Effective and Modern C++ Programming

Lab 10 – SFINAE

Exercise 1. SFINAE, enable_if & if constexpr

Implement template classes that use SFINAE to detect during compilation:

- **hasSize<T>** - if given type T has method size()
- **hasValueType<T>** - if given type T has member type value_type

```
cout << hasSize< int >::value << endl;  // false
cout << hasSize< vector<int> >::value << endl //true
cout << hasValueType< int >::value << endl;  // false
cout << hasValueType< vector<int> >::value << endl //true
```

Implement template function
size_t getSize(const T & x)
that:

- returns x.size() * sizeof(T::value_type) if T has method size and member type value_type,
- sizeof(x) otherwise.

Make two versions in separate namespaces to implement getSize function:

- v1 – use enable_if,
- v2 – use **if constexpr**

```
std::vector<int> v{1,2,3,4,5};
cout << v1::getSize(5) << endl;  // 4
cout << v1::getSize(v) << endl; // 20
cout << v2::getSize(5) << endl;  // 4
cout << v2::getSize(v) << endl; // 20
```

Exercise 2. Tag dispatching

Implement method
double median(Container set)

that finds **median** in given **set**. Container can be one of standard containers (list, forward_list, vector, deque).
Use iterator tags and tag dispatching to implement two versions one for random access containers (vector, deque) and second for general container with forward iterators (list, forward_list).

```
std::list<int> a{3, 2, 5, 1, 4};
cout << median(a) << endl; // 3
std::vector<int> v{3, 1, 4, 2};
cout << median(v) << endl; // 2.5
```

Exercise 3. Chrono timer

Implement class **Timer** that measures life span (time from creation to destruction) of its elements. On destruction it should print timer name and life span in seconds.
Provide also method duratioInNanoseconds() which will return life span of an object in nanoseconds (from construction to current moment).

Use: library std::chrono, steady_clock.