```
}

class DemoAbstract {

  public static void main(String[] args) {
    // A a = new A(); viga!
    B b = new B();
    b.esimeneTeade();
    b.teineTeade();
  }
}
```

Näeme, et kuigi klassist A ei ole võimalik genereerida isendit, on võimalik pöörduda tema meetodite poole alamklassi isendi abil.

Ülesanne 1

Koostada abstraktne klass Kujund, mis sisaldab kahte meetodit:

- täispindala()
- ruumala()

Muuta eelmises praktikumis koostatud klassi Risttahukas nii, et see pärineb klassist Kujund ja realiseerib selle kõik meetodid. Testida peaklassis uusi meetodeid.

Liidesed

Peale klasside võib Javas defineerida ka liideseid. Liides võimaldab määrata, mida klass peab tegema, jättes täpsustamata, kuidas seda teha. Süntaktiliselt on liidesed sarnased selliste klassidega, mille kõik meetodid on abstraktsed. Liides sisaldab ainult konstante ja meetodeid. Kui liides on loodud, on võimalik seda realiseerida (varustada sisuga) paljudes klassides ja vastupidi – üks klass võib realiseerida suvalise arvu liideseid.

Näide liidese kasutamise kohta:

```
interface Helista {
  void helista(int nr);
}

class Klient implements Helista {

  public void helista(int nr) {
    System.out.println("Helistatud numbril " + nr);
  }

  void helistaVeel() {
    System.out.println("Klassid, mis kasutavad liideseid" +
                       ", võivad sisaldada teisi meetodeid.");
  }
}
```

Võtmesõna implements väljendab asjaolu, et klass Klient realiseerib liidese Helista ning tema meetodi helista. Liidese kõik meetodid on vaikimisi avalikud (public). Liideses kirjeldatud meetodi ja klassis seda realiseeriva meetodi signatuurid peavad olema samad. Liidest realiseeriv klass võib sisaldada lisaks veel meetodeid, mida liideses ei ole kirjeldatud.

Liidest võib kasutada tüübina. Sel juhul hoitakse muutujas viidet isendile, mis realiseerib liidese.

```
class TestiLiidest {

  public static void main(String[] args) {
    Helista c = new Klient();
    c.helista(4646);
  }
}
```

Oluline on siin, et muutuja c on tüüpi Helista, mis on liides. Muutuja c abil saab pöörduda vaid nende meetodite poole, mis on loetletud liideses Helista. Olgu meil veel üks liidese Helista realisatsioon:

```
class TeineKlient implements Helista {

  public void helista(int nr) {
    System.out.println("Helista teine versioon.");
    System.out.println("Number ruudus = " + (nr * nr) + ".");
  }
}
```

Liidesetüüpi muutuja on sõltumatu liidest tegelikult realiseerivast klassist.

```
class TestiTeist {

  public static void main(String[] args) {
    Helista c = new Klient();
    TeineKlient d = new TeineKlient();
    c.helista(444);
    c = d; // c viitab nüüd klassi TeineKlient isendile
    c.helista(4);
  }
}
```

Kui klass kasutab (implements) liidest, kuid ei realiseeri siiski kõiki liidese meetodeid, siis tuleb klass deklareerida abstraktseks.