Classes, Flow Diagram, Management, and Timeline
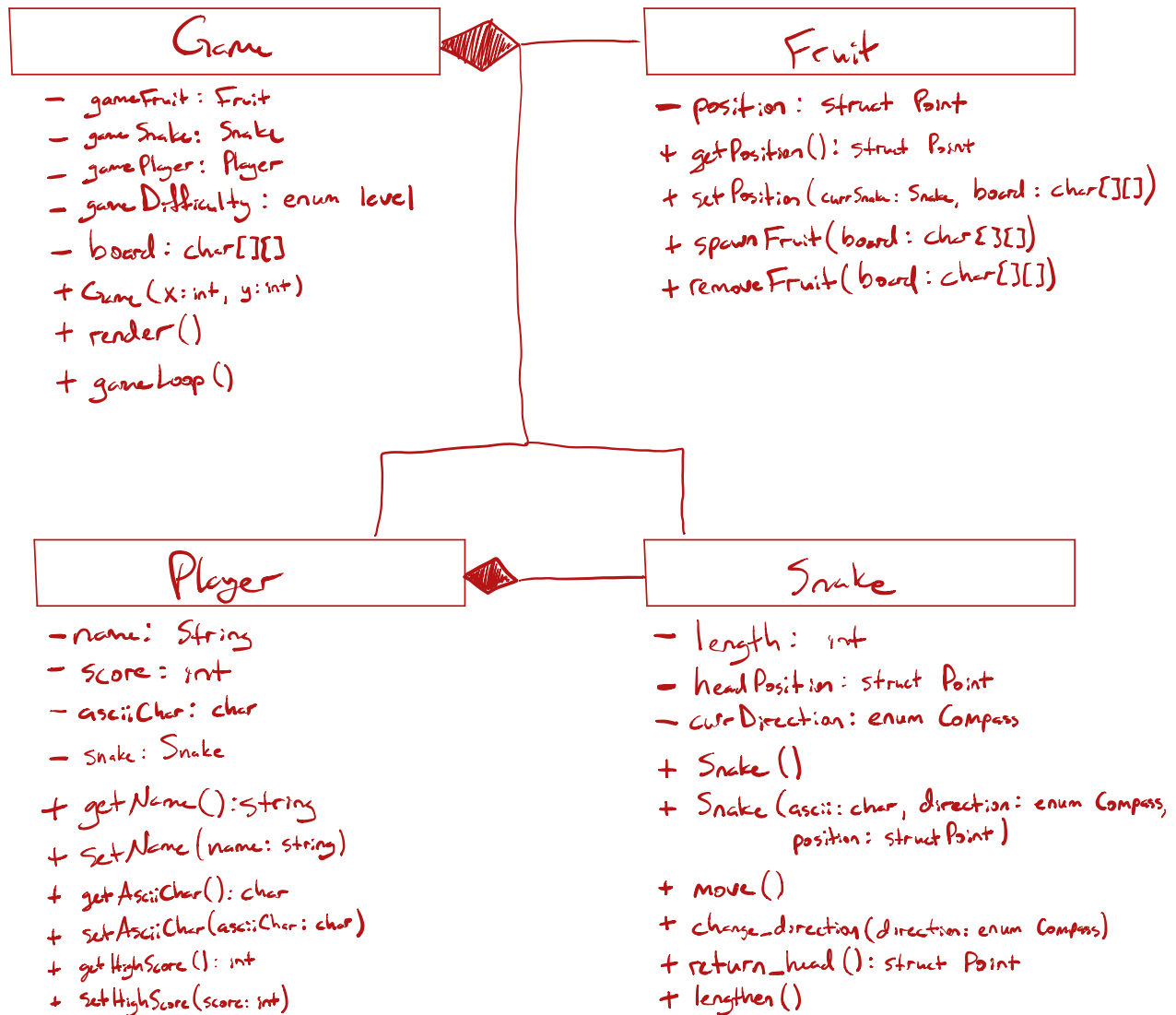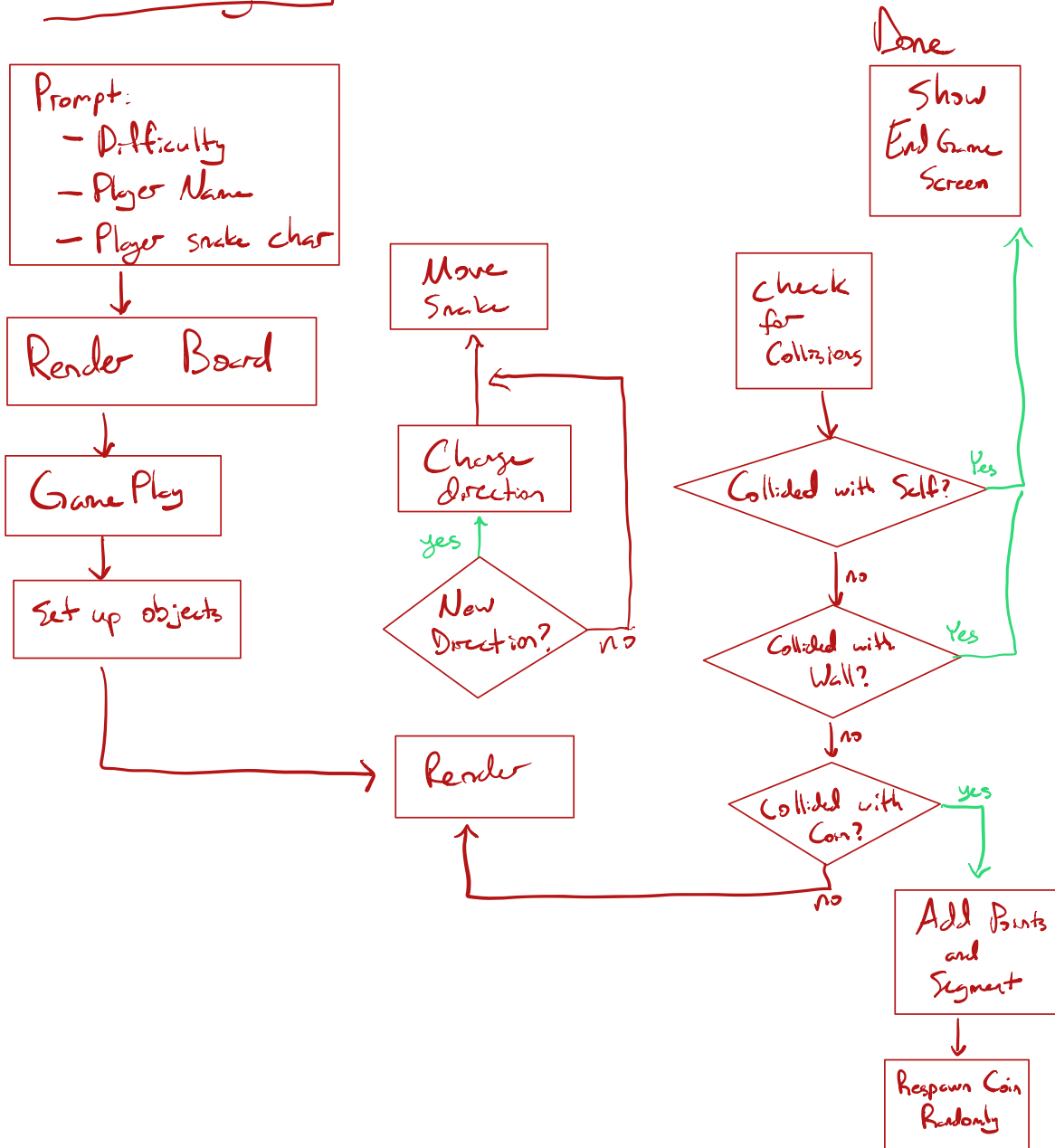
Luke Irvine, Hayden Tinker, Samuel Hernandez, Stefan DePaula

CPTR 142 Final Project

Game
- gameFruit : Fruit
- gameSnake: Snake
- gamePlayer: Player
- gameDifficulty : enum level
- board : char[][]
+ Game (x: int, y: int)
+ render ()
+ gameLoop ()

Fruit
- position : struct Point
+ getPosition (): struct Point
+ setPosition (currSnake: Snake, board : char[][])
+ spawnFruit (board: char[][])
+ removeFruit (board : char[][])

Player
- name: String
- score: int
- asciiChar: char
- Snake: Snake
+ getName (): string
+ setName (name: string)
+ getAsciiChar (): char
+ setAsciiChar (asciiChar: char)
+ getHighScore (): int
+ setHighScore (score: int)

Snake
- length: int
- headPosition: struct Point
- currDirection: enum Compass
+ Snake ()
+ Snake (ascii: char, direction: enum Compass, position: struct Point)
+ move ()
+ change_direction (direction: enum Compass)
+ return_head (): struct Point
+ lengthen ()

Flow Diagram

Prompt:
- Difficulty
- Player Name
- Player snake char

↓

Render Board

↓

Game Play

↓

Set up objects

↓

Render

Move Snake

↑

Change direction

↑ yes

New Direction? → no

Done

Show End Game Screen

check for Collisions

↓

Collided with Self? — Yes →

↓ no

Collided with Wall? — Yes →

↓ no

Collided with Coin? — yes ↓

no →

Add Points and Segment

↓

Respawn Coin Randomly

Challenges:
- Constant needed for number of columns when declaring array. How do we make a board of variable size then?
    - Solution: make constant a max value
- Unit testing all these functions
- Managing who will do what functions
    - Making sure work is evenly distributed
- Managing Snake body and how multiple directions are maintained simultaneously
    - keep cell "on" for loop iteration?
        - How does loop count change dynamically when fruit is encountered?
- Where/how should we render??
- Chicken or the egg principle

Team Management:
- Luke Irvine
    - Build Master (accepts merge requests)
    - Build unit tests throughout (for cptr245)
    - Assignments throughout
- Stefan DePaula
    - assignments throughout
- Hayden Tinker
    - assignments throughout
- Samuel Hernandez
    - assignments throughout

Timeline:

Phase 1
- Due Feb 24
- Set up classes with function prototypes
- Assignments:
  - Luke: Set up basic tests for mutators and accessors
  - Hayden: Player prototypes
  - Stefan: Game prototypes
  - Sam: Fruit and Snake prototypes

Phase 2
- Due Feb 28
- Build empty board
- Render (print) board
  - Start with terminal/txt file
    - Txt file if it updates dynamically
  - Move on to web
- Get snake of fixed length moving across board
- Change direction of snake with keys
  - Have it wrap around
- Assignments:
  - Luke: more unit tests
    - Also game loop
  - Hayden: Directions
  - Stefan: Snake
  - Sam: Print Board

Phase 4
- Due March 2
- Add fruit
  - Generate randomly across board
- Make fruit change size of Snake

Phase 5
- Due March 6
- End game if snake hits walls or itself

Phase 6
- Due March 9
- Output score to file
- Read high score from file

Phase 7
- DEBUG/slush time
- Due March 16