EXERCISES UNIT 4

1. Write a function template *sizeof_bits* for calculating the size of any variable in number of bits.

2. Write a function template for showing a vector of the STL.

3. Write a function template for showing any container of the STL. The function should receive as input parameters the *begin iterator* and *end iterator* of the container to show.

4. Write the class *Array2*, similar to *array* of the STL. It should have a nontype template parameter for the number of elements. Write the overloading of operator []. This operator should not be declared inline (inside the class). This operator will have to throw an *out_of_range* exception when appropriate.

5. Write a function template *resta()* that allows the substraction of 2 numbers of any numeric types. The 2 input parameter types can be different. Prevent the call to this function with a non-numeric type by using *static asserts*.

6. Write a function template for creating a vector of random integer numbers in the range given by the parameters *begin* and *end*. The template parameter is the type of the integer number. Prevent the call to this function with a type different from an integer by using *static asserts*.

7. Write a function template for creating a container of random numbers of *int* type in the range given by the parameters *begin* and *end*. The template parameter is the type of the container.

8. Write a variadic function template *num_param()* that returns the number of arguments passed to the function. You should not use the operator *sizeof…()*.

9. Write a variadic function template *suma()* that returns the addition of all the arguments passed to the function. All the arguments have to be of the same type.

10. Find the maximum value that can be stored in an **int**, a **long** and a **long long** using the library <**numeric_limits**>.