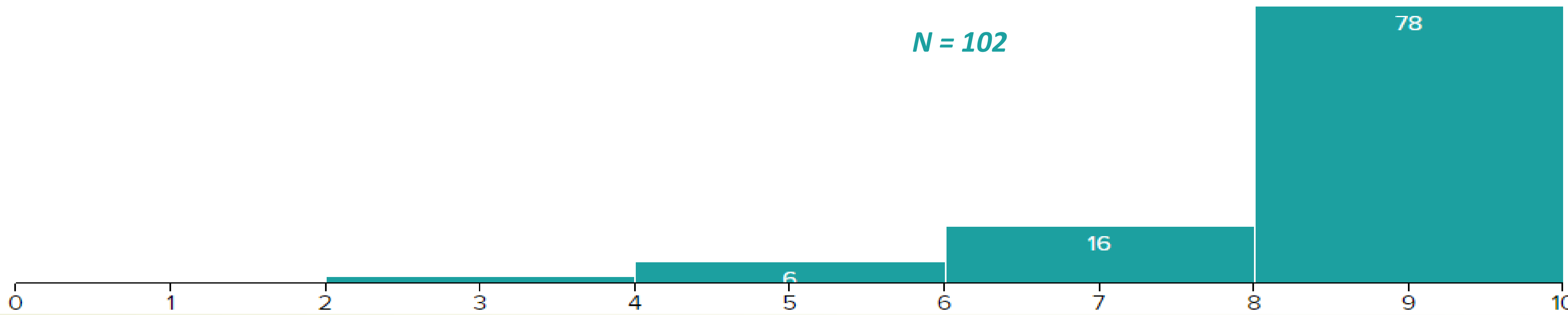
- Final Exam…

FINAL EXAM

- Comprehensive (everything we've done in this class)
- I've put up **practice questions** on our website
- ULA-led **review session** on **Thursday, Dec. 10th, 5 – 6 PM**
  - Zoom link will be posted on Piazza soon

- Exam will be on **Wednesday, Dec. 16th** on Gradescope
- Made available at **9:00 AM, for 24-hours**. Timed for **2 hours**.
- I will be online on Piazza to answer questions:
  - From **12:00 – 2:00 PM** and from **5:00 – 6:00 PM**

Quiz 8



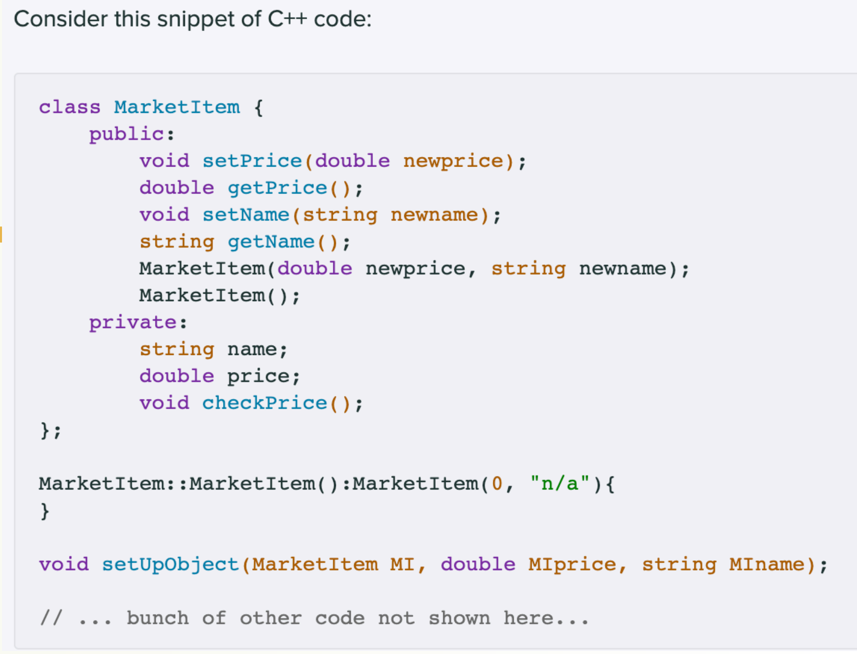
- Mean: **8.12/10**
- Median: **8/10**

Question 1.1

- Which of these is a permitted way to initialize an object of this class?

Not appropriate for a variable declaration, obv. ok for function

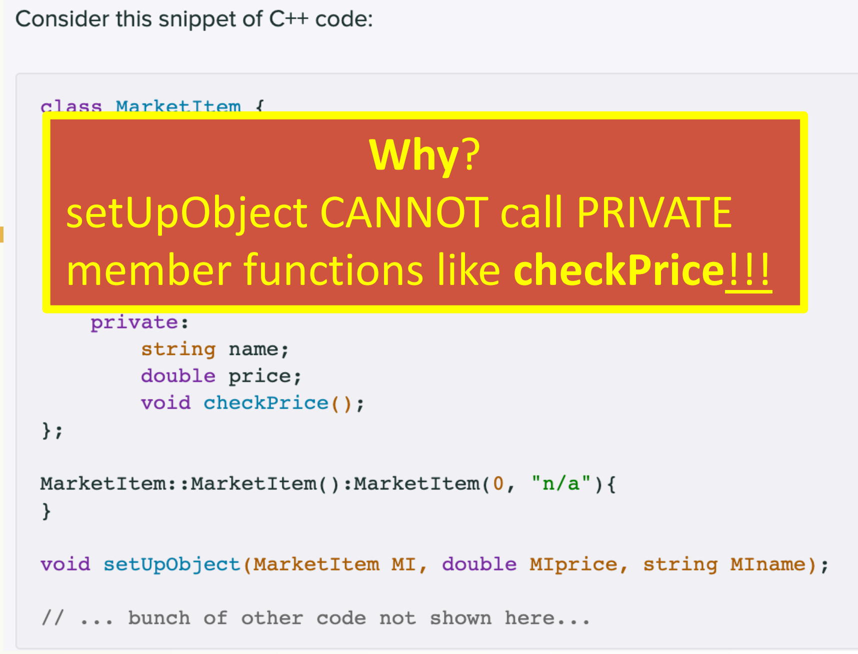
- X A. `MarketItem chocolateBar();`
- ✓ B. `MarketItem chocolateBar(1.5, "Twix");`
- ✓ C. `MarketItem chocolateBar;`
- Only A and B.
- [✓ Only B and C.]
- All of A, B and C.
- None of the above.


Consider this snippet of C++ code:

```
class MarketItem {
    public:
        void setPrice(double newprice);
        double getPrice();
        void setName(string newname);
        string getName();
        MarketItem(double newprice, string newname);
        MarketItem();
    private:
        string name;
        double price;
        void checkPrice();
};

MarketItem::MarketItem():MarketItem(0, "n/a"){
}

void setUpObject(MarketItem MI, double MIprice, string MIname);

// ... bunch of other code not shown here...
```

Q1.2

- Which of these function definitions can call **checkPrice()** without triggering a compile error?
  - A. `setPrice`
  - B. `setUpObject`
  - C. `getPrice`


Consider this snippet of C++ code:

```
class MarketItem {
    private:
        string name;
        double price;
        void checkPrice();
};

MarketItem::MarketItem():MarketItem(0, "n/a"){
}

void setUpObject(MarketItem MI, double MIprice, string MIname);

// ... bunch of other code not shown here...
```

Why?
setUpObject CANNOT call PRIVATE member functions like **checkPrice**!!!

- Alas, I did not have "**A and C**" as an option (my oversight), so everyone got these 2 points regardless of your answer (freebie).

Q1.3

- Write a definition for the member function **checkPrice()** that is supposed to check that the variable price is a positive non-zero number that's less than $100.

```
void MarketItem::checkPrice() {
   if(price >= 100 || price <= 0) {
      cerr << "Invalid Price";
      exit(1);
   }
}
```

Also acceptable:
- Else statement that prints out ”valid”

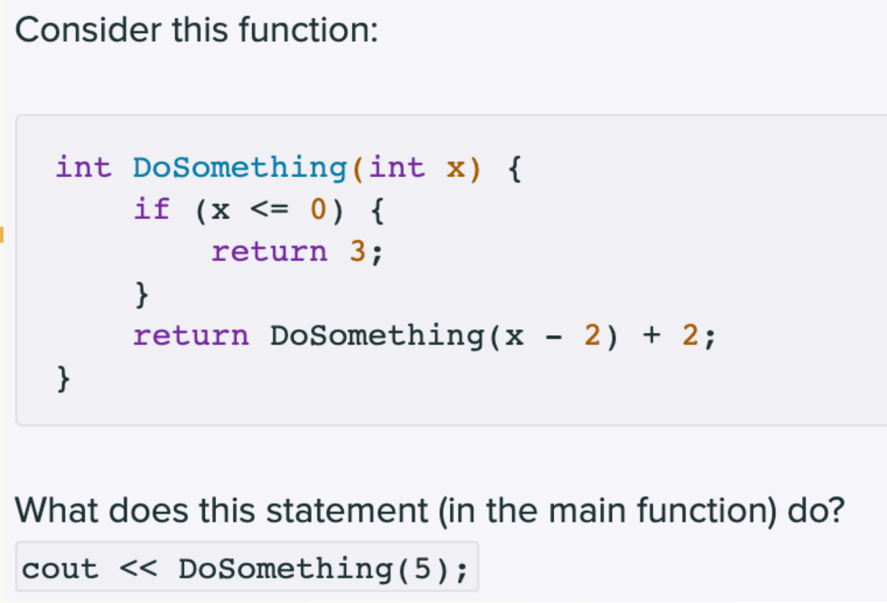
Considered “minor” (if only thing):
- Messing up the inequalities in the **if**
- Unnecessary code or weird design
- Using getPrice() instead of mem. variable
- Syntax error

Not acceptable:
- Re-purposing the given function to be a Boolean type (it’s not declared like that)
- Returning Booleans in a VOID function
- Messing up the class function header
- More than 1 “minor” mistake

Q3.6

- **DoSomething(5):**
  - Returns **DoSomething(3) + 2**
    - Returns **DoSomething(1) + 2** + 2
      - Returns **DoSomething(-1) + 2** + 2 + 2
        - » Returns **3** + 2 + 2 + 2

- So, it prints out **9**



Consider this function:

```
int DoSomething(int x) {
    if (x <= 0) {
        return 3;
    }
    return DoSomething(x - 2) + 2;
}
```

What does this statement (in the main function) do?

```
cout << DoSomething(5);
```

Lecture Outline

- Exercises!

High-Level List of Topics

This is NOT Comprehensive of Everything Discussed

- Standard I/O, basic var types & ops
- Flow Control (if/else, loops)
- Functions in C++
- Command-line arguments
- Basic UNIX commands
- Compiling Multiple Files / Makefile
- Debug Techniques, TDD

- Arrays
- Strings and Chars
- Search and Sort Algorithms
- File I/O
- Bin/Oct/Hex Conversions
- Structs, Classes
- Recursion

Exercise 1

- Write a function definition for a function called **ArrayMax** that takes a 2-dimensional double-type array with 10 rows and 10 columns as an argument, and returns the maximum of all the elements of the two-dimensional array as a result.
  - Assume that the smallest number in the array is no less than -65,535.
  - Pre-conditions? Post-conditions?

- What's your algorithm?

Exercise 2

- We all know what a factorial is (n!)
- For example, 10! = 10 × 9 × ... × 3 × 2 × 1 = **3628800**
- The sum of the digits in the number 10! is 3 + 6 + 2 + 8 + 8 + 0 + 0 = **27**.
- Let's program a way to sum the digits of any factorial
  - What are the limitations? Pre-conditions? Post-conditions?

- What's your algorithm?

Exercise 3

- Write a program that can find the **20th** number in the linear arithmetic series:

  1 25 217 1,753 14,041 …

  – What are the limitations? Pre-conditions? Post-conditions?

Exercise 3

- Assume we start the series at position 1
- How do we figure out the series?
  - If the general form is: $a(n) = M.a(n-1) + N$

- So, $a(1) = 1$, $a(2) = M + N = 25$, $a(3) = 25M + N = 217$
- 2 equations, 2 unknowns:
  - $M = 25 – N \Rightarrow 25(25 – N) + N = 217 \Rightarrow 625 – 24N = 217 \Rightarrow N = 17$
  - So, $M = 8$
  - So, the form is: $\mathbf{a(n) = 8.a(n-1) + 17}$

Exercise 4

- Write a program that will take a sentence and reverse the word order, for example: Oh Christmas tree!
  Becomes: tree! Christmas Oh
  - Assume that words are separated by space characters
  - What's the algorithm you'd use?

Exercise 4

- Start from the end of the string
- Collect the word backwards until you get to a space char.
- Then reverse the word and print it
- Starting from where you are in the string, **repeat**

- What's missing?
  - Devil's in the details…

YOUR TO-DOs

- Turn in your lab and homework assignments!
  - Remember: Lab9/Hwk9 are due by Friday (last day of the quarter).
  - **NO LATE SUBMISSIONS ALLOWED FOR THESE!! FRIDAY IS IT!**
- Take advantage of office hours this week!!
- Study for the final!
- Have a nice, HEALTHY break! ☺

</LECTURE>