**CSCI 1411 – Lab 12–Objects with Classes**

**Goals:**

* Understand how to build and use Classes
* Understand how to use arrays of objects built with classes

**Development Environment:** (all students must use Clion)

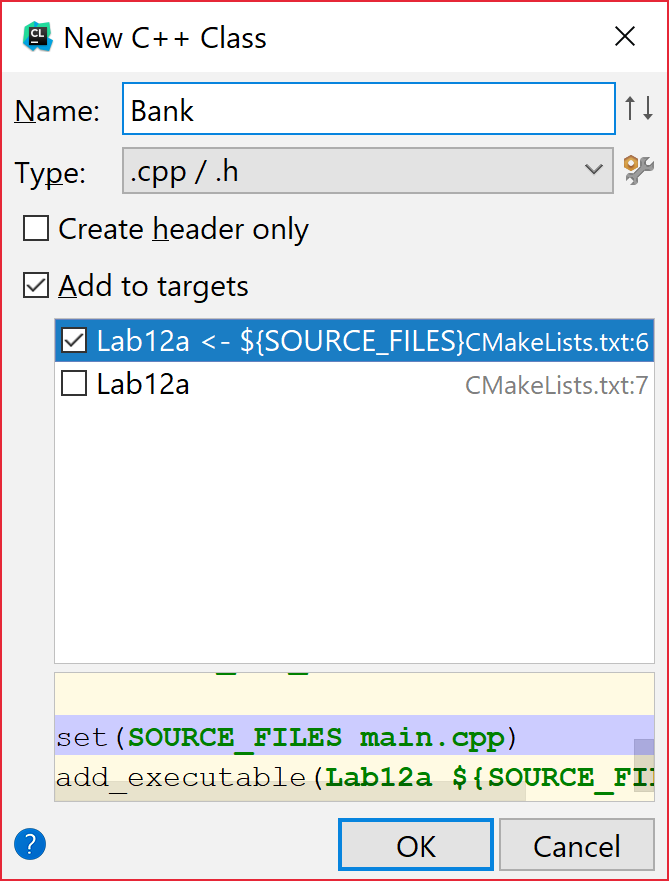
**Skills:** Classes, Default Constructor, Constructor, member functions, arrays of objects.

**Reading**: Chap 13

**Deliverables:** 1) This lab with 3 screen shots 2) lastnameFirstLab12.cpp, Scores.h, Scores.cpp

**Part I – Skills Practice (5 points)**

* Open a new project in CLion. Call the project Lab12a.
* For this project we are creating an array of Bank Accounts
* New->New C++ Class



* Add the name Bank. And accept the defaults Add to targets and Lab12a<-$(Source\_Files)
* This will create both a Bank.h file and a Bank.cpp file
* Double Click on the Bank.h file. Replace the entire text with this
* **#ifndef BANK\_H  
  #define BANK\_H  
    
  #include <string>  
  using namespace std;  
  class Bank  
  {  
  private:  
   string name;  
   int number;  
   float balance;  
  public:  
   Bank(); //default constructor  
   Bank(string \_name, int \_number, float \_balance); //constructor**

**string getName(){return name;}  
 void setName(string \_name){name=\_name;}  
 int getNumber() {return number;}  
 void setNumber(int \_number){number = \_number;}  
 float getBalance(){return balance;}  
 void increaseBalance(float amount);  
};  
#endif**

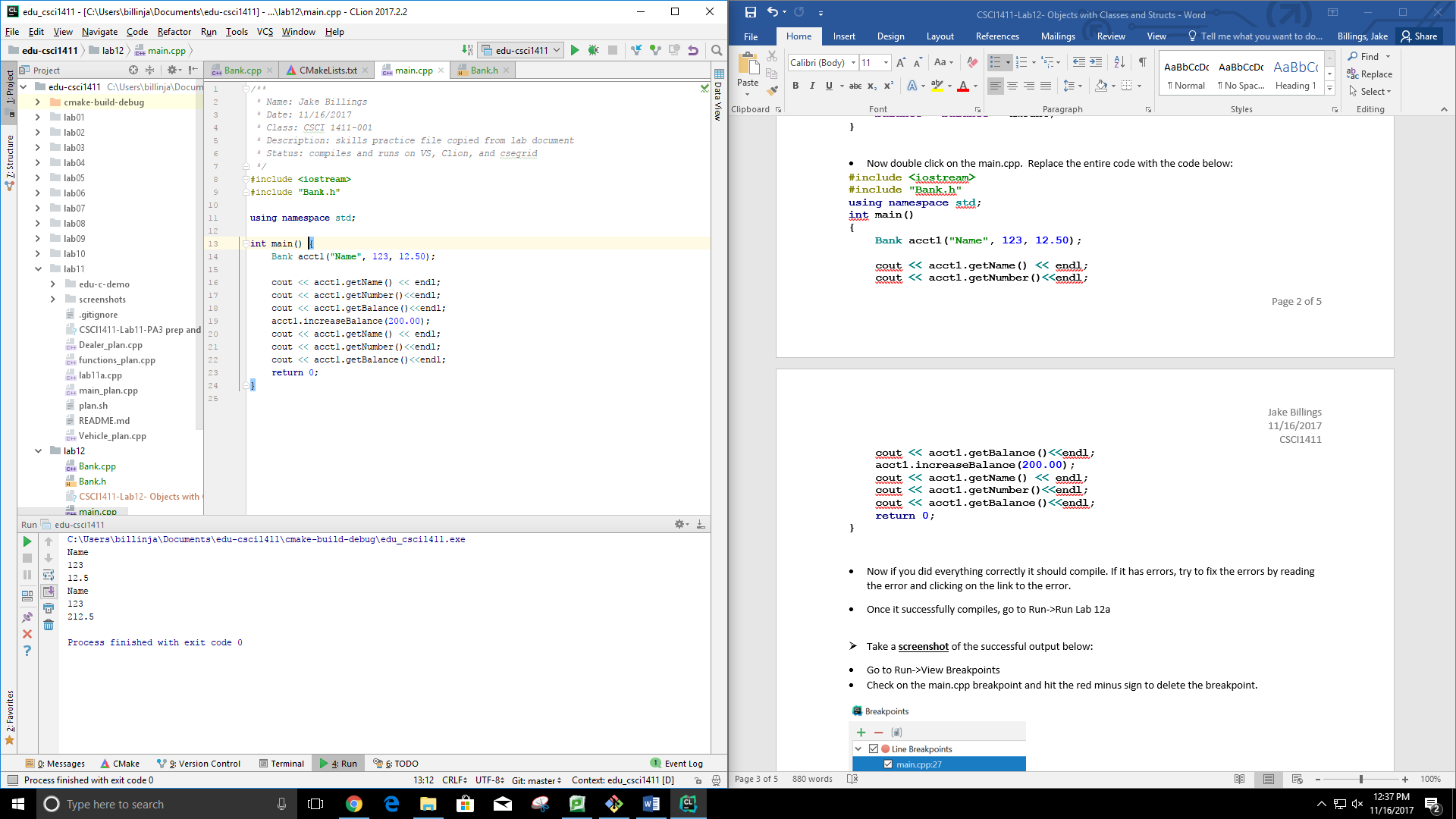
* Now double click on the Bank.cpp. Replace the entire code with the code below:

**#include "Bank.h"  
Bank::Bank()   
{  
 name = "";  
 number = 0;  
 balance = 0;  
}  
  
Bank::Bank(string \_name, int \_number, float \_balance)  
{  
 name = \_name;  
 number = \_number;  
 balance = \_balance;  
}  
  
void Bank::increaseBalance(float amount)   
{  
 balance = balance + amount;  
}**

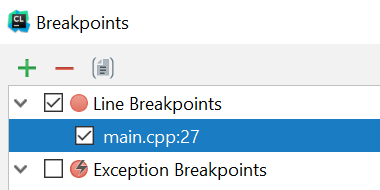
* Now double click on the main.cpp. Replace the entire code with the code below:

**#include <iostream>  
#include "Bank.h"  
using namespace std;  
int main()  
{  
 Bank acct1("Name", 123, 12.50);  
  
 cout << acct1.getName() << endl;  
 cout << acct1.getNumber()<<endl;  
 cout << acct1.getBalance()<<endl;  
 acct1.increaseBalance(200.00);  
 cout << acct1.getName() << endl;  
 cout << acct1.getNumber()<<endl;  
 cout << acct1.getBalance()<<endl;  
 return 0;  
}**

* Now if you did everything correctly it should compile. If it has errors, try to fix the errors by reading the error and clicking on the link to the error.
* Once it successfully compiles, go to Run->Run Lab 12a
* Take a **screenshot** of the successful output below:



* Go to Run->View Breakpoints
* Check on the main.cpp breakpoint and hit the red minus sign to delete the breakpoint.



* Some of the things that you should have noted in this example.
  + Classes should start with a capital letter.
  + The Class declaration should be in the .h file
  + The Class definitions should either be in line (in the .h file) if only one line, or in a separate .cpp file
  + The .h file should always start with the #ifndef, #define and end with the #endif. This ensures that code is only defined once in a compilation
  + Every class should have a default constructor. This is called whenever an object of that type is declared. The default constructor usually initializes the private variables
  + Any public member function can directly access the private variables.

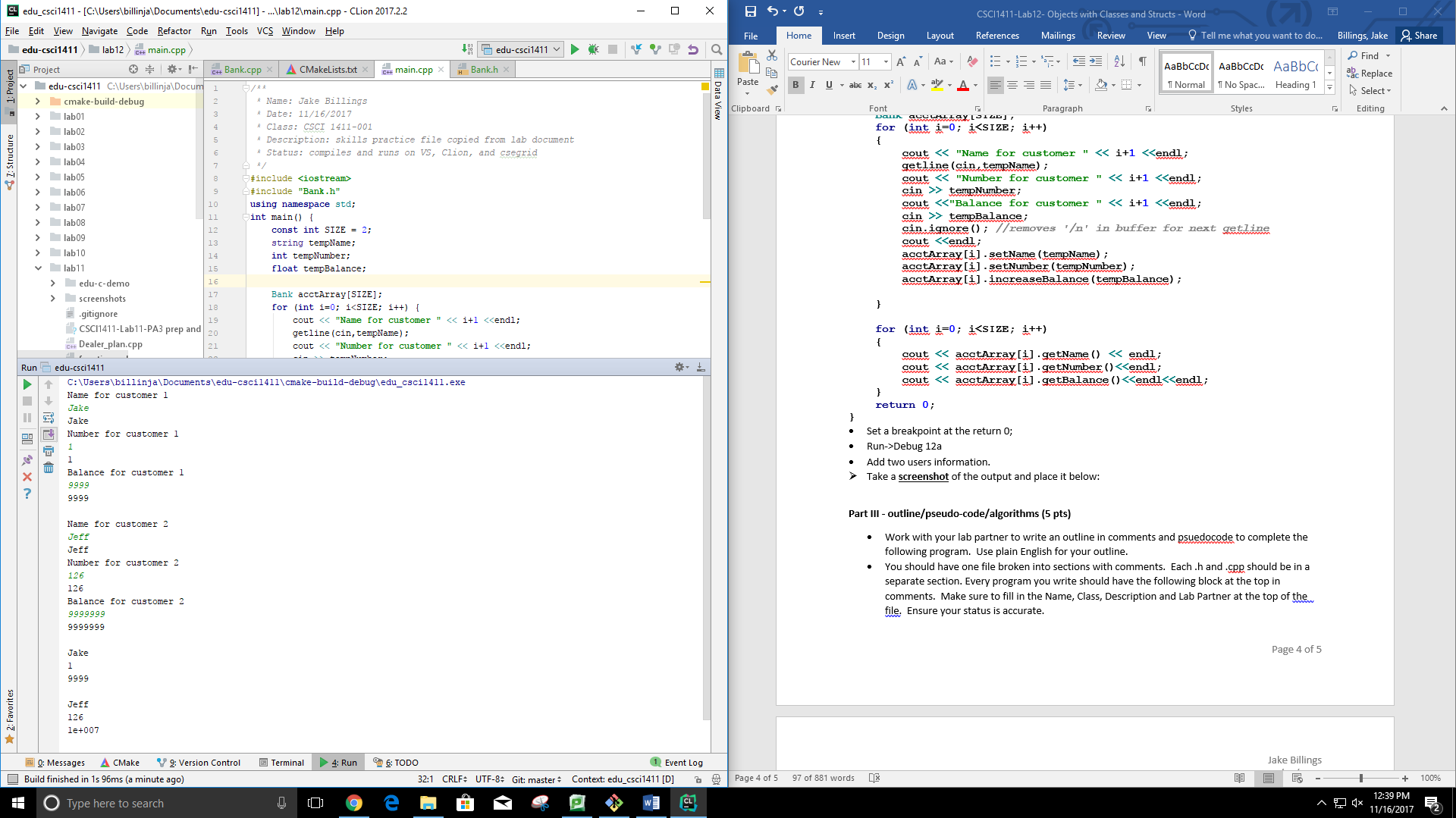
**Part II – More Skills Practice (5 points)**

Arrays of Classes. We will now create an array of Bank Accounts

Using the code from Part I. Change main.cpp to:

**#include <iostream>  
#include "Bank.h"  
using namespace std;  
int main()  
{  
 const int SIZE = 2;  
 string tempName;  
 int tempNumber;  
 float tempBalance;  
  
 Bank acctArray[SIZE];  
 for (int i=0; i<SIZE; i++)  
 {  
 cout << "Name for customer " << i+1 <<endl;  
 getline(cin,tempName);  
 cout << "Number for customer " << i+1 <<endl;  
 cin >> tempNumber;  
 cout <<"Balance for customer " << i+1 <<endl;  
 cin >> tempBalance;  
 cin.ignore(); *//removes '/n' in buffer for next getline* cout <<endl;  
 acctArray[i].setName(tempName);  
 acctArray[i].setNumber(tempNumber);  
 acctArray[i].increaseBalance(tempBalance);  
  
 }  
  
 for (int i=0; i<SIZE; i++)  
 {  
 cout << acctArray[i].getName() << endl;  
 cout << acctArray[i].getNumber()<<endl;  
 cout << acctArray[i].getBalance()<<endl<<endl;  
 }  
 return 0;  
}**

* Set a breakpoint at the return 0;
* Run->Debug 12a
* Add two users information.
* Take a **screenshot** of the output and place it below:

****

**Part III - outline/pseudo-code/algorithms (5 pts)**

* Work with your lab partner to write an outline in comments and psuedocode to complete the following program. Use plain English for your outline.
* You should have one file broken into sections with comments. Each .h and .cpp should be in a separate section. Every program you write should have the following block at the top in comments. Make sure to fill in the Name, Class, Description and Lab Partner at the top of the file. Ensure your status is accurate.

/\* Name:  
Class: CSCI 1411-00X  
Description: [fill in description]

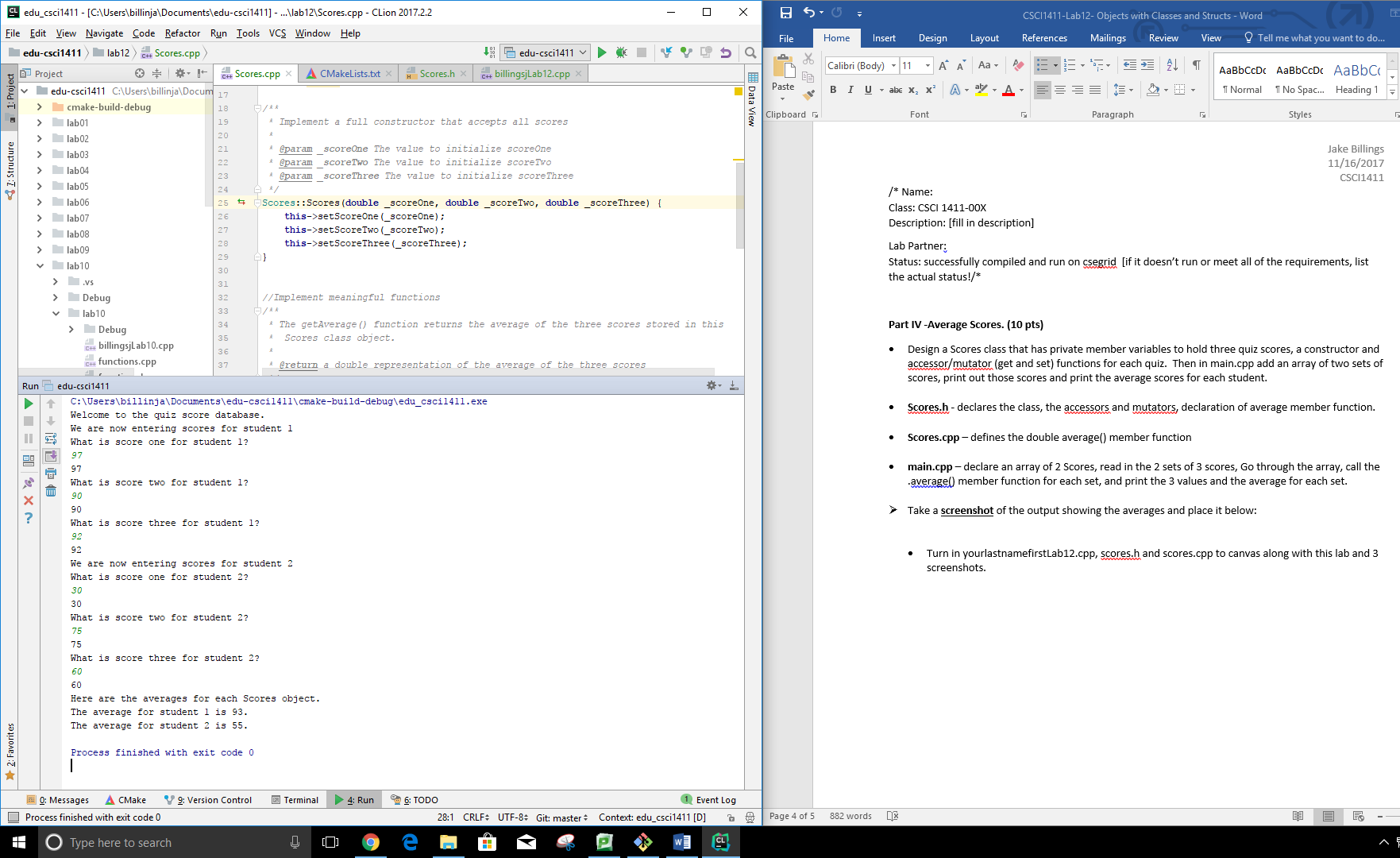
Lab Partner:  
Status: successfully compiled and run on csegrid [if it doesn’t run or meet all of the requirements, list the actual status!/\*

*/\*\*  
 \* Name: Jake Billings  
 \* Date: 11/16/2017  
 \* Class: CSCI 1411-001  
 \* Description: Scores class implementation; the scores class holds three scores  
 \* Status: compiles and runs on VS, Clion, and csegrid  
 \*/  
//Implement constructors  
/\*\*  
 \* Implement an empty constructor  
 \*/  
/\*\*  
 \* Implement a full constructor that accepts all scores  
 \*  
 \* @param \_scoreOne The value to initialize scoreOne  
 \* @param \_scoreTwo The value to initialize scoreTwo  
 \* @param \_scoreThree The value to initialize scoreThree  
 \*/  
//Implement meaningful functions  
/\*\*  
 \* The getAverage() function returns the average of the three scores stored in this  
 \* Scores class object.  
 \*  
 \* @return a double representation of the average of the three scores  
 \*/  
//Implement Getters and Setters for each Score*

*/\*\*  
 \* Name: Jake Billings  
 \* Date: 11/16/2017  
 \* Class: CSCI 1411-001  
 \* Description: declare an array of 2 Scores, read in the 2 sets of 3 scores, Go through the array, call the .average()  
 \* member function for each set, and print the 3 values and the average for each set.  
 \* Status: compiles and runs on VS, Clion, and csegrid  
 \*/  
//---------Dependency Imports---------  
//Include cout  
//Include the Scores class  
//Use the standard namespace  
//---------Declare Constants---------  
//-----------Main Function------------  
/\*\*  
 \* int main()  
 \*  
 \* The main entry point for the application.  
 \*  
 \* Returns 0.  
 \*/  
 //Declare a scores array of length SCORES\_COUNT (2)  
 //Print a welcome message  
 //Iterate once for each space in the scores object array  
 //Print a message telling the user which object the scores are going into.  
 // Add one so that humans don't get confused by our zero indexing  
 //Read score one  
 //Read score two  
 //Read score three  
 //Create the Scores object  
 //Place the object in the array  
 //Print a message stating we are entering the averaging-phase of the progam  
 //Iterate through the array again to print the averages*

**Part IV -Average Scores. (10 pts)**

* Design a Scores class that has private member variables to hold three quiz scores, a constructor and accessor/mutator (get and set) functions for each quiz. Then in main.cpp add an array of two sets of scores, print out those scores and print the average scores for each student.
* **Scores.h** - declares the class, the accessors and mutators, declaration of average member function.
* **Scores.cpp** – defines the double average() member function
* **main.cpp** – declare an array of 2 Scores, read in the 2 sets of 3 scores, Go through the array, call the .average() member function for each set, and print the 3 values and the average for each set.
* Take a **screenshot** of the output showing the averages and place it below:
* Turn in yourlastnamefirstLab12.cpp, scores.h and scores.cpp to canvas along with this lab and 3 screenshots.

****