# 24-783: Advance Engineering Computation Project

**Topic: "Bubble Bobble" Implementation in C++**

Group: HaveAGoodDay

1. **Overview:**

   Bubble Bobble is a classic Japanese arcade game first invented in 1986 by Taito. It was all the rage across the world. All members in the team HaveAGoodDay had played bubble bobble in their childhood. It was an unforgettable and valuable memory for our members. So we decided to use what we learned in the Advanced Engineering Computation course to port this classic arcade game into C++. The mechanism of this game is as in the figure 1 below: the player controls the little dinosaur to fight against the monster enemies by ejecting bubbles and colliding the bubbles. The score and the remaining life are shown at the top of the screen. The player will earn points when the little dinosaur collects the dropped fruit after defeating a monster. The reason why we chose this game is that it's not only a good practice of our homework elements but also a very challenging project.

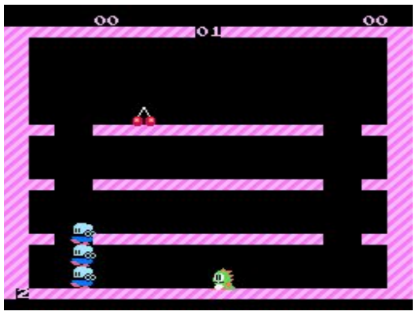

Figure 1: screenshot of the "Bubble Bobble"

2. **Features:**

   For the GUI components, we will have our little dinosaur and monsters in different orientation, fruit, and map tiles . This structure will be similar to our homework 4, so it's a good practice of what we learned in past homework.
   For code structure, we will include the libraries used in homework like simple_bitmap, tiles, and mmplayer.

   Here are the main features of our game:
   - Single-player mode game.
   - Background music for the entire game.
   - Player can move along platforms, fall to lower ones, and jump to higher ones and over gaps.
   - Level is limited to a single screen, with no left/right scrolling.
   - The player must blow bubbles to trap the enemies, then burst these bubbles by colliding with them.
   - Each enemy defeated by bubbles turns into a food item that can be picked up for extra points.

- All bubbles will float for a certain length of time before bursting on their own; players can jump on these and ride them to otherwise inaccessible areas.
- A player loses one life upon touching any free enemies.

3. **Components:**

Here is our project folder layout (Tentative):

```
- simple_bitmap/
        - template
        - map
- tiles/
        - dragon
        - enemy
        - fruit
        - bubble
        - trapped_enemy
- mmplayer/
        - music_samples
        - player
```

4. **Challenges:**

The following list are challenges that we might encounter. We think this will make sure our project has the appropriate level of difficulty. For map generating, the map would be loaded as it is in homework 4. While the data structure for storing the edges of the map and the map edges detection would be a challenge problem for us. This includes how the map will interact with other objects in the game. Meanwhile we also wanted to provide BGM during the game. The background BGM would be hard for us. We are planning to add sounds for actions like shooting bubbles and enemy disappearing. Also, We want to make the movement of the characters as smooth as possible and the animation to be attractive. For this we might try to optimize the shading logic or input data (something related to the memory miss/hit). As the mechanism of the game, we plan to hybridize the event driven and Time stamp concept for the backbone of the game. The time stamp is used for events such as the trapped enemies releasing after a certain time. How to manage the changing "time states" for each monster will need careful design. The event-driven design for the project requires a good system design before we get started.

List of technical challenges:

- Map generating
- Background music loading
- Moving mechanism
- Animation effects
- “Timer state” changing
- Event driven design