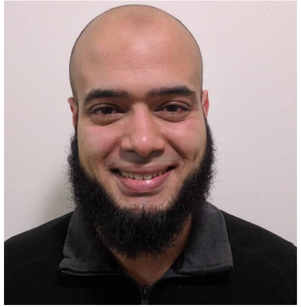
# C++ Programming

# Class Const, Static & Friend

# Homework 1

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

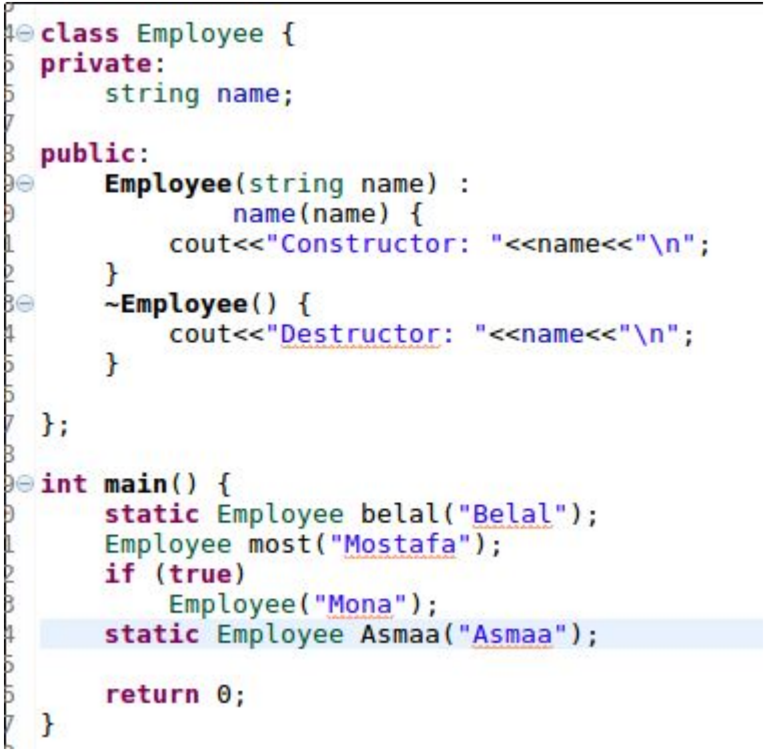
*Artificial Intelligence & Computer Vision Researcher*
*PhD* from Simon Fraser University - Canada
*Bachelor / Msc* from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



# Homework 01: Why!

- Why C++ doesn’t allow creating **const** static member function?
- Why C++ doesn’t allow using **this pointer** to access a static member data?

# Homework 02: Order



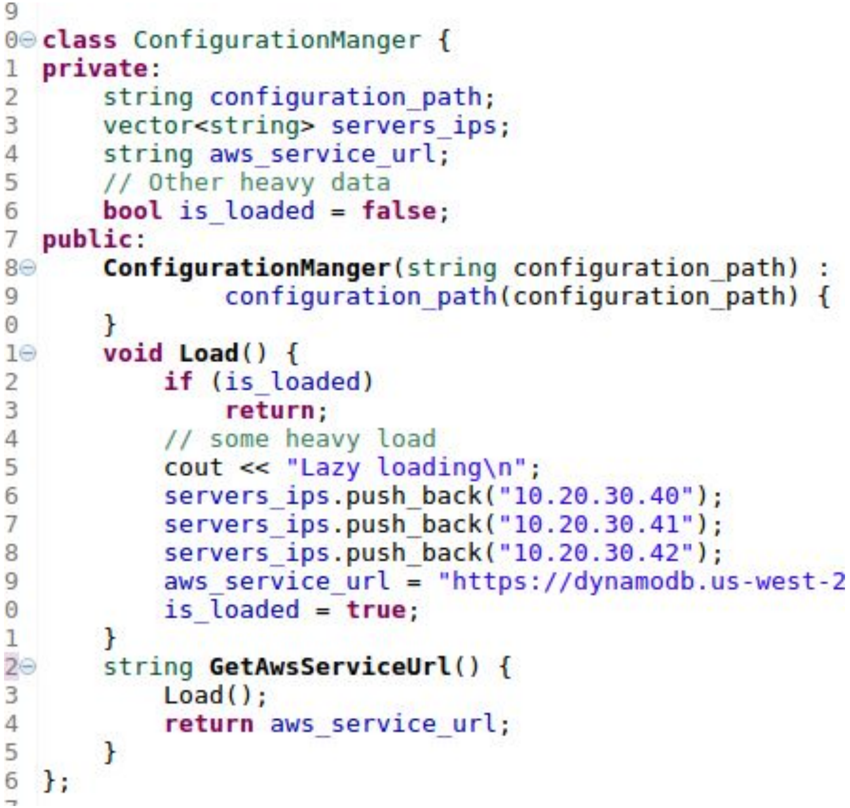
```
class Employee {
private:
    string name;

public:
    Employee(string name) :
            name(name) {
        cout<<"Constructor: "<<name<<"\n";
    }
    ~Employee() {
        cout<<"Destructor: "<<name<<"\n";
    }

};

int main() {
    static Employee belal("Belal");
    Employee most("Mostafa");
    if (true)
        Employee("Mona");
    static Employee Asmaa("Asmaa");

    return 0;
}
```

- What is the output of this program?
- Why?

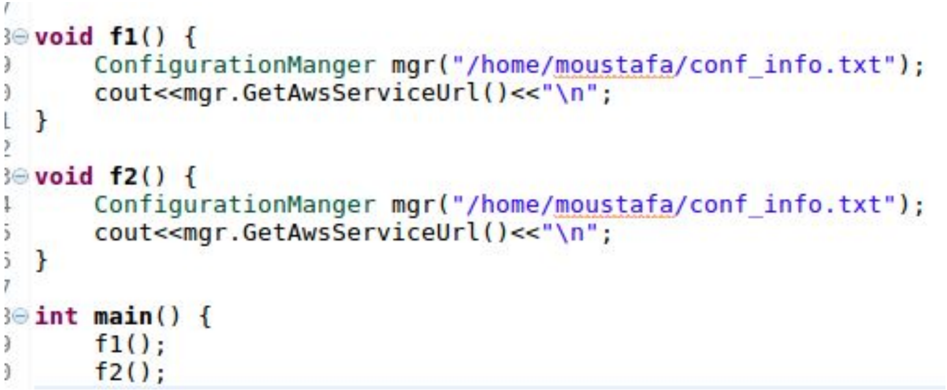
# Homework 03: The only one instance!

- Your team lead asked to develop a configuration manager component
  - This one loads system default values and critical information
  - E.g. Some servers IPs, Databases paths (& usernames & passwords)
  - E.g. also loads some **heavy** files and cache for frequent requests
- After you developed it, you noticed that several code parts create a seperate object
  - But is is actually loading the **same** things. It also **takes 30 minutes** to load data from disk!
- Your team lead suggested the following:
  - Change the class to allow ONLY **one shared** instance
    - No way to make 2 different objects in memory
  - This way the **class forces** any user to use the SAME object (the class itself!)
  - Change the next code to do such change

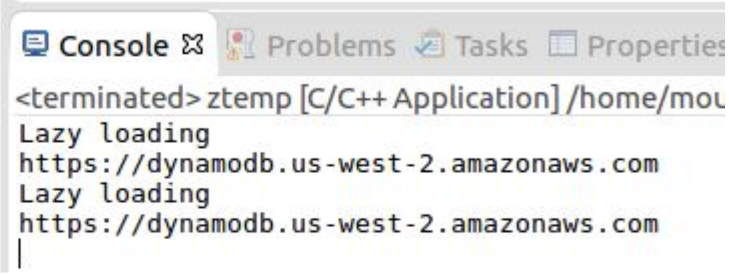
# Homework 03: The only one instance!



```cpp
class ConfigurationManger {
private:
    string configuration_path;
    vector<string> servers_ips;
    string aws_service_url;
    // Other heavy data
    bool is_loaded = false;
public:
    ConfigurationManger(string configuration_path) :
            configuration_path(configuration_path) {
    }
    void Load() {
        if (is_loaded)
            return;
        // some heavy load
        cout << "Lazy loading\n";
        servers_ips.push_back("10.20.30.40");
        servers_ips.push_back("10.20.30.41");
        servers_ips.push_back("10.20.30.42");
        aws_service_url = "https://dynamodb.us-west-2
        is_loaded = true;
    }
    string GetAwsServiceUrl() {
        Load();
        return aws_service_url;
    }
};
```



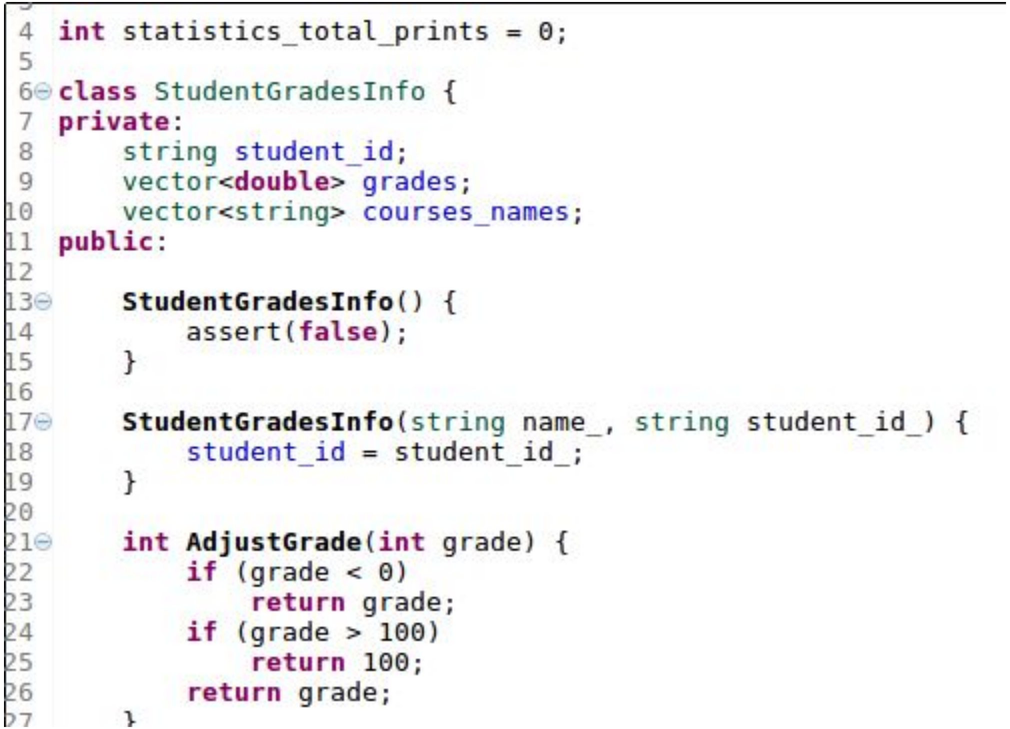
```cpp
void f1() {
    ConfigurationManger mgr("/home/moustafa/conf_info.txt");
    cout<<mgr.GetAwsServiceUrl()<<"\n";
}

void f2() {
    ConfigurationManger mgr("/home/moustafa/conf_info.txt");
    cout<<mgr.GetAwsServiceUrl()<<"\n";
}

int main() {
    f1();
    f2();
```



<terminated> ztemp [C/C++ Application] /home/mou

```
Lazy loading
https://dynamodb.us-west-2.amazonaws.com
Lazy loading
https://dynamodb.us-west-2.amazonaws.com
```

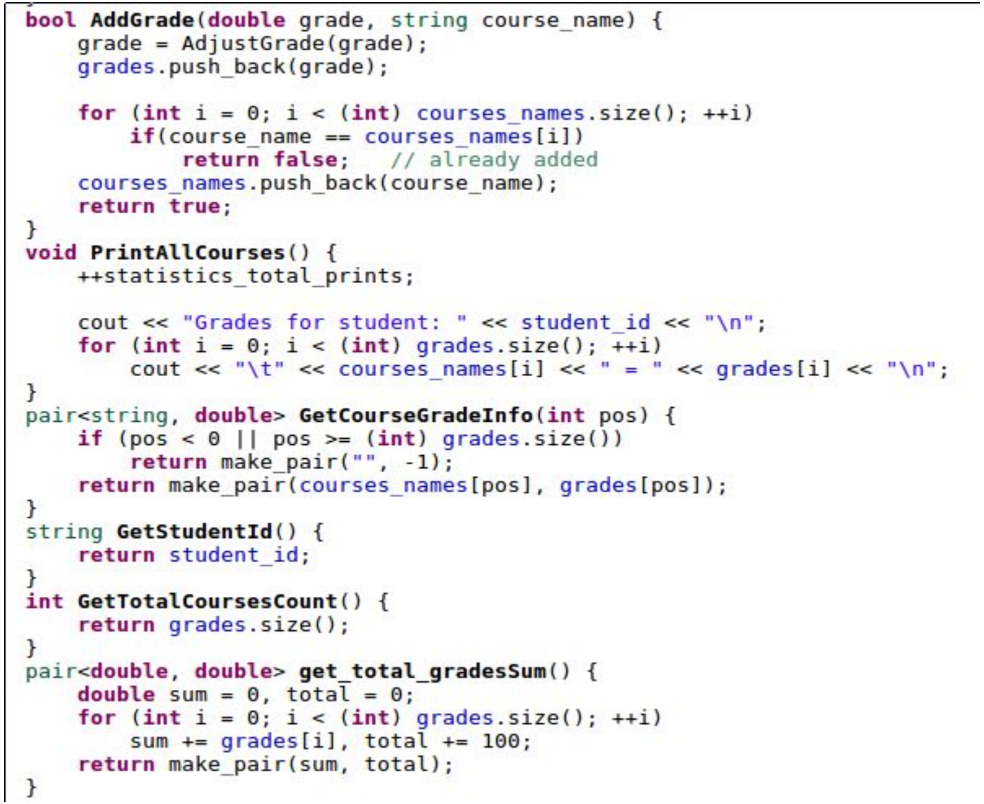
# Homework 04: Code Review



```
int statistics_total_prints = 0;

class StudentGradesInfo {
private:
    string student_id;
    vector<double> grades;
    vector<string> courses_names;
public:

    StudentGradesInfo() {
        assert(false);
    }

    StudentGradesInfo(string name_, string student_id_) {
        student_id = student_id_;
    }

    int AdjustGrade(int grade) {
        if (grade < 0)
            return grade;
        if (grade > 100)
            return 100;
        return grade;
    }
```

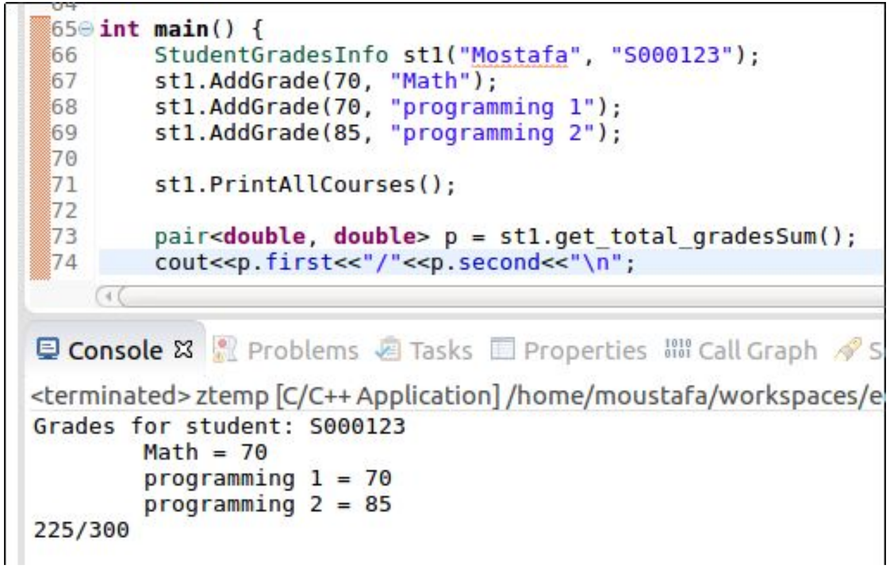
- Requirements
    - Class for a student and his grades per course
    - Grade max is 100 (e.g. 76.5/100)
    - Printing functionality or retrieve per course
    - Track how many times printing is called.
- Developer coded this/next

# Homework 04: Code Review



```
bool AddGrade(double grade, string course_name) {
    grade = AdjustGrade(grade);
    grades.push_back(grade);

    for (int i = 0; i < (int) courses_names.size(); ++i)
        if(course_name == courses_names[i])
            return false;   // already added
    courses_names.push_back(course_name);
    return true;
}
void PrintAllCourses() {
    ++statistics_total_prints;

    cout << "Grades for student: " << student_id << "\n";
    for (int i = 0; i < (int) grades.size(); ++i)
        cout << "\t" << courses_names[i] << " = " << grades[i] << "\n";
}
pair<string, double> GetCourseGradeInfo(int pos) {
    if (pos < 0 || pos >= (int) grades.size())
        return make_pair("", -1);
    return make_pair(courses_names[pos], grades[pos]);
}
string GetStudentId() {
    return student_id;
}
int GetTotalCoursesCount() {
    return grades.size();
}
pair<double, double> get_total_gradesSum() {
    double sum = 0, total = 0;
    for (int i = 0; i < (int) grades.size(); ++i)
        sum += grades[i], total += 100;
    return make_pair(sum, total);
}
```

# Homework 04: Code Review



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

    st1.PrintAllCourses();

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

```
<terminated> ztemp [C/C++ Application] /home/moustafa/workspaces/e
Grades for student: S000123
        Math = 70
        programming 1 = 70
        programming 2 = 85
225/300
```

- Figure out:
  - ○ 2 Coding bugs
  - ○ Several OO violations or design issues
  - ○ Something if we need to change, cause **changes** in several places
  - ○ Naming **inconsistency**
  - ○ 2 better **variables naming** in functions
- Suggest better declaration for
  - ○ pair<string, double> GetCourseGradeInfo(int pos)

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."