Student infographics created for the 2013 course:
Introduction to Computer Science
(1st year BA in Communication & IT, University of Copenhagen)

Teacher: Christina Lioma

Working with functions

What my friends think I do.

What my mom thinks I do.

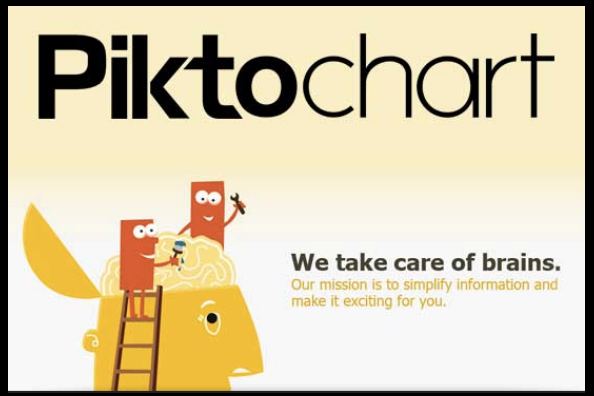
What society thinks I do.

population_growth_adv.py × paper_folding.py ×

```
x = 0
while (x<=30):
    width = (x+1)/200
    print('When folded ', x, 'times, the width is: ', width/100, 'meters. ')
    x += 1
input('Press Enter to exit program. ')
```

What my professor thinks I do in class.

What I think I do.



What I actually do

Moores Law

"The same amount of money would buy twice as much computing power, every 18th month"

Gordon E. Moore - 1965

1990

2012

1.

What we learned today

Computer science
- Hardware
- Software

Hardware

Secondary storage
Main memory
CPU
Input
Output

Software

Application software
System software

Output:
- Display
- Printer
- Speakers
- Etc.

Input:
- Keyboard
- Mouse
- Scanner
- Etc.

System software:
- Software development tools
- Utility software
- Operating systemt

Computer science is as working with recipes. One part of it is how recipes are written. Another part is about the units used in recipes. Third part is figuring out what kind of tasks the recipes can be written for. Fourth part is how well the recipes work. And the fifth and final is specialised recipes.

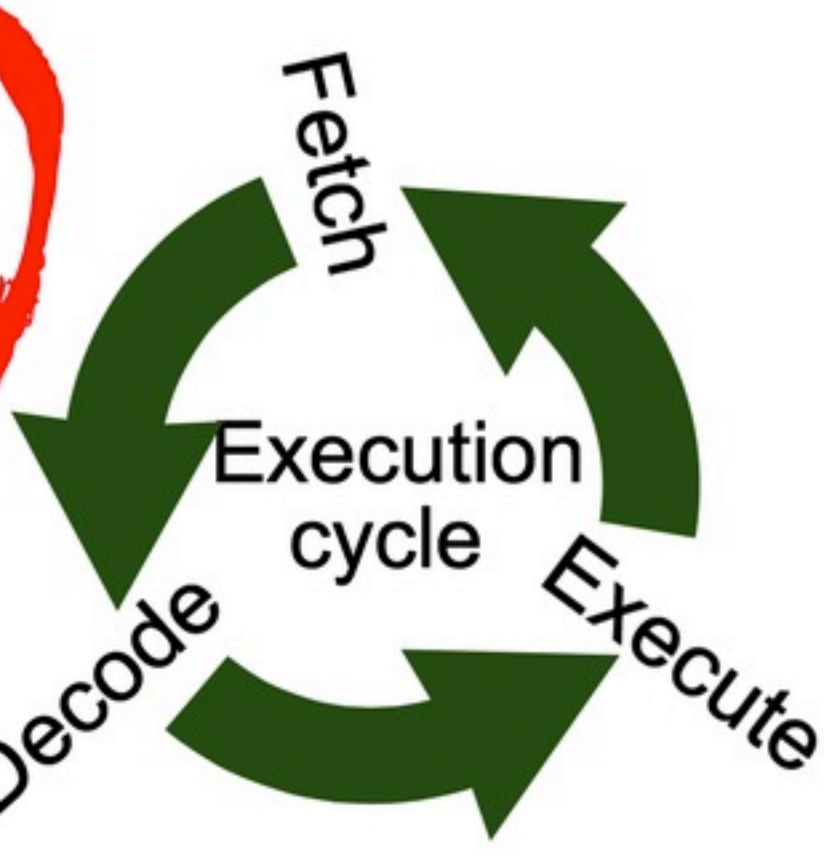
Basic notions:

Algorithm

Program

Software

GROUP 8



Recipe for Computers

Start by stirring your CPU.

Then add 8 GB main memory and season with in- and outputs of your own choosing.

Mix it all together in the second storage device, preheat the motherboard to a 85° and let it run for 45 min.

serving suggestions: serve it with windows software or Python code.

-Sebastian Boring

Hardware/The 5 major componenets

Main components

CPU

The Central Processing Unit runs the programs

Main memory

Main memory/ RAM is where the computer stores a program while the program is running

Seconday storage

Secondary storage/ROM holds data for long periods of time, even when the powers is off.

Input devices

Data the computer collects from people and other devices

(keyboard, microphones)

Output devices

Data produced by the computer for other people or devices

HOMERS PAYDAY

Homer just got a bonus from his work at the nuclear power plant, but he's having trouble figuring out how many beers he can afford....

```
>>> beer
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    beer
NameError: name 'beer' is not defined
>>>
```

SYNTAX ERROR

you have to remember to write input AND int

Think, think, think.....
salary=int(input('Insert salary'))
bonus = int(input('Insert bonus'))
pay = salary+bonus
print('This is your pay', pay)

YOU CAN AFFORD ONE MORE BEER!!!!

BEEEEEEEEEERRRRRR

THANK YOU FLANDERS...

.....and Python

Chapter 3, Python

Function: sequence of statements that has a name

Two parts to designing a function:
1) function definition
2) function invotation

Main function; one main function per program to control how all the other functions invokes.

Functions can contain variables:

Local variable: Defined inside function

Global variable: Can be used anywhere

Two types of arguments

Positional arguments

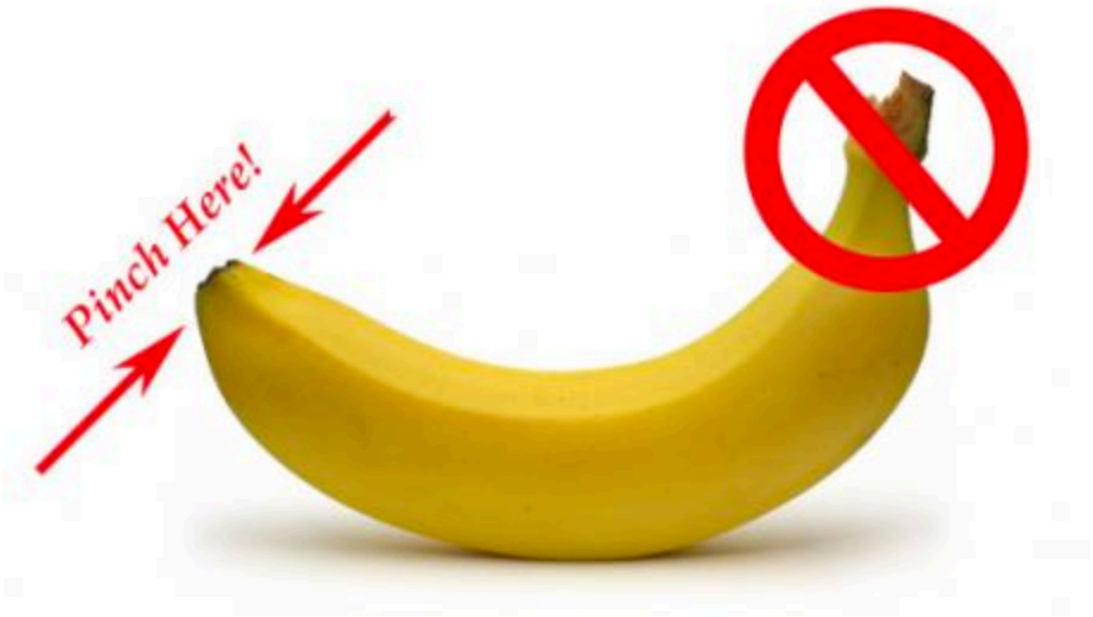
the use of them is determined by the position they have

Keyword arguments

have to be a keyword somewhere in the program

Group 8

HOW TO PEEL YOUR FUNCTION - TOP-DOWN



1. "Def" dat shit up and make your header!

2. Take some statements and put'em in yo block!

3. Call dat b*tich!

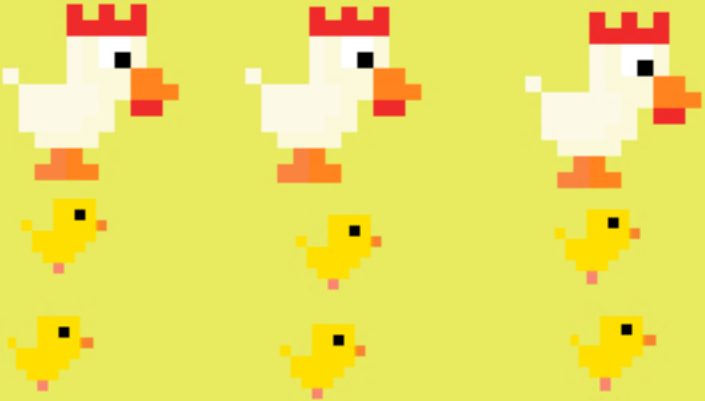
```
def peel_banana():
    pinch = input("where do you pinch (top or bottom)?: ")
    if pinch == ("top"):
        print ("wrong pinch bitch!")
        peel_banana()
    elif pinch == ("bottom"):
        print("Right on mofokker!")


peel_banana()
```

G-squad 10

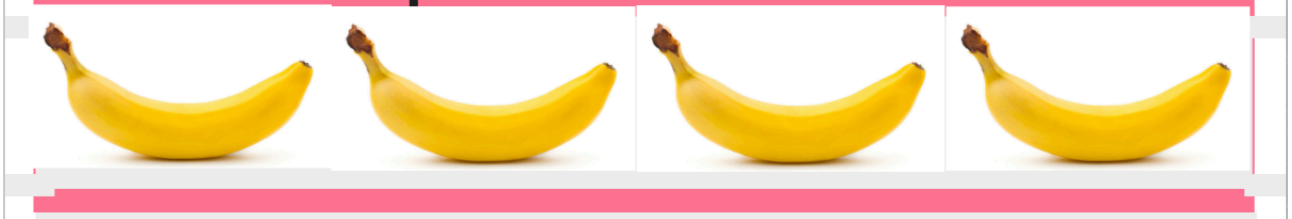
Main() should call all the functions in the program!

When you call a function(), you make it execute (call) all its statements!



But where do I come in??

Simple Functions



Creating a function

STEP 1 Define the function with a header

```
def main():
```

STEP 2 Continue the function with a block of indented statements

```
def main():
    statement
    statement
```

STEP 3 Now invoke the function in the code

```
main()
```

Arguments

Keyword

```
monkey(bananas=12)
```

Positional

```
monkey(bananas)
```

Variables

Local scope

```
def monkey():
    bananas = 12
```

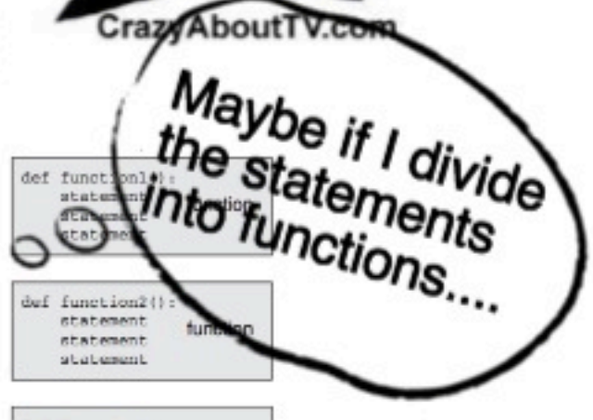
Global varibles

```
def monkey():
    global bananas
    bananas = 12
```

Dextors laboratory

Divide and conquer

```
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
statement   statement   statement
```




```
def function1():
    statement
    statement
    statement

def function2():
    statement
    statement
    statement

def function3():
    statement
    statement
    statement

def function4():
    statement
    statement
    statement
```

function

function

function

But why is that easy????

FIRST I DEFINE A MAIN FUNCTION, THEN I CAN MAKE SUB FUNCTIONS WITHIN THAT FUNCTION AND EVEN MORE SUB FUNCTIONS WITHIN THE SUBFUNCTIONS!!!!!!!

BECAUSE I CAN NOW CONQUER ANY PROGRAMING TASK!!