C++ Lab2: Conditionals

- *The source code needed for this lab is available on Moodle.*
- *A list of useful (English) programming vocabulary is given at the end of this lab.*

Objectives

1. Test your programs
2. Reasoning with conditionals
3. Respect the syntax and formatting rules

General instructions

- Read this lab description carefully before you start coding
- Each exercise has to be fully tested before you go on to the next one. Run a set of test cases that is as complete as possible.
- Add comments at the end of your programs describing the tests you used. One line per test: what values? For what purpose? Expected result?
- Use the C++ class notes for details concerning C++ syntax.

WORK TO HAND IN: Exercise 4, source code and test cases.

Exercise 1: putting things into order

Write a program that asks the user to enter two real numbers and that displays "increasing order" if the first is smaller or equal to the second, and "decreasing order" if the second is smaller than the first. Test all possible cases (ex: 1 and 5, 5 and 1, 5 and 5) and add them as comments at the end of your program.

Exercise 2: hoping for nice weather

Write a program that asks the user to input to enter a temperature and that displays "It's too hot!" if it is greater than 25, "It's too cold!", if it is smaller than 15, and "This is good!" if it ranges between 15 and 25. Start by writing a set of test cases, then write the program, adding your test cases as comments at the end of the file.

Exercise 3: seeking for extremums

Write a program that asks the user to input 3 numbers (**x**, **y** and **z**) and that displays the largest and the smallest. Use the variables shown below.

1. Write the set of test cases.
2. Write the program to display the max. Check your code with the test cases.
3. Add the statements to display the min. Check your code with the test cases.

```
int main()
{
     double min, max;
     double x, y, z;
     …
}
```

Exercise 4: Time for a show!

Upload to Moodle your commented program, along with a set of test cases

A concert hall uses the following chart to determine the price of a concert on the basis of age, subscription status, and type of show. We need a program that asks the user for the age of a person, their status and the type of show that is desired, and then displays the price of the corresponding ticket.

You need to follow these constraints:

- To be able to easily update concert prices, these will be stored in constants.
- Use a variable `price` to display the result a single time at the end of your program.
- Use embedded conditionals when possible to avoid having to test the same condition multiple times. In order to do so, test one condition at a time.

Follow these guidelines:

1. Write the code for asking for the type of the show, the age of the viewer, and whether they are a subscriber or not. For the type of show, declare a variable `typeShow` of type char. Tell the user to enter 's' or 'l' (for short or long). No checking, but change to lowercase (`typeShow = tolower(typeShow)` before testing. For subscriber, use a variable of type char named `subscriber` ('y' or 'n', for yes or no).
2. Add statements to compute and display the price without considering the show type. Test the different possible cases.
3. Add statements to consider the show type. Use a switch. Test all cases.

	Short show		Long show	
	Subscriber rate	**Normal rate**	**Subscriber rate**	**Normal rate**
Child (<18)	**6**	**8**	**10**	**12**
Adult	**15**	**18**	**18**	**22**

Exercise 5 (optional, for training)

Using the **`switch`** statement, write a program that asks the user to input a character ('s' for summer, 'f' for fall, 'w' for winter, and 'p' for spring), then displays the name of the corresponding season. An error message should be displayed if the character does not correspond to a season.

Exercise 6 (optional, for those who are fast)

In this exercise we consider "abbreviated ordinals", written with one or more digits followed by a suffix. For instance, "1st", "2nd", "3rd", "4th", etc., abbreviations of first, second, third, fourth. To determine the suffix, you need to look at the last digit of the number: if it's 1, the suffix is –st (321st for instance), if it's 2, the suffix is –nd (192nd), if it's 3, the suffix is –rd (53rd); in all other cases the suffix is –th (24th). There is one exception: if the next to last digit is 1, the suffix is always –th (213th).

Write a program that inputs a number, and displays the corresponding abbreviated ordinal.

Hint: the last digit of a number ***nb*** is nb%10 (and ***nb*** modulo ***n*** is the remainder of the integer division of ***nb*** by ***n***);

Programming Vocabulary Lab2

- assignment x=5 ; *affectation*
  - The assignment operator assigns a value to a variable. Here it assigns 5 to x.
  - We can also say that it sets the variable x to 2.
- conditionals and loops *conditionnelles et boucles*
  - conditionals and loops are two types of control structures :
    - conditionals execute certain statements if certain conditions are met
    - loops execute certain statement while certain conditions are met.

- a condition is an expression whose value is being tested, for instance : x>y. It uses two kinds of operators: relational (>= for instance) and logical (&& for instance). It can resolve to a value of true (we then say that the condition is met) or a value of false (the condition is not met)
- if-else statements : if the condition is met, the block corresponding to the ***if*** is executed, otherwise, the block corresponding to the ***else*** is executed.
- if-else if : is used to decide between two or more blocks based on multiple conditions

- nested control structures — *structures de contrôle emboitées*
  - it is possible to place ifs inside of ifs and loops inside of loops. They are referred to as nested statements.
- a set of test cases — *un jeu d'essais*
  - A set of input data, and of situations of use, applied to a program in order to test it. The goal is to provide input that will allow to check all possible cases, and to verify that the output in each one of those cases corresponds to what was expected.
- the modulo operation — *opération modulo*
  - Given two positive numbers, *a* (the dividend) and *n* (the divisor), *a* modulo *n* is the remainder of the division of *a* by *n*.
- integers and real numbers — *nombres entiers et réels*
  - An integer (also known as a "whole number"), is a number that can be written without a fractional component.
- even and odd numbers — *nombres pairs et impairs*
  - A number is even if it leaves no remainder when divided by 2, and odd if it is not even.
  - A number is even or odd according to whether its last digit is even or odd. That is, if the last digit is 0, 2, 4, 6 or 8, it is even; otherwise it is odd.
- punctuation

full stop, period	.
question mark	?
exclamation mark	!
colon	:
semicolon	;
comma	,
hyphen	-
dash	–
ellipsis	…
apostrophe	’
quotation marks :	
- double quotes	" "
- single quotes	' '
parentheses	( )
brackets :	
- square brackets	[ ]
- curly brackets	{ }
- angle brackets	⟨ ⟩

- general typography

ampersand	&
at sign	@
asterisk	*
slash	/
backslash	\
bullet	•
caret	^
tilde	~