C/C++ Coding Standards for Webots Controller

- General Format
  - ✓ Line Width : No line should be longer than 80 columns.
  - ✓ Functions : A function must perform a *single* task and in all but the most exceptional circumstances must not be longer than 30 lines.
  - ✓ File Size : A file should generally not exceed 1000 lines.
- Naming Conventions
  - ✓ Functions and Variables :All functions and variables must have descriptive, meaningful names.

    Ex) name for a function that displays a window on the screen

    “showWindow”(0) “shwd”(x) “sily”(x)
  - ✓ Constants : Use #define to give a meaningful name to any constant literal value that might conceivably change with future modifications of the program (such as the maximum length of a string or list) and to any constant literal value whose role will become clearer through such a name (such as PI).
  - ✓ Use of Case : Use lower case (combined with upper case for multi-word names) for function and variable names, beginning with a lower case letter.

    Ex) isEmpty, bigNumber, counter …. etc
  - ✓ Special Characters in Identifiers : The only "special" character (that is, a character that is neither a letter or a digit) that is legal in identifiers is the underscore character.

    Ex) is_allowed_character
  - ✓ CommentsExternal Documentation : External documentation consists of a top-down design and an overview of the specifications and function of the entire program, along with a description of the interface between files in a multi-file program.
  - ✓ Internal Documentation
    - File header comments

    At the beginning of each file, a comment block of the following form must appear:

```
/*********************************************
*Programmer:                                  *
*Program #:                                   *
*Lab Section:                                 *
*Date:                                        *
*Description:                                 *
*********************************************/
```

- Line comments

  These comments explain a single statement of the source code. They appear in the right margin of the code and follow the statement they explain. Your code will be more readable if you line them up. For example:

```
sum = 0;                        /* initialize the sum */
 while (sum < 100) {
    cout<< "Enter a positive whole number:”<<endl;
    cin>> i;                    /* read a user-entered */
                                /* number to add to the sum */
   sum += i;
}
```

- Structuring and Indentation

The overall structure of the program must be organized as follows

① Program header comment section

② # define statements

③ # include statements

④ Type definitions

⑤ Functions organized in a logical manner

⑥ The main function

- Forbidden Practices

✓ Do not use global variables unless the variable is used in virtually every function in the file

✓ Avoid using GOTO statements, but it can be used when it does not harm the control flow

Example

```
rc = Do_task_1( );
if (rc != 0)
   goto error_task_1;
rc = Do_task_2( );
if (rc != 0)
   goto error_task_2;
rc = Do_task_3( );
if (rc != 0)
   goto error_task_3;
return 0;

error_task_3:
undo_task_2( );
error_task_2:
undo_task_1( );
error_task_1:
return rc;
```