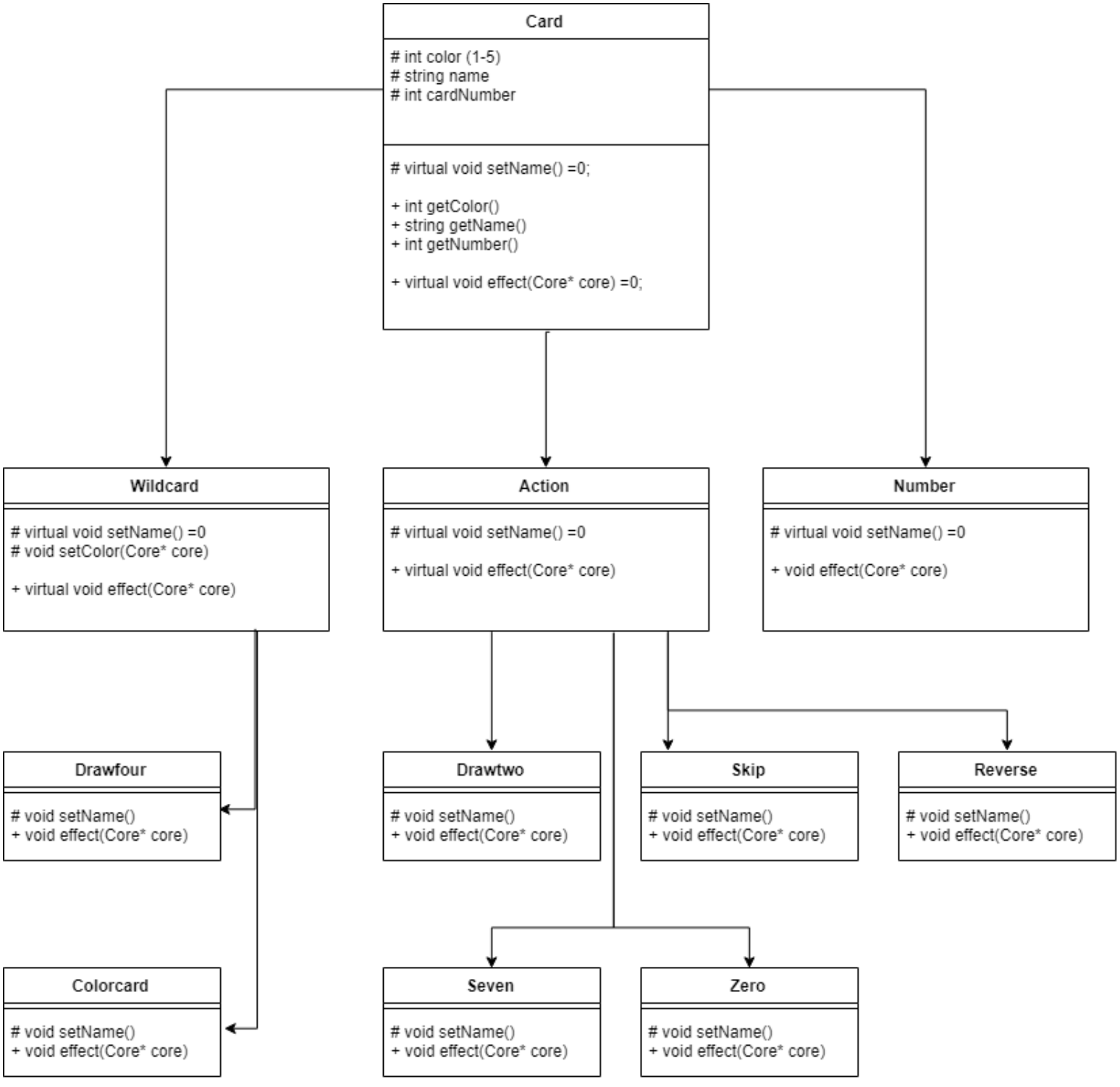
Uno design (Don’t want to make this, but I got to do it)

Part 1: The cards



1. Class Card
    - Int color: the color of the cards, from 1-5: RGBY & Unset wildcard color. Wildcard has default color is 5 and once it's played, it'll be set to other colors as player chooses
    - String name: the name of the card. Normal cards have their name as their color and their number (e.g. Yellow 5, Blue Skip…) but Wildcards have their name just their name (Wildcard & Draw 4 Wildcard)
    - Int cardNumber: each card has its own number: 0-9 for number card, 10 for Reverse, 11 for Skip, 12 for Draw 2, 13 for Wildcard (Colorcard) and 14 for Draw 4

    - Virtual void setName() =0: set the name of each card. Different types of cards have different way to set their name (as seen above)
    - Int getColor(), string getName(), int getNumber(): provide public access to those values
    - Virtual void effect() =0: effect of each card. All number cards will just going to have blank effect

2. Class Wildcard
    - Void setColor(Core* core): this function will be called in the effect of the two wildcards, prompting the player (either human or bot) to choose the color of the card. Color of the card will be changed from 5 to other colors after this function is called
    - Virtual void effect(): setColor will be called here. The two wildcards that inherit this function will automatically call setColor when the effect of those cards are called

3. Class Action
    - Virtual void setName(): Pre-construct the name of the card by adding the color in their name first. The inherited action cards will add their full name later

4. Class Number
    - Virtual void setName(): Fully construct the name of the cards by adding the color in front of their number. There won't be any further inheritance in this class
    - void effect(): This function is left blank

- NOTE: For the seven and zero card (in 0-7 rule), if that game mode is set to false, the main function will create these two cards as typical number cards. If the game mode is set to true, the main function will create the two cards as action card

5. Class Drawfour
    - Void setName(): name will be set to "Draw 4 Wildcard"
    - Void effect(): this function will set the number of cards that the next player is supposed to draw to 4. This is only used as an indication of which card that forces them to draw, the draw 2 or draw 4. What happen next is the matter of the core stackable function
      setColor will be called here, and next player will be skipped

6. Class Colorcard: Like Drawfour but effect is left blank

7. Class Skip:
    - Void setName(): name will be set to color + "Skip"

- Void effect(): tell core to set the state ‘nextTurn’ of the next player to false

8. Class Reverse:
   - Void setName(): name will be set to color + “Reverse”
   - Void effect(): tell core to swap the state ‘turnDirection’ (turnDirection = -1 or 1)

9. Class Drawtwo:
   - Void setName(): name will be set to color + “Draw two”
   - Void effect(): Like drawfour, but draw 2 cards instead and don’t call the setColor function

10. Class Seven:
   - Void setName(): name will be set to color + “Seven”
   - Void effect(): get the player that played the card from the core and tell that player to choose a player to swap with (how to do it depends on the human or bot). The function will then fetch the pointer to the hands of the two players and swap them with each other

11. Class Zero
   - Void setName(): name will be set to color + “Zero”
   - Void effect(): get the pointer to players’ hand from core and swap them with the next player

- NOTE: Despite the effects are hold in the card, it doesn’t relate to other state or behavior of the cards in any ways, but the core instead. Therefore, I might be able to move all of these effect

to core later on, though not sure if that’s a good idea or not. It reduces dependencies for the cost of the complexity of the core. Can still be an option for games like Uno with less than 20 types of cards, but out of question for games like Yugioh with over 22000 different types of cards