```cpp
#include <iostream>
#include <iterator>
#include <vector>
#include <algorithm>
#include <functional>

using namespace std;

struct Student {
  int id;
  string firstName;
  string lastName;

  Student() : id{}, firstName{}, lastName{} {}
  Student(int id, string first, string last) :
    id(id), firstName(first), lastName(last) {}
};

int main() {
  cout << std::boolalpha;

  vector<int> num1(100, 0);
  vector<int> num2(100, 0);

  bool e = equal(begin(num1), end(num1), begin(num2));
  cout << e << endl;

  e = equal(begin(num1), end(num1), begin(num2), end(num2));
  cout << e << endl;


  vector<Student> class1;
  vector<Student> class2;

  class1.push_back({0, "James", "Slocum"});
  class1.push_back({1, "Rodger", "Wright"});
  class1.push_back({2, "Jessica", "Alba"});
  class1.push_back({3, "Jessica", "Silva"});

  class2.push_back({0, "Frank", "Rogers"});
  class2.push_back({1, "Fred", "Rodgers"});
  class2.push_back({2, "Jamie", "Smith"});
  class2.push_back({3, "Jessica", "Silva"});

  function<bool(const Student&, const Student&)> id_only =
    [](const Student& s1, const Student& s2) {
      return s1.id == s2.id;
    };

  function<bool(const Student&, const Student&)> all_fields =
    [](const Student& s1, const Student& s2) {
      return s1.id == s2.id &&
        s1.firstName == s2.firstName &&
        s1.lastName == s2.lastName;
    };


  e = equal(begin(class1), end(class1), begin(class2), end(class2), id_only);
  cout << e << endl;

  e = equal(begin(class1), end(class1), begin(class2), end(class2), all_fields);
  cout << e << endl;

  return 0;
}
```