Effective and Modern C++ Programming

Lab 11– Concepts & STL

Exercise 1. User defined literals

Define user literals for time (ms, s, h) and distance (m, km, cm) they should convert literal to seconds and meters correspondingly.

```
/**
    Computes velocity in meters per seconds.
    @param distance  distance in meters
    @param time      time in seconds
    @return velocity in meters per seconds.
*/
double computeVelocity(double distance, double time){
    return distance/time;
}

int main(){
    cout << computeVelocity(100_m, 5_s) << endl;     //20
    cout << computeVelocity(360_km, 2.0_h) << endl;  //50
    cout << computeVelocity(3.6_km, 0.02_h) << endl; //50
    cout << computeVelocity(250_cm, 2.5_ms) << endl; //1000
    return 0;
}
```

Literature:
https://en.cppreference.com/w/cpp/language/user_literal

Exercise 2. Filesystem

Implement functions

```
/**
 * Prints content of directory given by path
 * Format
 *  [X]  file_name    file_size
 * where X equals D for directories, F for regular files, L for symlinks and space otherwise.
 * @param path directory path
 */
void printDirectory (std::string_view  path);


/**
 * Makes copies of all files matching fileNames regular expression in directory given by path
 * to files in the same directory but with changes extension to newExtension
 * @param path  directory path
 * @param fileNames  regular expression
 * @param newExtension new extension
 */
void changeExtension(fs::path path, std::string fileNames, std::string_view newExtension );
```

Exercise 3. Concepts

Implement concept Container, that describes containers that can be iterated using range base for loop, has member type value_type and its elements can be added using operator+.
Implement methods print(object) that prints information about object using member method print (i.e. object.print()) is object has one, otherwise use operator<<.

If both method and operator are present, prefer operator. Define appropriate concepts that check if obect has method print or operator<<.
If object is a Container, then print all its elements (using appropriate print method).

```
int main() {
  vector v{1,2,4,5};
  print(v);
  A<int> a{5};
  print(a);
  B<double> b{3.14};
  print(b);
  print(2.7);
  vector<A<int>> va{ 4, 5, 7, 9};
  vector<B<int>> vb{ 4, 5, 7, 9};
  print(va);
  print(vb);
  print( sum(v) );
  print( sum(vb) );
  return 0;
}
/**
* Expected output
0 : 1
1 : 2
2 : 4
3 : 5
-------
[5]
#3.14#
2.7
0 : [4]
1 : [5]
2 : [7]
3 : [9]
-------
0 : #4#
1 : #5#
2 : #7#
3 : #9#
-------
12
#25#
*/
```