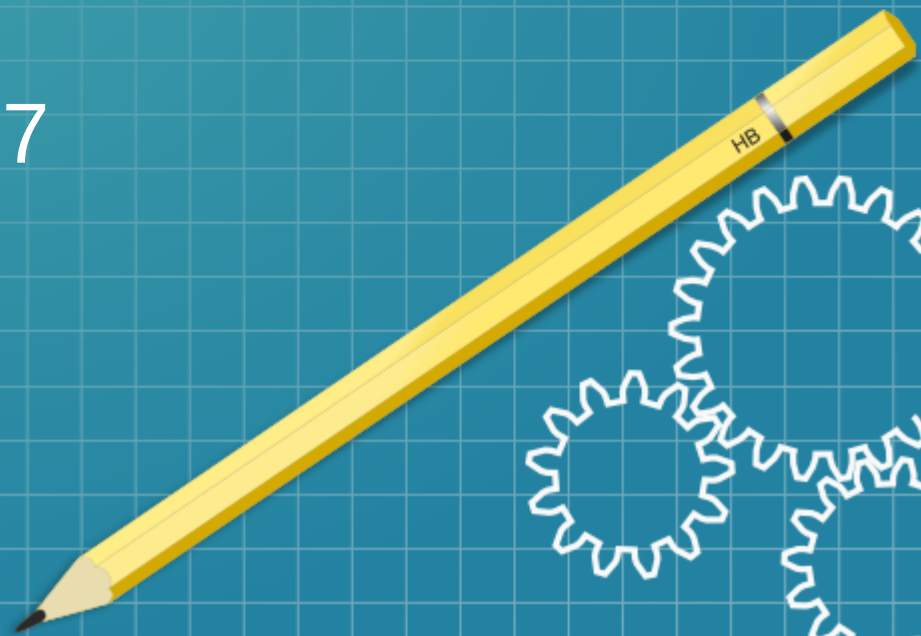
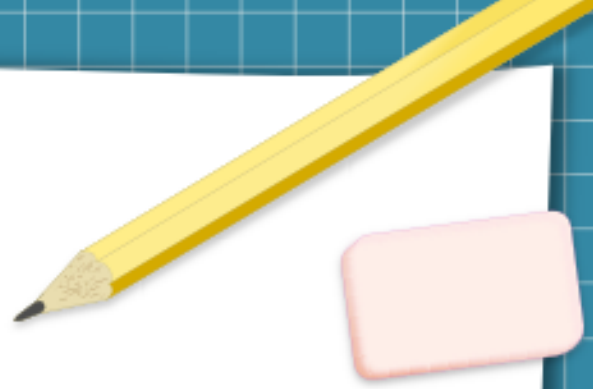
Class Templates `array` and `vector`; Catching Exceptions

Chapter 7



We will discuss:



- arrays
- vectors

Data Structures

Today we will discuss two *data structures*, i.e. collections of related data items: *arrays* and *vectors*.

Arrays are fixed-size collections of the data items of the same type.

Vectors are variable-size collections of the data items of the same type. The can grow and shrink dynamically at execution time.

Both, *array* and *vector* are C++ standard library *class templates*.

To use them we must include `<array>` or `<vector>` headers.

Arrays

An *array* is a series of elements of the same type placed in contiguous memory locations that can be individually referenced by adding an index to a unique identifier.

C++ has two types of arrays:
- *arrays* inherited from C language
- *container arrays* (*template* type)

`static local arrays` work in a similar way as `static local variables`, and when created all its elements are initialized to 0.

See arraysExample.cpp and arraysExample2.cpp

Visit http://www.cplusplus.com/doc/tutorial/arrays/ and https://en.cppreference.com/w/cpp/container/array for more

Arrays

Arrays can be passed as parameter, by value, to a function call.

- All modifications applied to C arrays inside the function, are affecting the original array.
- When C++ template/container array is passed to a function call by value, a copy is created and the original array is not affected by any changes inside the function.

See arraysExample3.cpp

Sorting and Searching Arrays

We can use the built-in C++ standard library

- `sort` function to sort the elements of the array in increasing order
- `binary_search` function to determine whether a value is in the sorted array

See arraysExample4.cpp

See https://www.cplusplus.com/articles/NhA0RXSz/ and https://en.cppreference.com/w/cpp/algorithm/binary_search for more information

Vectors

C++ standard library (STL)class template `vector` is similar to the class template `array`, but it supports *dynamic resizing*.

Don’t forget to include `<vector>` header.

See vect1.cpp , vect2.cpp and vect3.cpp

Visit http://www.cplusplus.com/reference/vector/vector/ for more

In-class work

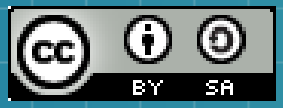
1. Write a program that will collect the ages of a village citizens from the keyboard input and output how many people of each age live in the village.
Do not display information if there are no people of that age.

2. (exercise 7.31) (***Print a String Backwards***) Write a recursive function `stringReverse` that takes a `string` and a *starting subscript* as arguments, prints the string backwards and returns nothing. The function should stop processing and return when the end of the string is encountered. Note that like an `array` the square brackets (`[]`) operator can be used to iterate through the characters in the `string`.

HW assignment

1) Exercise 7.13
3) Exercise 7.14
4) Exercise 7.28

Suggested exercises
(*not for grade, but the questions related to these will appear on a quiz or a test*):
1) Chapter 7,
   Summary and all Self-Review Exercises
2) Chapter 7, Exercises: 7.6 - 7.9, 7.11, 7.12, 7.17, 7.20



This work is licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License.
It makes use of the works of Mateus Machado Luna.