CS 152 Project:

Topic : Bomber Man

Team Members:-

Randhavane Tanmay Vitthalrao
Roll no. 110050010

Shivam H Prasad
Roll no. 110050041

Vivek Agrawal
Roll no. 110050042

Brief Description of Problem :

" Bomberman " is single player game. In this game you are the bomberman and are trapped in a maze. Use your bombs to destroy the enemies. Expand the maze by destroying blocks, not all blocks are affected by bombs and use these kind of blocks to shield yourself from the blasts which only are affective in directly perpendicular and horizontal directions.

Overall Design of program :

- **Key Concepts used in the program :-**
  - Lists and 2-D Vectors
  - Classes and Objects
  - Higher Order Functions
  - Random Generators
  - Probability Concepts
  - Simulation (in implementing the functioning of bomb and movements of bomberman and enemies)

- **Modules Used by us:-**
  - We have used 'graphics/graphics' module for implementing graphical user interface.
  - We have used 'racket/string' module to take string inputs from user as passwords to higher levels.
  - We have also used 'racket/gui' to implement user interactive windows.

- **Inner View:-**
  - The game-board used in our game simulates a 2d vector whose each element is an object of pos% class which is explained below. Each position on the game board has corresponding pos% object.

  - We have used lists to keep record of enemies. This list has enemy% class objects as its elements.

- We have created objects of following classes:
    - ♣ pos%
      Position (pos) class objects have been used to store all information associated with that position like whether that position has brick wall, permanent wall, enemy, bomberman, bomb etc
    - ♣ moving-object% class:
      We have used this as a parent class for all moving objects. It stores information regarding position, life status, image-address etc regarding that particular moving object.
    - ♣ bomberman%:
      It inherits from moving-object% class. There is only one object of this class which is used to implement bomberman.
    - ♣ enemy%:
      It too inherits from moving-object% class. It contains all fields of moving-object% class and additionally a field which contains position of bomberman.
    - ♣ bomb%:
      It is used to create a bomb. It has a timer and also stores bomb position and positions in range of bomb.

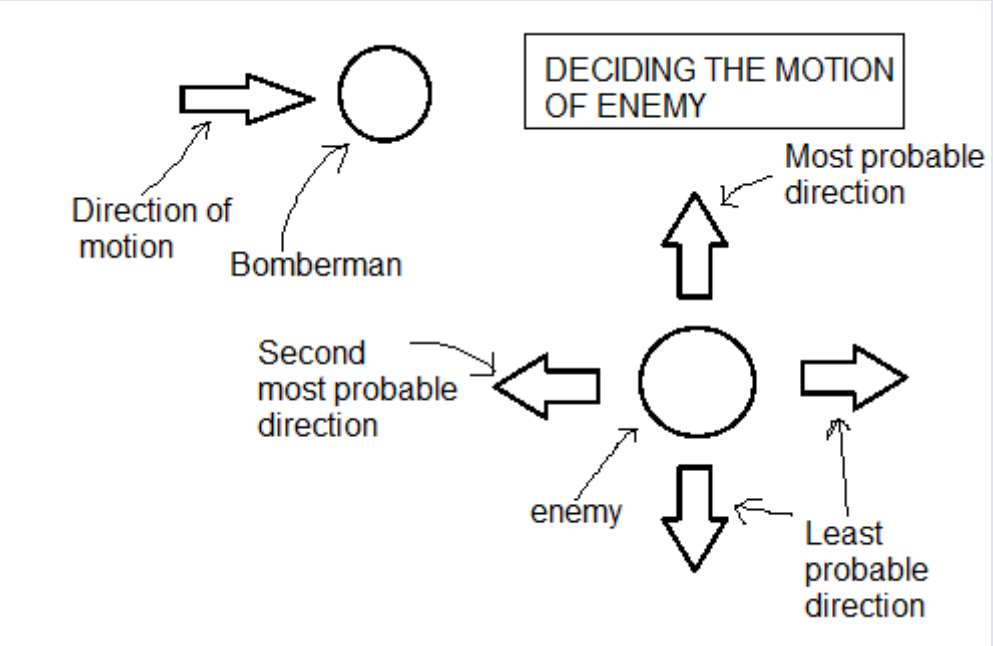
- We have used higher order function which takes level number as input and starts the game at that level. Also we have used a function which takes target position, enemy object and bomberman's movement direction and generates appropriate movement direction for that enemy.

- Each time the user plays our game the board is generated and set at that level randomly. Although the board is randomly generated we have ensured that it has uniformity. The board is divided into blocks and each block contains equal number of brick walls and permanent walls. The number of brick walls per block is constant for a particular level.

- We have set the motion of enemies such that they try to get closer to bomberman and they are repulsed by the bomb (if any). To ensure this we have used probability. If we wouldn't have used probability and given a fixed direction every time then there was a chance that the enemies would

be stuck forever at some positions so they will try to move in a direction that takes them closer to bomberman with higher probability and with lower probability in other directions. If the motion in a given direction is blocked then it will take another direction. If there is a bomb in line of motion then the enemy decides direction again and if he gets the same direction thrice then he proceeds in that direction.

Following diagram represents how movement of enemy is decided....



- We have used timers and sleep function from 'graphics/graphics' module to simulate the movement of bomberman, enemies and to implement bomb. When a bomb is placed a timer is generated which is updated after regular intervals. When this timer hits zero bomb explodes.

  At regular intervals a function is called which moves the enemies. A similar function is also designed for movement of bomberman which takes keyboard input at regular intervals and moves the bomberman accordingly. These timers influence the speeds of bomberman and enemies.

Limitations and Bugs in our program:-

i. The original bomberman game has powers but our bomberman does not include powers.
ii. Speeds of enemies and bomberman are fixed. User cannot change them.
iii. Sometimes it happens that when a bomb explodes but the images of the flame are not shown but all the hidden functionalities of bomb explosion are executed. We tried to fix this bug but could not find the reason behind it.
iv. Unlike the original bomberman, our bomberman can only place one bomb at a time.
