

Anwendungsfall-Diagramm // Aufgabe 3 // Auflegestapel

Interface Karten

Wert: number

Symbol: number

Wert: 7, 8, 9, 10, 11, 12, 13, 14

Symbol: 0, 0, +, -

Benutzer

Interaktion

Main-Main System

Karte ziehen
1x p. n

Karte legen
1x p. n

Karten sortieren
(Hand)
Jederzeit

Karte ziehen

SPACE

Click Deck

n = zufällige Number,
Abhängig von Anzahl
Karten im Deck []

Frage Deck [n]
unserer Hand [] holen

Entferne Karte Deck[n]
aus Deck []

Hand
Sortieren

Click
Brett

Sortiere Hand []
anhand von Wert

Sortiere Hand []
anhand von Symbol

Karte legen

Handkarten []
Handkarten [x]

Click
Card

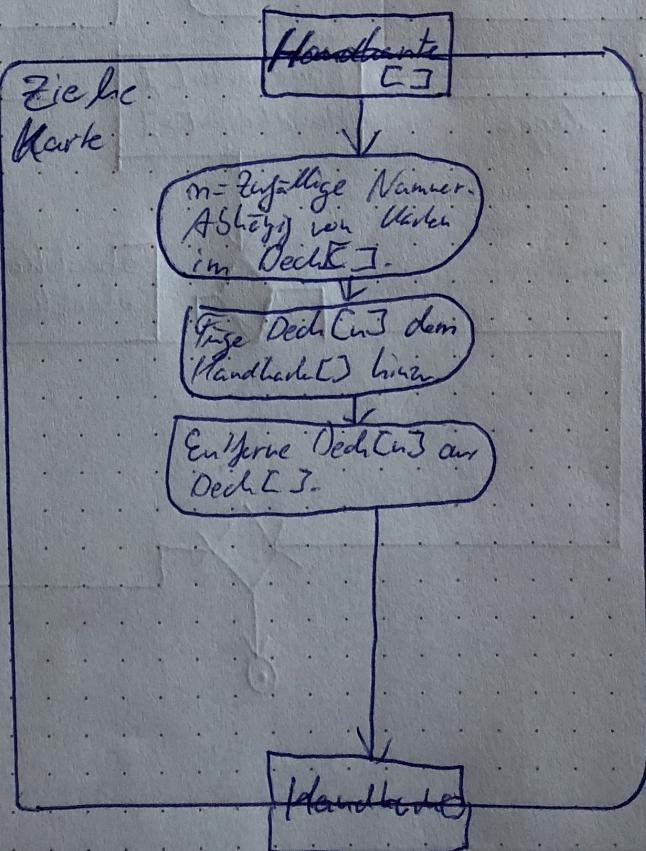
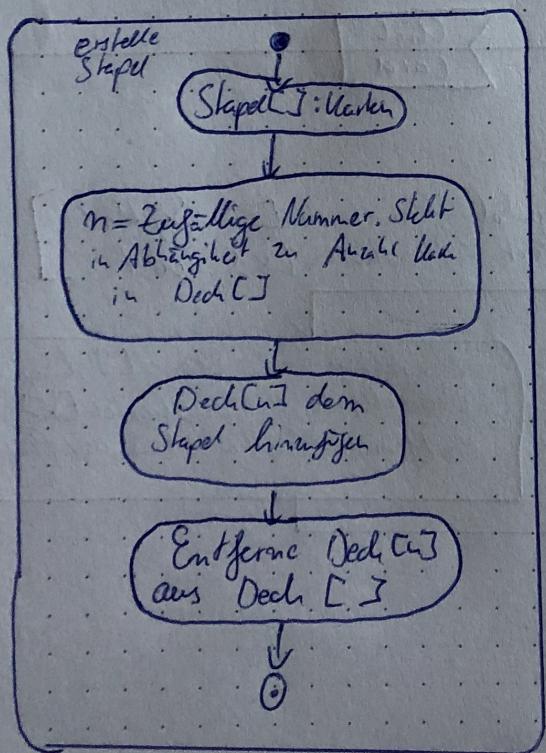
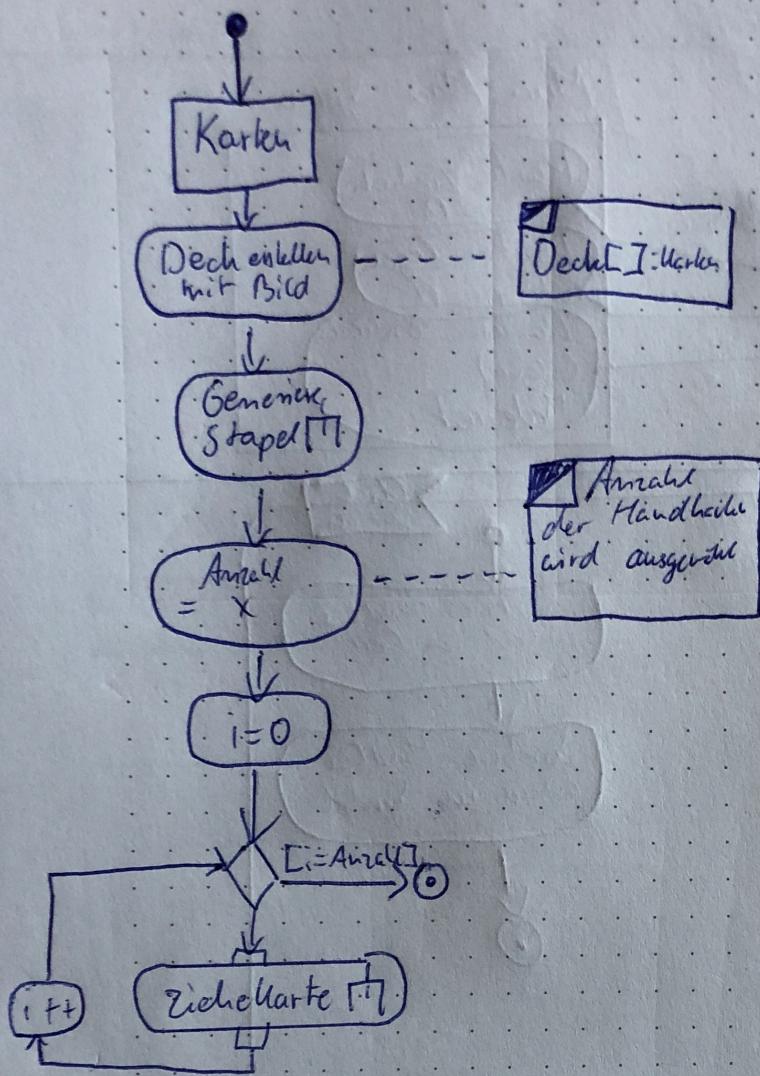
oberstKarte.Symbol = Handkarten [x].Symbol []
oberstKarte.Wert = Handkarten [x].Wert

oberstKarte wird Stapel [] hinzugefügt;
oberstKarte = Handkarten [x];
entferne Handkarten [x] aus Handkarten []

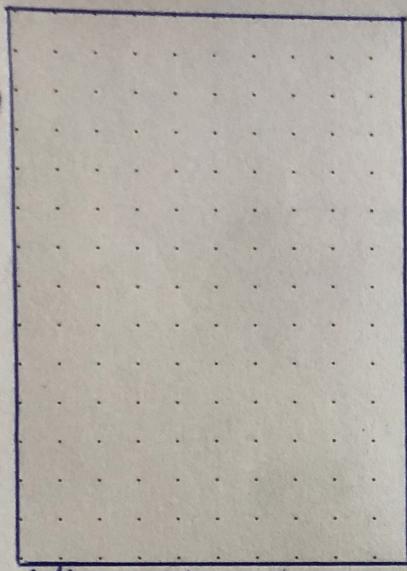
Aufgabe

I Man-Ma.

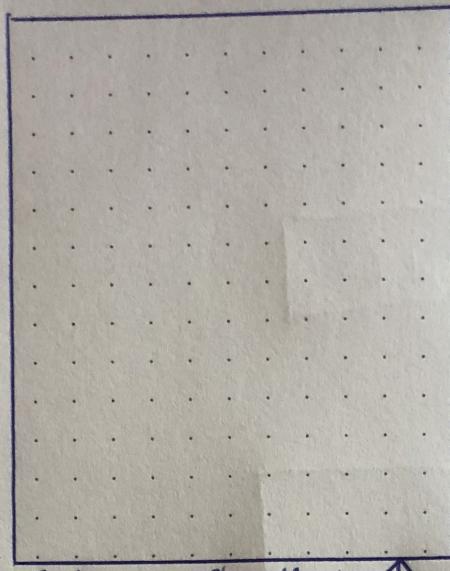
II Nachzieh-Stapel



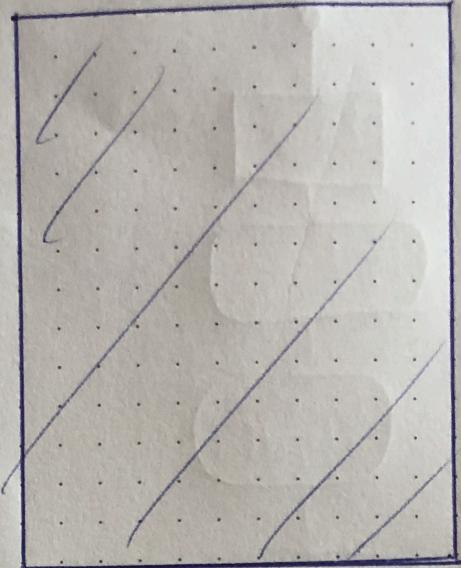
Slizze



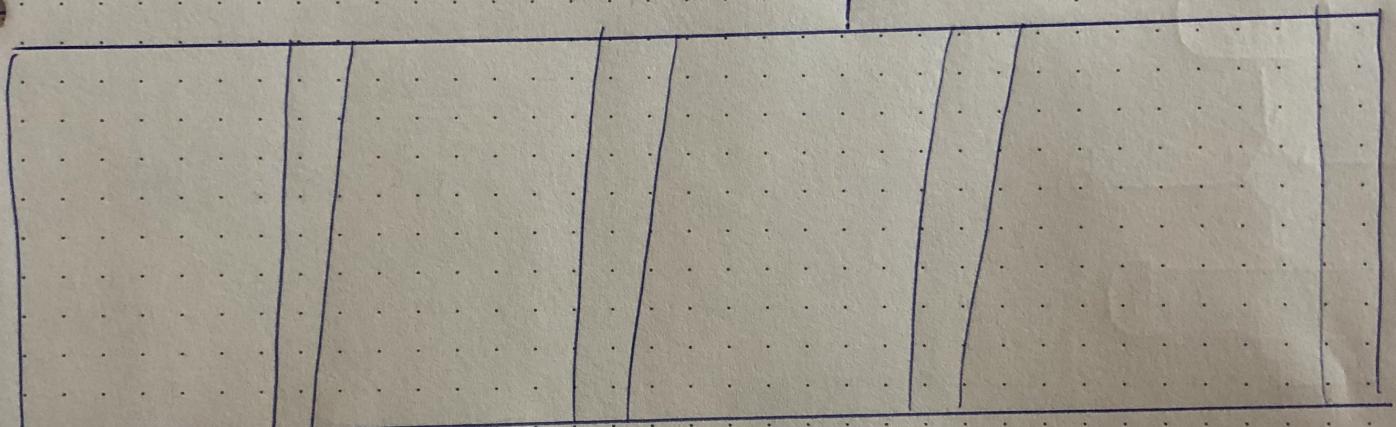
```
<div id="Deck">  
    ondlich(drawCard);</div>
```



```
<div id="Hand">
```



Wert
Symbol > Vergleich mit - Spieldat
Tippkarte - Ja/Nein



```
<div class="Hand">
```

Sor hren

Button ~~class~~
ondlich(sortHand);>

Hand.sort(jewelsila, {
 return a-b?});

> Klick auf Karte

Karte ziehen

> Klick auf Handkarte

Prüfen ob man legen darf

Karte ablegen

> Klick auf Sortieren

Karten sortieren