**Coding Styles Guide**

**Header Files**

Header files within the program will be self contained and always end in .h and will be stored at thetop of the page utilising #include “xyz”

**#Include libraries**

#Include <xyz> is used to call specific classes and libraries for use within that page of code. Include will be held underneath header declarations to ensure ease of access if issues arise due to missing libraries

**Namespaces**

Namespaces are used to avoid name conflicts within large projects and where possible

Using the term “using namespace x” will allow our code to remain clean and readable as well as automatically define names if appropriate . This statement will be placed just after the #include declarations

**Camel case**

For the sake of consistency are variable naming style will be the use of camel case

exampleVariable

**Tabbing Indentation**

A consistent tabbing system will should be enforced this is for clarity in coding and consistency it allows for easy readability for instance

A single tab key press is required per indent

Public Classname()

{

Start line here  
}

**Variable and Array Initialization**

You are permitted to use your personal preference of (),{} or =

String x = “hello”

String y(“hello”)

String z{“hello”}

**Naming**

Variables will be named consistently to allow for clarity and easy readability when coding. Variables will be named in correspondence with the classes they are contained and used within.

Classes will be named consistently to what they will contain or what function they will process and works in tandem with the variables

For examples for the class name and variable name for images:

Public image()

{

imageSize

imageShape

}

File names will utilise a similar clear structure being named appropriately and using \_ for space to ensure implementation within code.

**Braces**

Braces or curly brackets are essential for containing functions and code. The code will utilise a returned style to break up code as much as possible to ensure clear form throughout

Test Function()

{

Qwerty

qwertz

}

**Commenting**

Utilising // structure, Commenting will be constant within our code to ensure that any further contributing users can clearly understand our code as to better integrate themselves with the product without needless testing. Comments will be ended after every function or statement clearly describing the instance. Punctuation should be maintained and professional to ensure easy readability.

**Line Length**

Any line of code or comments must not exceed 80 characters long to ensure the lines can be read on majority of working layouts

Exclusions of this rule include

-Headers

-include statements

-comment that would harm the readability of the comment

**Function Call**

Function calls will be made on a single line unless exceed line length then an indented (Tab) new line can be utilised. But try to actively avoid excessive line use.

Function (Argument1, Argument2, Argument,3

Argument4)

**Operators**

Operators will always have spaces around them to ensure easy readability

int x = 4

Int y = 4 + 6 - 2