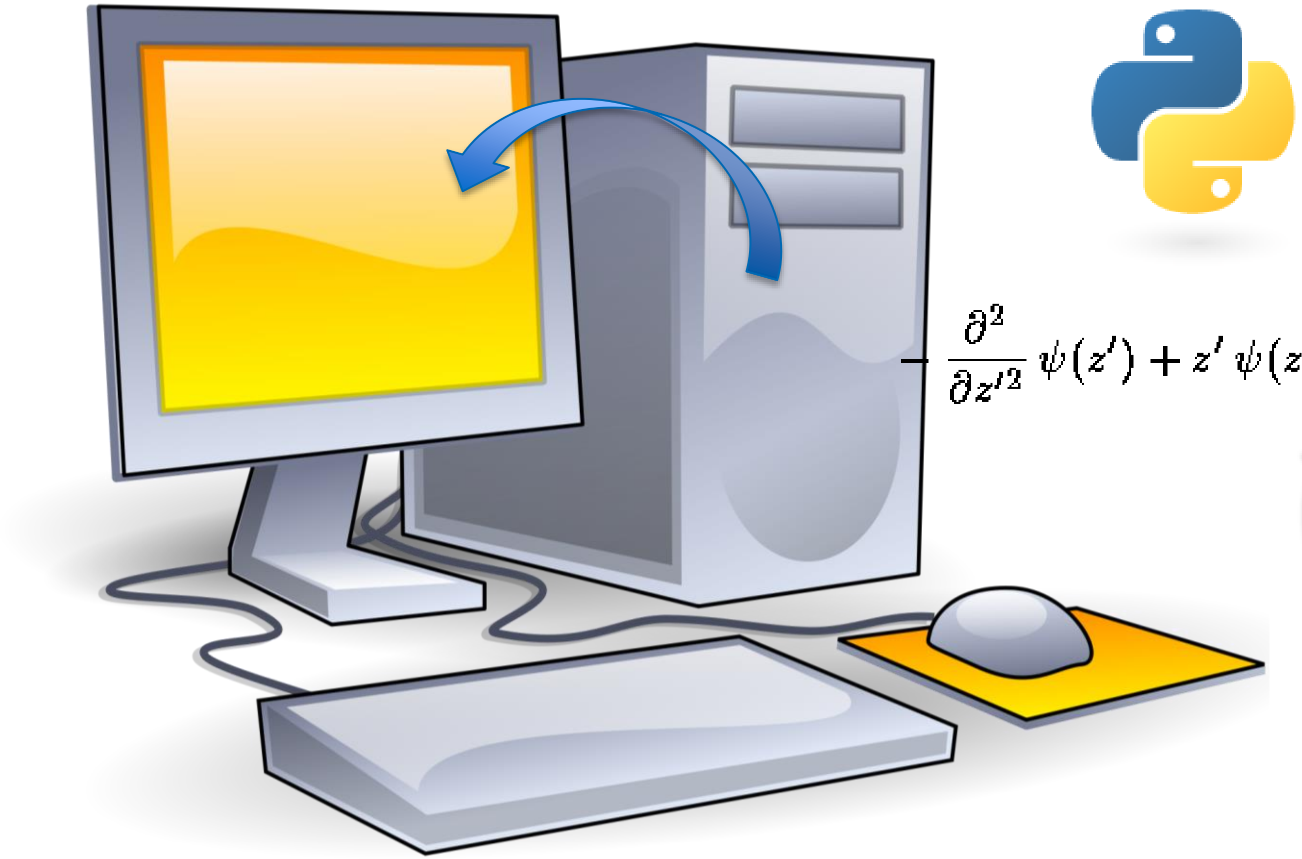
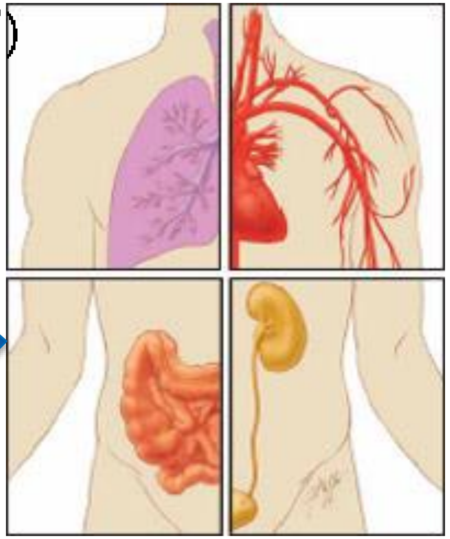
HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Anna Matuszyńska

V-Modul 494: Introduction to Mathematical Modelling of Biological Systems



$$-\frac{\partial^2}{\partial z'^2}\psi(z') + z'\,\psi(z') \;=\; E'\,\psi(z')$$



Quantitative und Theoretische Biologie
Heinrich-Heine-Universität Düsseldorf
Universitätsstraße 1
Gebäude: 25.32
Etage/Raum: 03.26

Our goal

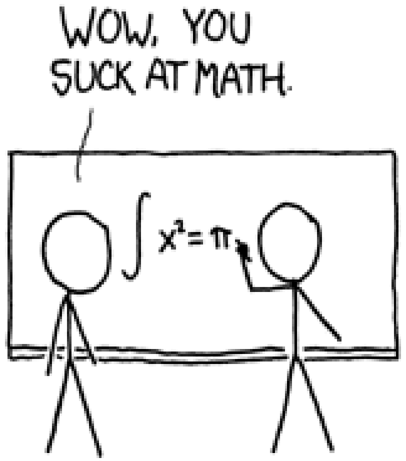
Learning outcomes

- You are familiar with simple mathematical models that describe the prototype of biological systems
- You can describe simple dynamic systems with equations
- You are able to simulate simple dynamic systems using Python language
- You can display the results graphically in different ways and interpret
- You are able to analyze simple mathematical models

You prove it with your final project and during the exam

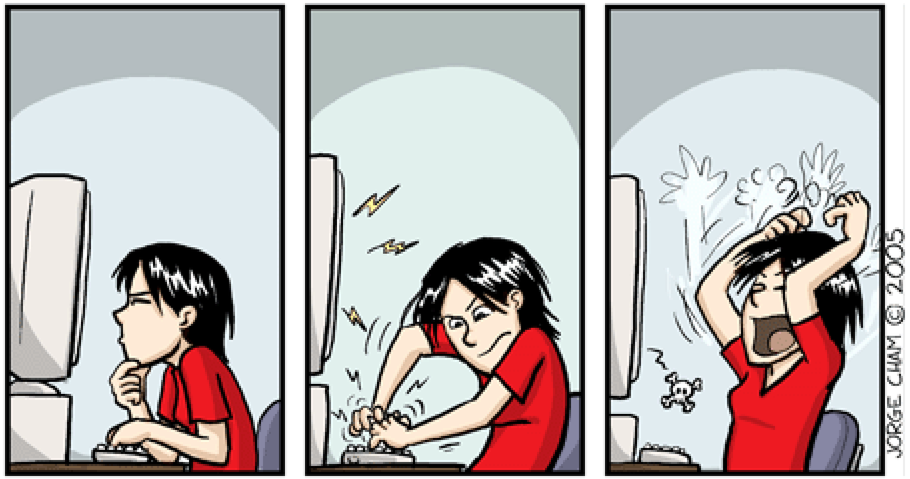
Scared of programming?

What is programming



+

=



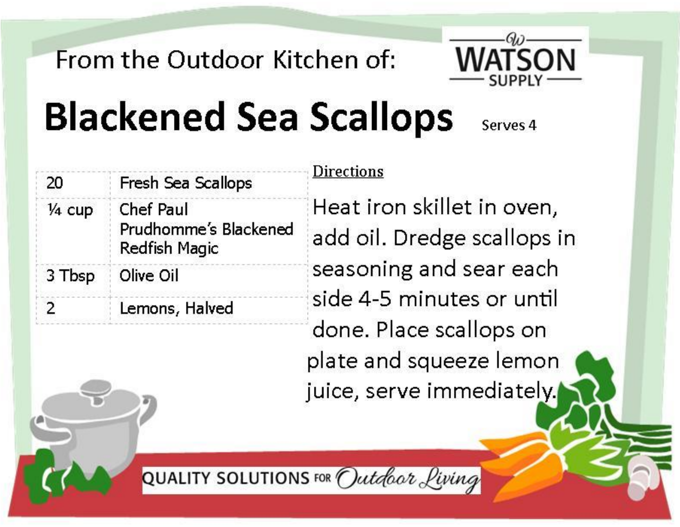
www.phdcomics.com

Let’s play: studio.code.org

Scared of programming?

Programming is like…

- … cooking: you prepare a list of variables (ingredients) and their data values (amount) and a list of stack manipulation instructions (recipe) so someone else can repeat it and obtain the same result (delicious dish)



From the Outdoor Kitchen of: WATSON SUPPLY

Blackened Sea Scallops Serves 4

20	Fresh Sea Scallops
¼ cup	Chef Paul Prudhomme's Blackened Redfish Magic
3 Tbsp	Olive Oil
2	Lemons, Halved

Directions

Heat iron skillet in oven, add oil. Dredge scallops in seasoning and sear each side 4-5 minutes or until done. Place scallops on plate and squeeze lemon juice, serve immediately.

QUALITY SOLUTIONS FOR Outdoor Living

- … training a dog: A computer is a simple device that knows how to remember things and how to look up those memories. It starts out with a blank slate of memory and relies on a human to tell it what to do with that memory.

Who uses Python

Organizations using Python

- Web Development: Google, Yahoo, Shopzilla
- Games: Battelfield 2, Civilization 4, Star Trek Bridge Commander
- Financial: Altis Investment Management
- Graphics: Walt Disney Feature Animation
- Software Development: Nokia, Red Hat
- Science: The National Research Council of Canada,

  Los Alamos National Laboratory Theoretical Physics Division

  NASA

Source: https://wiki.python.org/moin/OrganizationsUsingPython

Useful sources

- Python Software Foundation:

https://www.python.org/

- How to Think Like a Computer Scientist:

http://openbookproject.net/thinkcs/python/english2e/

- Scientific Tool for Python:

http://wiki.scipy.org/SciPy

- 2D Plotting Library:

http://matplotlib.org/

- 46 Simple Python Exercises:

http://www.ling.gu.se/~lager/python_exercises.html

- http://pythonforbiologists.com/

Before we start

Uniqueness of Python

- Python language provides constructs intended to enable clear programs on both a small and large scale
- The core philosophy of the language is summarized by the document "PEP 20 (The Zen of Python)”

```
>>> import this
```

- Python starts indexing from 0
- Python is space sensitive. Indentation required

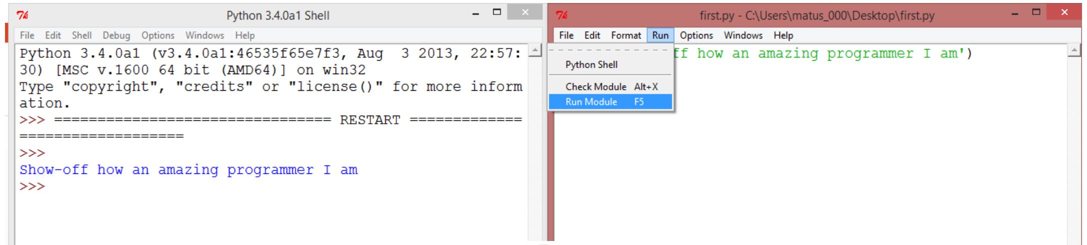
```
for a in range(50):
print ‘hello’ + a
```

```
for a in range(50):
     print ‘hello’+ a
```

Before we start

Editor and interpreter

- We will use the Python IDLE: the text editor for code execution
- IDLE is the Python IDE (integrated or interactive development environment).
- It is a software application that provides us comprehensive facilities for proper software development. It consists of:
  - Source code editor
  - Build automation tools
  - Debugger
  - Interactive interpreter (Python shell window)



```
Python 3.4.0a1 (v3.4.0a1:46535f65e7f3, Aug  3 2013, 22:57:
30) [MSC v.1600 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more inform
ation.
>>> ================================ RESTART =============
===================
>>>
Show-off how an amazing programmer I am
>>>
```

```
ff how an amazing programmer I am')
```

Python 3.4.0a1 Shell
File Edit Shell Debug Options Windows Help
first.py - C:\Users\matus_000\Desktop\first.py
File Edit Format Run Options Windows Help
Python Shell
Check Module Alt+X
Run Module F5

PYTHON

Gentle introduction to programming

Gentle introduction to programming

Syntax and semantix

```
x = 5
```

- The meaning of this term is to assign value 5 to a variable called x. We call it semantics.
- Programming languages offer different ways to provide the same semantix:

Python

```
x = 5
```

R

```
x <- 5
```

Pascal

```
x := 5;
```

- The syntax is the set of rules that defines how a program should be written. It is language-specific constraint on how we express semantics.

Gentle introduction to programming

All we need

1. Comments
2. Data Types:
   - Numbers
   - Strings
   - Lists
   - Tuples
   - Dictionaries
3. Variables
4. Operators
5. If Statement
6. For Loop
7. While Loop
8. Functions

Comments

Code Tells You How, Comments Tell You Why

- One line: `#` (hash) sign
- Multiple lines: 3 single `'''` or double `"""` quotes before and after

Ask yourself: "What questions would be asked by someone looking at this code for the first time?“ and write the comment so it includes the answer.

Data types

- Python has five standard data types:
  - Number
  - String
  - List
  - Tuple
  - Dictionary
- For example: my age would be stored as a numeric value,

my name as set of letters,
and address as mixture of numbers and letters (alphanumeric characters)

Strings

- A sequence of characters

```
>>> my_string = 'Anna'
>>> your_string = 'Student'
```

- Any characters:

```
>>> try_that = 'a2na_matuszYnska!'
```

- Python offers you several built-in methods. Investigate what they do:
  - my_string.count(‘a')
  - my_string.find(‘n')
  - my_string.lower()
  - my_string.upper()
  - my_string.replace('a', 'b')
  - my_string.strip()

Lists

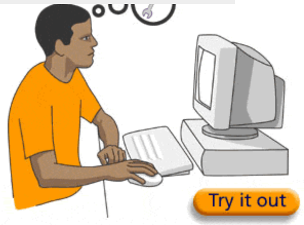
- The most basic data structure in Python is the **sequence**.
- Each element of a sequence is assigned a number - its position or index. Remember: The first index is 0, the second index is 1

```
>>> list = [1,2,3,4,5]
>>> list
[1, 2, 3, 4, 5]
>>> print(list)
[1, 2, 3, 4, 5]
>>> list[0]
1
>>> list[-1]
```

Lists

- We may wish to count number of characters in the list, sort it ascending, delete certain position or append.
- As for strings, Python offers set of built-in methods that can be applied to it. Find them using dir() function
- dir(my_object) is a provided method that attempt to return a list of valid attributes for that object.

```
>>> dir(list)
[… 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert',
'pop', 'remove', 'reverse', 'sort']
```



Lists: exercises

1. Define a list containing the age of all of your family members.
2. Sort this list from the youngest to the oldest.
3. Delete the youngest person from the list.
4. Add number 27 to your list.
5. Change the second number in the list to 14.
6. Revert the order of the list.
7. Create new list with only first two elements of your original list.
8. Concatenate two sorted lists into a new list.

Tuples

- Tuples are almost identical to lists but in contrary to the latter, they cannot be changed. Try it.

Hint: You can either try to append it or change selected value

```
>>> list = [1,2,3,4,5]
>>> tuple = (1,2,3,4,5)
```

Dictionaries

- A dictionary is mutable and is another type of a container that can store any number of Python objects.
- Dictionary in English: Wörterbuch in German

```
>>> dictionary = {'English': 'dictionary', 'Deutsch':
'Wörterbuch', 'Polski': 'Słownik', 'Italiano': 'dizionario'}
>>> dictionary
{'Italiano': 'dizionario', 'English': 'dictionary', 'Polski':
'Słownik', 'Deutsch': 'Wörterbuch'}
```

- How is dictionary in Italian?

Variables

- Variables are nothing but reserved memory locations to store values.
- Python supports dynamic name resolution (late binding), which binds method and variable names during program execution
- Python interprets and declares variables only when they are set equal to …

```
>>> a = 5
>>> type(a)
int
>>> b = “class”
>>> a = b
>>> type(a)
str
```

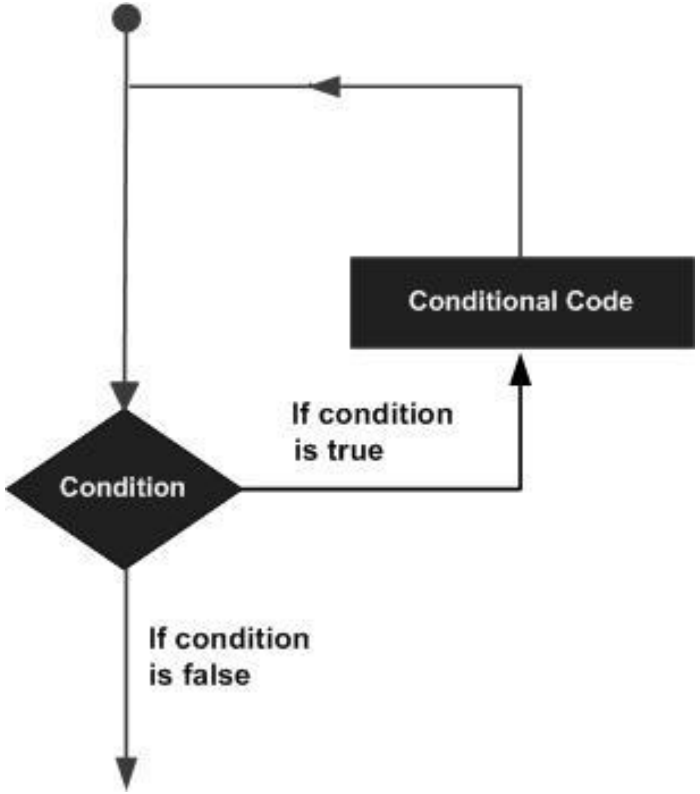
Operators and statements

- With every programming language we have operators:
  - Assignment: =
  - Arithmetic: well known: + , -, *, /
    and quite new: %, **, //, ./
  - Comparison: >, <
  - Logical: and, in, or, not
  - Increment/decrement: +=, -=
- Python, among others, include:

If, for, while, try, class, def, with, pass, assert, yield, import, print

If statement

- The heart of programming logic
- It is a conditional that when satisfied activates some part of code
- We use if statement all the time:
  If I pass it, I will come for a beer. If not, I won’t come.
  If I will fail, my parents will kill me.
  If I will make all exercises I will learn.
  If I won’t pay the bill, they will cut off my electricity.



Gentle introduction to programming

If statement

- Syntax: : colon after the condition, action in the new, indented line
  if for first condition,
  ____action
  elif for second condition and more,
  else for the last condition

```
if condition:
   do something
elif other condition:
   do something else

else:
    do something different
```

- Write a command that will append the list with number two only if it has an odd length. Example: list = [2, 3, 4] should be appended

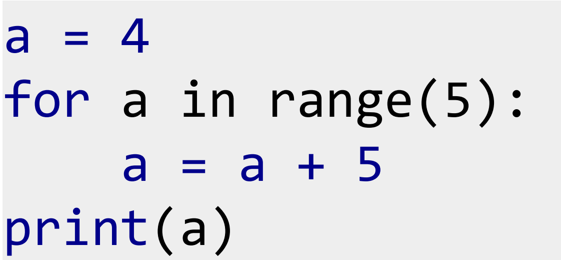
For loop

- If you wish to repeat some code certain number of times you will just loop through that code for desired number of times.
- Syntax: for how long:

____action

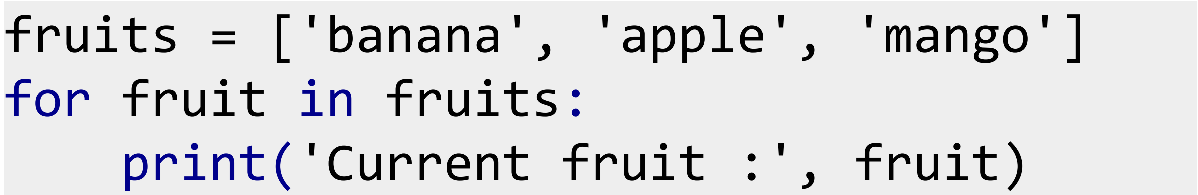
- How long can be expressed in many ways.
- Examples:

“Increase variable a by 5, for 5 times”



```
a = 4
for a in range(5):
    a = a + 5
print(a)
```

“Print all elements of the list”



```
fruits = ['banana', 'apple', 'mango']
for fruit in fruits:
    print('Current fruit :', fruit)
```

While loop

- You may also repeatedly execute a targeted statement as long as some given condition is true (you don’t know how many times)
- Syntax: while expression:

    _____action

```
count = 0
while (count < 9):
   print('The count is:', count)
   count += 1
```

- Be careful: a loop becomes infinite if a condition never becomes false.
- Try to write a loop that will never break.
  Now try to stop it.

Loops: exercises

1. Find all numbers dividable by three between 0 and 100.
   Hint: Use combination of an if statement with a for loop
   Hint: There is built in method that will quickly find numbers that divided by 3 return no reminder
2. Now change this code so the results will be stored in a list.
   Hint: You can declare an empty list called results before the loop.
3. Now select from this list only even numbers.

Functions

- Tired of repeating pieces of code? Any code that you think you will ever use againse, you should probably put in a function.
- Syntax: def name_of_function(arguments it takes):

____action

```
def my_first():
   print(‘first function’)

def my_second(a):
   a = a + 5
   return a
```

- How to call your function?
  Functions are always called by their name, followed with parenthesis and arguments inside. Example: `my_first(), my_second(7)`

Functions: exercises

1. Define a function that takes a list of numbers as an argument and returns the largest of them. Use the if statement that you have just learned.
   Comment: Python has a built in method called max(). Don't use it here.
2. Write a function that takes a number and returns a list of its digits.
3. Write a function that checks whether an element occurs in a list.
4. Define two functions that sums and multiplies respectively all the numbers in a list of numbers.
   Example: my_sum([2, 3, 4]) should return 9
5. Define a function find_longest_word() that takes a list of words and returns the length of the longest one.

More: http://www.ling.gu.se/~lager/python_exercises.html

First program

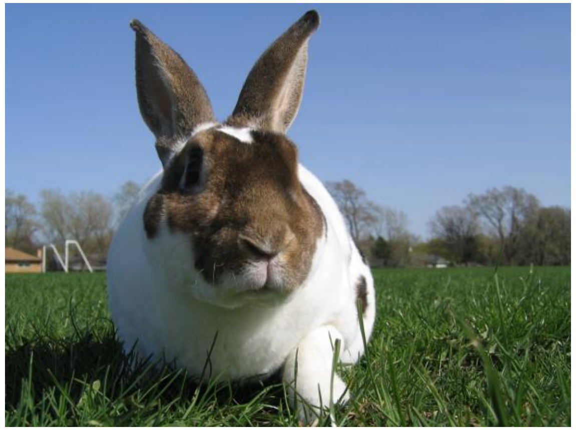
Fibonacci Numbers

We will consider an **idealized biological system** published in Liber Abaci (1202) by Leonardo of Pisa, known as Fibonacci.

Puzzle: We have rabbit population and wish to know how many rabbits will we have after one year.

Assumptions: a newly born pair of rabbits, one male, one female, are put in a field;

- rabbits are able to mate at the age of one month
- mating pair always produces one new pair
- rabbits never die

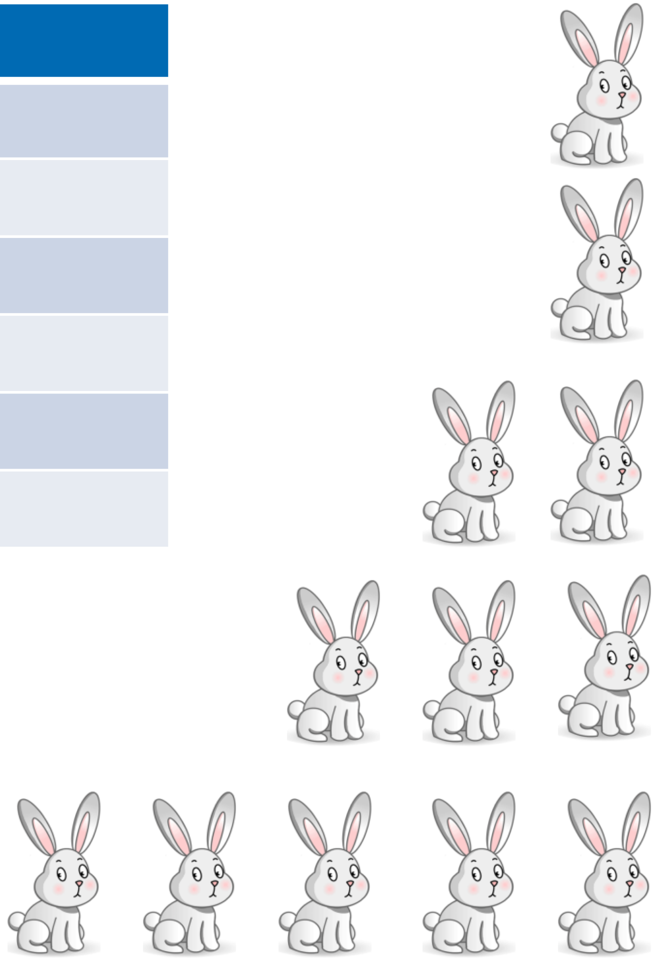


First program

Fibonacci Numbers

Month	Rabbit pairs
0	0
1	1
2	?
3	
4	
5	

$$F_n = F_{n-1} + F_{n-2},$$



First program

Recursion vs. iteration

- A recursive method is a method that calls itself either directly or indirectly

Approach to reduce problem to its simpler version:

1. A simple base case (or cases)
2. A set of rules that reduce all other cases toward the base case

```
if base:
    ____________
else:
    ____________
```

- An iterative method is the act of repeating a process with the aim of approaching a desired goal.

It is using the output from one iteration as the input to the next.

```
if ____:
   result = do input
   input = result
```

loop

Sum Up:
Iterative Algorithms = Fast Performance but Hard to write
Recursive Algorithms = Fast to write but Bad performance

How to program it?

Code

```
def fib_rec(n):
        """ assumes n an int >= 0
        returns Fibonacci of n
        """
        if n == 0:
                return 0
        elif n == 1:
                return 1
        else:
                return fib(n-1) + fib(n-2)

def testFib(n):
        for i in range(n+1):
                print ‘fib of’, i, ‘=‘, fib_rec(i)
```

```
def fib_iter(n):
        a,b = 0, 1
        for i in range(0,n):
                a,b = b, a+b
        return a
```

How to program it?

There is always another way

- Closed-form expression known as Binet's formula

$$F_n = \frac{1}{\sqrt{5}} \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n .$$

```
from math import sqrt
def fib(n):
     F = ((1+sqrt(5))**n-(1-sqrt(5))**n)/(2**n*sqrt(5))
     return F
```

```
from math import sqrt
def fib(n): return ((1+sqrt(5))**n-(1-sqrt(5))**n)/(2**n*sqrt(5))
```

Step ahead

Modules and packages

- Python comes with a library of standard modules.
- Some modules are built into the interpreter; these provide access to operations that are not part of the core of the language but are nevertheless built in:
  - We have used so far for instance: len(), type()
- Other modules you need to call in order to use their function:

```
from math import sqrt
```

- Packages are collection of modules. They provide a way of structuring Python’s module namespace by using “dotted module names”.

Over next days we will be using lots of such packages.

First program

Variations on the Fibonacci numbers: exercise

1. Define a function that instead of a particular Fibonacci number will return the list of the first n Fibonacci numbers.
   Example: fib_seq(5) should return 1, 1, 2, 3, 5
2. Define a function that will return Fibonacci numbers between given range.
   Example: fib_seq(3, 5) should return 2, 3, 5

```
def fib(n):
        a, b = 0, 1
        while a < n:
                print a
                a, b = b, a+b
        print()
```

SOLVING DIFFERENTIAL EQUATIONS

You can simulate simple dynamic systems

First differential equation

Simple Logistic Growth Equation

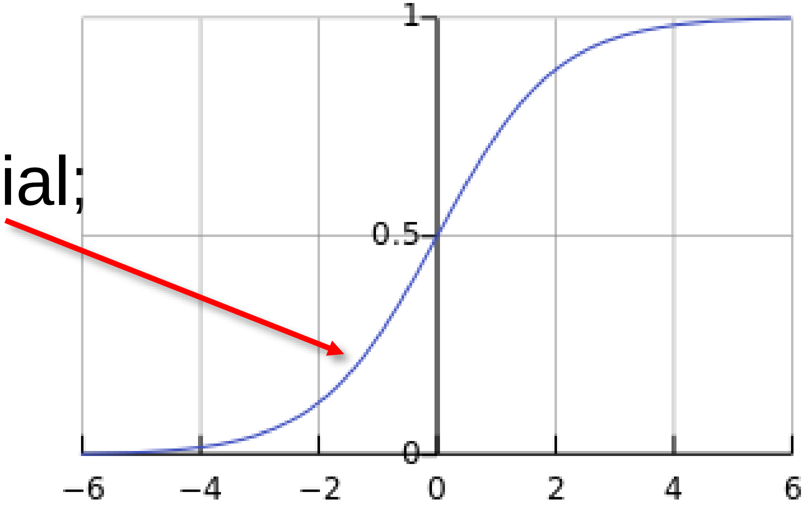
- Letting N represent population size and t represent time, this model is formalized by the differential equation:

$$\frac{dN}{dt} = rN(1 - \frac{N}{k})$$

N represents population size; r defines the growth rate; k is carrying capacity

- The function is a sigmoid curve.

The initial stage of growth is approximately exponential; as saturation begins, the growth slows, and at maturity, growth stops.



First differential equation

Tasks for today

- Goal: Your goal is to write two Python scripts to solve two one-dimensional problems: the logistic equation and the logistic equation with punishment for low population.

How to start:

- Mathematical description
- Initial conditions
- Parameter set
- Necessary tools for integration and plotting

Before we start

Import

```
import numpy as np
import scipy
import matplotlib.pyplot as plt
import pylab
```

We will be using functions that are not built in as a standard one. Therefore we need to import specific packages.

NumPy is a Python extension module that provides efficient operation on arrays.

SciPy is a set of open source (BSD licensed) scientific and numerical tools for Python.

Matplotlib is a python 2D plotting library which produces publication quality figures.

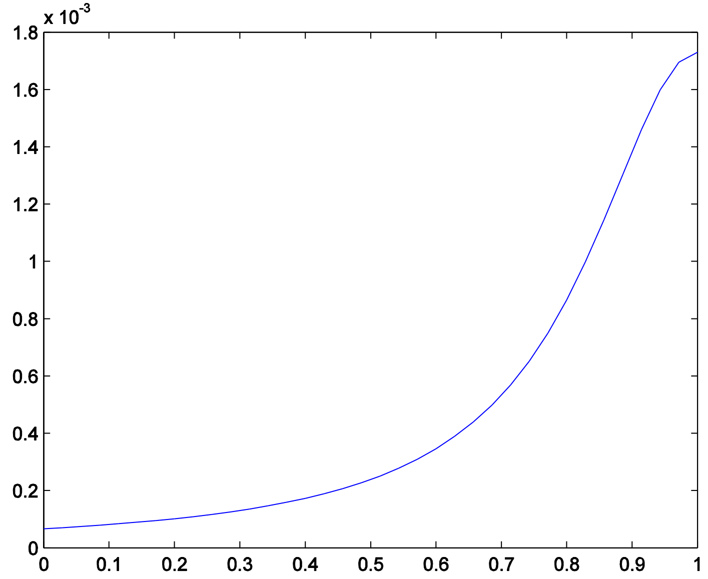
GRAPHICAL PACKAGE

You can display the results graphically

Using graphical package

How to represent my data

~~`from pylab import *`~~



Create a figure.

Create your data. We need X and Y values.

```
X = [1,2,3]               import numpy as np
X = range()               X = np.linspace()
                          X = np.arange()
```

Use the pylab package to plot your results.

```
import pylab as pl
pl.plot()
```

Show results of screen

```
pl.show()
```

Using graphical package

How to represent my data

1. Plot function f(x) = x.
2. Plot sine and cosine functions on the same plot.
3. Change the colours of the plot to red and blue.
4. Add legend, title and axes names.
5. Try to plot two plots next to each other.

scipy-lectures.github.io/intro/matplotlib/matplotlib.html

SOLVING DIFFERENTIAL EQUATIONS

You can simulate simple dynamic systems

First differential equation

Simple Logistic Growth Equation

```
def logistic(y, t0, r, K):
    """ returns population  growth """
    dY = r * y[0] * (1 - y[0]/ float(K))
    return
```

Note that t0 is not used by the function. Why do we supply it as an argument?

First differential equation

Simple Logistic Growth Equation

```
def logistic(y, t0, r, K):
        """ returns population  growth """
        dY = r * y[0] * (1 – y[0]/ float(K))
        
        return
```

Note that t0 is not used by the function. Why do we supply it as an argument?

Provide values of r and K and starting population size.

```
params = (0.3, 10)
Y = [1]
```

What is the type of the params value?

First differential equation

Simple Logistic Growth Equation: integration

```
growth = scipy.integrate.odeint(func, y0, t, args=(), …)
```

First differential equation

Simple Logistic Growth Equation: integration

```
growth = scipy.integrate.odeint(func, y0, t, args=(), …)
```

```
t = range(0,1000)
growth = scipy.integrate.odeint(logistic, y, t, args=params)

plt.plot(t, growth)
plt.show()
```

Second differential equation

Modified Verhulst Equation

```
def modifiedVerhulst(N, t0, c=5., g =3, d = 1.,r= 2., k = 40.):
        """ returns the population growth rate
        modified Verhulst equation with the factor
        governing population behaviour at small sizes   """
        dY = r * N * (g/r) * N/(N+c) – d/r – N/k

        return
```

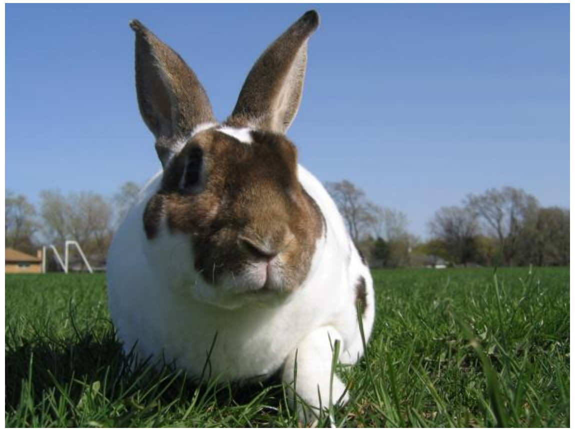
Tasks for today:

- Integrate the model over the time of 1 year, how the population evolves?
- Find the stationary states of the model. Are they stable?
- Plot the bifurcation diagram for the new bifurcation parameter c.

Coupled differential equations

Lotka-Voltera Model

- Lets consider population of two species that interact.
- One is a predator and second a prey.
- How to describe their dependent dynamics?



Coupled differential equations

Lotka-Voltera Model

- b is the natural growing rate of rabbits, when there are no wolfs
- d is the natural dying rate of rabbits, due to predation
- c is the natural dying rate of wolfs, when there are no rabbits
- f is the factor describing how many caught rabbits let create a new wolf

Coupled differential equations

Lotka-Voltera Model

- b is the natural growing rate of rabbits, when there are no wolfs
- d is the natural dying rate of rabbits, due to predation
- c is the natural dying rate of wolfs, when there are no rabbits
- f is the factor describing how many caught rabbits let create a new wolf

rabbits → `dr/dt = b*r - d*r*w`

wolfs → `dw/dt = -c*r + f*b*w*r`

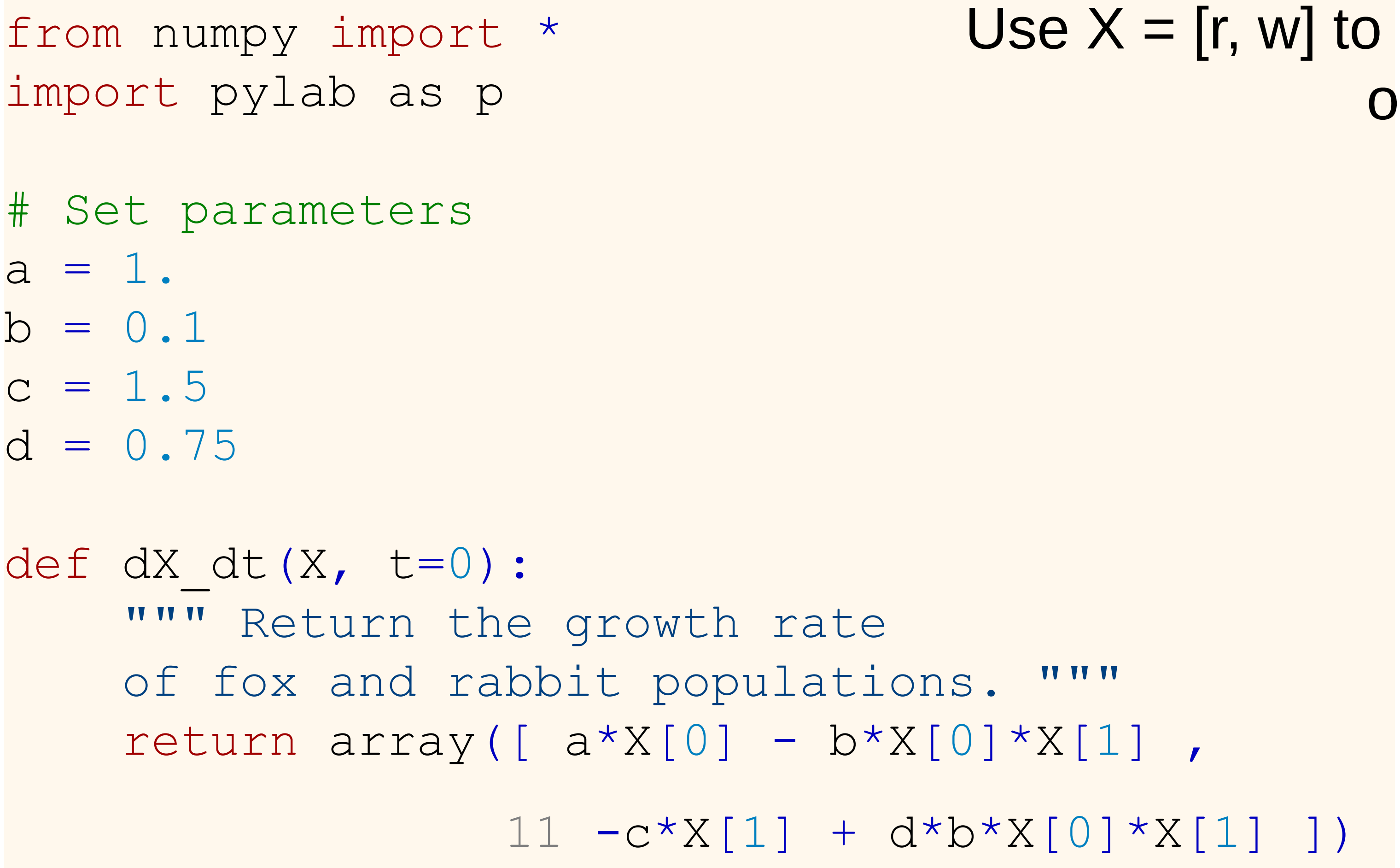
Coupled differential equations

Task

- Solve the Lotka-Volterra model (also known as the predator-prey) and create plots of the evolution of the population for following cases:
  - a. r = d = c = f = 1 for variety of initial conditions
  - b. r = 1, d = 0.1, c = 1.5, f = 0.75 and t = 1000, for R = 10 and W = 5
- What does it mean that the population size is stable over the time?
- Play with parameters and initial conditions so different species will survive.

Coupled differential equations

Lotka-Voltera Model



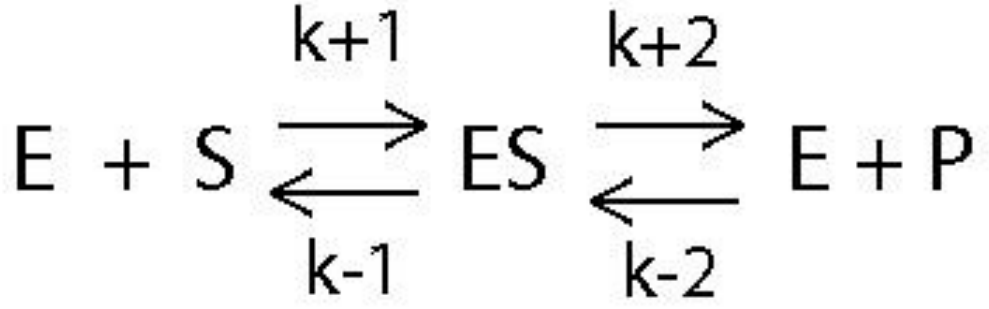
```
from numpy import *
import pylab as p

# Set parameters
a = 1.
b = 0.1
c = 1.5
d = 0.75

def dX_dt(X, t=0):
    """ Return the growth rate
    of fox and rabbit populations. """
    return array([ a*X[0] - b*X[0]*X[1] ,

                   -c*X[1] + d*b*X[0]*X[1] ])
```

Use X = [r, w] to describe the state of both populations

Modelling biochemical reactions

Simple biochemical network

- We will use differential-equation-based models for biological regulatory networks to ulitmately simulate the change in concentrations over the time of substrate (S), enzyme €, substrate-enzyme complex (ES) and the final product (P) in the following reaction:



FINAL PROJECT

You can do it! Prove it with your final project

Final Project

Your task

Using your knowledge about ordinary differential equations (ODE) and programming in Python, you are asked to build an n compartment model that will give a good approximation to the pharmacokinetics of a selected drug or its compound.

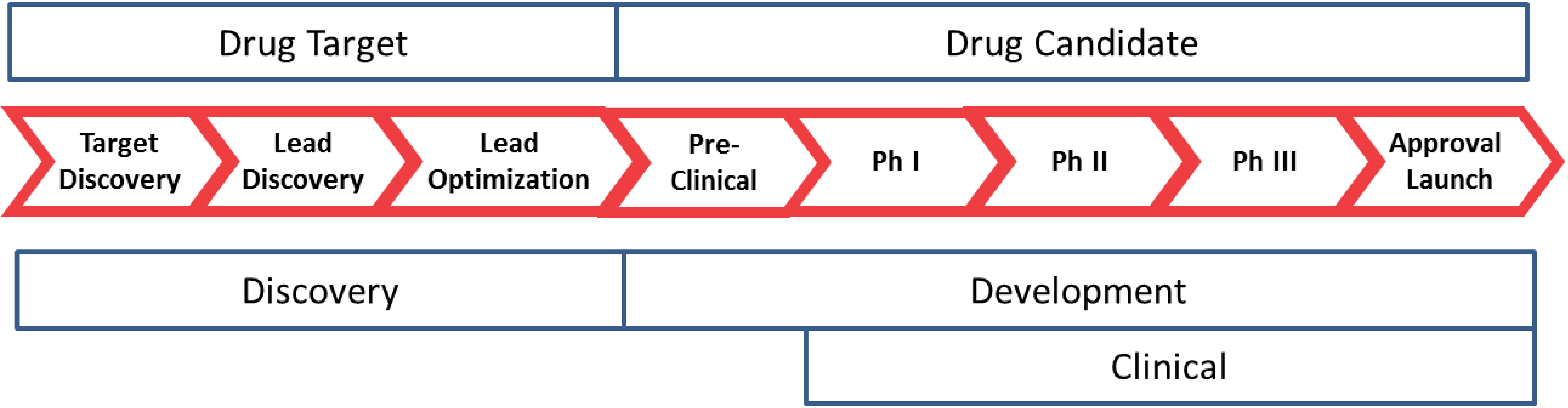
Introduction to pharmacokinetics

Drugs on the market

400 mg
Advil®
PER ZACHTE CAPSULE 400 mg IBUPROFEN
Liquid-Caps 400
20 stuks
Pijnstillend - Koortsverlagend - Ontstekingsremmend

BAYER
ASPIRIN®
Tablette mit 500 mg Acetylsalicylsäure
50 Tabletten
Zum Einnehmen mit reichlich Flüssigkeit.
Nicht über 30 °C lagern.

Drug Discovery and Development



Introduction to pharmacokinetics

Pharmacokinetics

- the science of studying drugs in the body and how they are affected by different processes
- describes the behavior of an administered drug in the body over time
- uses mathematical equations to relate different variables to each other
- uses this to make predictions about drug behaviour in the body
- used to administer the drug appropriately and safely

Mathematical modeling of pharmacokinetics / pharmacodynamics (PKPD) is an important and growing field in drug development.

Vocabulary

- Compartment – a concept, it can be a tissue or organ or an entire body
- Dose – amount of drug administered
- Concentration – amount of the drug in a given volume of plasma
- Clearance –the volume of plasma cleared of the drug per unit of time
- Half-life – time it takes for a substance to lose half of its pharmacological activity

Example of a 2 compartment model

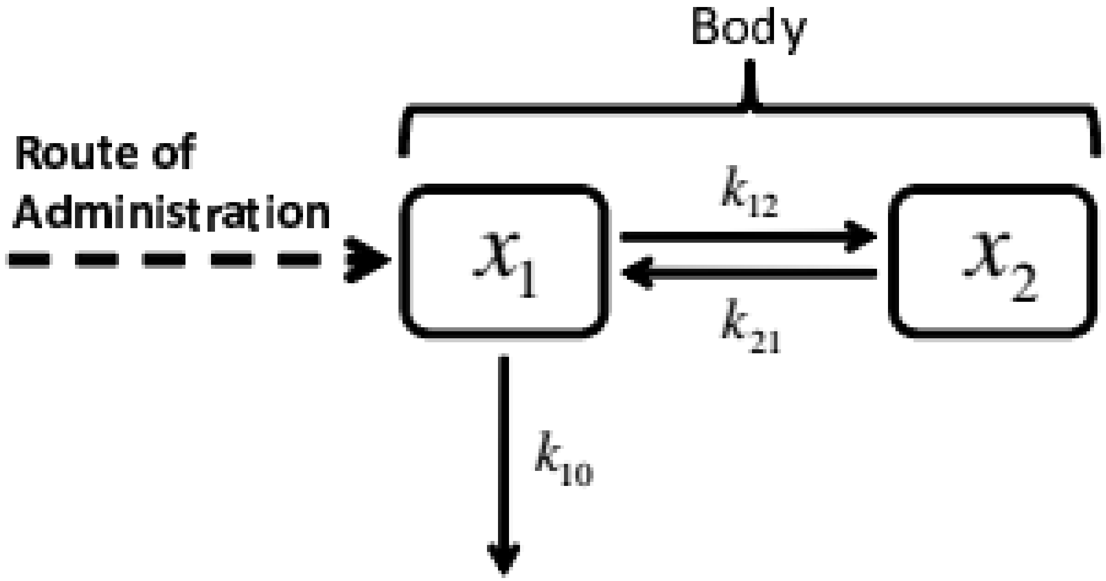
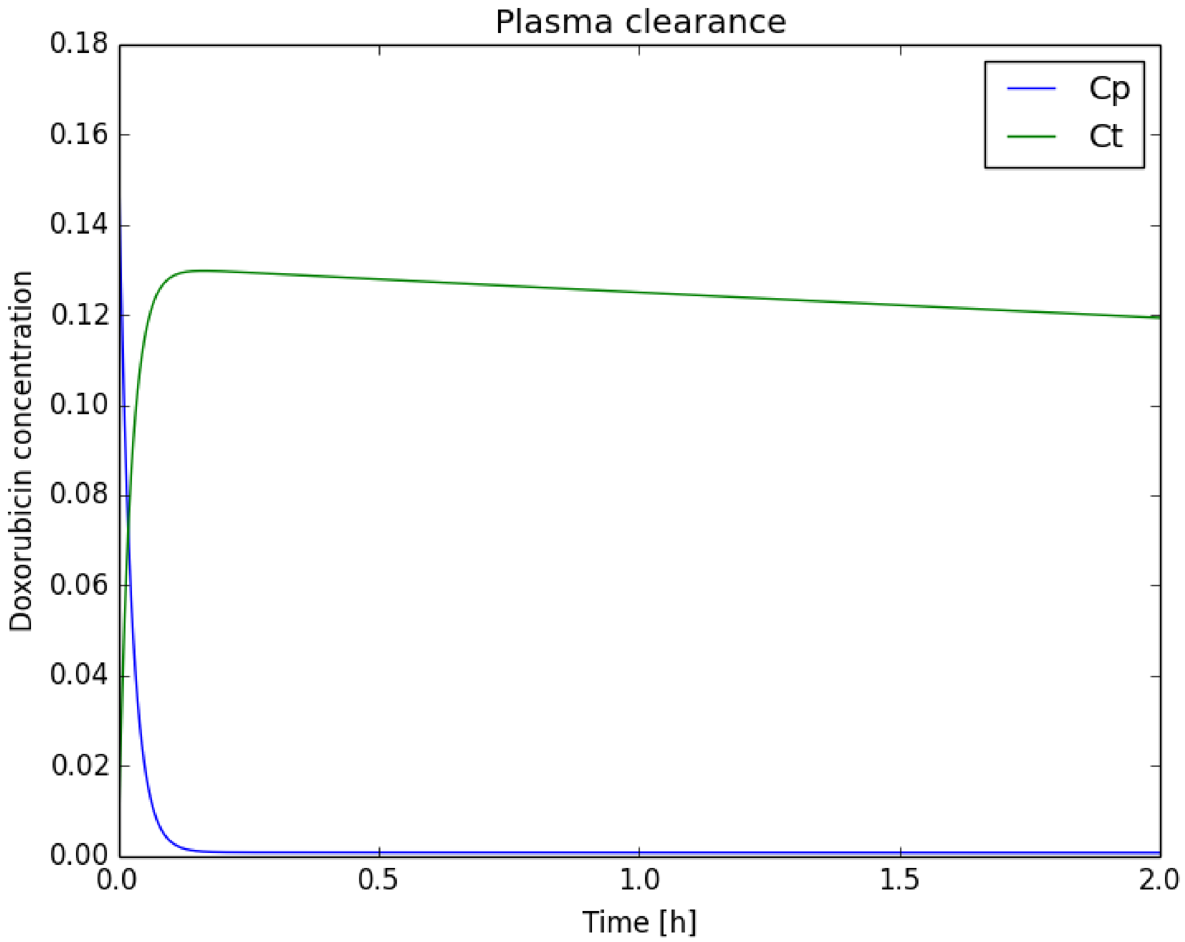


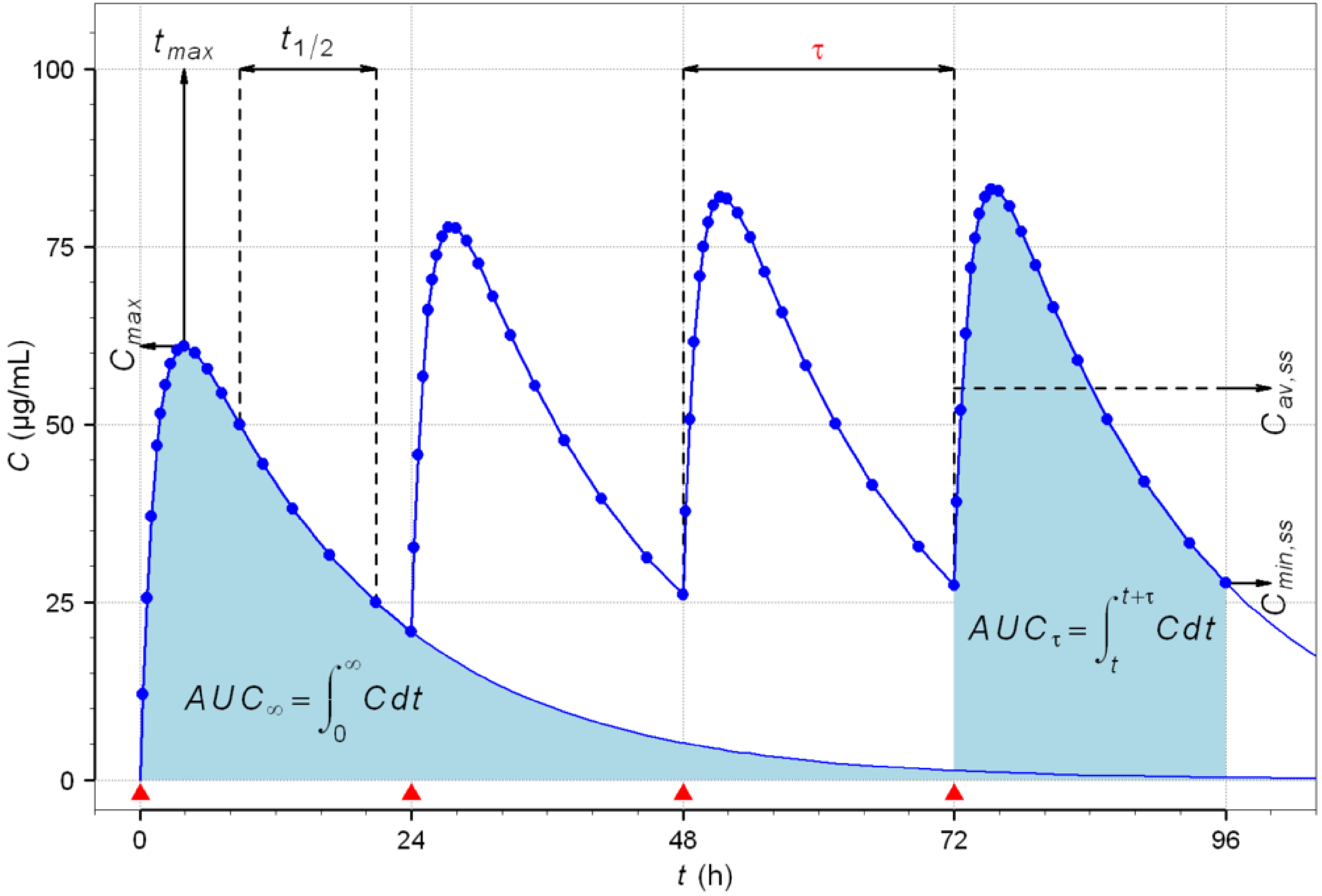
Figure 3.1: *General scheme of the two-compartment model.*

Source: Gilbert Koch, Modeling of Pharmacokinetics and Pharmacodynamics with Application to Cancer and Arthritis

Single administration



Administration every 24 hours



The project

Group division

- You will be working in three groups:

1. Melanie, Constantin, Oussamah, Pascal
2. Ann – Katharin, Christine, Dario, Julius
3. Alina, Nima, Simon

You are supposed to submit one code but seperate reports for each member of the group.

Deadline: 12th December 2014, 1 pm.