Lezione 3

```
(* Successione di Fibonacci relativa alla crescita dei conigli:

   ogni coppia adulti genera una coppia di conigli al mese
   e questi diventano a loro volta adulti in due mesi *)
```

In[1]:=

```
Fibonacci[0] = 1; (* si inizia con una sola coppia di conigli ... *)
Fibonacci[1] = 1; (* ... non ancora adulti fino al secondo mese *)

Fibonacci[m_] := Fibonacci[m] =  (* per memorizzare i valori calcolati *)
  Fibonacci[m - 1] +  Fibonacci[m - 2]

(* Fibonacci[m - 1] = numero delle coppie nel mese precedente
   Fibonacci[m - 2] = numero delle coppie adulte nel mese precedente *)


(* controlliamo ... *)
```

In[5]:=

```
?? Fibonacci
```

```
Global`Fibonacci

Fibonacci[0] = 1

Fibonacci[1] = 1

Fibonacci[m_] := Fibonacci[m] = Fibonacci[m - 1] + Fibonacci[m - 2]
```

In[6]:=

```
Fibonacci[5]
```

Out[6]=

```
8
```

```
(* vediamo se ha memorizzato ... *)
```

In[7]:=

```
?? Fibonacci
```

```
Global`Fibonacci

Fibonacci[0] = 1

Fibonacci[1] = 1

Fibonacci[2] = 2

Fibonacci[3] = 3

Fibonacci[4] = 5

Fibonacci[5] = 8

Fibonacci[m_] := Fibonacci[m] = Fibonacci[m - 1] + Fibonacci[m - 2]
```

In[8]:=

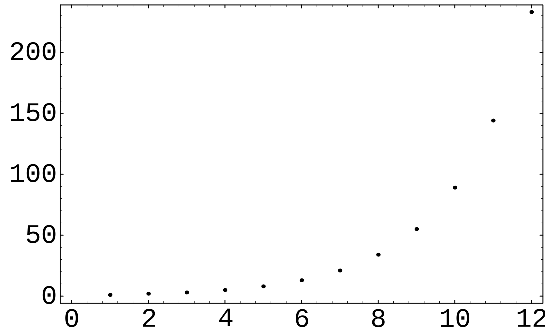
```
Table[Fibonacci[m],{m,1,12}]  (* in un anno ... *)
```

Out[8]=

```
{1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233}
```

In[9]:=

```
ListPlot[%,Axes->False,Frame->True]
```



Out[9]=

```
-Graphics-
```

In[10]:=

```
Table[Fibonacci[m]/Fibonacci[m-1] - 1,{m,1,36}] (* tasso di crescita *)
```

Out[10]=

$$\left\{0,\ 1,\ \frac{1}{2},\ \frac{2}{3},\ \frac{3}{5},\ \frac{5}{8},\ \frac{8}{13},\ \frac{13}{21},\ \frac{21}{34},\ \frac{34}{55},\ \frac{55}{89},\ \frac{89}{144},\ \frac{144}{233},\ \frac{233}{377},\ \frac{377}{610},\ \frac{610}{987},\ \frac{987}{1597},\ \frac{1597}{2584},\ \frac{2584}{4181},\ \frac{4181}{6765},\ \frac{6765}{10946},\ \frac{10946}{17711},\ \frac{17711}{28657},\ \frac{28657}{46368},\ \frac{46368}{75025},\ \frac{75025}{121393},\ \frac{121393}{196418},\ \frac{196418}{317811},\ \frac{317811}{514229},\ \frac{514229}{832040},\ \frac{832040}{1346269},\ \frac{1346269}{2178309},\ \frac{2178309}{3524578},\ \frac{3524578}{5702887},\ \frac{5702887}{9227465},\ \frac{9227465}{14930352}\right\}$$

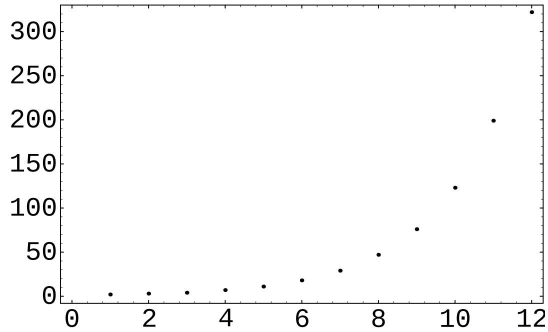
In[11]:=

```
N[%,10]
```

Out[11]=

```
{0, 1., 0.5, 0.6666666667, 0.6, 0.625, 0.6153846154, 0.619047619,
  0.6176470588, 0.6181818182, 0.6179775281, 0.6180555556,
  0.6180257511, 0.6180371353, 0.6180327869, 0.6180344478,
  0.6180338134, 0.6180340557, 0.6180339632, 0.6180339985, 0.618033985,
  0.6180339902, 0.6180339882, 0.618033989, 0.6180339887, 0.6180339888,
  0.6180339887, 0.6180339888, 0.6180339887, 0.6180339888,
  0.6180339887, 0.6180339887, 0.6180339887, 0.6180339887,
  0.6180339887, 0.6180339887}
```

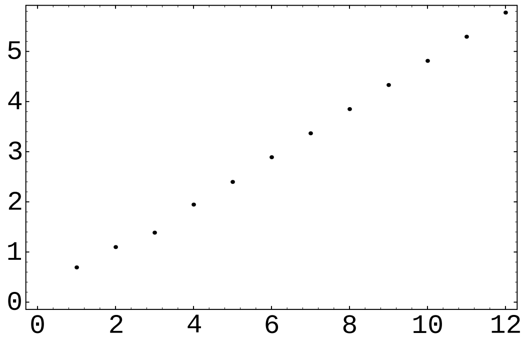
```
(* il tasso di crescita tende al rapporto aureo - 1 ...  *)
```

In[12]:=
```
N[(1 + Sqrt[5])/2 - 1,10]
```
Out[12]=
```
0.6180339887
```
```
(* Modello di crescita a tasso costante:
   i = popolazione iniziale, k = tasso di crescita *)
```
In[13]:=
```
TassoCostante[i_,k_] := (Clear[p];
                         p[0] = i;
                         p[t_] := p[t] = (1 + k) p[t-1])
```
```
(* costruiamo il modello con il tasso ottenuto sopra ... *)
```
In[14]:=
```
TassoCostante[1,N[(1 + Sqrt[5])/2 - 1]]
```
In[15]:=
```
Table[p[t],{t,1,12}]
```
Out[15]=
```
{1.61803, 2.61803, 4.23607, 6.8541, 11.0902, 17.9443, 29.0344,
  46.9787, 76.0132, 122.992, 199.005, 321.997}
```
In[16]:=
```
Round[%]
```
Out[16]=
```
{2, 3, 4, 7, 11, 18, 29, 47, 76, 123, 199, 322}
```
In[17]:=
```
ListPlot[%,Axes->False,Frame->True]
```


Out[17]=
```
-Graphics-
```
```
(* l'andamento è simile a quello della successione di Fibonacci:
   la crescita della popolazione è di tipo esponenziale ... *)
```

In[18]:=

```
ListPlot[Map[Log,%%],Axes->False,Frame->True]
```



Out[18]=

```
-Graphics-
```

In[19]:=

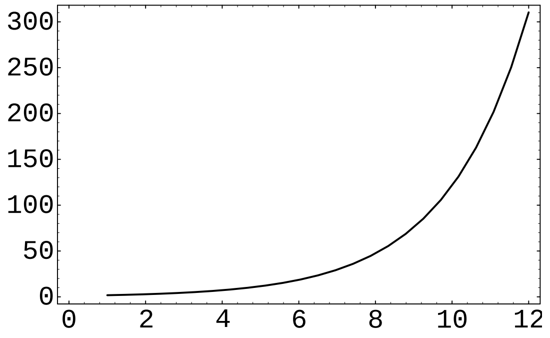
```
Fit[N[Map[Log,%%%]],{1,t},t]
```

Out[19]=

```
0.0991273 + 0.469884 t
```

In[20]:=

```
Plot[E^%,{t,1,12},Axes->False,Frame->True]
```

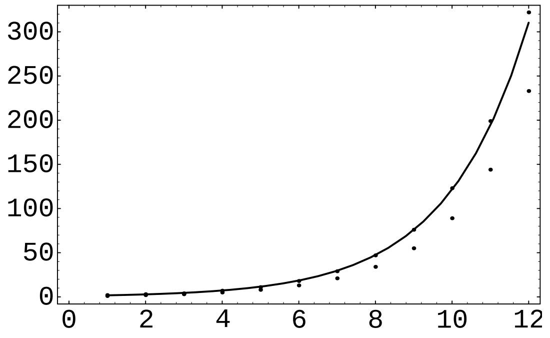


Out[20]=

```
-Graphics-
```

In[21]:=

```
Show[%,%9,%17] (* Fibonacci resta un po' al di sotto ... *)
```



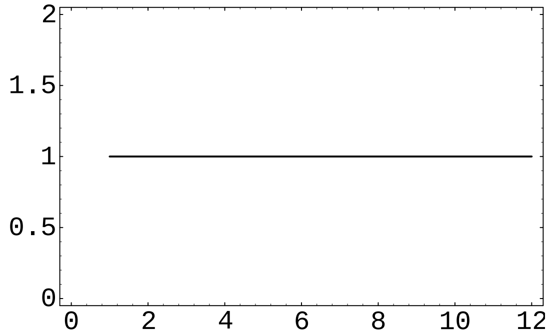
Out[21]=

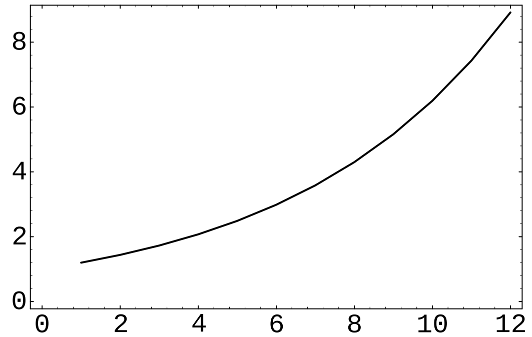
```
-Graphics-
```

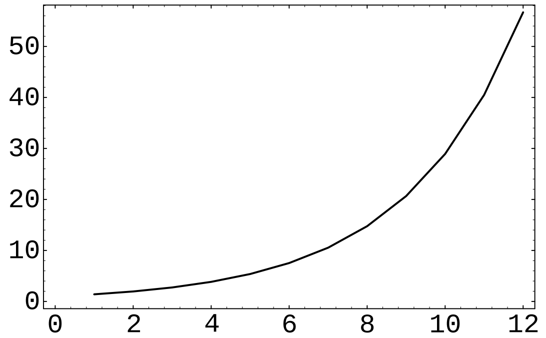
```
(* vediamo gli andamenti corrispondenti a diversi valori del tasso ... *)
```

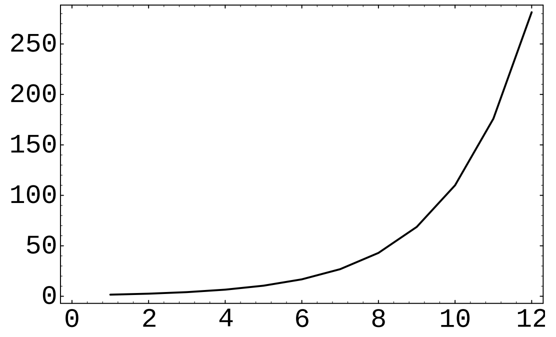
In[22]:=

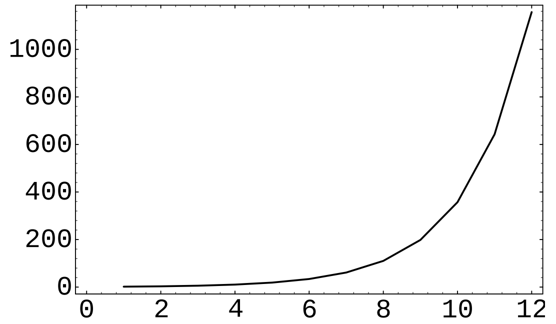
```
Table[ListPlot[TassoCostante[1,k];Table[p[t],{t,1,12}],
              Axes->False,Frame->True,
              PlotJoined->True], (* per congiungere i punti *)
      {k,0.,1.,.2}]
```

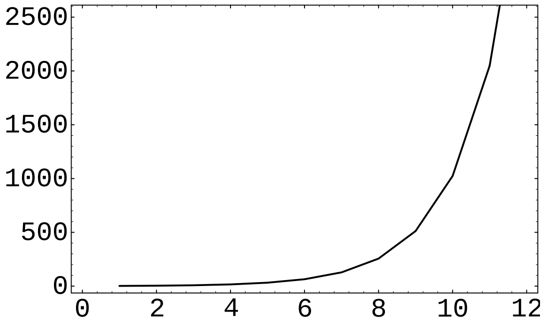










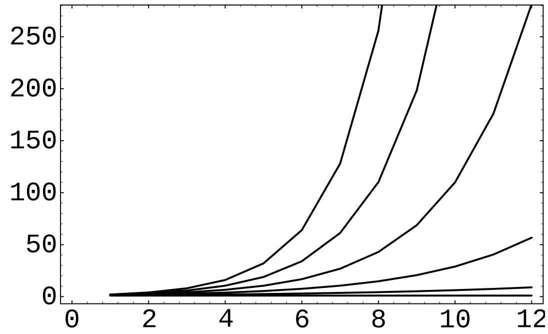


Out[22]=

```
{-Graphics-, -Graphics-, -Graphics-, -Graphics-, -Graphics-,
  -Graphics-}
```

In[23]:=

```
Show[%]
```



Out[23]=

-Graphics-

```
(* Modello di crescita di Verhulst:
   i = popolazione iniziale, m = popolazione ottimale
   k = tasso teorico di crescita senza limiti di sviluppo *)
```

In[24]:=

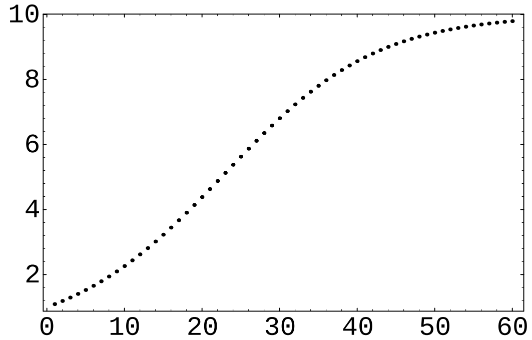
```
Verhulst[i_,m_,k_] := (Clear[p];
                       p[0] = i;
                       p[t_] := p[t] = (1 + k (1 - p[t - 1]/m)) p[t - 1])
```

In[25]:=

```
Verhulst[1,10,.1] (* con k piccolo la popolazione si avvicina al
                     valore ottimale secondo una "logistica" ... *)
```

In[26]:=

```
ListPlot[Table[p[t],{t,1,60}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[26]=

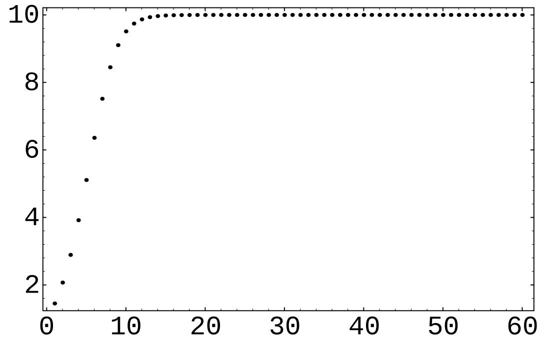
-Graphics-

In[27]:=

```
Verhulst[1,10,.5] (* k maggiore la popolazione si avvicina
                     più rapidamente al valore ottimale ... *)
```

In[28]:=

```
ListPlot[Table[p[t],{t,1,60}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[28]=

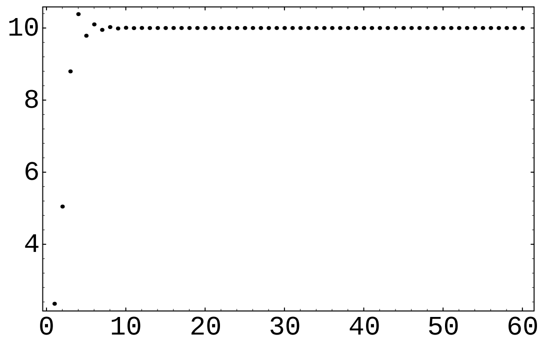
-Graphics-

In[29]:=

```
Verhulst[1,10,1.5] (* se k > 1 la popolazione oscilla intorno al
                      valore ottimale (superandolo) prima di
                      stabilizzarsi ... *)
```

In[30]:=

```
ListPlot[Table[p[t],{t,1,60}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[30]=

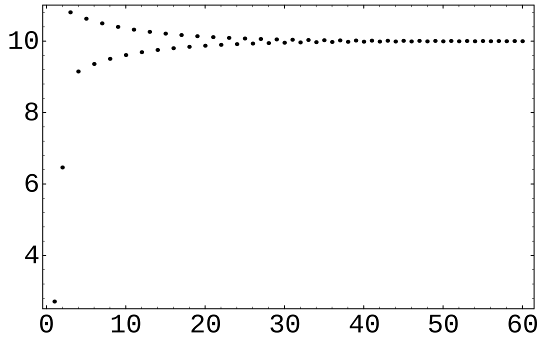
-Graphics-

In[31]:=

```
Verhulst[1,10,1.9] (* aumentando ancora k le oscillazioni diventano
                      più ampie e la stabilizzazione più lenta ... *)
```

In[32]:=

```
ListPlot[Table[p[t],{t,1,60}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[32]=

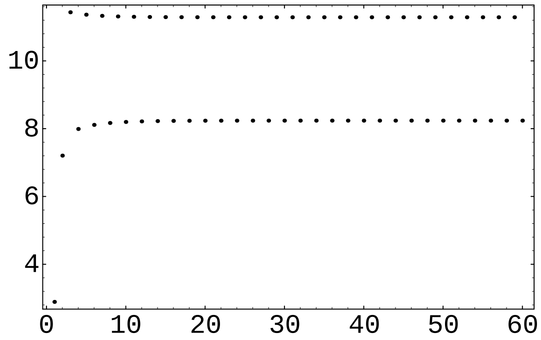
-Graphics-

In[33]:=

```
Verhulst[1,10,2.1] (* appena k supera 2 le oscillazioni
                      diventano stabili ... *)
```

In[34]:=

```
ListPlot[Table[p[t],{t,1,60}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[34]=

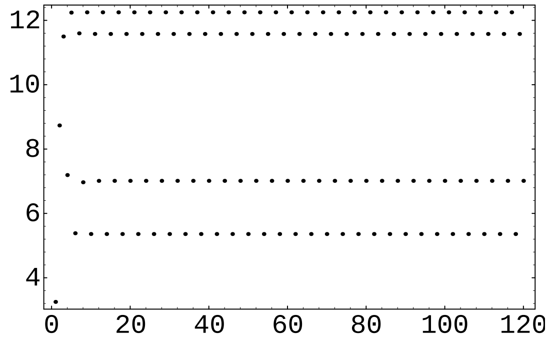
-Graphics-

In[35]:=

```
Verhulst[1,10,2.5] (* ora si oscilla tra quattro valori ... *)
```

In[36]:=

```
ListPlot[Table[p[t],{t,1,120}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[36]=

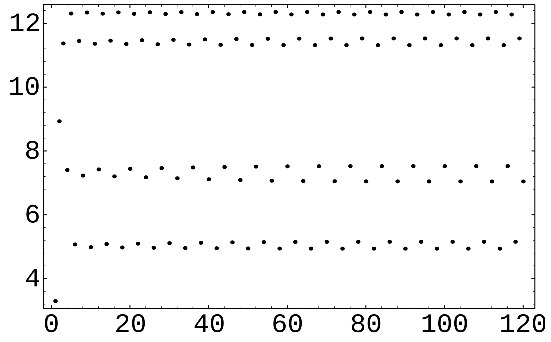
-Graphics-

In[37]:=

```
Verhulst[1,10,2.55] (* ... tra otto valori ... *)
```

In[38]:=

```
ListPlot[Table[p[t],{t,1,120}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[38]=

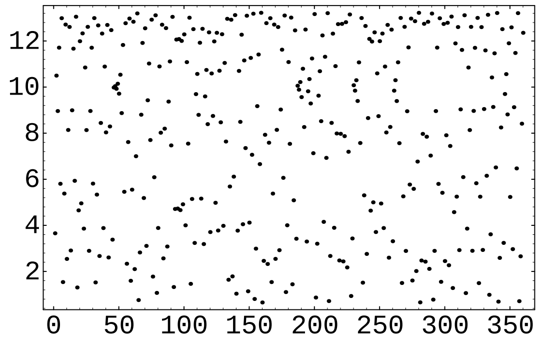
```
-Graphics-
```

In[39]:=

```
Verhulst[1,10,2.95] (* ora l'andamento è caotico ... *)
```

In[40]:=

```
ListPlot[Table[p[t],{t,1,360}],
  Axes->False,Frame->True,PlotRange->All]
```



Out[40]=

```
-Graphics-
```

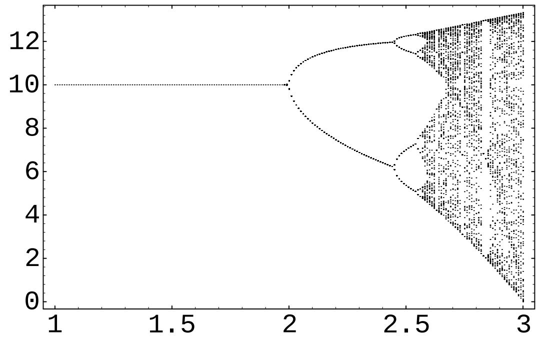
```
(* valori ai quali si "stabilizza" la popolazione al variare di k ... *)
```

In[41]:=

```
Table[Verhulst[1.,10.,k];
      Take[Table[{k,p[i]},{i,1,400}],-100],{k,1.,3.,.01}];
```

In[42]:=

```
ListPlot[Apply[Join,%],
        Axes->False,Frame->True,PlotStyle->{PointSize[0.002]}]
```



Out[42]=

```
-Graphics-
```