Spelet tjugoett

- Stort, svårt exempel från slutet av kursen.
- Kursboken, kap 10.10
- Visar vad vi kan vid kursens slut - inrymmer många av kursens begrepp.
- Det finns *många* lösningsstrategier - inget "rätt svar". Vald lösning finns i kursboken.
- Fokusera på idéer, inte på programspråkskonstruktioner

Spelet tjugoett

Du spelar mot banken

```
Välkommen till tjugoett
Du fick Ruter Ess och har 14 poäng
Ett kort till? j
Du fick Hjärter 6 och har 20 poäng
Ett kort till? n
Datorn fick Klöver Ess
Datorn fick Klöver 2
Datorn fick Klöver 10
Datorn fick Spader 9
Datorn har 22 poäng
Du vann!
Nytt parti? j
Du fick Hjärter 8 och har 8 poäng
Ett kort till? j
Du fick Spader Kung och har 21 poäng
Du vann!
Nytt parti? j
Du fick Spader 9 och har 9 poäng
Ett kort till? j
Du fick Spader 5 och har 14 poäng
Ett kort till? j
Du fick Klöver 9 och har 23 poäng
Du förlorade!
Nytt parti? n
```

"Vilken Java" finns i programmet?

- Någon form av minnesfunktion
- Någon form av repetion
- Någon form av kommunikation mellan objekt
- Någon form av in och utmatning
- Någon form av felhantering
- Lite annat


```
public class Tjugoett {
  Kortbunt lek = new Kortbunt();
  Manniska du  = new Manniska(lek);
  Dator    jag = new Dator(lek, du);
```

Här ska metoden Tjugoett() vara,
som startar spelet och avgör vem som vinner

```
  public static void main(String[] arg)
{
    Tjugoett tj = new Tjugoett();
  }
}
```

```
  Tjugoett() {
    System.out.println("Välkommen
           till tjugoett");
    boolean nyttPartiÖnskas = true;
    while (nyttPartiÖnskas) {
      lek.nyKortlek();
      lek.blanda();
      du.spela();
      if (du.poäng() > 21)
      System.out.println("Du
             förlorade!");
      else if (du.poäng() == 21)
        System.out.println("Du vann!");
      else { // datorn måste spela
        jag.spela();
        if (jag.poäng() <= 21 &&
            jag.poäng() >= du.poäng())
        System.out.println("Du
            förlorade!");
        else
        System.out.println("Du
            vann!");
      }
      System.out.print("Nytt parti? ");
      nyttPartiÖnskas = du.svararJa();
    }
}
```



```
public abstract class Spelare {
  protected Kortbunt leken;

  protected Kortbunt hand =
     new Kortbunt();
  protected int p;

  public Spelare(Kortbunt k) {
    leken = k;
  }

  public abstract void spela();

  public void nyttSpel() {
    hand.slängKorten();
    p = 0;
  }

   public int poäng() {
    return p;
  }
```

```
import java.io.*;

public class Manniska extends Spelare {
  BufferedReader
     myIn = new BufferedReader
     (new InputStreamReader(System.in));

  public Manniska(Kortbunt k) {
    super(k);
  }

  public void spela() {
    boolean nyttKortÖnskas = true;
    nyttSpel();
    while (p < 21 && nyttKortÖnskas) {
      System.out.println("Du fick " +
      nyttKort() +
      " och har " + p + " poäng");
      if (p < 21) {
        System.out.print("Et till? ");
        nyttKortÖnskas = svararJa();
      }
    }
  }
```



```
  public boolean svararJa() {
    System.out.flush();

    String s = "";
    try {
       s = myIn.readLine();
    }
    catch (IOException e) {}
    return s.equals("") ||
           s.equals("j") ||
           s.equals("ja");
  }
}
```

```
public class Dator extends Spelare {
   private Spelare motspelare;

   public Dator(Kortbunt k, Spelare mot){
     super(k);
     motspelare = mot;
   }

   public void spela() {
     nyttSpel();
     while (p < 21 && p <
            motspelare.poäng()) {
       System.out.println("Datorn fick "
       + nyttKort());
     }
     System.out.println("Datorn har " + p
     + " poäng");
   }
 }
```



```
 public Kort nyttKort() {
    Kort k = leken.geÖversta();
    hand.läggÖverst(k);
    int antalEss = 0;
    p = 0;
    for (int i=1;
     i<=hand.antalKort(); i++) {
      int v = hand.tittaPå(i).valör();
      if (v == 1) { // ett Ess
        p += 14;
        antalEss++;
      }
      else
        p += v;
    }
    for (int j=1; j<=antalEss && p>21;
                  j++)
      p -= 13;
    return k;
  }
}
```

```
public class Kortbunt {
  private Kort[] bunten = new Kort[52];
  private int antal = 0;

  public int antalKort() {
    return antal;
  }

  public void slängKorten() {
    antal = 0;
  }

  public Kort tittaPå(int nr) {
    return bunten[antal-nr];
  }

  public void läggÖverst(Kort k) {
    bunten[antal++] = k;
  }

  public  Kort geÖversta() {
    return bunten[--antal];
  }

  public void nyKortlek() {
    slängKorten();
    for (int f=1; f<=4; f++)
      for (int v=1; v<=13; v++)
        läggÖverst(new Kort(f,v));
  }
```



```
  public void blanda() {
    for (int i=1; i<1000; i++) {
      int n1 = (int) (Math.random() *
           antal);
      int n2 = (int) (Math.random() *
           antal);
      Kort temp  = bunten[n1];
      bunten[n1] = bunten[n2];
      bunten[n2] = temp;
    }
  }
}
```

Vad såg vi?

Programspråkskonstruktioner för att ….

- Spara värden
- Repetera
- Definiera objekt
- Beskriva relationer mellan objekt
- Mata in eller ut data
- Felhantering

Objektorientering

- Sätt att modellera världen med ”saker”
- Klasser beskriver typer av objekt
- Objekten interagerar
- Objekten har egenskaper/tillstånd och kan handla

Varför lär man sig detta?

- Detta ≠ ”Bara” ren programmering.
- Förstå datorer/datoranvändning är en del av ingenjörskompetens.
- Dator/programmeringskunskap med grundbegrepp är nyttigt i sig och ganska stabilt. Tillämpningsprogram förändras blixtsnabbt
- Ger en problemlösningsstrategi.
- Speciellt för STS: Kunna kommunicera med olika specialister.
- Intressant för en generalist.