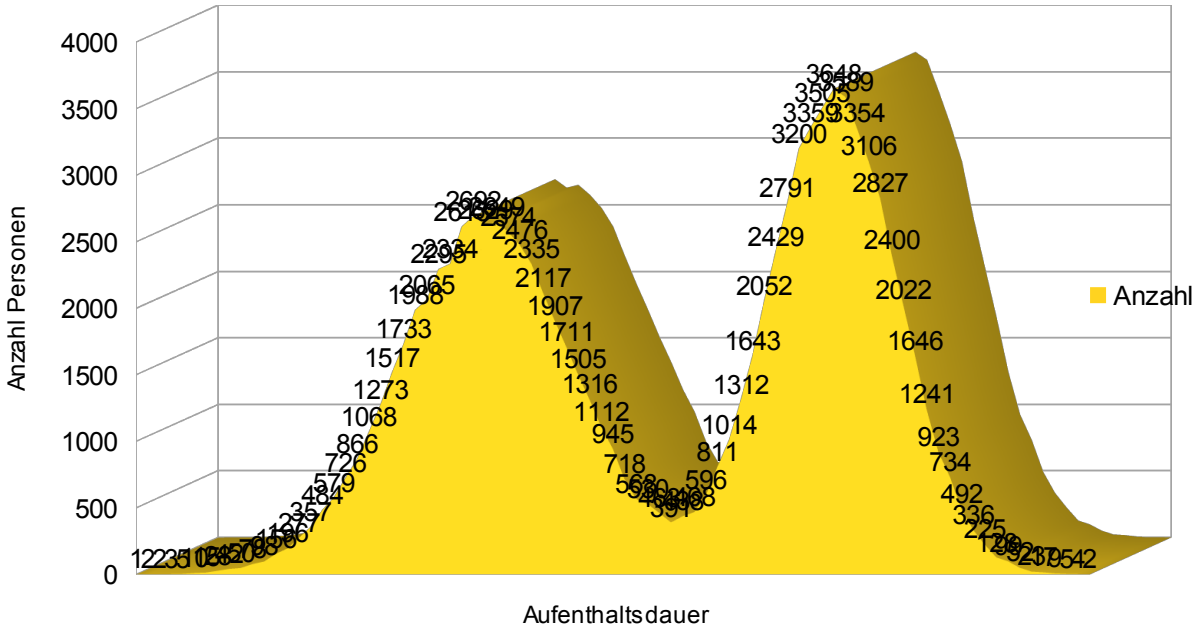
Intelligente Systeme, Aufgabe 1, Markus Krebs, Kim Schweikert

Aufgabenstellung:
Fitnessstudio möchte unmittelbar benachbarte Schrankbelegung zur Umzugszeit vermeiden.
Wahl zwischen: 20 neuen Schränken oder Belegungssoftware
Testsystem implementieren
Zufallsstrategie testen
Eigene Strategie entwickeln
Eigene Strategien testen

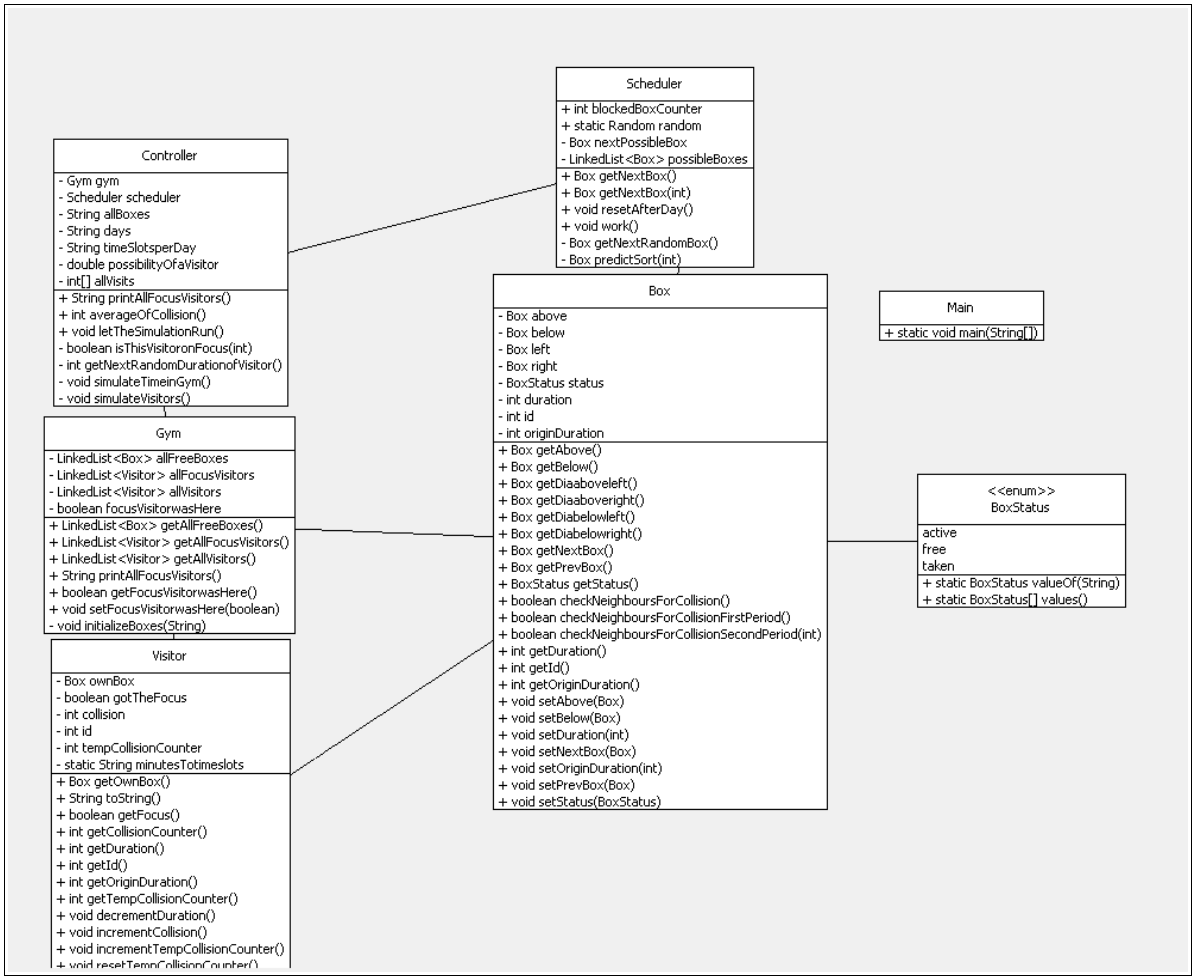
Gegeben:
150 Schränke in 2 Reihen(Variabel einstellbar)
Liste mit Besuchshäufigkeiten / Personen
Tage der Simulation (Variabel einstellbar)
Öffnungszeiten des Studios
Bisherige Verteilung: Zufall

Analyse der gegebenen Daten
Aufenthaltsdauer nach Häufigkeit



Intelligente Systeme, Aufgabe 1, Markus Krebs, Kim Schweikert

Implementierung / Klassendiagramm



Main

Führt die Simulation aus

Erstellt einen Controller mit parametrisierten Bedingungswerten

Controller

Simuliert den Tagesablauf im Gym mit n besuchern und m Schränken o mal

Aktuelle Besucher

Besucherauftreten / Wahrscheinlichkeit des auftretenden Benutzers

Aktuelle Zeit

Zählen von Kollisionen

Box

Enhält Referenzen auf die jeweils nächsten Schränke links, rechts, oben und Unten

Gibt es die referenz physikalisch nicht, ist sie null

Getter/Setter für Referenzen und Diagonalverweise über vorhandene Referenzen

Enthält die Zeit des Boxbenutzenden Visitors sobald die Box belegt ist

Erhält einen Boxstatus

BoxStatus

Free: Der Schrank ist frei

Intelligente Systeme, Aufgabe 1, Markus Krebs, Kim Schweikert

Active:	Der Schrank ist gerade in aktiver Benutzung
Taken:	Der Schrank ist belegt

Gym
- Enthält die Schränke, Öffnungszeit, aktuelle Besucher
- Initialisiert Schränke

Scheduler
- Controller fordert nächstmögliche Box vom Scheduler
- Weist die nächst beste Box, je nach Strategie zu

Visitor
- Enthält Referenz auf ihm zugeteilte Box
- Fokusperson ja/nein?
- Aufenthaltszeit (Total, Aktuell)

Intelligente Systeme, Aufgabe 1, Markus Krebs, Kim Schweikert

Gegenüberstellung der Strategien

Implementiert wurden 3 Strategien

Random:
Dem neuen Kunden wird eine zufällige freie Box zugewiesen

Round Robin (Simple):
Die freien Schränke werden nacheinander verteilt
Wurde ein Schrank vergeben, wird er ans Ende der Liste gehängt

Eigene Strategie:
Der nächste freie Schrank wird gesucht
3 Und-Verknüpfte Kriterien in 3 Iterationen, bestmöglichstes wird returnt:
- Im aktuellen Moment keine Kollision?
- Während der Umziehzeit des Besuchers keine Kollision?
- Während der Umziehzeit der anderen Besucher keine Kollision?

Kein Kriterium möglich? Random freien Schrank oder null returnen

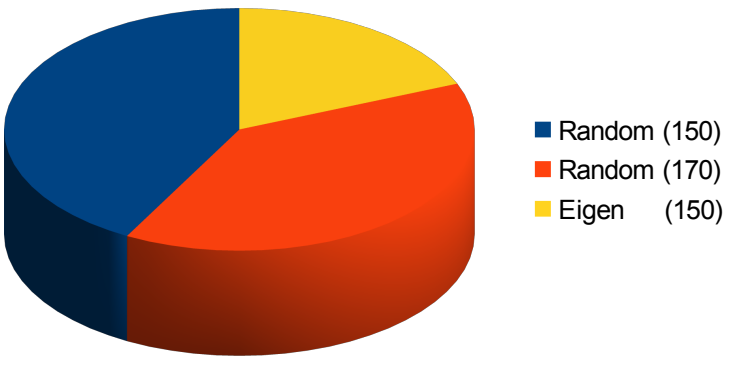
Round Robin in Vergleich zu Random doppelt so kollisionslastig. Nicht weiter verfolgt.

Ergebnis : 3 Simulationen pro Strategie mit je 10000 simulierten Tagen (ca. 83 Jahre):

Random (150 Schränke):
Simulation 1: Durchschnittlich 0,634 Kollisionen pro Tag : 634 Kollisionen gesamt
Simulation 2: Durchschnittlich 0.656 Kollisionen pro Tag : 656 Kollisionen gesamt
Simulation 3: Durchschnittlich 0.647 Kollisionen pro Tag : 647 Kollisionen gesamt

Random(150+20 Schränke):
Simulation 1: Durchschnittlich 0,521 Kollisionen pro Tag : 521 Kollisionen gesamt
Simulation 2: Durchschnittlich 0.542 Kollisionen pro Tag : 542 Kollisionen gesamt
Simulation 3: Durchschnittlich 0.589 Kollisionen pro Tag : 589 Kollisionen gesamt

Eigene Strategie (150 Schränke):
Simulation 1: Durchschnittlich 0,247 Kollisionen pro Tag : 247 Kollisionen gesamt
Simulation 2: Durchschnittlich 0.284 Kollisionen pro Tag : 284 Kollisionen gesamt
Simulation 3: Durchschnittlich 0.247 Kollisionen pro Tag : 247 Kollisionen gesamt



Intelligente Systeme, Aufgabe 1, Markus Krebs, Kim Schweikert

Ausblick:

Belegungsstrategie Bruteforcen?
- Strategie parametrisieren
- Random Seeds iterieren und mit random Parametern testen
- Random Seeds der besten Simulation speichern
- => Benötigt zu viel (unvorhersagbare) Rechenzeit

Besseres Predictionsystem

Laufzeitoptimierungen
- Iterationen vermeiden
- Liste im Scheduler sortieren

Anfangsvergabeverteilung ändern
- Abwechselnd links+rechts
- Mittig nach außen
- Einen Schrank überspringen