**Anabelle Lee (kaylalee)**

**15-112 Term Project Proposal**

**School Flirting Game**

**Project Description**

My project “School Flirting Game” will be a point-and-click style game where the user will play as a high school girl trying to win over the hearts of as many boys as possible in the given time limit. There will be obstacles along the way, such as other high school girls trying to steal the user’s targets, as well as teachers roaming the halls who can hinder the user’s progress. Depending on their popularity, different boys will be more difficult to win over than others. The user can also enter “Beauty Time” mode if they win over enough boys, giving them major buffs that will help them win over more boys’ hearts for a short time frame. There is no ultimate objective – there will be a local scoreboard that stores all previous scores.

**Similar Projects**

I’m basing my project over an old flash game I played a lot as a kid called “School Flirting Game” (you can find an emulated version [here](https://www.gamenora.com/game/school-flirting-game/)). Mechanically, I’m hoping the game will be identical, with the exception of using the mouse position to move the character – rather, movement will be locked to WASD, and laser beams will be triggered with the mouse buttons. The only feature I’m hesitant to include is the multi-level school. I’m not sure on how I would implement this, and feel like I can still make a sizable TP with side-scrolling only. I’ve looked at other projects in the TP Gallery with side-scrolling (for example, [Ruslana Fogler’s project](https://www.youtube.com/watch?v=VeJvD36yryo&ab_channel=RuslanaFogler)), and am fairly confident that this will be sufficient for my own TP.

**Structural Plan**

Classes:

* Main Character
  + targetLock()
  + attack()
  + beautyTime == False
  + beautiful()
* Love Interest
  + inLove == False
  + followLove()
* Rivals
  + battleMode == False
  + battle()
* Teachers
  + collide == False

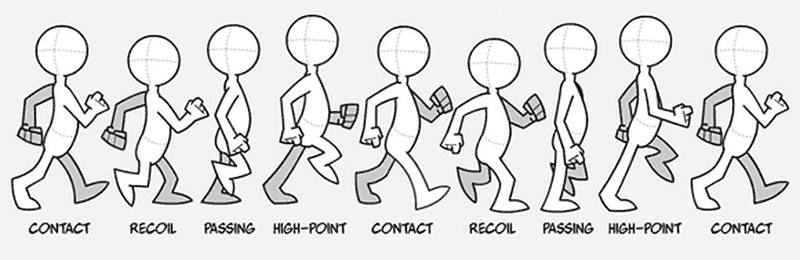
Functions:

* Collision trackers (if laser beam hits target, score ++)
* Battle (if rival also targeting love interest)
* Scores (if special love interest, score increases more)

Graphics:

* Running animations
* Laser beam animations

**Algorithmic Plan**

* Fancy Graphics
  + Running Animations
    - Every character object will have an attribute “walking animations” which is a *list of frame-by-frame walking images* (see below for example)*.* I will use the variable “index counter” to loop through each of these images and display them on the screen, thus making it appear as if they are walking.
    - 
  + Scrolling Map Image
    - Need to think about this more, but I think I will have a super long PNG image of the school that scrolls depending on the main character’s position.
      * So, if MC is located in the CENTER of the screen, image will not scroll
      * If MC is on the FAR LEFT or FAR RIGHT, image will scroll towards that direction.
* Battle Mechanics
  + Getting Love Interests to follow the Main Character once they’ve been defeated
    - Every Love Interest will have a boolean attribute “inLove” and method “followMC()”
      * if self.inLove: self.follow()
      * self.follow() – changes Love Interest’s XY coordinates to be dependent on the MC, as opposed to independent.
  + Teacher Collisions
    - If MC.position == teacher.position:
      * Lose points
      * Collision animation plays
      * Lose time
* End Game Trigger
  + If Timer == 0:
    - Stop all mechanics
    - Play ending animation
    - Show local leaderboard

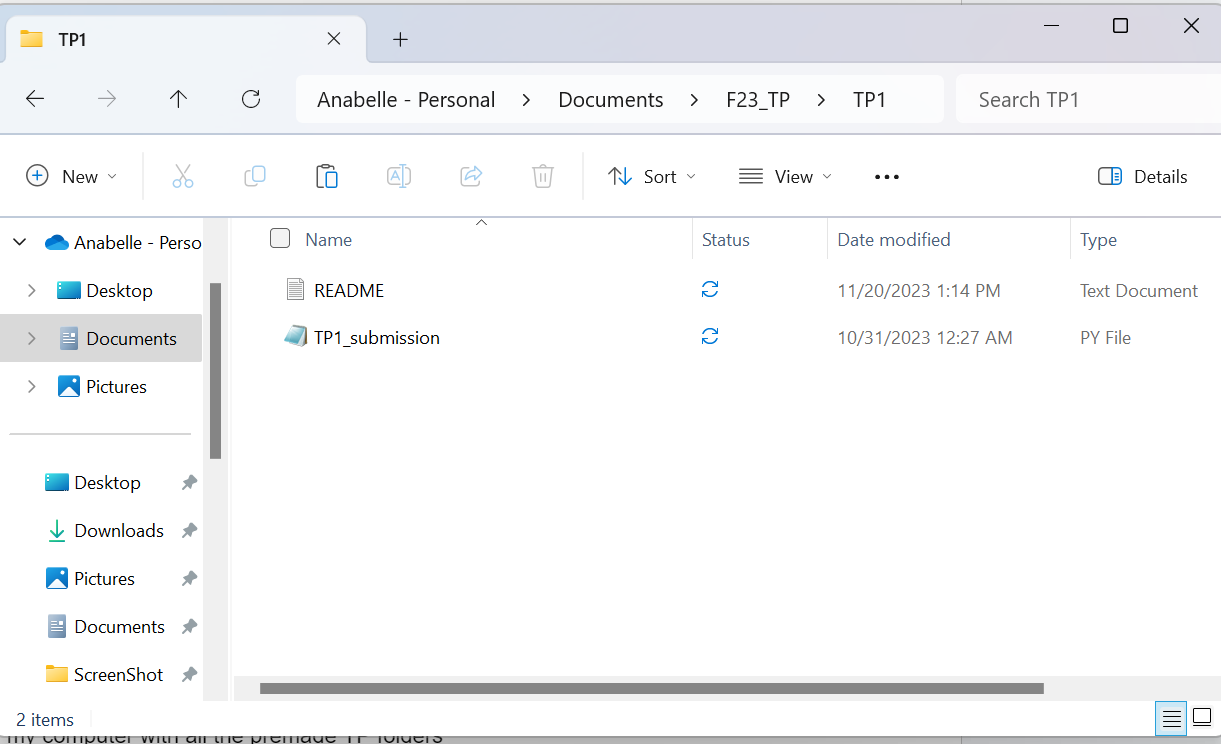
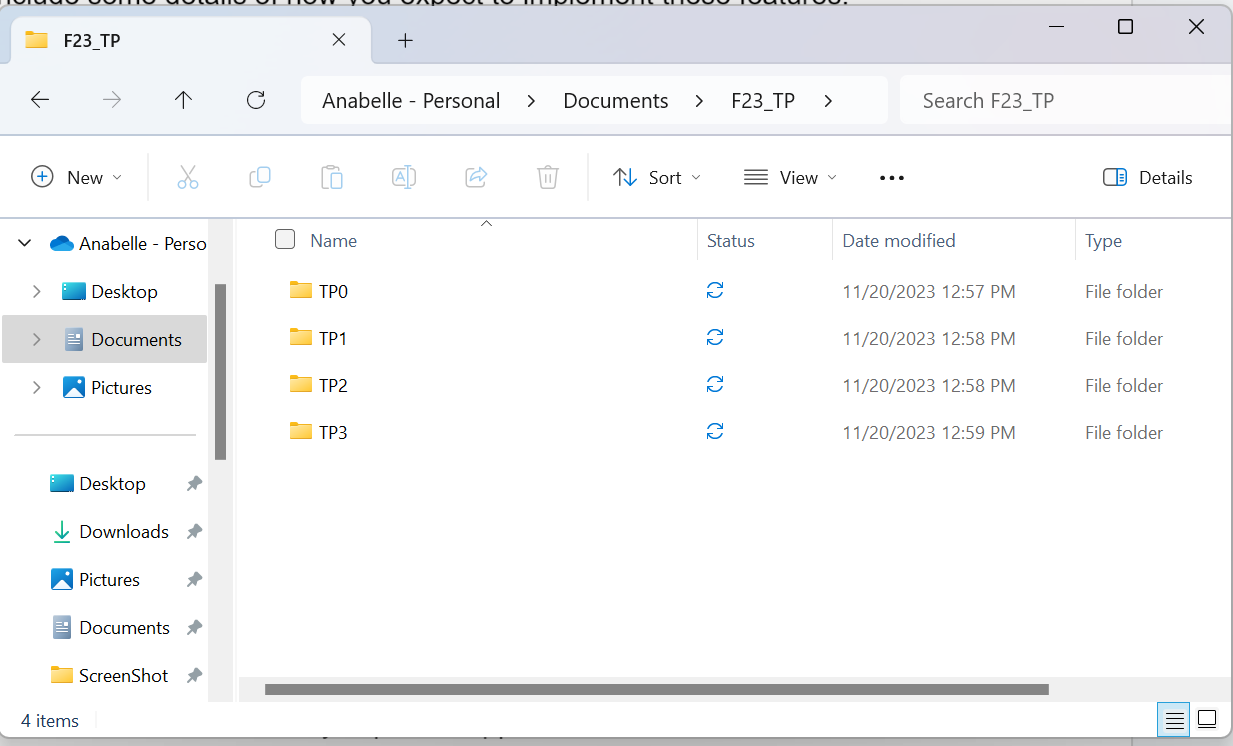
**Timeline Plan**

* TP1
  + All classes written
  + School map drawn and completed
  + Main character is fully functional – can run with WASD, shoot laser beams with mousePressed
* TP2
  + Battle mechanics implemented – if Rival witnesses battle, will join battle.
  + After Love Interest defeated, will follow Main Character around
  + Teachers can collide with Main Character
  + *Side scrolling must be implemented by TP2*
* TP3
  + Fancy graphics – detailed running animations
  + Local scoreboard implemented
  + Possible sound effects and background music

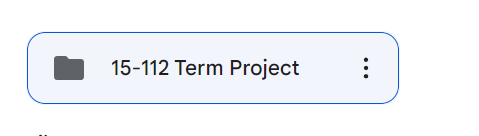
**Module List**: NONE!

**Version Control Plan:**

Every single TP submission will be in a *separate VS Code Python File,* each of which have also been *zipped and uploaded to Google Drive* for further security.



^^ Folder on my computer with all the premade TP folders



^^ This Google Drive folder will be populated with my imported zipped files submitted for each TP assignment

**TP1 Update:** None!

**TP2 Update:** None!

**TP3 Update:** None!