# Lösningsförslag till tentamen*

| | |
|---|---|
| **Kursnamn** | **Objektorienterad programutveckling** |
| **Tentamensdatum** | **2001-03-06** |
| **Program** | **DAI 2** |
| **Läsår** | **2000/2001, lp III/IV** |
| **Examinator** | **Uno Holmer** |

* se även `www.chl.chalmers.se/~holmer/` där finns det mesta av kursmaterialet.

**Uppgift 1** (12 p)

Ingen lösning ges. Se kurslitteraturen.

**Uppgift 2** (12 p)

```
class NoisyDie : public Die {
public:
    NoisyDie( int Pitch = 1000, int Duration = 100,
                  int Pause = 1000 )
      : thePitch(Pitch),theDuration(Duration),thePause(Pause) {}
    int Value() const {
        for ( int I = 0; I < Die::Value(); I++ ) {
            sound( thePitch );
            delay( theDuration  );
            nosound();
            delay( thePause );
        }
        return Die::Value();
    }
protected:
    int thePitch;
    int theDuration;
    int thePause;
};
```

**Uppgift 3** (3+6+3 p)

a) (3 p)

```
class IsLeapYear {
public:
    bool operator() ( unsigned int X ) {
        return ( X % 4 == 0 && X % 100 != 0 ) || X % 400 == 0;
    }
};
```

b) (6 p)

```
template <class A,class Condition>
bool Exists( const List<A> & Src, Condition Cond ) {
    ListItr<A> Sitr(Src);
    while ( +Sitr ) {
        if ( Cond( Sitr() ) )
            return true;
        Sitr++;
    }
    return false;
}
```

c) (3 p)

```
List<int> L;
...
if ( Exists( L, IsLeapYear() )
   cout << "Listan innehåller ett skottår";
```

**Uppgift 4** (7+5 p)

a) (7 p)

Funktionen F har en värdeparameter X av typ Base vilket innebär att basklassdelen skärs ut vid kopiering av argumentet till X. Således kommer alltid Base::H att anropas i F, oavsett om argumentet har typ Base eller Sub. G däremot hanterar argumentet via en basklassreferens. Eftersom H är deklarerad som virtuell i Base kommer därför anropet G(B) att leda till att Base::H anropas, men Sub::H för G(S). Utskrifterna blir enligt nedan.

```
void main() {
   Base B(2);    // X initieras till 2
   Sub S(3);     // X initieras till 3, Z till 6
   F( B );       // Base::H 2
   G( B );       // Base::H 2
   F( S );       // Base::H 3
   G( S );       // Sub::H 9
}
```

b) (5 p)

Rätt svar är alt. c. Många A-objekt kan känna till samma B-objekt (delning). Varje C-objekt har ett B-objekt som i sin tur känner till det ägande C-objektet, vilket motsvarar en oriktad aggregatrelation. Alt. a faller på att aggregatrelationen är riktad, alt b för att relationen mellan C och B ej är en aggregatrelation, samt att den är riktad, alt d på att multipliciteten N anges för fel klass, samt att relationen mellan C och B endast anges som en dubbelriktad association, vilket är betydligt lösare än vad koden antyder.

**Uppgift 5** (12 p)

```
class Shape {
public:
    Shape( int Size, Point Pos )
        : mySize(Size), myCentre(Pos) {}
    void Resize( int NewSize ) { mySize = NewSize; }
    int GetSize() const { return mySize; }
    virtual void Paint() = 0;
    virtual void Undo() = 0;
protected:
    Point myCentre;
    int mySize;
};

class Square : public Shape {
public:
    Square( int Size, Point Pos ) : Shape(Size,Pos) {}
    void Paint() { /* paint the square*/ }
    void Undo() { /* undo the square*/ }
};

class Circle : public Shape {
public:
    Circle( int Radius, Point Pos ) : Shape(Radius,Pos) {}
    void Paint() { /* paint the circle */ }
    void Undo() { /* undo the circle */ }
};

// denna kan alt. definieras i basklassen
void ZoomOut( Shape & S ) {
    while ( S.GetSize() > 0 ) {
        S.Paint();
        // delay( 10 ); // ms
        S.Undo();
        S.Resize( S.GetSize() - 1 );
    }
}

void main() {
    Square S( 100, Point( 100, 200 ) );
    Circle C( 200, Point( 300, 400 ) );
    ZoomOut( S );
    ZoomOut( C );
}
```