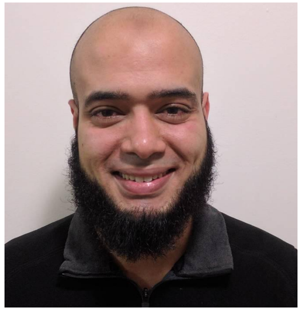
C++ Programming

Class Const, Static & Friend

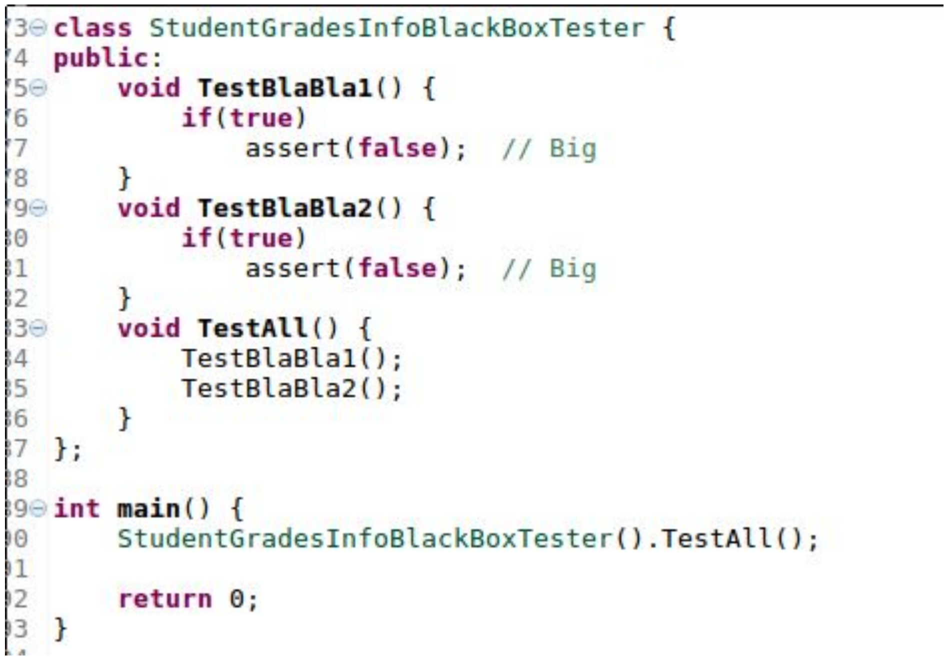
Homework 2

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Homework 05: BlackBox Testing



```
class StudentGradesInfoBlackBoxTester {
public:
    void TestBlaBla1() {
        if(true)
            assert(false);  // Big
    }
    void TestBlaBla2() {
        if(true)
            assert(false);  // Big
    }
    void TestAll() {
        TestBlaBla1();
        TestBlaBla2();
    }
};

int main() {
    StudentGradesInfoBlackBoxTester().TestAll();

    return 0;
}
```

- In black box testing, we test the public functionality of a class
    - Focus on what not how
    - No care of internals
- Develop a class that test our previous class
    - Try the old code
    - Then the fixed code

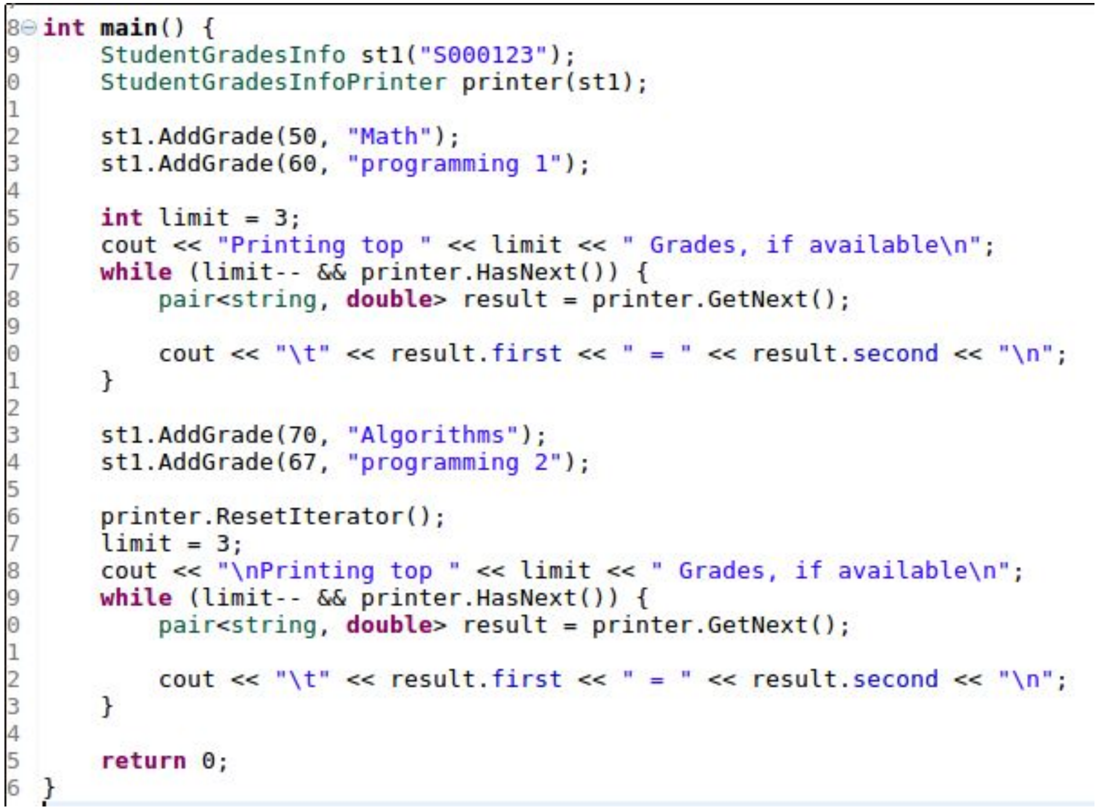
Homework 06: WhiteBox Testing

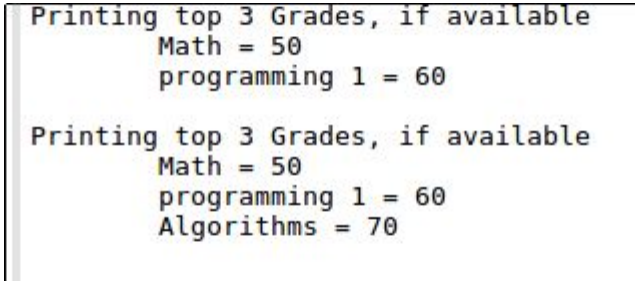
- In white box testing, we care about really what happens **internally**.
    - The private variables and methods and their updates
- But how to test the class from outside?
    - Several approaches
    - One way Define **friend class StudentGradesInfoWhiteBoxTester**; inside the class StudentGradesInfo
        - Please follow this approach for the homework
    - Now, the tester can access all internals and do deeper testing
- About white testing & using friend: Future reading

Homework 07: Code Extension

- Sadly current print has 2 issues
  - Print to console / Print all content!
- For some reasons, we can’t change the code
  - Another idea is to **extend** its functionality!
- Your team lead asked to develop a class that satisfy the following main
  - Mainly a new class that works on an object from StudentGradesInfo
  - It satisfies 2 critical functions for **iterating** on the StudentGradesInfo courses:
    - HasNext: That tell us if there is more to retrieve
    - GetNext: Return actual element in turn to retrieve
  - Also Reset method in case we wanna iterate from scratch again
- Develop a class that satisfies this main

Homework 07: Code Extension

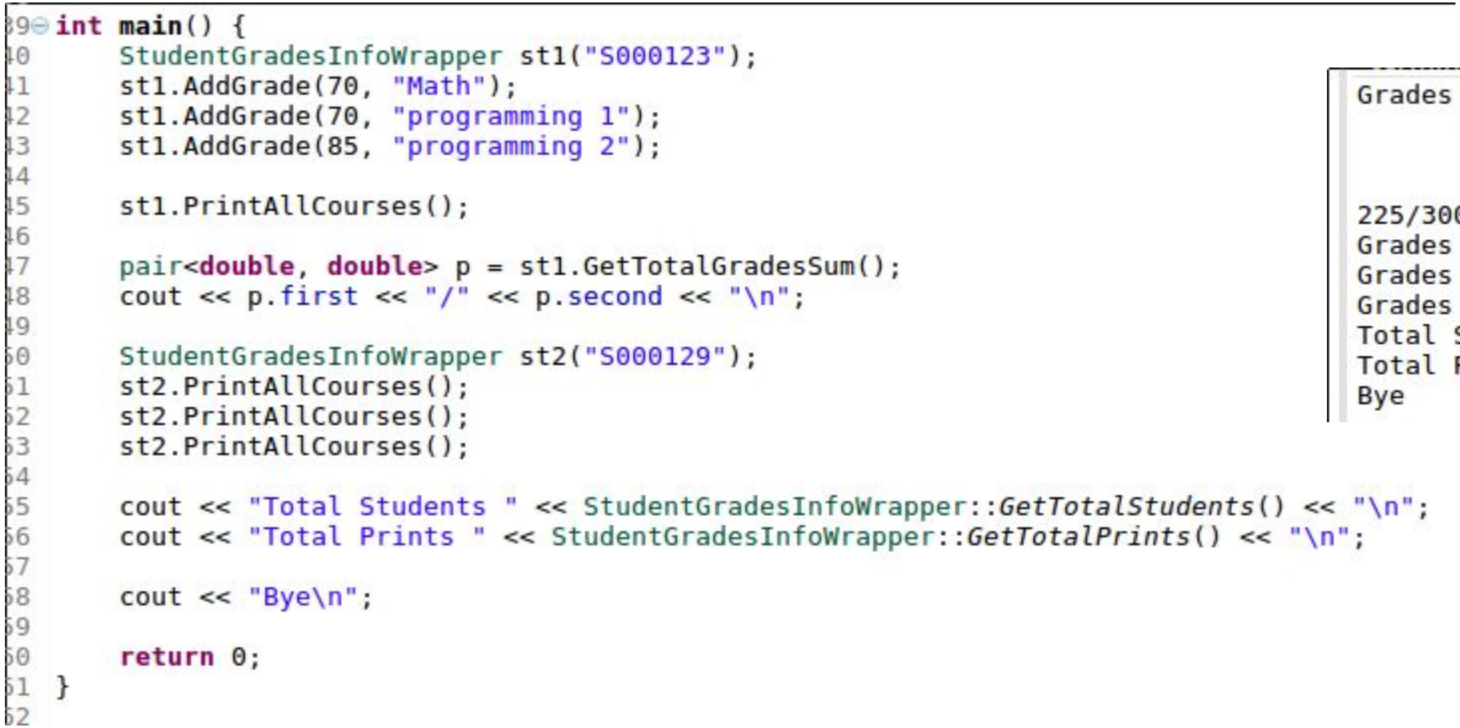

```
int main() {
    StudentGradesInfo st1("S000123");
    StudentGradesInfoPrinter printer(st1);

    st1.AddGrade(50, "Math");
    st1.AddGrade(60, "programming 1");

    int limit = 3;
    cout << "Printing top " << limit << " Grades, if available\n";
    while (limit-- && printer.HasNext()) {
        pair<string, double> result = printer.GetNext();

        cout << "\t" << result.first << " = " << result.second << "\n";
    }

    st1.AddGrade(70, "Algorithms");
    st1.AddGrade(67, "programming 2");

    printer.ResetIterator();
    limit = 3;
    cout << "\nPrinting top " << limit << " Grades, if available\n";
    while (limit-- && printer.HasNext()) {
        pair<string, double> result = printer.GetNext();

        cout << "\t" << result.first << " = " << result.second << "\n";
    }

    return 0;
}
```


```
Printing top 3 Grades, if available
        Math = 50
        programming 1 = 60

Printing top 3 Grades, if available
        Math = 50
        programming 1 = 60
        Algorithms = 70
```

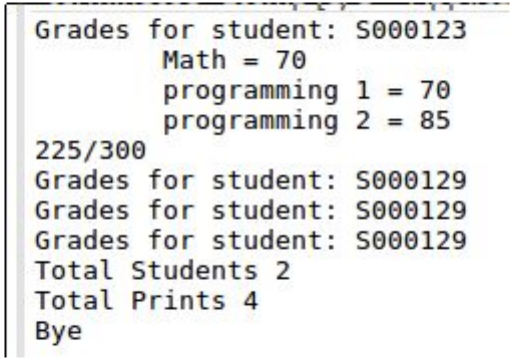
Homework 08: Wrapper

- StudentGradesInfo is coming from an open source library. Good to save time
  - Your team lead is afraid from hidden bugs or stopping the maintenance
  - What if we have 20 classes that use it and then we decided to replace or write our own!
  - Any change in this class => change in all of them!
- Your team lead suggested building a wrapper
  - The idea is create another class StudentGradesInfoWrapper
    - It provides the same public functionality as StudentGradesInfo
  - It has object from type StudentGradesInfo
  - With every call to StudentGradesInfoWrapper, just call same function in ur local object
  - Now all your code depends on the wrapper not on the open source code that may change
- Please also maintain 2 public methods to tell us total
  - User prints and total # of created students info

Homework 08: Wrapper



```
int main() {
    StudentGradesInfoWrapper st1("S000123");
    st1.AddGrade(70, "Math");
    st1.AddGrade(70, "programming 1");
    st1.AddGrade(85, "programming 2");

    st1.PrintAllCourses();

    pair<double, double> p = st1.GetTotalGradesSum();
    cout << p.first << "/" << p.second << "\n";

    StudentGradesInfoWrapper st2("S000129");
    st2.PrintAllCourses();
    st2.PrintAllCourses();
    st2.PrintAllCourses();

    cout << "Total Students " << StudentGradesInfoWrapper::GetTotalStudents() << "\n";
    cout << "Total Prints " << StudentGradesInfoWrapper::GetTotalPrints() << "\n";

    cout << "Bye\n";

    return 0;
}
```

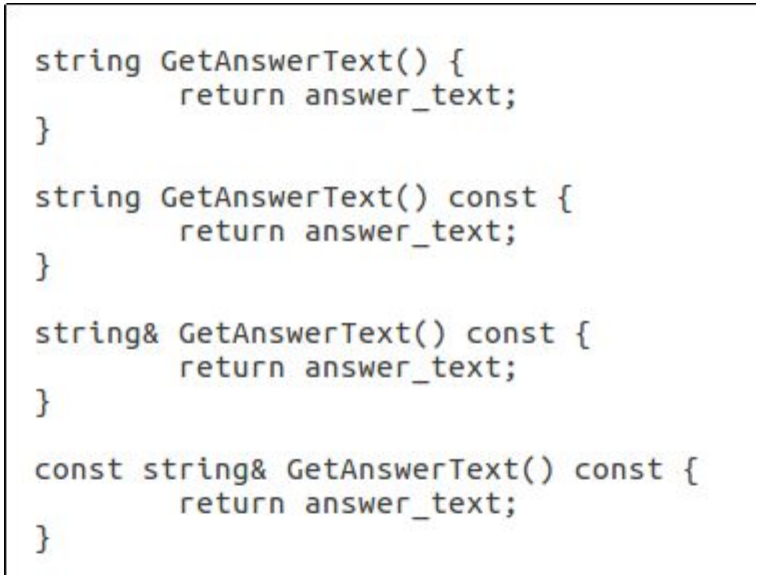


```
Grades for student: S000123
        Math = 70
        programming 1 = 70
        programming 2 = 85
225/300
Grades for student: S000129
Grades for student: S000129
Grades for student: S000129
Total Students 2
Total Prints 4
Bye
```

Homework 09: Future Features

- A fresh developer approached the team leader with the following suggestion
    - From an informal discussion with a customer, it seems after 6 months we will need:
        - Several printing styles & streams (file, console)
        - Maintaining statistics about every used function and providing getters for them
    - He suggests to implement these extensions now to save future time for other features
- As a leader
    - Do you accept? Or Reject? Or Suggest an alternative?
    - Why?

Homework 10: Returning objects



```
string GetAnswerText() {
        return answer_text;
}

string GetAnswerText() const {
        return answer_text;
}

string& GetAnswerText() const {
        return answer_text;
}

const string& GetAnswerText() const {
        return answer_text;
}
```

- We have a string and providing a getter for it
- You are debating with a fresh developer about the differences between these styles
- Discuss the differences

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."