National Research University Higher School of Economics
Bachelor's Program in Data Science and Business Analytics (DSBA)

Workshop 28 22.04.2021 – Groups 203-1 and 204-2
Exercises

Task 1. Templates, Enums, and Bitwise Operations

Code a template function called executeTask<T> with input arguments:
- a std::vector v with elements of data type T.
- an unsigned integer of 16 bits (uint16_t) indicating a *mode*.
- an variable called *x* of type T to insert and/or find in v.

The function executeTask will do *up to* 4 *actions* according to the bits of *mode*.

Operations

PRINT_SIZE	= 8	1000	prints the size of the vector v
PRINT_ELEMENTS	= 4	0100	print all elements of the vector v
INSERT_ELEMENT	= 2	0010	insert element *x* in the vector v
FIND_ELEMENT	= 1	0001	look for element x in the vector v

You'll read each four bits of the variable *mode* **from left to right** to see the next action to do.

EXAMPLE #1

```
std::vector<int> v1 = {100, 200, 300, 400, 500, 600, 700, 800};

uint16_t mode  = 0x2408                                  //          0010 0100 0000 1000

executeTasks<int>(v1, mode, 999);
```

Output:

```
INSERT_ELEMENT
PRINT_ELEMENTS
100 200 300 400 500 600 700 800 999
PRINT_SIZE
9
```

EXAMPLE #2

```
std::vector<std::string> v2 = {"Avocado", "Lemon", "Pepper", "Onion", "Lettuce"};

uint16_t mode = 0x0014;                                  //          0000 0000 0001 1000

executeTasks<std::string>(v2, mode, "Onion");
```

Output:

```
FIND_ELEMENT
found!
PRINT_ELEMENTS
Avocado Lemon Pepper Onion Lettuce
```

Task 2. “Function-like” Objects with overload of operator() + STL Algorithms

1) Create a class **CityDistanceChecker** that will compare if the distance of a city to Moscow is less than a double _d

The class has the following:

- a private attribute double _d
- a public constructor CityDistanceChecker(double d) that assigns the input d to attribute _d.
- overload of () with argument City c, and returns bool value distanceToMoscow(c) < _d.

2) Create a class **CityPrinter** that has:

- overload of () with argument City c. Inside, the function prints a the name of a city and its distance to Moscow as follows:

```
std::cout << c.name << "==>" << distanceToMoscow(c) << "kms" << std :: endl;
```

3) Inside the task() function:

- Create objects CityDistanceChecker and CityPrinter.
For example: CityDistanceChecker checker(2000.0); CityPrinter printer;

- Create a CityVector v2. Use function std::copy_if passing as argument the CityDistanceChecker object.

See: http://www.cplusplus.com/reference/algorithm/copy_if/

- Print elements of CityVector v2 using the function std::for_each passing as argument the CityPrinter object.

See: http://www.cplusplus.com/reference/algorithm/for_each/

EXAMPLE

```
void task(CityVector v)
{
        CityDistanceChecker checker(2000.0);
        CityPrinter printer;
        …
}
```

Output:

```
Moscow==>0kms
Istanbul==>1756.68kms
Ankara==>1794.3kms
Saint Petersburg==>634.99kms
Berlin==>1610.36kms
Kyiv==>755.762kms
Bursa==>1841.6kms
```

Baku==>1930.63kms
Minsk==>675.55kms
Bucharest==>1502.94kms
Vienna==>1669.76kms
Hamburg==>1781.33kms
Warsaw==>1150.09kms
Budapest==>1569.11kms
Diyarbakir==>1993.89kms
Munich==>1960.1kms
Yekaterinburg==>1417.18kms