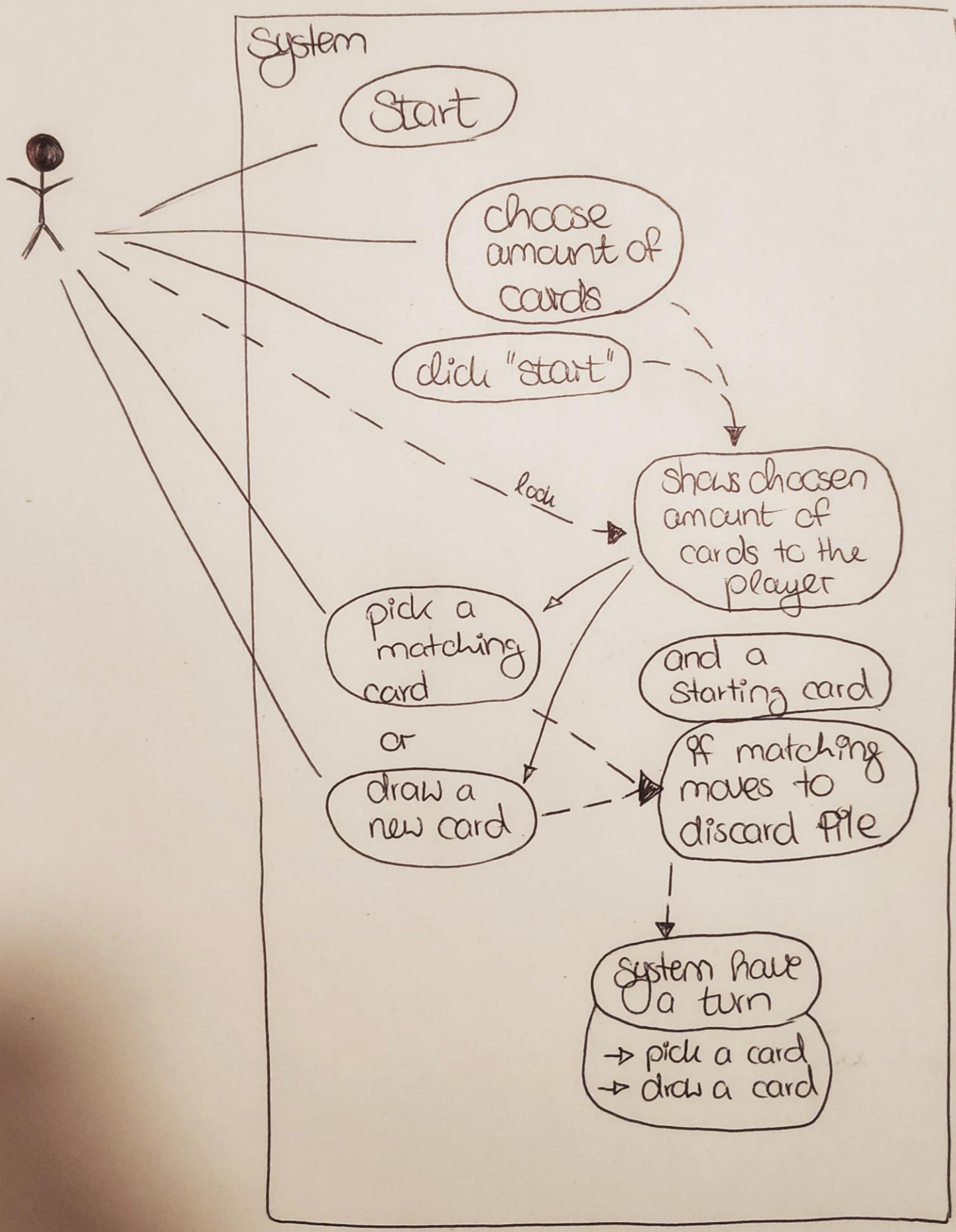
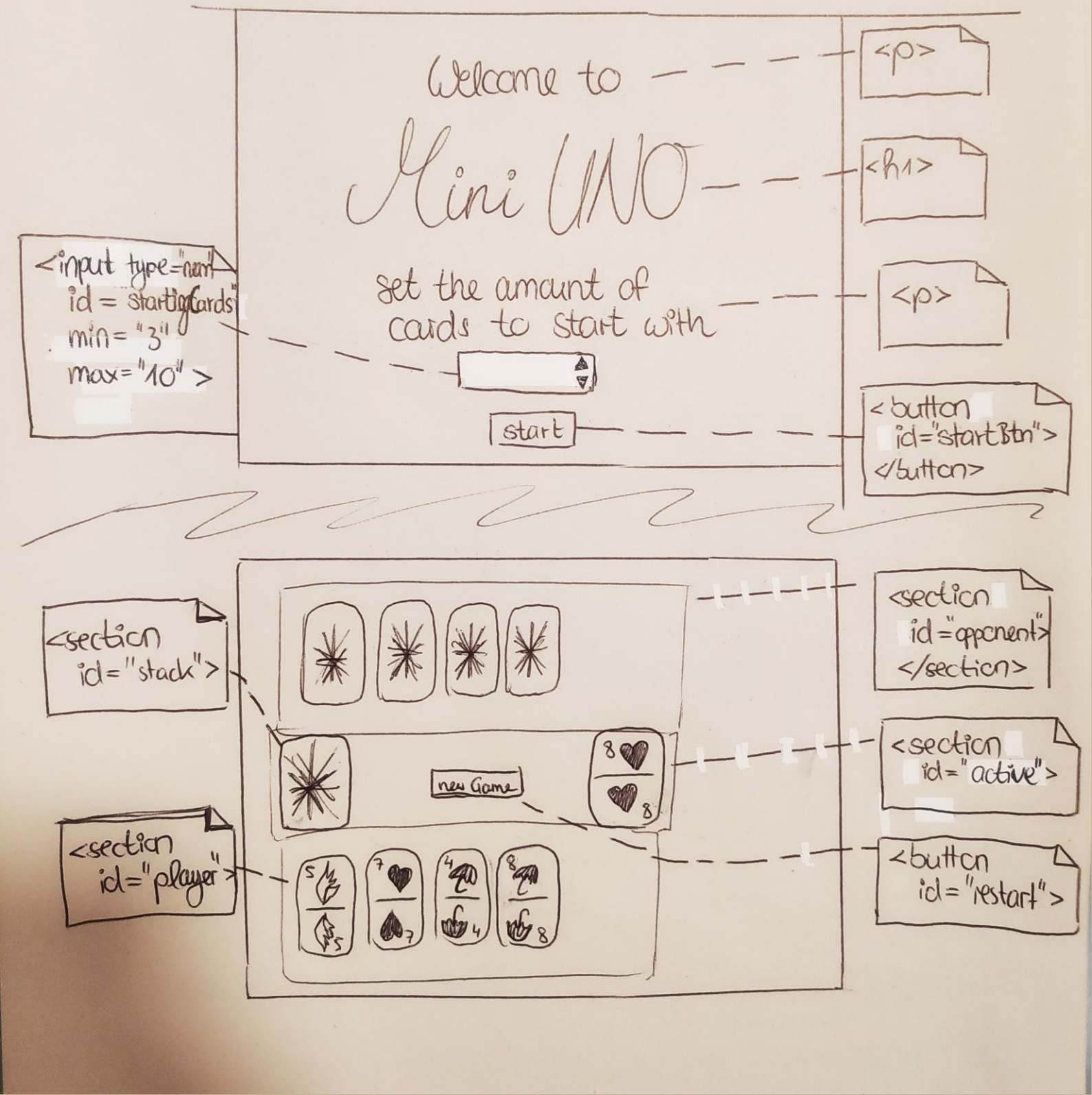
Use-Case-Diagramm „Mini UNO"



User-Interface „Mini UNO"



Activity-Diagramm „MiniUNO"

touch/click on start-button

createCards

setMain

```
interface Card { cardColor : string ;
                 cardValue : number }

cardStack : Card[] = []
playerCards : Card[] = []
computerCards : Card[] = []
discardStackCards : Card[] = []
```

setMain

```
w : HTMLElement = Element with ID "startWindow"
m : HTMLElement = Element with ID "main"
```

w style display = none → m style display = contents

createCards

```
getInput : HTMLElement = Element with ID startingCards
numberOfCards : number = parseFloat(getInput.value)
```

clearAll

generateCardStack

shuffle(cardStack)

push one card from CardStack into discardStackCards

updateHtml
createCards in Html

addEventListener onto stack

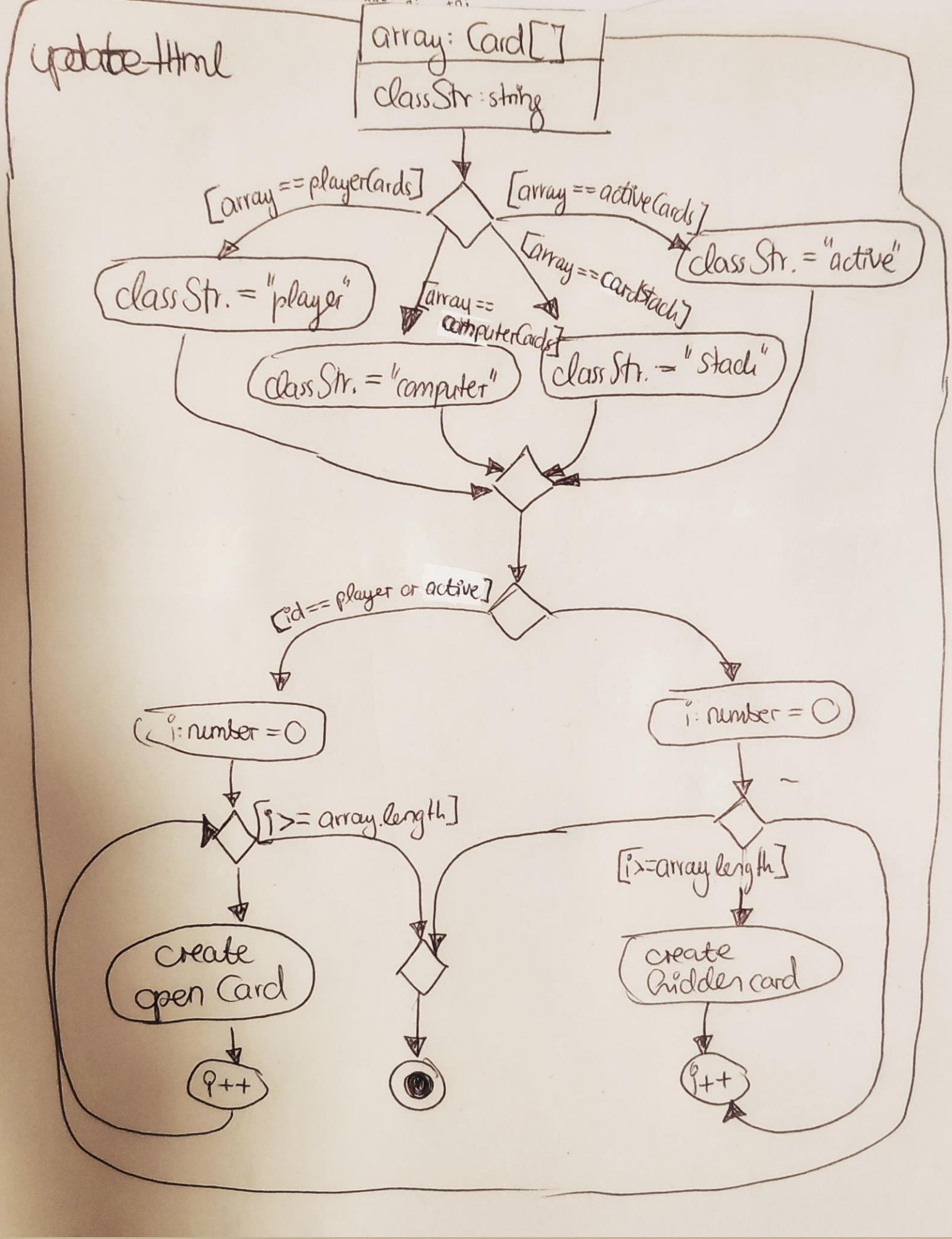
push 0 from CardStack into playerCards

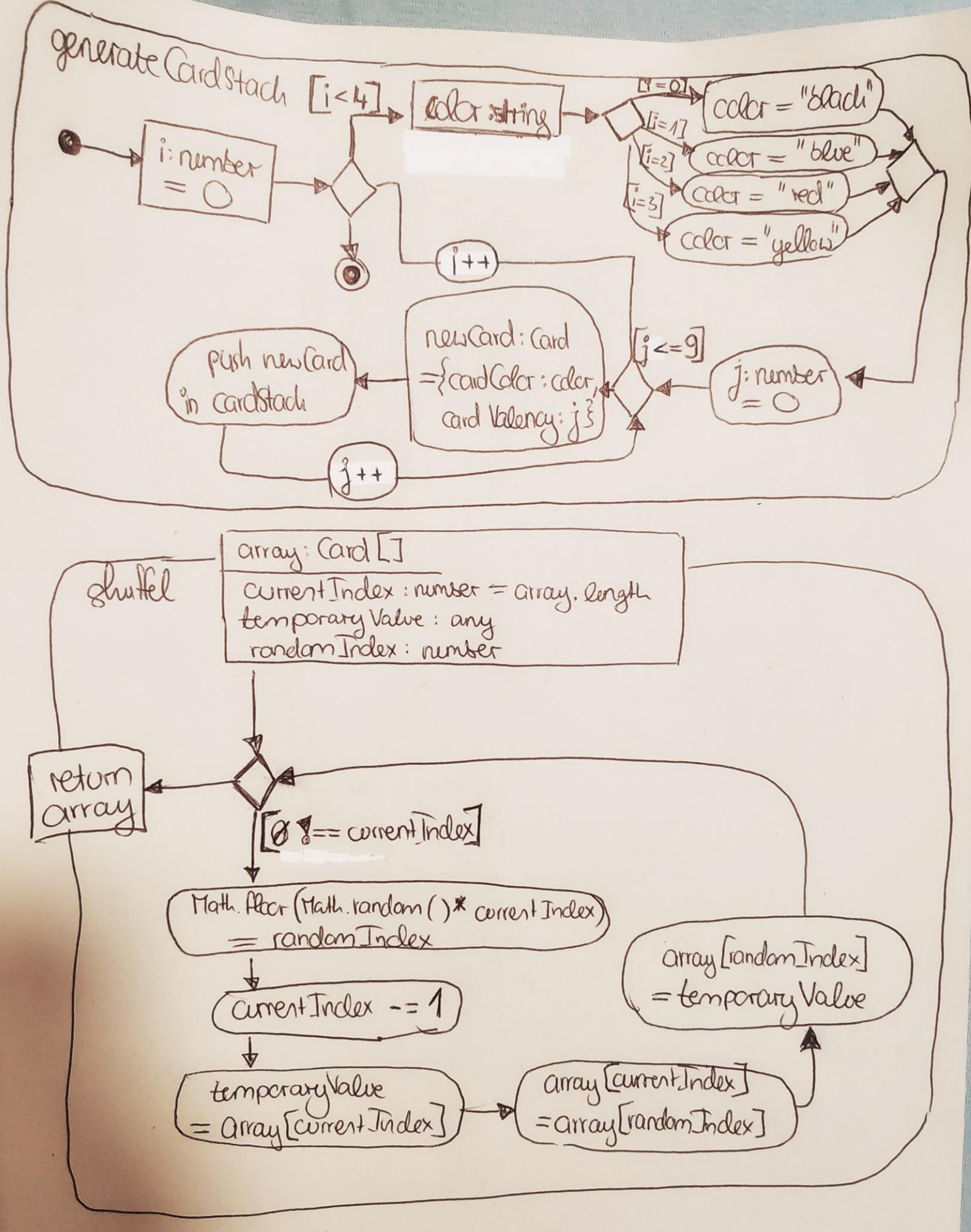
slice(0,1) from cardStack

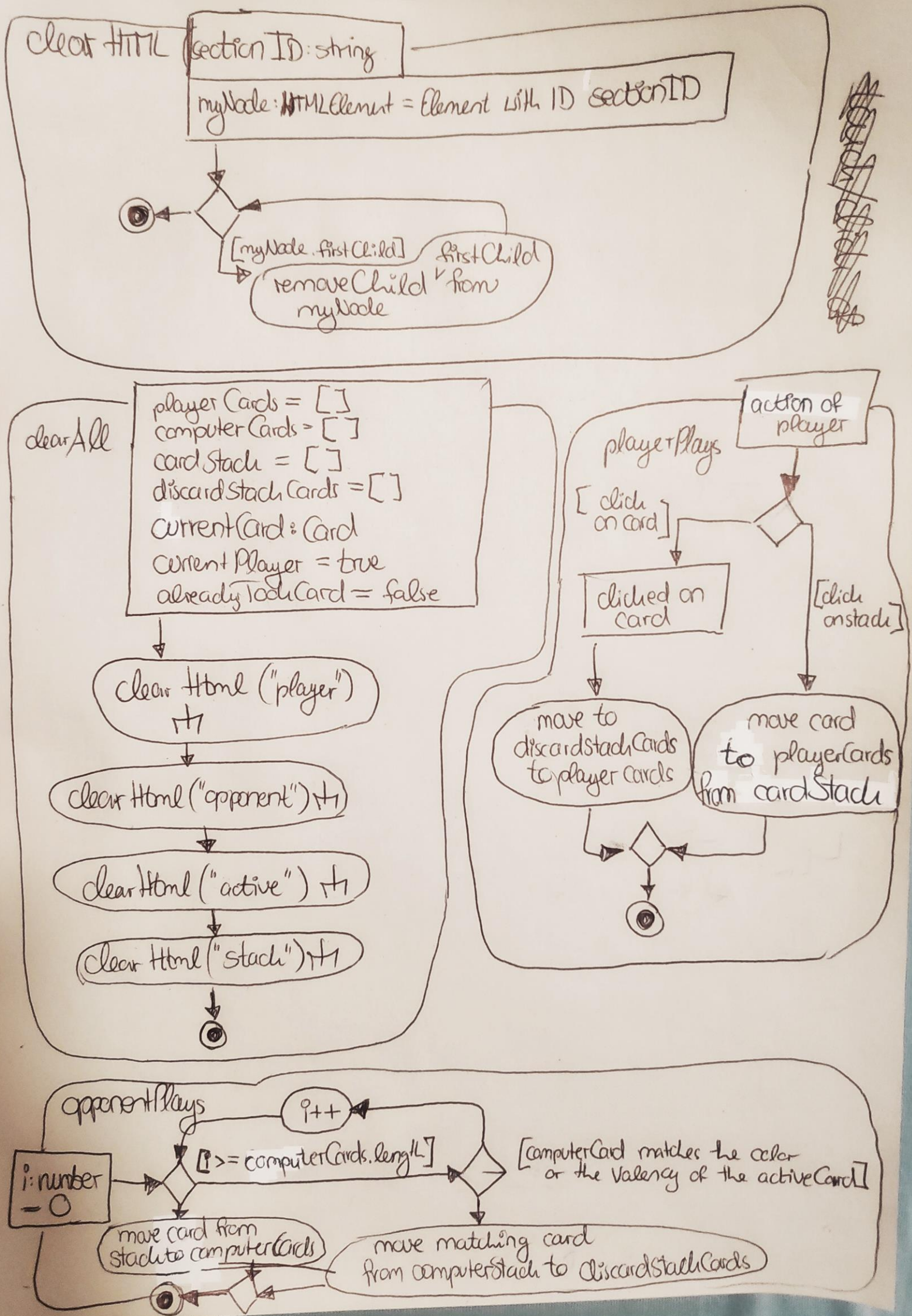
Same with opponentCards

[numberOfCards ≠ 0]

decrease numberOfCards


update Html
array: Card[]
classStr: string
[array == playerCards]
[array == activeCards]
[array == cardStack]
[array == computerCards]
classStr. = "player"
classStr. = "active"
classStr. = "computer"
classStr. = "stack"
[id == player or active]
i: number = 0
i: number = 0
[i >= array.length]
[i >= array length]
create open Card
create hidden card
i++
i++


generateCardstack
[i<4]
color :string
i: number = 0
[i=0]
color = "black"
[i=1]
color = "blue"
[i=2]
color = "red"
[i=3]
color = "yellow"
i++
newCard: Card ={cardColor: color, cardValency: j}
[j<=9]
push newCard in cardstack
j: number = 0
j++
shuffel
array: Card[]
currentIndex: number = array.length
temporaryValue: any
randomIndex: number
return array
[0 !== currentIndex]
Math.floor(Math.random()* currentIndex) = randomIndex
currentIndex -= 1
temporaryValue = array[currentIndex]
array[currentIndex] = array[randomIndex]
array[randomIndex] = temporaryValue


clear HTML
sectionID: string
myNode: HTMLElement = Element with ID sectionID
[myNode.firstChild]
firstChild
removeChild from myNode
clearAll
playerCards = []
computerCards = []
cardStack = []
discardStackCards = []
currentCard: Card
currentPlayer = true
alreadyTookCard = false
clearHtml("player")
clearHtml("opponent")
clearHtml("active")
clearHtml("stack")
playerPlays
action of player
[click on card]
clicked on card
[click on stack]
move to discardStackCards to playerCards
move card to playerCards from cardStack
opponentPlays
i: number = 0
i++
[i >= computerCards.length]
[computerCard matches the color or the valency of the activeCard]
move card from stack to computerCards
move matching card from computerStack to discardStackCards