Checks for coding standards

1. Indentation using 1 tab/indentation

```
function foo()
{
    if ($maybe) {
        do_it_now();
        again();
    } else {
        abort_mission();
    }
    finalize();
}
```

2. Don't put multiple statements on a single line
3. Don't leave whitespace at the end of lines.
4. switch to have a default case with break.
5. Limit line length to 80 characters:
6. Put the opening brace at the end of the line for all non-function statement blocks (if, switch, for, while, do). E.g:

```
   if (x == true) {
         …

   switch (action) {
   case KOBJ_ADD:
         …
   }

   while (ptr) {
     …
```

7. Do not use a brace if there is only one statement as part of the if statement or the loop.
8. In case of functions the open curly brace will be on the new line as shown below.

```
int function(int x)
{
   body of function
}
```

9. Closing brace is on a line of its own, except in the cases where it is followed by a continuation of the same statement, ie a ``while`` in a do-statement or an ``else`` in an if-statement, like this:

```
do {
      //body of do-loop
} while (condition);
```

```
if (x == y) {
..
} else if (x > y) {
...
  else {
....
}
```

10. Spacing rules:
    a. Add space after the following keywords: if, switch, case, for, do, while…
    b. Add spaces before and after the parenthesis “(“, “)”, but not inside it. e,g
        `if (x == y) {`
        should not be written as:
        `if ( x == y ) {`
        or as this:
        `if(x == y){`
    c. Add spaces on either side of most of the binary and teriary operators such as: `=   +   -   <   >   *   /   %   |   &   ^   <=   >=   ==   !=   ?   :`
    d. Do not add spaces with unary operators like: `&   *   +   -   ~   !   sizeof   typeof   alignof   __attribute__   defined ++ --`
    e. Do not add spaces with structure member operators like: `.  ->`
    f. When declaring pointer data or a function that returns a pointer type, the preferred use of ``*`` is adjacent to the data name or function name and not adjacent to the type name.
    Examples:

```
char *linux_banner;
unsigned long long memparse(char *ptr, char **retptr);
char *match_strdup(substring_t *s);
```

11. All local variables must be initialized to the extent possible.
12. All global variables that won't be accessed from any other files must be declared as static
13. All C header files must be protected against multiple inclusions by using the `#pragma once` pre-processor directive.
    Example:

```
#pragma once
…
```