Game name: Jo: The Stardust Crusader

Game desc:

Play as Jo, who shoots his gun (Star Platinum) to destroy the enemies (minions). You get points when you destroy enemies, after a certain amount of points, you will get upgrades to your gun. Survive for 1 minute and acquire as many points as possible before facing the big boss (dio).

# Things to Note:

1. Try to get as many points as possible.
2. At 120 points, you get a damage increase.
3. At 240 points, you get a health bonus and a small damage bonus.
4. At 300 points, +health, +dmg, +move speed bonus.

# How to Win/Lose:

1. You lose if your hp drops to 0, you lose hp when a minion or dio touches you.
2. You win if you defeat dio (the red rectangle).

# General Tips:

1. Try to get all the upgrades to increase your chances of winning.
2. DO NOT underestimate Dio. Once his health drops under 20, he becomes enraged, moving much quicker and doing more damage.

# Controls:

Movement:

* Left: left arrow key
* Right: right arrow key
* Float: up arrow key
* Fly: up arrow key (hold)

Combat:

* Shoot: spacebar

# Characters:

1. Jo (blue rectangle) -

# Game details:

* Jo (Player):
  + 5 hearts – upgraded through points
  + 1 gun – upgraded through points
  + (optional) Ora-load: overload Jo’s gun to shoots a wide spread of bullets that deal 2 damage per bullet, similar to a shotgun shell.
  + Can crouch
  + Maybe can jump??
* Dio (Enemy Boss):
  + 8 hearts
  + 2 guns
  + Can randomly spawn enemies
  + (optional) Vampiric Grab: perform a special attack (melee) that does 2 hearts of damage and heals dio for 2 heart of damage (30s cd?).
* Minions (enemies):
  + Cant shoot
  + Tries to collide with player to deal 1 damage.

# Classes

* Sprite: base model for all the physical entities on the map, responsible for loading spritesheets.
  + loadSprite(filename) – read a spritesheet .png, returns a bufferedimage.
  + getSprite(int xGrid, int yGrid) – return subImage from the loaded spritesheet to get the specified sprite from the sprite sheet.
* Frame: hold an image for a duration
  + getFrame(): return this.frame;
  + setFrame(bufferedimage frame): set the frame to a frame;
  + getDuration() - return this.duration;
  + setDuration() – this.duration = duration;
* Animation: renders frames for the game from a list of frames
  + Start(): start the game
  + Stop(): stops the game
  + Restart(): reset frame size and curr frame
  + Reset(): reset frame to initialized values.
  + addFrame(bufferedimage frame, int duration): add a frame
  + update(): go to next frame.