Klausur C++ #2

Jahr: 2001; Dozent: Dipl.Ing. Sorber

1. Aufgabe:

Wie viele Konstruktoren kann eine Klasse besitzen? Begründung.

Beliebig viele, unterschieden werden sie durch die Parameterliste.

2. Aufgabe:

Was ist der Unterschied zwischen einem protected Member und einem private Member einer Klasse. (Member: Datenelement oder Methode einer Klasse)

protected Zugriff auf protected Daten und Methoden kann auch aus anderen Klassen zugegriffen werden.

private Zugriff auf private Daten und Methoden kann nur innerhalb der Klasse erfolgen.

3. Aufgabe:

Wie und warum wird der Bereichsoperator :: (scope) in Klassendefinitionen verwendet? Geben Sie ein Beispiel.

Der scope Operator wird zum definieren von Methoden außerhalb der Klasse genutzt.

```
class Cache
{
     private:   int cache;
     public:    disable ();
};

void Cache::disable ()
     {
      if (cache==1) cache.disable();
     }
```

4. Aufgabe:

Implementieren Sie eine Klasse Zeit. Jede Instanz dieser Klasse repräsentiert eine spezifische Uhrzeit und speichert Stunden, Minuten und Sekunden als int-Werte. Schreiben Sie einen Konstruktor, eine Methode advance (int h, int m, int s) zur Fortschreibung der akt. zeit einer existierenden Instanz, eine Methode reset(int h, int m, int s) zum Rücksetzen der akt. Zeit einer existierenden Instanz und eine Methode print() zur Anzeige der Zeit einer existierenden Instanz. Berücksichtigen sie jeweils die Überläufe bei Sekunden, Minuten (jeweils 60) und bei Stunden (24) durch eine Methode normalize().

```
# include <iostream.h>
# include <conio.h>

class zeit
{
 private:  int sec, min, hr;

 public:   zeit () {sec=0; min=0; hr=0;}

            void normalize ()
            {
                if (sec >= 60){sec-=60; min++;}
               if (min >= 60){min-=60; hr++;}
               if (hr >= 24){hr-=24;}
            }

            void advance (int h, int m, int s)
            {
               hr+=h;
               min+=m;
               sec+=s;

               normalize();
            }

            void reset (int h, int m, int s)
            {
               hr=h;
               min=m;
               sec=s;

               normalize();
            }

            void print ()
            {
                cout << hr << " : " << min << " : " << sec << endl;
            }
};

int main ()

{
 zeit time1;

 time1.advance(25,70,80);
 time1.print();
 time1.normalize();
 time1.print();
 time1.reset(0,0,0);
 time1.print();
 getch ();

 return 0;
}
```

5.Aufgabe:

Wie verhalten sich Standardkonstruktoren in einer Vererbungshierarchie? Wie erfolgen Ihre Aufrufe?

Sie werden immer alle aufgerufen. Der Compiler führt sie automatisch bei der Objekterstellung aus. Man kann sie nicht einzeln direkt aufrufen.

6.Aufgabe:

Warum wird die Verwendung der Operatoren << und >> „formatierte“ und die Verwendung der Funktionen put(), get(), write(), und read() „unformatierte“ Ein- und Ausgabe genannt?

Bei >> und << heißt es formatierte Ausgabe, da Sonderzeichen wie "\n" in die Ausgabe geschreiben werden können die das Verhalten beeinflussen. Funktionen wie put(),get(), ... können nur Werte übernehmen => man kann die Aus- bzw. Eingabe nicht beeinflussen.

7.Aufgabe:

Schreiben Sie die Quelltextzeilen, die cout so formatieren, dass float-Werte in wissenschaftlicher, also Expotenzialschreibweise mit 12-stelliger Genauigkeitig ausgegeben werden können.

```
cout.setf (ios::scientific);
cout precision (12);
```

8.Aufgabe:

Welche Werte besitzen die Elemente eines Arrays, wenn es deklariert wird und die Initialisierungsliste weniger Werte als die Anzahl der Elemente dieses Arrays umfasst.

Die Elemente besitzen den Wert 0.

9.Aufgabe:

Formulieren Sie die Bildschirmausgabe des folgenden Programmfragments:

```
for(int i=0;i<8;i++)
 if (i%2==0) cout<<i+3<<endl;
 else if(i%3==0) cout<<2*i-1<<endl;
 else if(i%5==0) cout<<i*i<<endl;
 else cout<<i<<endl;
```

```
3
1
5
5
7
25
9
7
```

10.Aufgabe:

Welche Bildschirmanzeige realisiert der folgende Quelltext. Geben Sie eine kurze Begründung.

```
char *s1="ABCDE";
char *s2="ABC";
if(strcmp(s1,s2) < 0) cout<<s1<<"<"<<s2<<endl;
else cout<<s1<<">="<<s2<<endl;
```

Ausgabe: "ABCDE>=ABC" (ohne Gänse).
Mit strcmp werden die zwei Strings "ABCDE" und "ABC" verglichen. Durch die if else Anweisung wird die entsprechende Ausgabe realisiert.

11.Aufgabe:

Nennen Sie den Unterschied zwischen den beiden folgenden Anweisungen, wenn s1 und s2 vom Typ char sind:*

```
1.s1=s2;
2.strcpy(s1,s2);
```

1.Beide zeigen auf die gleiche Adresse (s2).
2.Adressen unterschiedlich, Werte gleich (s2).

12.Aufgabe:

Schreiben Sie eine Funktion zum Kopieren eines Strings in einer zweiten, die dasselbe leistet, wie die Funktion strcpy(). Die Funktion habe den Prototyp
char stringcopy(char* s2, const char* s2)*

```
int ls1=strlen (&s1); i ;
int ls2=strlen (&s2);

if (s1<=s2)
   {for (i=0;i<ls1;i++)
     {&s1[i]=&s2[i];}
   }
else {for (i=0;i<ls2;i++)
     {&s1[i]=&s2[i];}
     }
```

nicht die optimale Lösung, aber es funktioniert.