CS205 C/ C++ Programming - Lab Assignment 7

Name: 邱煜（Qiu Yu）

SID: 11611127

Part1 - Analysis

In this lab, we are asked to implement a template to determine whether a given type is incrementable according to if `++obj` and `obj++` work by using template metaprogramming technique.

Part2 - Code

template.hpp

```
//
//  template.hpp
//  lab7
//
//  Created by 邱煜 on 2019/5/26.
//  Copyright © 2019 邱煜. All rights reserved.
//

#ifndef template_hpp
#define template_hpp

// primary template
template <typename, typename = std::void_t<> >
struct is_incrementable : std::false_type{ };

// specialization
template <typename T>
struct is_incrementable<T,
std::void_t<decltype(std::declval<T&>()++),
decltype(++std::declval<T&>())>
> : std::true_type { };

#endif /* template_hpp */
```

Part 3 - Result & Verification

main.cpp

```cpp
//
//  main.cpp
//  lab7
//
//  Created by 邱煜 on 2019/5/26.
//  Copyright © 2019 邱煜. All rights reserved.
//

#include <iostream>
#include <vector>
#include <type_traits>
#include "template.hpp"

using namespace std;

int main(){
    cout << boolalpha << is_incrementable<int>() << endl;
    cout << boolalpha << is_incrementable<double>() << endl;
    cout << boolalpha << is_incrementable<float>() << endl;
    cout << boolalpha << is_incrementable<char>() << endl;
    cout << is_incrementable<std::string>() << endl;
    cout << is_incrementable<std::vector<int> >() << endl;

    return 0;
}
```

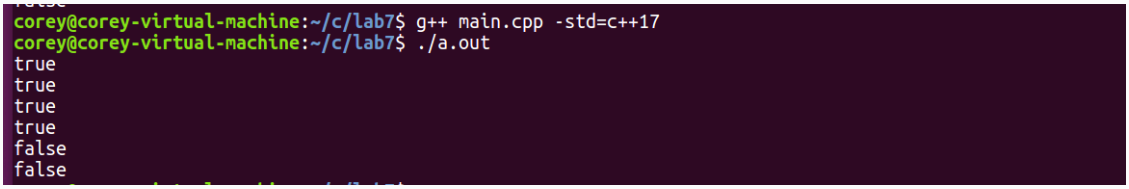
Using command `g++ main.cpp -std=c++17`

Expected Output

```
true
true
true
true
false
false
```

Real Output

```
true
true
true
true
false
false
```



H2 Part 4 - Difficulties & Solutions

1. This approach will trigger complier error when the **g++ version is <= 5.4**!

   `gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.10)`

   and clang 5.0

   But it works in the

   LLVM environment `Apple LLVM version 10.0.1 (clang-1001.0.46.4)` and in the

   g++ 7.4.0 `gcc version 7.4.0 (Ubuntu 7.4.0-1ubuntu1~16.04~ppa1)` and g++ 8.1

   And it should be in **c++ 17** !