Prof.Dr.H.P.Weber	Praktische Prüfung PAD1 WS 13/14	h_da HOCHSCHULE DARMSTADT UNIVERSITY OF APPLIED SCIENCES fbi FACHBEREICH INFORMATIK

Name: MatNr: User:

Hinweis: Denken Sie während der Klausur auch an die **Datensicherung!**
Empfehlung: Arbeiten Sie in kleinen Schritten und sorgen Sie dafür, dass Sie jederzeit ein lauffähiges Anwendungssystem haben.

Allgemeine Vorgaben:

- **Benutzen Sie ausschließlich MS Visual Studio Professional 2013.**
- Verwenden Sie möglichst keine globalen Variablen oder Objekte und nur solche, die als **const** deklariert sind.
- Programmieren Sie C++-standardkonform (z.B. keine Verwendung von **<conio.h>**, kein Aufruf von **system**, keine Ein-/Ausgabe im C-Stil!)
- Verwenden Sie die vorgegebenen Bezeichner; zusätzliche eigene Bezeichner sind 'sprechende Bezeichner'.
- Beachten Sie das 'Principle of least privilege' (z.B. 'const-correctness').
- Die öffentliche Schnittstelle der Klassen darf durch **operator<** und **operator==**-Funktionen erweitert werden.
- Alle Variablen/Objekte werden zur Übersetzungszeit erzeugt und direkt mit ihrem Namen angesprochen, d.h. in Ihrem Programm kommt kein **new**-Operator und keine Zeiger-Variable vor.
- Trennung von Klassendefinitionen und –implementierungen in .h- und .cpp-Dateien,

Zur Bewertung:
Insgesamt können Sie 100 Punkte bekommen.
Die Note 'sehr gut' (1,0 und 1,3) wird für mindestens 93 Punkte vergeben.
Bestanden haben Sie mit mindestens 50 Punkten.
Werden die 'Allgemeinen Vorgaben' nicht eingehalten, gibt es Punktabzug.

Aufgabenstellung:
Sie sollen ein Ticketverkaufsprogramm für ein Theater mit 10 Sitzreihen und 15 Plätzen pro Reihe erstellen. Ihr Programm soll die Klassen **Theatre** und **Ticket** (s. Klassendiagramme) enthalten:

Theatre
+ numberOfTickets: int - tickets: Ticket[numberOfTickets]
+ Theatre() + sellTicket(Ticket): bool + sortByPrice(): void + sortBySeat(): void + numberOfSoldTickets(): int + totalRevenues(): double + soldTickets(): string + showSeats(): string

Ticket
- row: int - seat: int - price: double - occupied: bool
+ Ticket() + Ticket(int,int,double,bool) + isOccupied(): bool + toString(): string + *<erforderliche setter/getter>*

Besonderheiten der Datenelemente und Elementfunktionen sind:

- Der Standardkonstruktor von **Ticket** soll einen unbesetzten Sitzplatz erzeugen (Die Reihennummer **row** und die Platznummer **seat** haben den Wert **0**, der Ticketpreis **price** den Wert **0.0** und die Information **occupied** - ob der Platz besetzt ist - den Wert **false**). Ein zweiter Konstruktor übernimmt Werte für **row**, **seat**, **price** und **occupied** von außen.
- **toString** soll alle Datenelemente eines **Ticket**-Objekts in einer geeigneten Formatierung als **string** zurückgeben.
- Das **Theatre** hat 10 Sitzreihen und 15 Plätze pro Reihe, d.h. die (konstante!) Klassenvariable **numberOfTickets** hat den Wert 150.
- Der Standardkonstruktor von **Theatre** soll ein unbesetztes Theater erzeugen, dessen Tickets (d.h. also die **Ticket**-Objekte im Array **tickets**) aber schon die richtigen Reihennummern und Platznummern enthalten. Außerdem soll der Preis für jedes Ticket nach folgendem System gesetzt werden: Die äußeren Plätze in der letzten Reihe (also Platz 1 und 15 in der Reihe 10) kosten beide 10 Euro. Für jede Reihe weiter vorn erhöht sich der Ticketpreis um 1 Euro und gleichzeitig erhöht sich der Ticketpreis für jeden Sitz in Richtung Mitte um 0,50 Euro. Der teuerste Platz im Theater ist also der Platz 8 in der ersten Reihe mit 22,50 Euro.

- **sellTicket** soll ein vom Anwendungsprogramm übergebenes **Ticket**-Objekt auf folgende Weise verarbeiten:
    - Falls der Platz mit der **row** und dem **seat** des übergebenen Objekts in dem **Theatre**-Objekt schon besetzt ist, soll **false** zurückgegeben und das übergebene Objekt nicht weiter verarbeitet werden.
    - Falls nicht, wird der **occupied**-Wert des passenden Platzes auf **true** gesetzt und außerdem **true** zurückgegeben.
- **sortByPrice** soll die **Ticket**-Objekte innerhalb des Arrays **tickets** nach fallendem Preis sortieren. Hierbei soll das Sortierverfahren 'Direktes Auswählen (Selection-Sort)' benutzt werden.
- **sortBySeat** soll die **Ticket**-Objekte innerhalb des Arrays **tickets** aufsteigend nach Reihennummer und innerhalb der gleichen Reihe nach Platznummer sortieren. Sie können ein Sortierverfahren Ihrer Wahl (außer 'Selection-Sort', s.oben) benutzen.
- **numberOfSoldTickets** soll die Anzahl der verkauften Tickets ermitteln und zurückgeben.
- **totalRevenues** soll die Gesamteinnahmen in Euro ermitteln und zurückgeben.
- **showSeats** soll den Saalplan des Theaters im unten angegebenen Format als **string** zurückgeben. Hierbei wird für einen besetzten Sitz ein **'x'** ausgegeben und für einen freien Sitz ein **'-'**.

```
Verkaufte Tickets: 50
Gesamteinnahmen: 812.50 Euro
Saalplan:
    1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
 1  -  -  -  -  -  -  x  -  x  x  -  -  -  x  x
 2  x  -  -  -  x  -  -  x  x  -  x  -  -  -  -
 3  -  -  x  x  -  -  -  x  -  x  x  -  -  -  -
 4  x  -  x  -  -  -  x  x  -  -  -  -  -  x  -
 5  -  -  -  -  -  x  x  -  -  -  -  x  -  -  -
 6  -  -  x  -  -  -  -  x  x  x  x  -  -  -  x
 7  x  -  x  -  -  -  x  x  -  -  -  -  x  x  -
 8  -  x  -  x  -  -  x  -  -  -  -  -  -  x  x
 9  -  -  x  -  -  x  x  -  -  x  -  x  -  x  x
10  -  -  -  -  -  x  -  -  -  -  x  x  -  -  -
```

Das **Anwendungsprogramm (main)** soll die beiden Klassen testen. Dazu sollen der Reihe nach folgende Aktionen ablaufen (eine Menüsteuerung soll also nicht realisiert werden):

- Erzeugung eines **Theatre**-Objektes und Ausgabe des Saalplans des leeren Theaters.
- Einfügen von **Ticket**-Objekten für zufällig ausgewählte Plätze in das **Theatre**-Objekt, bis insgesamt 50 Plätze besetzt sind. (Falls der gewählte Platz schon besetzt ist, soll eine entsprechende Meldung ausgegeben werden und ein weiterer Platz zufällig gewählt werden).
- Ausgabe der Anzahl der verkauften Tickets, der erzielten Gesamteinnahmen und des Saalplans des Theaters (entsprechend der Abbildung oben).
- Ausgabe aller verkauften Tickets in Form einer nach dem Ticketpreis sortierten Liste.
- Danach: Ausgabe aller verkauften Tickets in Form einer nach Reihe und Sitzplatz sortierten Liste.

Ergebnis:

Als Ergebnis erwarte ich in jedem Fall ein **ausführbares** Anwendungssystem.
Speichern Sie alle Ihre Projektdateien mit Ausnahme des Debug-Verzeichnisses in Ihrem persönlichen Netzverzeichnis.
Das Anwendungssystem muss sich mit diesen Dateien auf "Knopfdruck" erzeugen lassen.
Löschen Sie Ihr Projekt am Klausurende **nicht** und fahren Sie den Rechner **nicht** herunter.
Lassen Sie das **Aufgabenblatt** (mit Namen, Matrikel- und Usernummer) an Ihrem Platz liegen.

Bewertung (Punkte):	**maximal**	**erreicht**
Klassendefinitionen	12	
Methoden **Ticket**	10	
Konstruktor **Theatre**	12	
sellTicket	8	
sortByPrice	10	
sortBySeat	10	
numberOfSoldTickets	5	
totalRevenues	5	
soldTickets	4	
showSeats	12	
main	12	

Gesamtpunkte:	**Note:**