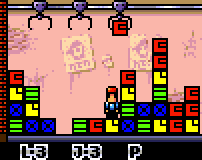
Konzept

Stack Attack:

Ein Siemens Handy Spiel. Man verschiebt kisten und bekommt dafür Punkte, sobald man eine Reihe Kisten vollgeschoben hat (Oder man paare von Kisten zusammen stehen)









<https://play.google.com/store/apps/details?id=com.pgames.saclass>

Kräne die zufällig (einzeln) farbige Kisten fallen lassen. ( Kräne und ihre Geschwindigkeit sind Schwierigkeitsgrad)

Rot/Grün/Blau /Gelb Kisten sind verschiebbar  
Schwarze Kisten (Ambos?) nicht verschiebbar

Spieler:  
normal: springt ein hoch kann eine Kiste verschieben  
groß: Springt 2 hoch, kann 1 schieben  
klein: Springt 1 hoch, kann 1 schieben

Ein Spieler ist doppelt so hoch wie eine Kiste  
Ein Spieler hat 1 Leben am Anfang  
Ein Leben wird verloren wenn eine Kiste auf den Kopf des spielers fällt

Powerups (möglichkeiten):  
Herz: ein zusätzliches leben  
Bombe : sprengt kisten um einen herum (muss gelegt werden)  
Bombe (Farbe): Sprengt alle Steine einer farbe  
Blitz: springt von kiste zu kiste und zerstört diese  
Farben Tausch ändert alle Kistenfarben  
Springbonus? (Feder)  
Stärkebonus? (Hantel)  
(Geschwindigkeitsbonus (schuhe))  
magnet (ermöglicht es kisten zu ziehen / zieht kisten mit Kran ?)

Kisten verschwinden wenn:  
Die Reihe voll ist  
3 gleichfarbige Kisten in einen 3er + verbund stehen  
Spezialfall Powerup : kann von Spieler eingesammelt werden, verhält sich sonst wie Block.

Punkte setzten sich daraus zusammen:  
Wie viel Linien abgearbeitet wurden  
Wie viel boxverbunde abgearbeitet wurden (gibt mehr Punkte!)  
Highscore & Highscore Board

1. **Einleitung** (inkl. Durchführende und Verantwortlichkeiten)
2. Erläuterung des **Spielkonzepts** und der Spielregeln, Ziel des Spiels, ggf. Levelkonzept
3. Überblicksartige Darstellung ihrer **Architektur** (MVC)
4. **Model**
   1. Überblick über ihr Model
   2. Aus dem Spielkonzept abgeleitete Entities (vermutlich Klassen)
   3. Erläuterung des aus den Spielregeln abgeleiteten konzeptionellen Models inkl. abgeleiteter Entities (vermutlich Klassen)
   4. Beschreibung des Zustands der konzeptionellen Modelentities
   5. Beschreibung des dynamischen Verhaltens des Models
   6. Erläuterung der zustandsverändernden Schnittstelle des Models (für den Controller)
   7. Erläuterung der zustandslesenden Schnittstelle des Models (für den View)
   8. ggf. Erläuterung wie Spiellevel definiert werden
5. **View**
   1. Überblick über ihren View
   2. Erläuterung des statischen HTML Dokuments zur Darstellung des Spielfelds
   3. Erläuterung der Model-View Schnittstelle
   4. Erläuterung der Wirkungsweise des View-Updates (dies sollte insbesondere beinhalten wie der DOM-Tree der in 1 definierten HTML-Dokuments verändert wird)
   5. Gestaltung des Spiels mittels CSS
6. **Controller**
   1. Überblick über ihren Controller
   2. Erläuterung der Model-Controller Schnittstelle
   3. Erläuterung der Interaktionsmöglichkeiten des Nutzers
   4. Erläuterung des Timers (welche Trigger erzeugen Zustandsübergänge des Models)
   5. Abbildung der Nutzerinteraktionen auf Modelinteraktionen (Controler - Model Schnittstelle, Teil 1)
   6. Abbildung der Timer-Events auf Modelinteraktionen (Controler - Model Schnittstelle, Teil 2)
7. **Zusammenfassung und Ausblick**

**Anlage:** ggf. Datenformat zur Beschreibung eines Spiellevels (erforderlich falls sie so vorgegangen sind)