1 Organisatorisches

1.1 Team

- Reinhard Penn, s1110306019
- Bernhard Selymes, s1110306024

1.2 Aufteilung

- Reinhard Penn
  - Planung
  - Klassendiagramm
  - Implementierung der Klassen CarRental, ConcreteCar und Unterklassen
  - Testen aller Klassen
- Bernhard Selymes
  - Planung
  - Klassendiagramm
  - Implementierung der Klassen ICar, Decorator und Unterklassen
  - Dokumentation

1.3 Zeitaufwand

- geschätzte Mh: 12
- tatsächlich: Reinhard (5h), Bernhard (6h)

2 Systemspezifikation

Eine Sofware für die Verwaltung von Kraftfahrzeuge in einer Autovermietung soll entworfen werden. Die Kraftfahrzeuge gehören zu einer Klasse, die den Preis des Fahrzeugs bestimmt. Ein Kraftfahrzeug kann zusätzlich Sonderausstattungen haben, die zusätzlich etwas kosten.

3 Systementwurf

3.1 Klassendiagramm

3.2 Komponentenübersicht

- Klasse ”Object”:
  Basis aller Basisklassen.
- Interface ”ICar”:
  Schnittstellen der Funktionen.
- Klasse ”ConcreteCar”:
  Basisklasse für die einzelnen konkreten Kraftfahrzeuge.
- Klassen ”SmallCar, MiddleRangeCar, PremiumCar und SUV”:
  Konkrete Klassen von Kraftfahrzeugen.
- Klasse ”Decorator”:
  Basisklasse für die konkreten Sonderausstattungen.
- Klassen ”AirConditioner, Navi, Speedometer und Xenon”:
  Abgeleitet von MusicComponent.
- Klasse ”MusicCollection”:
  Abgeleitet von MusicComponent.
- Interface ”Visitor”:
  Abstrakte Basisklasse für Visitors.
- Klasse ”TimeVisitor”:
  Abgeleitet von Visitor. Visitor für Spieldauer.
- Klasse ”SearchVisitor”:
  Abgeleitet von Visitor. Visitor für die Suche nach Medien.
- Klasse ”PlayVisitor”:
  Abgeleitet von Visitor. Visitor für das Abspielen von Medien.
- Klasse ”MusicPlayer”:
  Klasse die die Medien verwaltet und alles mögliche mit ihnen machen kann.

4 Komponentenentwurf

4.1 Klasse ”Object”

Abstrakte Basisklasse aller Klassen. Von ihr werden alle anderen Klassen abgeleitet. Beinhaltet einen virtuellen Destruktor.

4.2 Enumeration ”TMusicKind”

- TSong
- TAlbum
- TMusicCollection

4.3 Klasse ”MusicFactory”

Kann Musikmedien dynamisch anlegen und wieder freigeben. In einer Liste werden die Referenzen auf die Objekte gespeichert. Die Methoden legen die Objekte je nach Medientyp mit den übergebenen Parametern an und fügen sie der Liste hinzu. Die Factory darf nur nach dem Musikplayer gelöscht werden, es kann nicht überprüft werden ob die Referenzen auf Objekte noch im Musicplayer gespeichert sind, weil keine Verbindung zwischen den beiden Klassen besteht.

4.4 Klasse ”MusicComponent”

Bietet die Schnittstellen für die Methoden ”Accept”, ”GetType”, ”GetNumberOfEntries” und ”AddMusic”, ”GetName” wird implementiert. Hat zwei Member die Namen und Type speichern.

Methode ”GetName”:
Schnittstelle:
Rückgabetyp: string
Get Funktion.

4.5 Klasse ”Song”

Hat Member die den Namen des Albums und des Interpreten und die Spieldauer in Sekunden speichert. Wir haben uns für Sekunden entschieden, weil die Übergabe leichter ist (keine extra Struktur), das Addieren von mehreren Zeiten umständlich wäre und die Sekunden für die Ausgabe und weitere Verwendung leicht umgerechnet werden können. Hat einige Getter Funktionen.

Methode ”Accept”:
Schnittstelle:
Parameter: Visitor*
Rückgabetyp: void
Ruft die Funktion Visit vom übergebenen Visitor mit sich selbst auf.

4.6 Klasse ”Album”

Hat Member die den Namen des Interpreten und eine Liste die die Lieder des Albums speichert.
Hat einige Getter Funktionen.

Methode ”Accept”:
Schnittstelle:
Parameter: Visitor*
Rückgabetyp: void
Ruft die Funktion Visit vom übergebenen Visitor mit sich selbst auf.

Methode ”ForwardVisitor”:
Schnittstelle:
Parameter: Visitor*
Rückgabetyp: void
Ruft für alle Lieder die Funktion ”Accept” mit dem Visitor auf.

Methode ”GetTime”:
Schnittstelle:
Rückgabetyp: void
Ruft für alle Lieder die Methode ”Accept” mit dem Visitor auf.

Methode ”AddMusic”:
Schnittstelle:
Parameter: MusicComponent*
Rückgabetyp: void
Fügt ein Lied (und nur ein Lied) zur Liste hinzu.

4.7 Klasse ”MusicCollection”

Hat eine Liste die alle Musikmedien, die in der Kollektion enthalten sind.

Methode ”Accept”:
Schnittstelle:
Parameter: Visitor*
Rückgabetyp: void
Ruft die Funktion Visit vom übergebenen Visitor mit sich selbst auf.

Methode ”ForwardVisitor”:
Schnittstelle:
Rückgabetyp: void
Ruft für alle Musikmedien die Methode ”Accept” mit dem Visitor auf.

Methode ”GetNumberOfEntries”:
Schnittstelle:
Rückgabetyp: size_t
Ruft für alle Musikmedien die Funktion ”GetNumberOfEntries” auf und addiert sie.

4.8 Interface ”Visitor”

Definiert die Schnittstellen der Methoden.

Methoden ”Visit”:
Schnittstelle:
Parameter: Song* oder Album* oder MusicCollection*
Rückgabetyp: void
Pure virtual function.

4.9 Klasse ”TimeVisitor”

Hat einen Member der die gesamte Dauer speichert.

Methode ”Visit”:
Schnittstelle:
Parameter: Song*
Rückgabetyp: void
Addiert zum Gesamtdauermember die Dauer vom Lied.

Methode ”Visit”:
Schnittstelle:
Parameter: Album*
Rückgabetyp: void
Ruft die Funktion ”ForwardVisitor” vom Album mit sich selbst (Visitor) auf.

Methode ”Visit”:
Schnittstelle:
Parameter: MusicCollection*
Rückgabetyp: void
Ruft die Funktion ”ForwardVisitor” von der Kollektion mit sich selbst (Visitor) auf.

4.10 Klasse ”SearchVisitor”

Hat einen Member der den gesuchten Namen speichert und eine Liste die Referenzen zu den gefundenen Objekten speichert.

Methode ”Visit”:
Schnittstelle:
Parameter: Song*
Rückgabetyp: void
Schaut ob der gesuchte Name Teil des Namens vom Lied ist und speichert in ggf. in der Liste.

Methode ”Visit”:
Schnittstelle:
Parameter: Album*
Rückgabetyp: void
Ruft die Funktion ”ForwardVisitor” vom Album mit sich selbst (Visitor) auf.

Methode ”Visit”:
Schnittstelle:
Parameter: MusicCollection*
Rückgabetyp: void
Ruft die Funktion ”ForwardVisitor” von der Kollektion mit sich selbst (Visitor) auf.

4.11 Klasse ”PlayVisitor”

Hat einen Zähler der die Nummer der Lieder bestimmt.

Methode ”Visit”:
Schnittstelle:
Parameter: Song*
Rückgabetyp: void
Gibt die Daten des Liedes formatiert auf der Konsole aus und erhöht den Zähler.

Methode ”Visit”:
Schnittstelle:
Parameter: Album*
Rückgabetyp: void
Legt einen TimeVisitor an. Dieser wird vom Album accepted. Die Informationen des Albums werden auf der Konsole ausgegeben. Danach wird der Visitor den Elementen im Album weitergegeben.

Methode ”Visit”:
Schnittstelle:
Parameter: MusicCollection*
Rückgabetyp: void
Legt einen TimeVisitor an. Dieser wird von der Kollektion akkzeptiert. Die Informationen der Kollektion werden auf der Konsole ausgegeben. Danach wird der Visitor den Elementen in der Kollektion weitergegeben.

4.12 Klasse ”MusicPlayer”

Hat einen Liste mit Musik-Medien.

Methode ”GetTime”:
Schnittstelle:
Parameter: MusicComponent* const
Rückgabetyp: size_t
Prüft zuerst ob das Element überhaupt vorhanden ist. Dann wird via eines TimeVisitors die Dauer des Elements ermittelt.

Methode ”GetTotalTime”:
Schnittstelle:
Rückgabetyp: size_t
Via eines TimeVisitors die Dauer aller Elemente ermittelt.

Methode ”Play”:
Schnittstelle:
Rückgabetyp: void
Via eines PlayVisitors wird die gesamte Abspielliste ausgegeben.

Methode ”Search”:
Schnittstelle:
Parameter: string const&
Rückgabetyp: void
Via eines SearchVisitors wird der Name überall gesucht. Der Name der gefundenen Elemente wird danach ausgegeben.

5 Source Code

6 Testausgaben

```
Testcase0: Empty testcase with NULL pointer.
Add: error in MusicPlayer::Add(): no valid pointer
 ...done
GetTime: error in MusicPlayer::GetTime(): no valid pointer
 ...done
GetTotalTime:  ...done
Play:
 ...done
Remove: error in MusicPlayer::Remove(): no valid pointer
 ...done
Search: error in MusicPlayer::Search(): no valid name
 ...done
Delete MusicPlayer:  ...done
Delete MusicFactory:  ...done


Testcase1: Normal testcase with valid objects.
Create Objects:  ...done
Put Albums together:  ...done
Put an album into an album: Error in Album::AddMusic:
Tried to add a wrong object type to the song list
 ...done
Put Collection together:  ...done
Put a collection into itself: Error in MusicCollection::AddMusic:
no valid pointer
 ...done
Add stuff to the player:  ...done
GetTime: error in MusicPlayer::GetTime(): component doesnt exist in list
0 seconds ...done
GetTotalTime: 1359 seconds in player ...done
Play:
1. Staring At The Sun << 02:12 << The Offspring
Collection: MyPlayList (3 Song(s))  << 10:51
Album: Americana (2 Song(s))  << 06:08
2. Staring At The Sun << 02:12 << The Offspring
3. Have You Ever << 03:56 << The Offspring
4. Psychosocial << 04:43 << Slipknot
Album: Conspiracy Of One (1 Song(s))  << 03:28
5. Living In Chaos << 03:28 << The Offspring
Album: Americana (2 Song(s))  << 06:08
6. Staring At The Sun << 02:12 << The Offspring
7. Have You Ever << 03:56 << The Offspring
 ...done
Search: found medias: (search for "in")
Staring At The Sun
Staring At The Sun
```

```
Living In Chaos
Staring At The Sun
 ...done
Remove:  ...done
Play after remove:
1. Staring At The Sun << 02:12 << The Offspring
Album: Conspiracy Of One (1 Song(s))  << 03:28
2. Living In Chaos << 03:28 << The Offspring
 ...done
Delete MusicPlayer:  ...done
Delete MusicFactory:  ...done
```