**Summative Proposal**

By Team NoOffer

May 21, 2014

ICS4U1-01

**Team Members :**

●Frank Han ●Sylvia Tsai

●Wilson Li ●Kitty Cao

**Table of Contents**

**Program Description**

●Overall **- - - - - - - - - - - - - - - - - - - - - - - - - -- - - - - - - -**

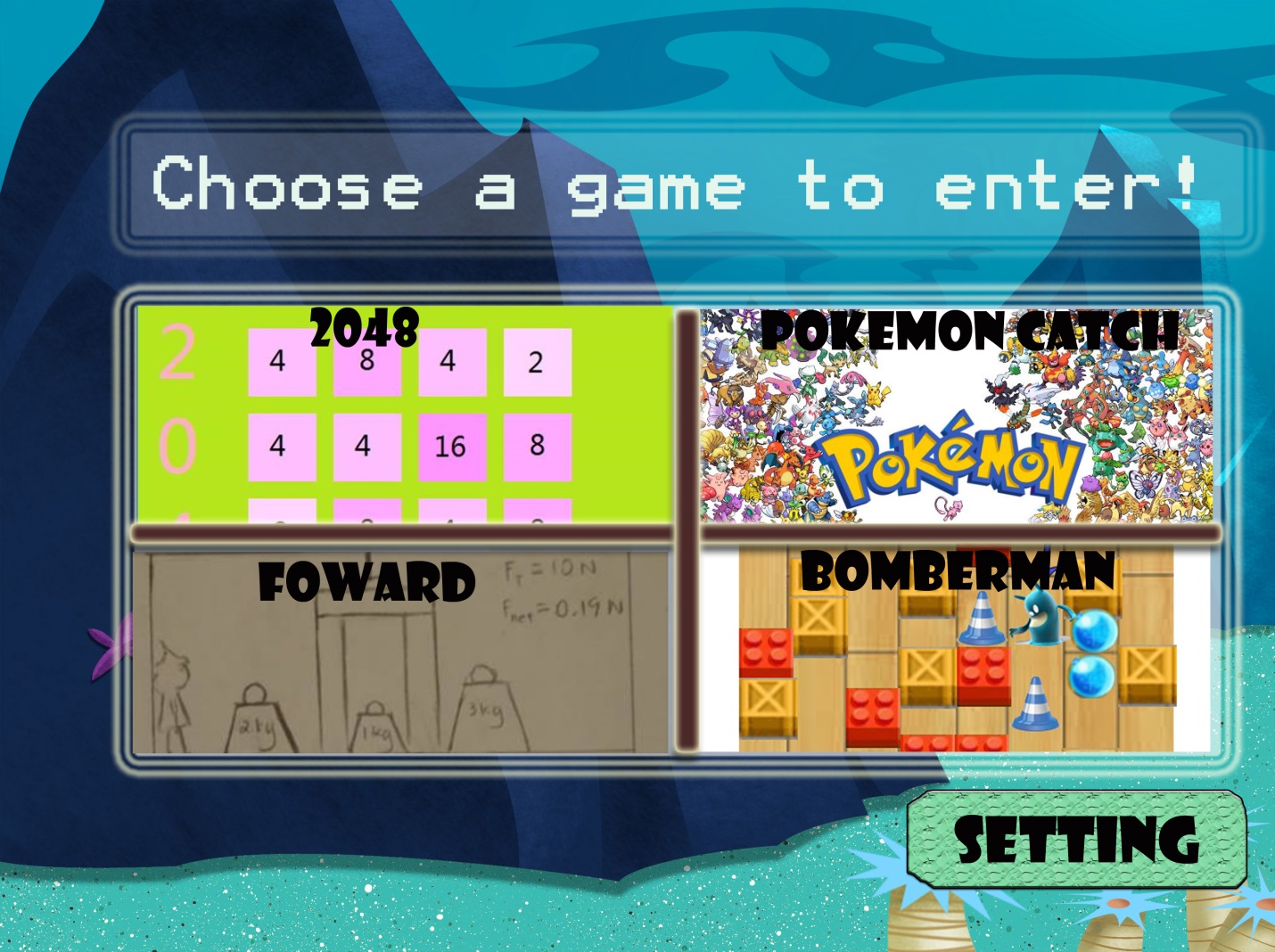
●2048

●Pokemon Catch

●Forward

●Bomberman

Main Program Description:

This program contains a series of single and multiplayer games that are designed to entertain the end users. The main program will act as a platform for users to choose games to enter. When the cursor is above the game, it will provide a simple game summary on either top or bottom. 

It will also have a setting button which provides extra functions including changing background, playing music and adjusting volume.

The main program will be controlled by mouse actions. Each Page will have a back button to go back to the previous page.

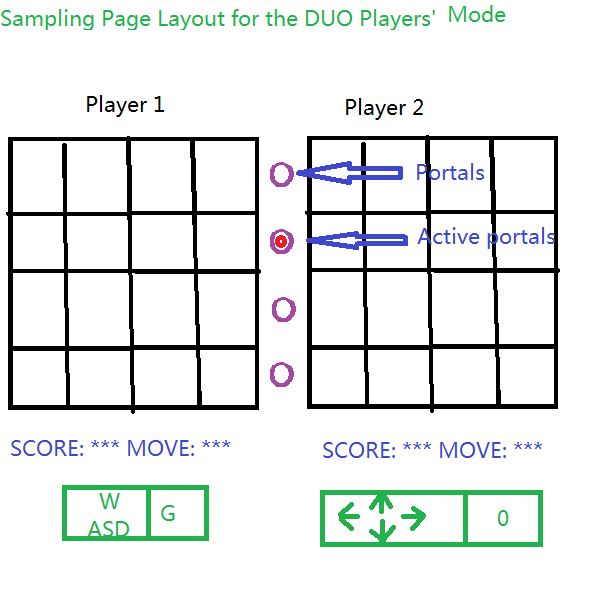
Individual Game Description:

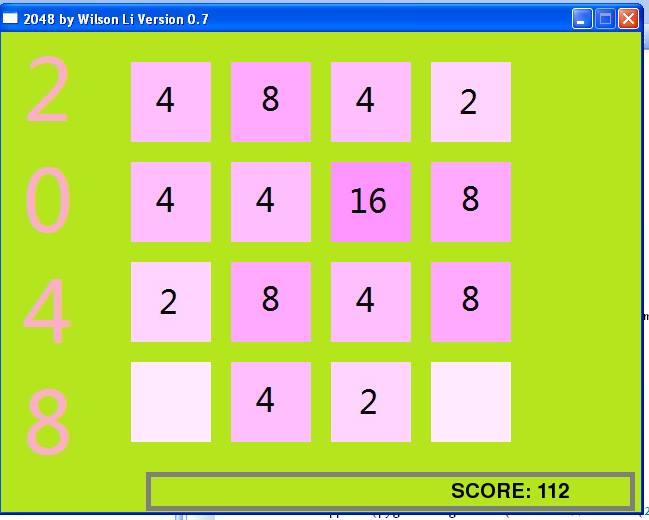
●Wilson – 2048:

2048 is a once-fashionable tile game that involves in combining tiles with the same face value together through moving all the tiles up, down, left, or right, to form a new tile with twice the value. This program is to be created as a replica of the already successful version of the game, developed by Gabriele Cirulli. Thus, starting from zero, a program will be created to be as similar as possible as Mr. Cirulli’s version.

That is not it. Something original, namely duo-player combating mode of 2048, will be added to my version. Two players will each have their own grid, and play side-by-side against each other. The original ideology is to allow tile exchange between the two players, through portals that could be open randomly between two players’ grid. If both players agree for an exchange while one portal is active, a transaction will be made, and two players will swap their number at the gird closest to the portal. The objective is to get a higher score than your opponent. One must play a modified strategy in order to succeed in this Duo players’ mode.

Programming concepts, such as modular programming, will be reflected with proficiency.

Single Player’s mode:



●Sylvia – Pokemon Catch:

Pokémon Catch is a single player puzzle game that is played with a mouse. When the player selects to play this game, they will be given the options to select “Play”, “Setting” or “Help”. If the user selects “Play”, a 10\*10 grid will appear with 5 different kinds of Pokémon inside each of the boxes. The target score will be displayed at the top right corner of the interface whereas the number of moves that the players is allowed with will be displayed at the bottom. The player will be required to create rows or columns with three Pokémon of the same kind to remove them. The player can make special combinations with 4 or 5 candies to generate Pokeballs. When the Pokeballs are combined, they will eliminate the row or the column of where they are and give out bonus. If the player reaches the target score before the player uses up his moves, he wins and a congratulation message will be displayed. If not, the player loses, a consolation message will be displayed and the player can choose to “Play Again” or to “Exit”. During the game, there will three buttons on the bottom left corner. The first button controls the music, the second button controls the sound effect and the last button allows the user to end the game. Going back the the main menu of Pokémon Catch, if the player selects “Setting”, he will be able to adjust the volume of the music and the sound effects. If the player selects “Help”, a tutorial will be displayed to assist him with the game. If the player selects “Exit”, the game will stop and it will go back to the main menu where the player can select one of the four games to play.

●Kitty- Forward:

Forward is a game that tests the physics knowledge of Newton’s Laws. The gamer controls an avatar that walks around using the left and right arrow keys. To advance in the game, the user must solve physics problems. There are three levels ranging from easy to difficult. To advance to the next level via an exit block, the user must have successfully solved all the problems on one level.

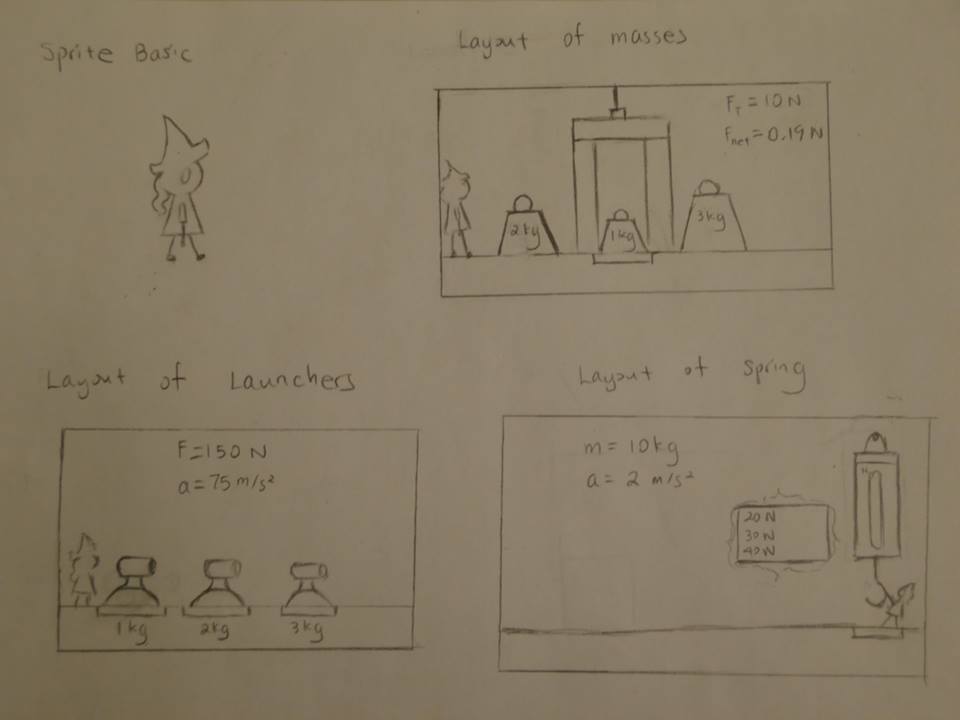
Net force is equal to force of friction plus force applied. Force is equal to mass times acceleration. These two equations are used for problems in level one. When dealing with force at an angle, force in the horizontal direction is equal to cosine theta times force applied. Net force is then the sum of force in the horizontal direction and force of friction, this is the equation used for level two. There are also the elevator physics questions where net force equals force of tension and force of gravity. The elevator questions are reserved for level three.

In the first level, there will be ten problems to solve. The number of problems decreases by two every level but increases in difficulty. The final level is the third level. The user will get only one try to complete a problem. If the user does not solve the problem, the gamer will have to complete another problem of equal difficulty. The values are randomly generated, as are the problem sets. The user may wish to left-click the restart button for a different problem or the hint button. There is no time limit.

When the solution requires a mass, the gamer must select one of the various labelled masses around the room. The player can pick up the mass by pressing P when the character is “colliding” with the mass. Once the mass is picked up, the player is considered to have given an answer.

When the solution requires a force in Newton, the player must pull on a spring to match the Newton on the spring to the answer. When the player is standing directly under the spring in a designated area, the avatar will automatically grab the spring. A window will appear showing the multiple choices of N. The player clicks on an answer.

When the solution requires acceleration in m/s2, the question becomes a multiple choice. There will be three launchers with different masses but the same force. The masses will be labelled, and the gamer will be informed of the force. The player can select one of the launchers by pressing K on a highlighted platform that is near each launcher. When the button is pressed, the player is considered to have given the answer.

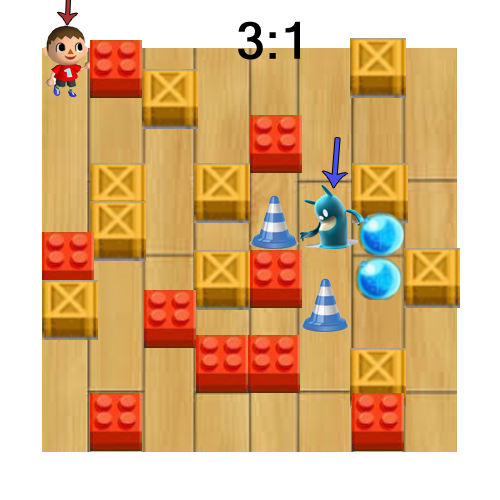


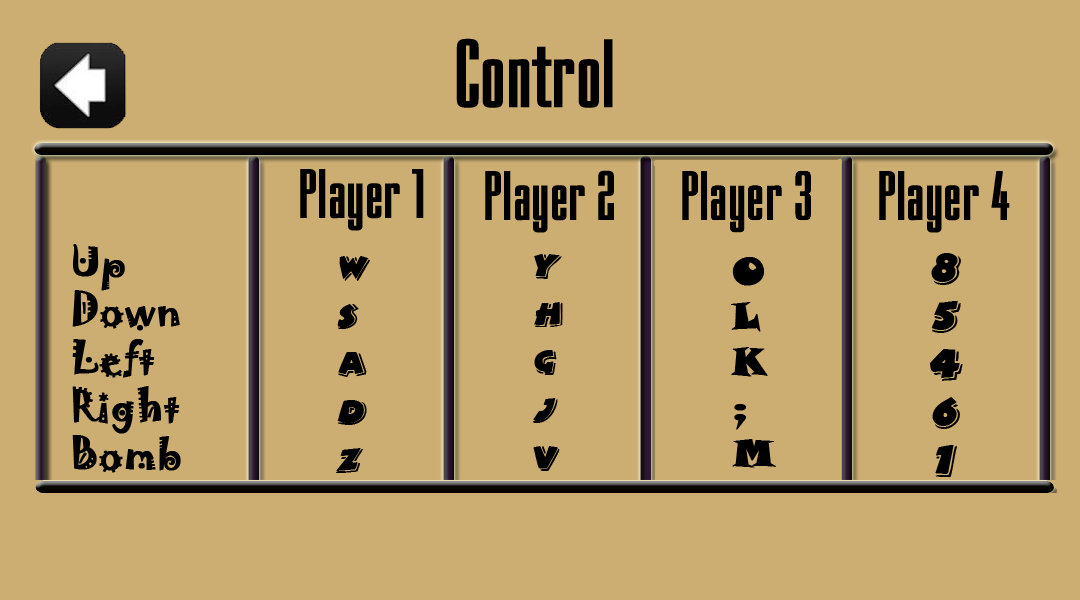
●Frank – Bomberman:

Bomberman is the game for 2- 4 players on one computer. Each player will control a character by a specific set of keys on keyboard. Each character can put down bombs that will explode after several seconds. Any player that gets hit in horizontal or vertical line will lose one life point. It is a remake for the classic game. However, the user will be able to jump over some walls, use items (ie. banana peel) and select unique characters with special abilities(ie. Bomb explode in a bigger range) to help them winning the game.

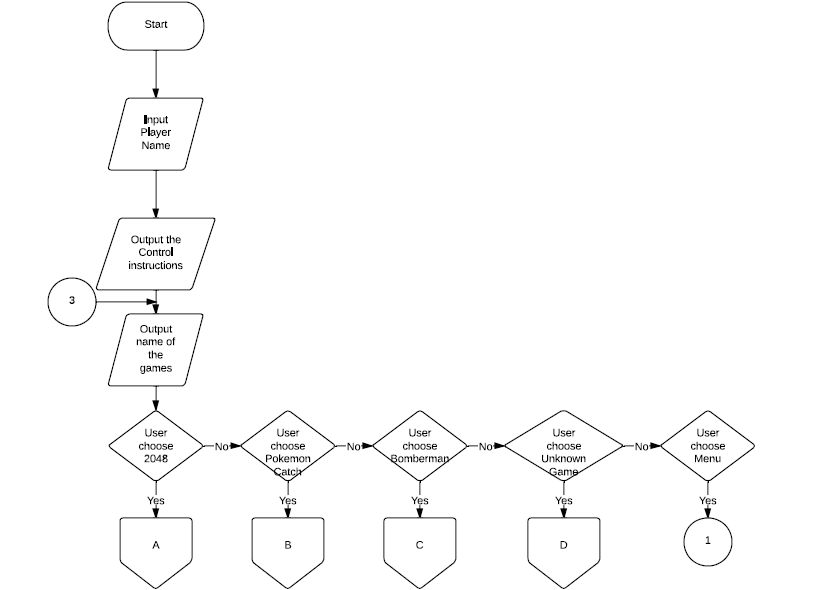
The game welcome page will have 4 buttons: play, control, game instruction and exit. The control function will output the specific control keys for each of four players. Instructions will be given in the instruction function with detailed explanation to rules of the game. And exit allows user to return to the game selection part.

When play button is selected, it will enter the pre-game setting which will decides the life point, number of players and the map for the current game. Each user will then select his character’s ability and appearance.

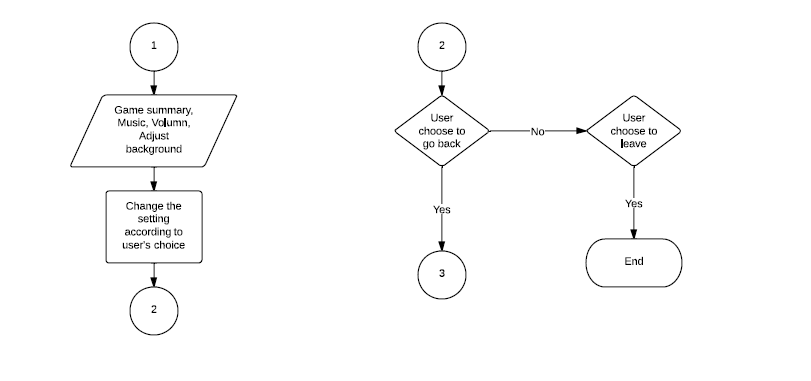
While in game, there will be colored arrows that identify each player above the characters. Each character will spawn at the specific location in the map. Any character cannot go through each other. When one’s life point dropped to 0, he will be eliminated from the game. If there is only one person standing alive, he wins the game. And the user will be able to choose whether they want a rematch or go back to the game menu.



Main Program Flowchart:



NO



●Frank – Bomberman:

Anyone hit by a bomb

Remain player =0

No

No

1

YES

YES

Life point of that player-1

Life point =0

No

YES

Player remain-1

Player is dead

1

No

No

If the user choose to bomb

If the user choose to move

YES

YES

No walls/characters are met

No

Place the bomb there

YES

Move Position according to speed

Move Position according to speed

Group Schedule:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [◄ March](http://www.wincalendar.com/March-Calendar/March-2014-Calendar.html) | |  |  | **~ April 2014 ~** | | | | | |  |  | [May ►](http://www.wincalendar.com/May-Calendar/May-2014-Calendar.html) | |
| **Sunday** | | **Monday** | | **Tuesday** | | **Wednesday** | | **Thursday** | | **Friday** | | **Saturday** | |
|  |  |  |  | **1** |  | **2** |  | **3** |  | **4** |  | **5** |  |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
| **6** |  | **7** |  | **8** |  | **9** |  | **10** |  | **11** |  | **12** |  |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | | Submit Proposal | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
| **13** |  | **14** |  | **15** |  | **16** |  | **17** |  | **18** |  | **19** |  |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | | **1st Progress Report** | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
| **20** |  | **21** |  | **22** |  | **23** |  | **24** |  | **25** |  | **26** |  |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | | Background and game summary feature completed | |  | |  | | Music and Volumn added in the main | |  | |
|  | |  | |  | |  | |  | |  | |  | |
| **27** |  | **28** |  | **29** |  | **30** |  | Notes: | |  |  |  |  |
|  | |  | |  | | **2nd Progress Report** | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | | Individual Game finished | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [◄ April](http://www.wincalendar.com/April-Calendar/April-2014-Calendar.html) | |  |  | **~ May 2014 ~** | | | | | |  |  | [June ►](http://www.wincalendar.com/June-Calendar/June-2014-Calendar.html) | |
| **Sunday** | | **Monday** | | **Tuesday** | | **Wednesday** | | **Thursday** | | **Friday** | | **Saturday** | |
|  |  |  |  |  |  |  |  | **1** |  | **2** |  | **3** |  |
|  | |  | |  | |  | |  | | User help guide (small book )finished | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | | Debug in the group | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
| **4** |  | **5** |  | **6** |  | **7** |  | **8** |  | **9** |  | **10** |  |
|  | |  | | Individuals Code fully documented submitted. | |  | |  | | Test log /beta test log submitted | |  | |
|  | |  | |  | |  | |  | | Adjustment made according to the feedback | |  | |
|  | |  | | Codes are put together into the main | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
| **11** |  | **12** |  | **13** |  | **14** |  | **15** |  | **16** |  | **17** |  |
|  | |  | | **In Class testing and Debug if needed** | |  | |  | | Presentation Design & Practice done | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
| **18** |  | **19** |  | **20** |  | **21** |  | **22** |  | **23** |  | **24** |  |
|  | |  | | **Final Presentation Date** | | **Final Submission of the project** | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |  | |  | |

●Frank – Bomberman:

**April**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sunday** | **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** | **Saturday** |
| **6** | **7** | **8**  Proposal Submit | **9** | **10** | **11**  **Game welcome page made** | **12** |
| **13** | **14** | **15**  **Instructions/Control function made** | **16** | **17** | **18**  Pre-Game selection and character selection completed | **19** |
| **20** | **21** | **22**  Jump/Move/Collusion completed with sound | **23**  **Bomb and items function completed** | **24** | **25**  **Connect it with the main program** | **26** |
| **27** | **28** | **29**  **Finish program** | **30** |  |  |  |

**May**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sunday** | **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** | **Saturday** |
|  |  |  |  | **1** | **Debug** | **3** |
| **4** | **5** | **6**  Program fully Commented | **7**  **Beta testing** | **8** | **9**  Edit | **10** |
| **11** | **12** | **13** | **14** | **15** | **16**  **Prepare for the presentation** | **17** |
| **18** | **19** | **20** | **21** |  |  |  |

●Frank – Bomberman:

|  |
| --- |
| **Map** |
| Length:int  Width:int  wholeMap: 3D array with length x width  startPosition: list of integer  playerRemain:int |
| Def Additem(class):void  Def deleteitem(class):void |

|  |
| --- |
| **Wall** |
| Position:list  Height:float  Destructable: False |
|  |

|  |
| --- |
| **Character** |
| Position:list  Life:int  Size: list  Jumping: bool  Speed:float  jumpHeight:float  CharacterType:int  Playernumber:int  NumberOfGrenade:int  NumberOfBanana:int |
| Def bomb():void  Def jump():void  Def move(direction):void  Def banana():void  Def grenade(direction):void |

2 . . 4 0 . . \*

|  |
| --- |
| **Destructable Wall** |
| Destructable: True |
| Def destruct():void |

|  |
| --- |
| **Grenade (Range bomb)** |
| Target:list  Speed:float |
| Def fly(position,target):void |

|  |
| --- |
| **Banana** |
| Position:list |
| Def slip(character class)  : void |

|  |
| --- |
| **Bomb** |
| Position:list  StartTime=int |
| \_\_init\_\_():void  Def explode(startTime, currentTime):void  1. . 4 |

1..4

0 ..4