CSE 331-503 Project 1
Bomberman
Due Date: 12/11/2023 23:59 - Teams

In this Project you will implement a very primitive version of famous Bomberman games. The details of this Bomberman Game can be found in the following link:

https://www.hackerrank.com/challenges/bomber-man/problem

You will implement that on MIPS, but we are compassionate, therefore, we make this problem easier for you:

Different Submission Versions

1. In version 1, you will have a fixed size of matrix and it is 16x16, the simulation will always end up at the end 3rd second. The maximum grade for that version is **85pts**.
2. In version 2, the matrix can be in any size according to the user input. The simulation again ends at 3 seconds. Max grade for that is **100pts**.
3. Version 3 is same as the implementation in the above link. Thus, everything including the number of seconds is taken from the user. Max grade for that is **115pts**.
4. Version 4 is for the venturesome. You will detect at a given simulation instance whether there is an open path from [0, 0] to [max_i, max_j]. A path is open if one can pass from one matrix position to another without stepping over a bomb. Max grade for that is **130pts**.

Rules

- The details of the Project will be explained in next PS. You can ask your questions in that PS.
- You will use MARS ISS.
- Pseudo instructions are allowed.
- You MUST comment each line of your assembly, explaining why you wrote that line.
- You will at least implement 3 assembly subroutines. Implement them wisely.
- It is better to have a working lower version than to have a not working high version. So be sure to submit a working code. One step at a time and keep every version. Because, not simulating submissions or totally wrong results can get 30pts at most. Therefore, do not target Version 4 in the beginning. Start with the first version and update it to the next versions.
- Obey the contract.
- This is not a rule but strong advice: First, implement the solution in C, then test that implementation and then write your assembly with the same algorithm. Because rewriting in C is much easier.
- Each late day after the due date results in 20% grade loss.
- If you use compiler you get 0, if you cheat you get 0. If you cannot fully explain your own code you get 0. Moreover, those who attempt any of these will cause a highly negative impression over the instructors.

Aims

- Learn very well to use assembly for generic programming.
- Understand how such simple instructions can solve generic problems.
- Think similar to the MIPS CPU.
- Understand better how compilers work.

This project is not hard if you start on time, yet procrastination will surely make you work harder or be loser.