

CS-A113 Basics in Programming Y1

3rd Lecture
28.09.2021



The Lecture

- **Join with Video** – Makes my life nicer!
- Feel free to open your microphone and ask questions
- Feel free to write questions into the chat
- We will record the sessions and put it unlisted on youtube.

Exercises Slack Update

We would like to have a better structure / overview of the questions posed in Slack for the respective rounds. I created separate channels for each round.

Please join all of the channels and post your questions accordingly.

The exercise-session call will stay on the general #exercises channel. General questions concerning the exercises can still be posted there.



Discussion of Exercises

Tuesdays at 16:15 start two different exercise sessions:

- Recap of submitted exercises (same as lecture Zoom link – you can just stick with it – will most likely not last 2 hours)
 - Presentation of a model solution
 - Common mistakes
 - Discussion in the group about pros and cons of different solutions
 - Questions about this particular round
- The usual 1:1 exercise session where you post your Zoom-Link on Slack



Interactions Today:



Go to:

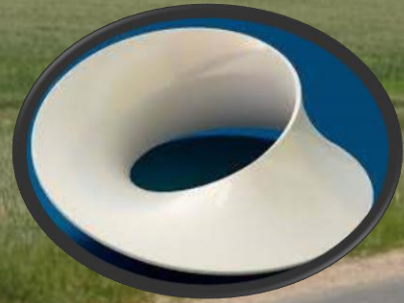
<http://presemo.aalto.fi/csa1113>

Link can also be found in the myCourses page directly below the lecture Zoom link.

Timeline

myStyle

Style
Revisited



Loops revisited



Who are you?

Who are you?

We now know a bit better 😊



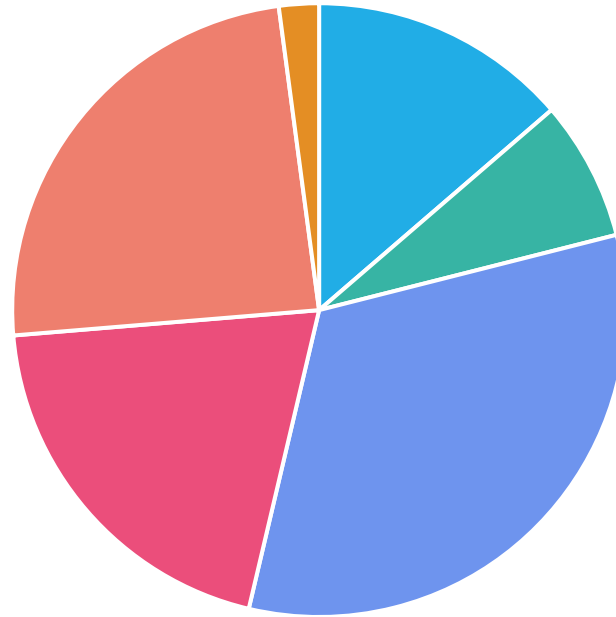
NATIVE



ORIGIN



School



■ Science ■ Engineering ■ Business ■ Chemical Engineering ■ Electrical Engineering ■ Arts, Design and Architecture



Go to:

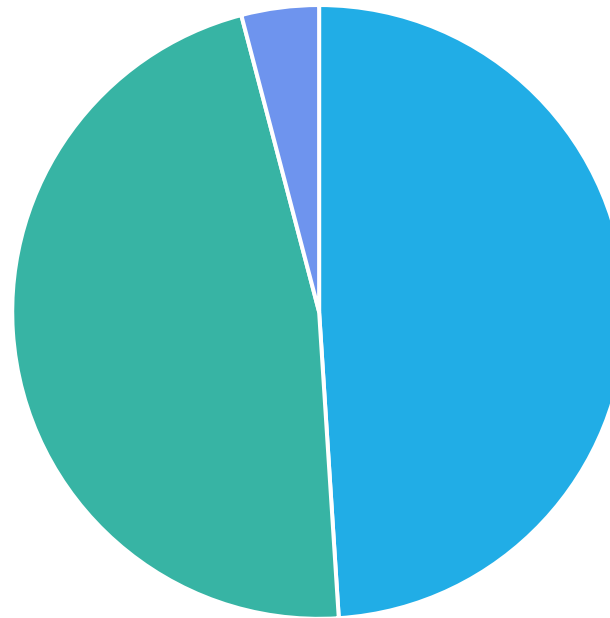
<http://presemo.aalto.fi/csa1113>



Did you take a
programming course
before?

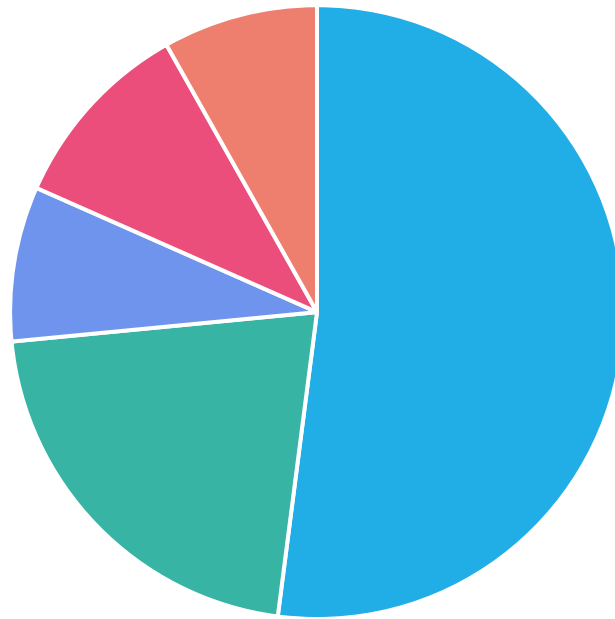
55% did

Gender



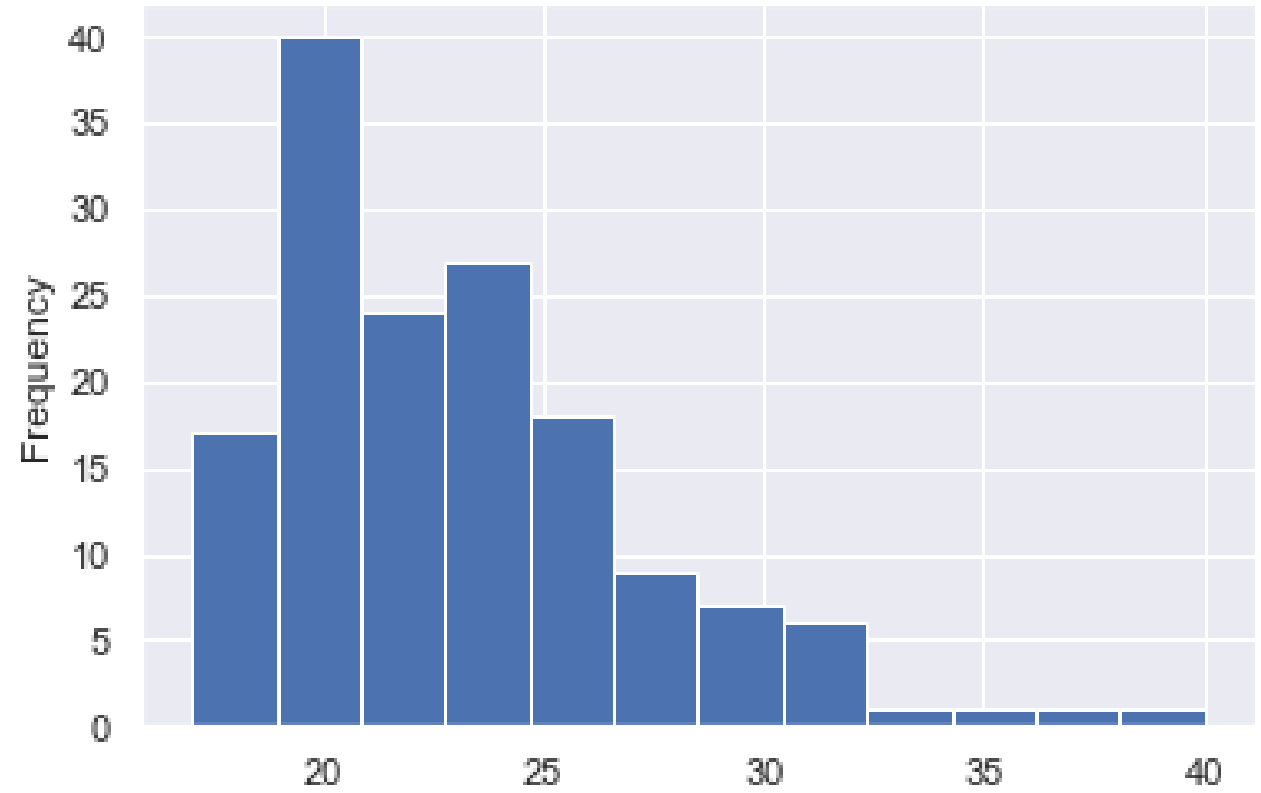
■ Female ■ Male ■ Other/Prefer not to say

Study Year

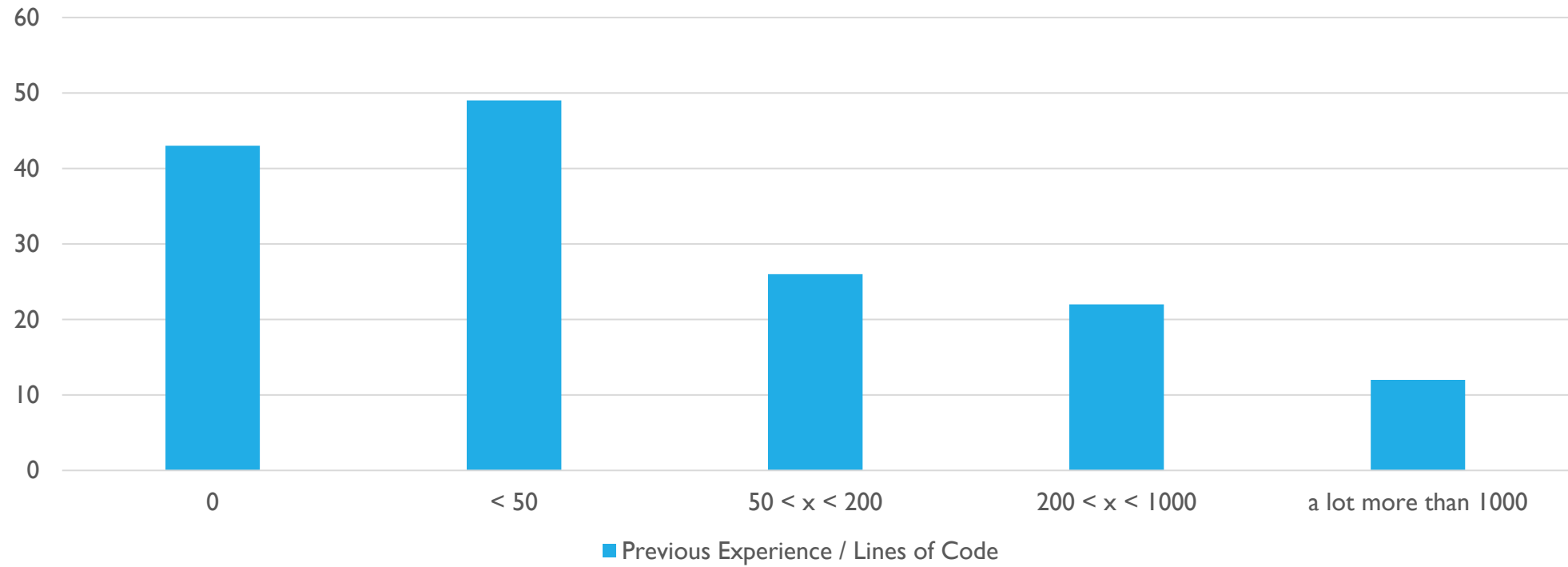


■ 1st ■ 2nd ■ 3rd ■ 4th ■ 5th +

HOW OLD ARE WE?



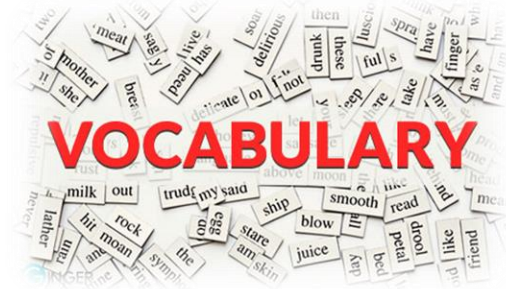
Previous Experience / Lines of Code





For-Loop and Range

Vocabulary and Format



Topics Today



Coding Style & Debugging 101



Go to:

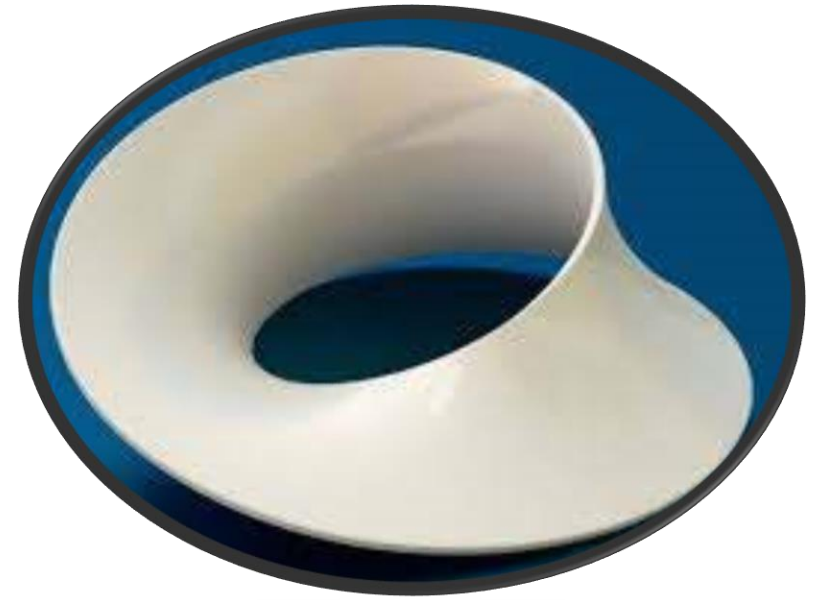
<http://presemo.aalto.fi/a1113>



LOOPS

Loops Recap:

```
def superLoop():  
    i = 1  
    j = i  
    while i < 5:  
        print(i)  
        j += 1
```



For-Loop with Range-Function

```
def superLoop1():  
    for i in range(8):  
        print(i)
```

0,1,2,3,4,5,6,7

```
def superLoop2():  
    for i in range(0,8):  
        print(i)
```

0,1,2,3,4,5,6,7

```
def superLoop2():  
    for i in range(2,8):  
        print(i)
```

2,3,4,5,6,7



for i in range(x,y) ->

i = x

i = x+1

i = x+2

...

i = y-1

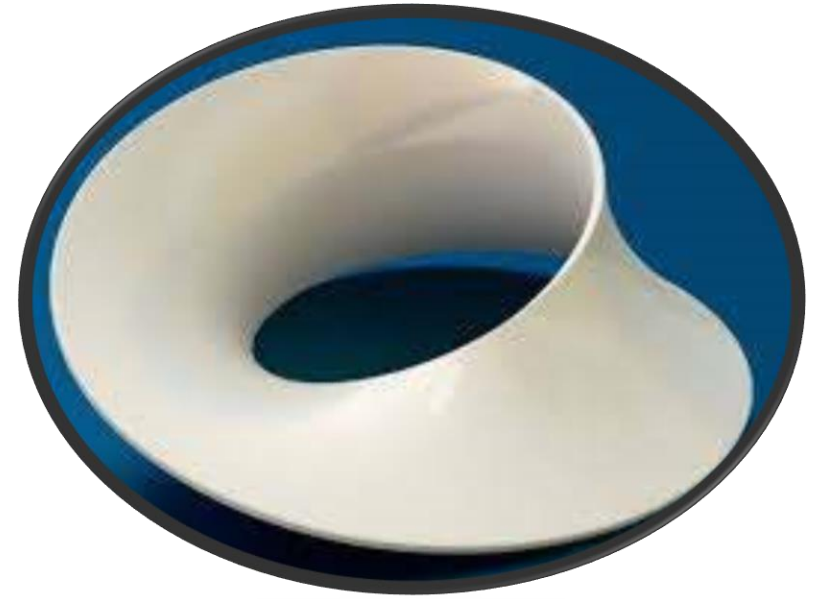


Go to:

<http://presemo.aalto.fi/csa1113>

Loops

```
def superLoop():  
    myNumber = int(input('Say a number.\n')):  
    for i in range(myNumber)  
        print(i)
```



Range in more Detail

```
range(5) = range(0,5,1)
```

```
range(x,y,z)
```

```
x,  
x+z,  
x+2z,  
x+3z,  
...  
x+wz < y
```

We want to print all positive, odd numbers until 100.

```
def superLoop():  
    for i in range(x,y,z):  
        print(i)
```



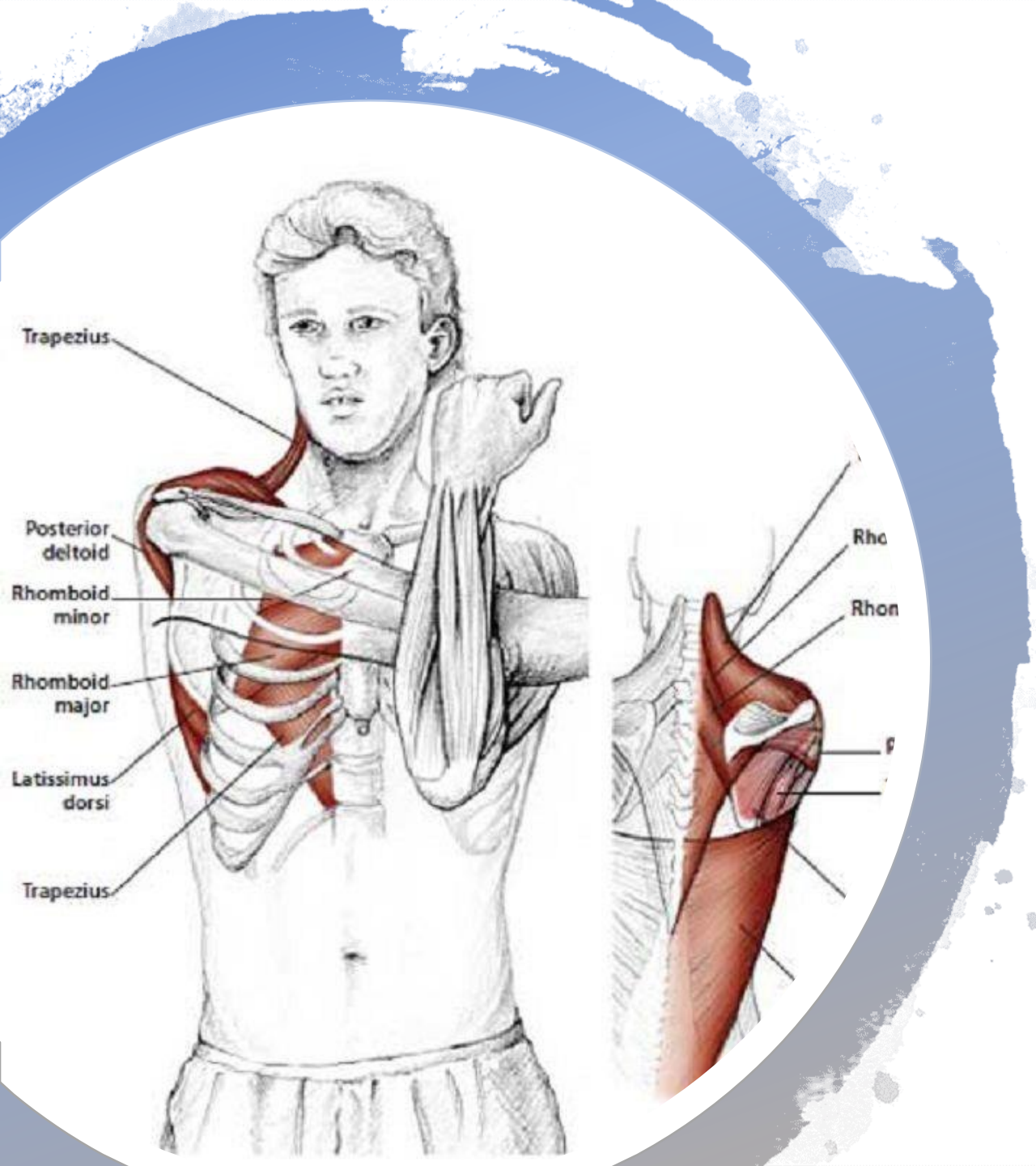
Loops

```
def superLoop():  
    for i in range(x,y,z):  
        print(i)
```

We want all negative Numbers greater than -20 which are divisible by 3 without remainder

We want all numbers in a descending order between 20 and 0 that are divisible by 3 without remainder





Break: Move your
Shoulders

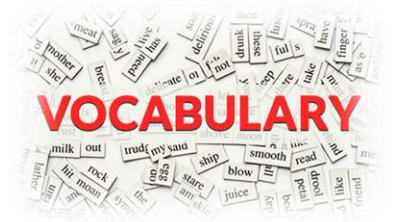
VOCABULARY

and Format ;)



Go to:

<http://presemo.aalto.fi/a1113>



Incomprehensive, Non-Accurate List

Syntax: 'structure' of the program e.g., `for x in range(n):`

Semantic: What the program actually does e.g., loop through 0 to n-1

Algorithm: A way of solving a problem

Program: An instruction for the computer to follow

Input: What is given to a program

Output: What the program returns

String: one or several characters

Integer: a 'normal' number

Float: a number that can have a decimal point

List: a list of elements with an order

Bool: a logical value, True or False

Comment: Something the computer ignores when running a program

IDE: Integrated Development Environment (e.g. PyCharm, Eclipse)

Python File: A file with the endin ".py" should contain python code

Debugging: Correcting the code to do what one wants it to do

```
>>> number = 5 >>> factor = 8 >>> result = number * factor >>>
```

```
(f"{number:3d} times {factor:3d} is {result:6d}") 5 times 8 is 40
```

Format String

Ugly outputs:

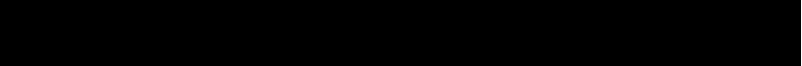
```
print("Good: "+firstname+" earned "+salary+" Euros in "+month+". ")
```




Good: Alex earned 100.126735486649 Euros in September.

Good: Joe earned 10.5 Euros in August.

Help:

```
().format(firstname,salary,month)
```

```
("  ").format(firstname,salary,month)
```

```
("Good:  earned  Euros in  .").format(firstname,salary,month)
```

```
("Good: {:} earned {:} Euros in {:} .").format(firstname,salary,month)
```

Good: Alex earned 100.126735486649 Euros in September.

Format String

Ugly outputs:

Good: Alex earned 100.126735486649 Euros in September.

Good: Joe earned 10.5 Euros in August

Help

```
("Good: {:} earned {:} Euros in {:} .").format(firstname,salary,month)
```

Good: Alex earned 100.126735486649 Euros in September.

```
print(("Good: {:15s} earned {:} Euros in {:} . ").format(firstname,salary,month))
```

Good: Alex earned 100.126735486649 Euros in September.

```
print(("Good: {:15s} earned {:5f} Euros in {:} . ").format(firstname,salary,month))
```

Good: Alex earned 100.126735 Euros in September.

```
print(("Good: {:15s} earned {:5.2f} Euros in {:} . ").format(firstname,salary,month))
```

Good: Alex earned 100.13 Euros in September.

```
("Good: {:15s} earned {:5.2f} Euros in {:9s} .").format(firstname,salary,month)
```

Good: Alex earned 100.13 Euros in September.

Good: Joe earned 10.50 Euros in August .

Alternative f-strings

Since python 3.6 you can use f-strings (slightly less confusing)

```
number = 5
```

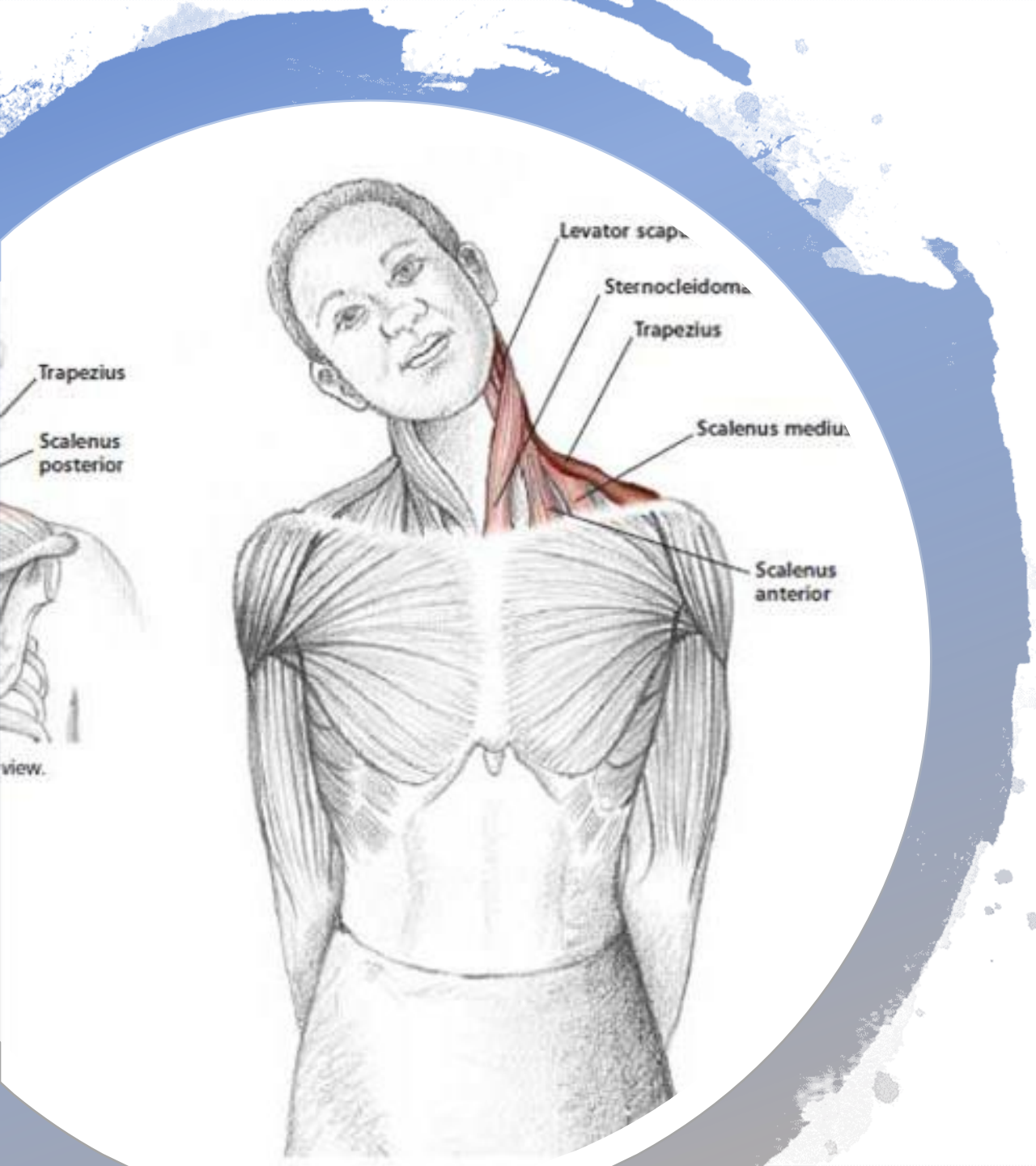
```
factor = 8
```

```
result = number * factor
```

```
print(f"{number:3d} times {factor:3d} is {result:6d}")
```

```
5 times 8 is 40
```

```
print(("{:3d} times {:3d} is {:6d}").format(number, factor, result))
```

Break:
Move your Neck!

ERROR !!!

Coding Style and Debugging 101



We will Look at Your Style on Some Point!



Keep it clean ;)

Naming, Naming, Naming

- variables: use reasonable and self-describing names, not too long
- index variables: i,j,k
- x,y are usually used for axes in a plot
- Structure your program, keep similar things together
- Use variables for values you need several times
- Read a style Guide

Comment your code

What does your code do?

What does it expect as input, which format?

Write your code for someone else

(you will be someone else in a few months ;))

Try not to swear or be inappropriate ;)

Always code as if the person who ends up maintaining your code is a violent psychopath who knows where you live.



TRICKS

Coding Examples

```
def main():  
    a = int(input("Enter your age!\n"))  
    if (a < 18):  
        print("You cannot legally drink in Finland!")  
    else:  
        print("Enjoy your drink!")
```

```
def main():  
    age = int(input("Enter your age!\n"))  
    if (age < 18):  
        print("You cannot legally drink in Finland!")  
    else:  
        print("Enjoy your drink!")
```




Go to:

<http://presemo.aalto.fi/a1113>

Coding Examples

```
def main():
```

```
    a = float(input("Enter number!\n"))
```

```
    b = float(input("Enter number!\n"))
```

```
    c = a*b
```

```
    print("Your result is: "+c)
```

```
# This program calculates the product of two input factors
```

```
def main():
```

```
    factor1 = float(input("Enter your first factor!\n"))
```

```
    factor2 = float(input("Enter your second factor!\n"))
```

```
    product = factor1*factor2
```

```
    print("Your product is: "+product)
```

Coding Examples

```
def main():
```

```
    aFloatNumberToCalculateTheProduct =float(input("Enter a floating point number!\n"))
```

```
    anotherFloatNumberToCalculateTheProduct = float(input("Enter a floating point number!\n"))
```

```
    variableToStoreTheProductOfTheNumbers = aFloatNumberToCalculateTheProduct*
```

```
    anotherFloatNumberToCalculateTheProduct
```

```
    print("Your product from your two floating point numbers is" + variableToStoreTheProductOfTheNumbers)
```

Coding Examples

```
def example1():
    nofYears = 10
    startCapital = 50
    interest = 0.01
    currentCapital = startCapital
    for i in range(nofyears):
        currentCapital += currentCapital*interest
    print("after " + nofYears + " your starting capital of "+startCapital+"
has become "+currentCapital)
```

```
def example2():
    while i<10:
        if i==0:
            money=50
        money += money*0.01
    print("after 10 years your starting capital of 50 has
become "+money)
```




“That’s all Folks!”

lsberg