Klausurvorbereitung

Polymorphie
Dynamischer Typ / Statischer Typ

Clicker: Wie ist die Ausgabe der 2 Print-Aufrufe?

```
class Mitarbeiter {
   public:
   virtual float Gehalt() { return 25; }
   virtual int    Unterg() { return 0; }
};

class Chef: public Mitarbeiter {
   public:
  virtual float Gehalt() { return 400; }
  virtual int  Unterg() { return 20; }
};
```

```
void Print (Mitarbeiter& pers) {
   cout << pers.Gehalt()<<
         "  " << pers.Unterg();
}
...
Mitarbeiter m;
Chef ch;
Print(m);
Print(ch);
```

Alles ohne Gewähr.
Ich – und Ihre Nachfolger – sind für Fehlerhinweise dankbar.
Im Zweifelsfall haben Sie einen Compiler oder meine Mail-Adresse.

1. 25 0 25 0
2. 25 0 400 0
3. 25 0 400 20
4. 400 0 400 0

Mit _-_ im Folgenden immer die Lösung gekennzeichnet

Ergebnis:
__ 1 _ 2 _-_ 3 __4

Clicker: Wie ist die Ausgabe der 2 Print-Aufrufe?

```
class Mitarbeiter {
   public:
   virtual float Gehalt() { return 25; }
   virtual int    Unterg() { return 0; }
};

class Chef: public Mitarbeiter {
   public:
  virtual float Gehalt() { return 400; }
  virtual int  Unterg() { return 20; }
};
```

```
void Print (Mitarbeiter pers) {
   cout << pers.Gehalt()<<
         "  " << pers.Unterg();
}
...
Mitarbeiter m;
Chef ch;
Print(m);
Print(ch);
```

1. 25 0 25 0
2. 25 0 400 0
3. 25 0 400 20
4. 400 0 400 0

Ergebnis:
_- _ 1 _ 2 __ 3 __4

Clicker: Wie ist die Ausgabe der 2 Print-Aufrufe?

```
class Mitarbeiter {
   public:
   virtual float Gehalt() { return 25; }
   int    Unterg() { return 0; }
};

class Chef: public Mitarbeiter {
   public:
  virtual float Gehalt() { return 400; }
  int  Unterg() { return 20; }
};
```

```
void Print (Mitarbeiter& pers) {
   cout << pers.Gehalt()<<
         "  " << pers.Unterg();
}
...
Mitarbeiter m;
Chef ch;
Print(m);
Print(ch);
```

1. 25 0 25 0
2. 25 0 400 20
3. 25 0 400 0
4. 400 0 400 0

Ergebnis:
__ 1 _ 2 _-_ 3 __4

Clicker: Wie ist die Bildschirm-Ausgabe?

```
class Mitarbeiter  {
public:
   Mitarbeiter()     {cout << "+M ";}
   ~Mitarbeiter()   {cout <<"-M ";}
};
class Chef : public Mitarbeiter  {
public:
    Chef()  {cout << "+C ";}
   ~Chef()  {cout << "-C ";}
};
```

```
void Poly() {
        Mitarbeiter m1
        Chef* ph1;  // Zeiger
}
```

1. +M -M
2. +M +M -C
3. +M +M +C
4. +M +M +C –C –M -M

Ergebnis:
_-_ 1 _ 2 __ 3 __4

Clicker: Wie ist die Bildschirm-Ausgabe?

```
class Mitarbeiter  {
public:
   Mitarbeiter()     {cout << "+M ";}
   ~Mitarbeiter()   {cout <<"-M ";}
};
class Chef : public Mitarbeiter  {
public:
    Chef()  {cout << "+C ";}
   ~Chef()  {cout << "-C ";}
};
```

```
void Poly() {
   Mitarbeiter* ph1 = new Chef();
   delete ph1;
}
```

1. +M +C –C -M
2. +M +C -M
3. +M +C
4. +C +M

Ergebnis:
__ 1 _-_ 2 __ 3 __4

Clicker: Wie ist die Bildschirm-Ausgabe?

```
class Mitarbeiter  {
public:
   Mitarbeiter()     {cout << "+M ";}
   virtual ~Mitarbeiter()   {cout <<"-M ";}
};
class Chef : public Mitarbeiter  {
public:
    Chef()  {cout << "+C ";}
   ~Chef()  {cout << "-C ";}
};
```

```
void Poly() {
   Mitarbeiter* ph1 = new Chef();
   delete ph1;
}
```

Ausgabe unabhängig von virtual bei Destruktor von Chef

1. +M +C –C -M
2. +M +C -M
3. +M +C
4. +C +M

Ergebnis:
_-_ 1 _ 2 __ 3 __4

Dynamischer Typ von ph1?

```
class Mitarbeiter  {
public:
   Mitarbeiter()     {cout << "+M ";}
   virtual ~Mitarbeiter()   {cout <<"-M ";}
};
class Chef : public Mitarbeiter  {
public:
    Chef()  {cout << "+C ";}
   ~Chef()  {cout << "-C ";}
};
```

```
void Poly() {
   Mitarbeiter* ph1 = new Chef();
   delete ph1;
}
```

1. Zeiger auf Mitarbeiter
2. Zeiger auf Chef
3. Mitarbeiter
4. Chef

Ergebnis:
__ 1 _-_ 2 __ 3 __4

Statischer Typ von ph1?

```
class Mitarbeiter  {
public:
   Mitarbeiter()     {cout << "+M ";}
   virtual ~Mitarbeiter()   {cout <<"-M ";}
};
class Chef : public Mitarbeiter  {
public:
    Chef()  {cout << "+C ";}
   ~Chef()  {cout << "-C ";}
};
```

```
void Poly() {
   Mitarbeiter* ph1 = new Chef();
   delete ph1;
}
```

1. Zeiger auf Mitarbeiter
2. Zeiger auf Chef
3. Mitarbeiter
4. Chef

Ergebnis:
_-_ 1 _ 2 __ 3 __4

Clicker: Wie ist die Bildschirm-Ausgabe?

```
class Mitarbeiter  {
public:
   Mitarbeiter()     {cout << "+M ";}
   virtual ~Mitarbeiter()   {cout <<"-M ";}
};
class Chef : public Mitarbeiter  {
public:
    Chef()  {cout << "+C ";}
   ~Chef()  {cout << "-C ";}
};
```

```
void Poly() {
   auto_ptr<Mitarbeiter> p(new Chef());
}
```

1. +M +C –C -M
2. +M +C -M
3. +M +C
4. +C +M

Ergebnis:
__ 1 _ 2 __ 3 __4

Aufgaben

```
class Saeuger {
public:
   void ton() {datei << " knurr ";}
   virtual int beine() {return 0;}
   };
class Hund: public Saeuger{
public:
   void ton() {datei << " wau ";}
   virtual int beine() {return 4;}
private:
   };
class Katze: public Saeuger{
public:
   void ton() {datei << " miau ";}
private:
   };
```

```
void funk6() {

   Saeuger* s1 = new Hund();
   Hund* h2 = new Hund();
   Saeuger* s3 = new Saeuger();
   Saeuger* s4 = new Katze();

   // Zeigerarray initialisieren
   Saeuger* arr[] = {s1, h2, s3, s4};

   for (int i=0; i<4; ++i) {
      datei << "\n" << i << " "
            << arr[i]->beine();
       arr[i]->ton();
       delete arr[i];
      }
   }
```

```
0 4 knurr
1 4 knurr
2 0 knurr
3 0 knurr
```

Aufgaben

```
class Saeuger {
public:
   void ton() {datei << " knurr ";}
   virtual int beine() {return 0;}
   };
class Hund: public Saeuger{
public:
   void ton() {datei << " wau ";}
   virtual int beine() {return 4;}
private:
   };
class Katze: public Saeuger{
public:
   void ton() {datei << " miau ";}
private:
   };
```

```
void funk7() {

   Saeuger s1;
   Hund h2;
   Katze k3;

   datei << "\n" << s1.beine();  s1.ton();
   datei << "\n" << h2.beine();  h2.ton();
   datei << "\n" << k3.beine();  k3.ton();
   }
```

0 knurr
4 wau
0 miau

Aufgaben

```
class Saeuger {
public:
   void ton() {datei << " knurr ";}
   virtual int beine() {return 0;}
   };
class Hund: public Saeuger{
public:
   void ton() {datei << " wau ";}
   virtual int beine() {return 4;}
private:
   };
class Katze: public Saeuger{
public:
   void ton() {datei << " miau ";}
private:
   };
```

```
void h(Saeuger& s) {
   datei <<  s.beine();  s.ton();
    }
void g(Saeuger* s) {
   datei <<  s->beine();  s->ton();
    }
void f(Saeuger s) {
   datei <<  s.beine();  s.ton();
    }

void funk8() {
   Hund h2;        Katze k3;
   h(h2);   h(k3);
   g(&h2);   g(&k3);
   f(h2);   f(k3);

}
```

4 knurr
0 knurr

4 knurr
0 knurr

0 knurr
0 knurr

Übung 7: Welche Ausgabe ergibt sich? Aufgabe c

```
class Figur   {
   int dummy;
public:
   Figur()  {dummy=0; datei << "+F";}
   ~Figur() {datei << "-F";}
class Ring: public Kreis  {
   int rRad;  // Innenkreis-Radius
public:
      Ring(int r)  {
         rRad=r;
         datei << "+R" << rRad << " ";       }
      Ring(int r, int rr):Kreis(r), rRad(rr) {
         datei << "+R" << rRad << " ";       }
      ~Ring() {
         datei << "-R" << rRad << " ";       }
 /* …*/
};
```

```
class Kreis: public Figur  {
   protected:      enum {PI=3};
      int rad;  // Außenkreis-Radius
   public:
      Kreis(): rad(4)  {
         datei << "+K" << rad << " ";        }
      Kreis(int r) {
         rad=r;
         datei << "+K" << rad << " ";        }
      ~Kreis() {
         datei << "-K" << rad << " ";        }
 /* …* /
};
```

```
void funk3()   {
   Kreis* k1[5];
   datei << " xxx ";
   k1[4]=NULL;
   datei << " yyy ";
   Kreis  k2[2];
   }
```

Übung 7: Welche Ausgabe ergibt sich? Aufgabe d

```
class Figur   {
   int dummy;
public:
   Figur()  {dummy=0; datei << "+F";}
   ~Figur() {datei << "-F";}
class Ring: public Kreis  {
   int rRad;  // Innenkreis-Radius
public:
      Ring(int r)  {
         rRad=r;
         datei << "+R" << rRad << " ";       }
      Ring(int r, int rr):Kreis(r), rRad(rr) {
         datei << "+R" << rRad << " ";       }
      ~Ring() {
         datei << "-R" << rRad << " ";       }
 /* …*/
};
```

```
class Kreis: public Figur  {
   protected:      enum {PI=3};
      int rad;  // Außenkreis-Radius
   public:
      Kreis(): rad(4)  {
         datei << "+K" << rad << " ";         }
      Kreis(int r) {
         rad=r;
         datei << "+K" << rad << " ";         }
      ~Kreis() {
         datei << "-K" << rad << " ";         }
 /* …* /
};
```

```
void funk4()    {
  Kreis* p = new Kreis(12);
  delete p;
   }
```

Übung 7: Welche Ausgabe ergibt sich? Aufgabe e

```
class Figur   {
   int dummy;
public:
   Figur()  {dummy=0; datei << "+F";}
   ~Figur() {datei << "-F";}
class Ring: public Kreis  {
   int rRad;  // Innenkreis-Radius
public:
      Ring(int r)  {
         rRad=r;
         datei << "+R" << rRad << " ";       }
      Ring(int r, int rr):Kreis(r), rRad(rr) {
         datei << "+R" << rRad << " ";       }
      ~Ring() {
         datei << "-R" << rRad << " ";       }
 /* …*/
};
```

```
class Kreis: public Figur  {
   protected:      enum {PI=3};
      int rad;  // Außenkreis-Radius
   public:
      Kreis(): rad(4)  {
         datei << "+K" << rad << " ";         }
      Kreis(int r) {
         rad=r;
         datei << "+K" << rad << " ";         }
      ~Kreis() {
         datei << "-K" << rad << " ";         }
 /* …* /
};
```

```
void funk4()    {
  Figur* p = new Kreis(12);
  delete p;
   }
```

Übung 7: Welche Ausgabe ergibt sich? Aufgabe i

```
class Figur   {
   int dummy;
public:
   Figur()  {dummy=0; datei << "+F";}
   ~Figur() {datei << "-F";}
   virtual double flaeche() = 0;
};

class Ring: public Kreis  {
   int rRad;  // Innenkreis-Radius
public:
      Ring(int r)  {
         rRad=r;
         datei << "+R" << rRad << " ";       }
      Ring(int r, int rr):Kreis(r), rRad(rr) {
         datei << "+R" << rRad << " ";       }
      ~Ring() {
         datei << "-R" << rRad << " ";       }
virtual double flaeche() {return (rad * rad –
    rRad*rRad) * PI - ;]
 /* …*/
};
```

```
class Kreis: public Figur  {
   protected:      enum {PI=3};
      int rad;  // Außenkreis-Radius
   public:
      Kreis(): rad(4)  {
         datei << "+K" << rad << " ";         }
      Kreis(int r) {
         rad=r;
         datei << "+K" << rad << " ";         }
      ~Kreis() {
         datei << "-K" << rad << " ";         }
virtual double flaeche() {return rRad*rRad *
              PI;]

 /* …* /
};
```

Warum kann von den Klassen Figur und Ring kein Array erzeugt werden?