Write all classes and structures required so that the following main function:

```
int main()
{
    Catalog c = "math";
    ((c += {"Popescu", 10}) += {"Ionescu", 9}) += {"Popescu",8}; // Popescu is added
twice, only the first time

    printf("Catalog: %s (%d students ==> average: %.2f)\n", (const char *)c, (int)c,
(float)c);
    for (auto a : c)
        a.Print();

    if (c == "Ionescu")
        printf("Ionescu exists in catalog !\n");
    else
        printf("Ionescu DOES NOT exists in catalog !\n");

    if (c == "Georgescu")
        printf("Georgescu exists in catalog !\n");
    else
        printf("Georgescu DOES NU exists in catalog !\n");

    c += {"Marin", 8};
    c += {"Popovici", 7};
    c -= "Popescu";

    printf("Catalog: %s (%d students)\n", (const char *)c, (int)c);
    for (auto a : c)
        a.Print();

    return 0;
}
```

will print on the screen:

```
Catalog: math (2 students ==> average: 9.50)
Student: Popescu => Grade: 10
Student: Ionescu => Grade: 9
Ionescu exists in catalog !
Georgescu DOES NU exists in catalog !
Catalog: math (3 students)
Student: Ionescu => Grade: 9
Student: Marin => Grade: 8
Student: Popovici => Grade: 7
```

Read the next page for information about the how this paper will be graded:

Observations:

- You are not allowed to use STD in this exam
- You are not allowed to use string function (strcpy, strncpy, etc) in this exam

Points:

- 3 files (main, a header and a cpp implementation file for the classes and structures) ➔ 5 points
- A function to compare two strings (string = ***const char** ** type) ➔ 5 points (it is required to see if a student with a certain name exists in the catalog. It is used in the operator== implementation)
- "operator+=" ➔ 13 points
  - If "`((c += {"Popescu", 10}) += {"Ionescu", 9}) += {"Popescu", 8};`" works as it should (ads only the first two students and it correctly compiles written like this) ➔ 13 points
  - If "`((c += {"Popescu", 10}) += {"Ionescu", 9}) += {"Popescu", 8};`" compiles, but adds Popescu twice or uses the last grade for Popescu ➔ 7 points
  - If "`((c += {"Popescu", 10}) += {"Ionescu", 9}) += {"Popescu", 8};`" does not compile, however "`c += {"Marin", 8};`" does compile ➔ 5 points
  - If "`c += {"Marin", 8};`" does not compile ➔ 0 pints
- "operator-=" ➔ 10 points
- "operator==" ➔ 7 points
- operator to cast to ***int*** ➔ 1 point
- operator to cast to ***float*** ➔ 5 points
- operator to cast to ***const char ** ➔ 1 point
- constructor ➔ 2 points
- for-each support ➔ 3 points
- Print function defined in an internal object used to store information ➔ 3 points
- Code compiles and prints the expected output ➔ 5 points

Attention:

- Using STD or string function will be penalized in the following way: all methods / operators that depend on this will NOT be graded. If the code compiles and prints the expected output, but this is the result of using STD of string function only 2 pointes will be provided.