C/ C++ Programming Style 101

To have a readable program:

- Every program must have a header as follows:

```
// Name: Sally Brown
// Date: August 30, 1996
// Last Update: January 8, 1997
// Problem statement: This C++ program computes the volume of Snoopy's house.
```

- Use blank lines to separate logical sections.
- Use spaces around '=' and around operators and after commas and semicolons.
  For example:
  ```
   int weight, height;
   weight = 3.0 + height * 2.7;
  ```

- Use comments to describe major sections of program or where something needs to be clarified.
  For example:
  ```
  …
  int main()
  {
          …
          // reading records from the input file
          ReadRecFromFile(…);
  }
  ```
- For names of objects (variables) you will use lower case letters and capitalize the first letter of the second and succeeding words.
  For example:
  ```
  int noOfElement;
  float realPartNumber;
  ```

- For constants (including enumeration values), the identifier should be all capital letters (uppercase) using underscore to separate words.
  For example:
  ```
  const float PI = 3.14159;
  float letterGrade;
  cons tint COLOR_RED= 25;
  ```

- Names representing methods or functions must be verbs and written in mixed case starting with lower case.
  For example:
  getName(), computeTotalWidth()

- Names representing namespaces should be all lowercase.
  For example:
  model::analyzer, io::iomanager, common::math::geometry

- Names representing template types should be a single uppercase letter.
  template<class T> ...
  template<class C, class D> ...

- The names of classes should start with an upper case letter.
  ```
  class Fun
  {
      // stuff to define class
  };
  ```

- Use descriptive object and class names which relate the program to the problem.
  For example
  // to compute average for int numbers
  AverageInts(…);
  // to sorting strings by lexical order
  SortByLexicalOrder(…);

- A class should be declared in a header file and defined in a source file where the name of the files match the name of the class.
  MyClass.h, MyClass.cpp

- Indent if, for and do-while as shown:

```
if( weight > 200)
{
    cout << "Too Big!\n";
}
else
{
```

```
    cout << "Ok.\n";
}


for(i = 1; i <=   n; i ++)
{
    s =   s + i;
    cout << "     Don't Panic!\n";
}


do
{
    i = i - 1;
} while (i > 0);
```

- All functions must have a series of comments which state the intent and the pre and post conditions. A pre-condition is a sentence or two which states what must be true before the function is called. The post-condition states what is true after the function is called.

  For example:

```
    // Intent: To sum the positive integers from 1 to n.
    // Pre:        The variable n must have a value and n > 0.
    // Post:      The function returns the sum from 1 to n.
    int Sum(int n)
    {
        // code for Sum
    }
```

- Header files must contain an include guard

```
#ifndef COM_COMPANY_MODULE_CLASSNAME_H
#define COM_COMPANY_MODULE_CLASSNAME_H
:
#endif // COM_COMPANY_MODULE_CLASSNAME_H
```

References

- https://google.github.io/styleguide/cppguide.html