**RISKS ANALYSIS**

**Group Members:**

Chan, Charmaine Kaye

Gutierrez, Vernon

Macam, Francis

Tan, Mark Joshua

**Risks Involved (1 being the most critical):**

1. Being unable to finish
2. Faulty design of the software
3. Errors in the implementation
4. Lack of features
5. Miscalculation of formula
6. Loss of unsaved data or code
7. Interface is not user-friendly
8. Overestimation/Underestimation of Time
9. Inability to run in slower computers
10. Conflicts among group mates
11. Sleeping/laziness
12. Brownout while saving
13. Computer crash
14. Destroyed hard drive
15. Random school events
16. Crappy and buggy downloaded code
17. Natural calamities
18. No common time to meet up
19. Slow program
20. Lacking public approval
21. Inability to work together
22. Absences/tardiness in meetings
23. Not matching the mood to the melody
24. Not being able to find the "right" formula
25. Errors in the algorithm
26. Non-functioning GUI
27. Inaccurate documentation
28. Not feasible
29. Readings/distractions
30. Lacking requirements
31. A member meets an accident
32. Lack of equipment/facilities
33. Messy codes/Hard to read codes
34. Miscommunication among team members
35. Change of requirement
36. Inability to support other software
37. Incompatible with other OS
38. Procrastination of members
39. Overestimation/Underestimation of Cost
40. Debugging Errors
41. Code could not be understood by others
42. Software crash
43. Inaccuracy of the GUI
44. Problems linking the front-end to back-end
45. Error in programming logic
46. The program is difficult to operate
47. Inefficient number of LOCs
48. Inability to find the proper software
49. Problem with the GIT
50. Limited skills of the group members

**How would you attempt to mitigate each risk (Top 10 Risks)?**

1. Being unable to finish
   * Prioritize the tasks which are more important
2. Faulty design of the software
   * Make the code more flexible
3. Errors in the implementation
   * Massive Debugging Session
4. Lack of features
   * Make Code Easily Extensible
5. Miscalculation of formula
   * Try to double check the calculation of the formula
6. Loss of unsaved data or code
   * Try to save code or work in intervals
7. Interface is not user-friendly
   * Let kids or non-groupmates test and modify accordingly
8. Overestimation/Underestimation of Time
   * Hard to Avoid. Give allowance to the estimated time.
9. Inability to run in slower computers
   * Not our problem. We will try to make the program more efficient.
10. Conflicts among group mates
    * Everyone must learn anger management. Settle conflicts fairly and calmly.