**Introduction**

This document provides the coding convention used in our project. It will work as a standard according to which all the programming in python for the purpose of this project will be done. There are a couple of motivations for this standardization: firstly, it will help the group members to be in sync; secondly, it will be easier for someone outside the project group to understand and possibly modify the work done here.

**Code Layout**

**Indentation**

As a standard python convention, 4 spaces should be used for indentation where required. In the case of large function names and lines within a function, indentation should clearly identify the continuation lines. That is the structure of indentation should be able to distinguish between continued and discontinued lines. Tabs should be used for all the indentation required and automated by python. Spaces should be used for the indentation done for the continued lines.

**Line Length**

All lines should be less than 99 characters. In case a line reaches this upper limit, it should be broken into multiple lines. This should be done through either parentheses or backslashes. If a binary operation is involved in the breaking of lines, the operation should be the first character of the continued line.

**Blank Lines**

Top-level functions and class definitions should be surrounded by at least two blank lines. Method definitions inside a class should be surrounded by at least a single blank line. Blank lines may be used within functions in order to distinguish between logical and related groups in functions.

**Code Structure**

**Imports**

Imports should be always at the top of a file. Each line of import should correspond to a single class. Absolute imports (import <module>) are recommended. Though, relative imports (from <module> import <function>) are acceptable. Wildcard imports (from <module> import \*) should be avoided. Imports should be grouped together as following: absolute imports followed by relative imports followed by library-specific imports (from <module.function> import <function>).

**String Quotes**

Double quotes should be used for representing strings. If the string itself contains double quotes, single quotes should be used.

**Whitespaces**

Conventions from English language should be used here. Whitespaces should be avoided immediately after inside parentheses, between a comma and close parentheses, immediately before a comma or colon, immediately before the opening parentheses of an argument or slicing. Statements should be aligned at the start of the line. Trailing whitespaces should be avoided.. Binary operators of assignment, augmented assignment, comparisons, and Booleans should be surrounded by a single space on either side. In the cases where operators with different priorities are used, surround the ones with whitespaces which have the lowest priorities. Multiple statements on a single line should be avoided.

**Comments**

**General Guidelines**

Use English for writing comments. Comments should be complete phrases. The first work of comments should be capitalized unless it is an identifier name. Comments containing complete sentences should have periods after the end of each sentence.

**Block Comments**

Block comments should start with a # followed by a single space. It should contain information about the code to be followed. It should be indented to the same level as the code to be followed.

**Inline Comments**

Inline comments should start with a # followed by a single space. It should be on the same line as the code statement and should follow the code. It should be at least 2 spaces further from the end of the statement. It should be used as sparingly as possible and only used in the cases where it is necessary.

**Documentation Strings**

Documentation Strings should start and end with “““, where the ending quotes should be on a new line. It is highly recommended for all the public modules, functions, classes and methods. It should describe what the corresponding entity does.

**Naming Conventions**

**Naming Styles**

Variable names should use the style *mixedCase*. That is, variable names should start with a lowercase letter and whenever there is a change in word, an uppercase letter should be used to indicate it. Class names should use the style CapitalizedWords. That is, class names should start with a uppercase letter and and uppercase letter should be used to indicate the change in words. Function and method names should be in mixedCase. CapitalizedWords should also be used as exception names. In case of name clashing, different styles may be used. Other styles may be used if they are consistent with the particular modules or classes used.

**Prescriptions**

Single character names, especially “O” and “I”, should not be used. The use of global variables should be avoided except for defining constants. Use “self” as the first argument of instance methods.