

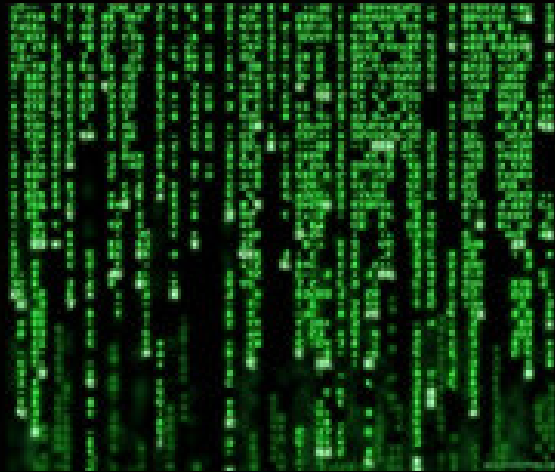
Introduktion till Programmering

Dagens föreläsning

1. **Programmering** – Vad är det egentligen?
2. **Programmeringsspråk** – Python?
3. **Programmeringsmiljö?**
 1. IDLE
 2. Terminalen/Konsolen
4. Att börja **programmera**
 1. Hur skriver man?
 2. Var skriver man?
 3. Hur kör man sin kod?
5. Introduktion till datatyper & variabler i Python

Frågor innan vi börjar?

PROGRAMMING



WHAT PEOPLE THINK I DO



WHAT MY MOTHER THINKS



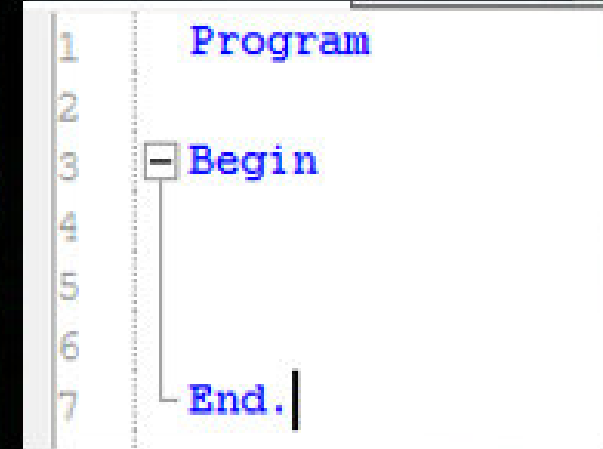
WHAT MY FRIENDS THINK I DO



WHAT MY TEACHER THINKS



WHAT I THINK I DO



WHAT I ACTUALLY DO

Programmering

- Programmering går ut på att ge instruktioner till dator
- Datorn gör det du säger till den, och inget annat
- Datorn behöver specifika instruktioner



STÄDNING PÅ TOALETTERNA

Månad : SEPTEMBER 2016

Datum	Dag	Städning		Städning	
		KL	SIGN	KL	SIGN
1	Torsdag	10.5	RM	14.00	RM
2	Fredag	9:30	BAH		
3	Lördag	13:00	BAH		
4	Söndag				
5	Måndag	9:50	RM	13:40	RM
6	Tisdag	09:30	RM	14:00	RM
7	Onsdag	9:30	RM	14:00	RM
8	Torsdag	9:30	RM	14:00	RM
9	Fredag	10:00	RM	13:30	RM
10	Lördag				

samhall

Var vänlig och ring till Teamledare om det fattas toalettpapper, pappershanddukar och tvål

19	Måndag				
20	Tisdag				
21	Onsdag				
22	Torsdag				
23	Fredag				
24	Lördag				
25	Söndag				
26	Måndag				
27	Tisdag				
28	Onsdag				
29	Torsdag				
30	Fredag				

Var vänlig och ring till Teamledare om det fattas toalettpapper, pappershanddukar och tvål

Exempel på instruktioner

- Matematiska instruktioner (beräkningar)
 - $5 + 5$
 - $10 * 2$
 - $20 / 4$

```
>>> 5 + 5
```

```
10
```

```
>>> 10 * 2
```

```
20
```

```
>>> 20 / 4
```

```
5
```

Exempel på instruktioner

- Instruktioner att skriva ut saker (output)
 - `print ("Hello World!")`
 - `print ("Tjena kexet, sitter du här och smular?")`

```
>>> print("Hello World")
Hello World
>>> print("Tjena kexet, sitter du här och smular?")
Tjena kexet, sitter du här och smular?
```


Exempel på instruktioner

- Instruktion att hämta data från användare (input)
 - `input("Hej, vad heter du?")`
 - `input("Vilket är Sveriges bästa fotbollslag?")`

```
>>> input("Vad heter du?")
Vad heter du?Anton
'Anton'
>>> input("Vilket är Sveriges bästa fotbollslag?")
Vilket är Sveriges bästa fotbollslag?Elfsborg
'Elfsborg'
```

Hur fungerar ett program?

- Program är en (ofta väldigt stor) samling av maskininkod – som innehåller instruktioner till datorn
- Det är väldigt ovanligt att en person skriver maskinkod direkt, då detta inte är så enkelt...

```
00000000 0000 0001 0001 1010 0010 0001 0004 0128
00000010 0000 0016 0000 0028 0000 0010 0000 0020
00000020 0000 0001 0004 0000 0000 0000 0000 0000
00000030 0000 0000 0000 0010 0000 0000 0000 0204
00000040 0004 8384 0084 c7c8 00c8 4748 0048 e8e9
00000050 00e9 6a69 0069 a8a9 00a9 2828 0028 fdfc
00000060 00fc 1819 0019 9898 0098 d9d8 00d8 5857
00000070 0057 7b7a 007a bab9 00b9 3a3c 003c 8888
00000080 8888 8888 8888 8888 288e be88 8888 8888
00000090 3b83 5788 8888 8888 7667 778e 8828 8888
000000a0 d61f 7abd 8818 8888 467c 585f 8814 8188
000000b0 8b06 e8f7 88aa 8388 8b3b 88f3 88bd e988
000000c0 8a18 880c e841 c988 b328 6871 688e 958b
000000d0 a948 5862 5884 7e81 3788 1ab4 5a84 3eec
000000e0 3d86 dcb8 5cbb 8888 8888 8888 8888 8888
000000f0 8888 8888 8888 8888 8888 8888 8888 0000
00001000 0000 0000 0000 0000 0000 0000 0000 0000
```

Lågnivå- och högnivåspråk

- Maskinkod var ju inte så enkelt att förstå - men det gör inget!
- Maskinkod är ett **lågnivåspråk**
 - Binärt och hexadecimalt
- Vi kommer att jobbat med **högnivåspråk**
 - T.ex. Python, JavaScript, C#, C++, Java, PHP, Ruby, etc.

Resultat (3 körningar)

```
# Asks user for their name
name = input("Please enter your name :")
if len(name) < 4:
    # If the users name has less than 4 characters
    print(name + " is a short name")
elif len(name) < 7:
    # If the users name has less than 7 characters (and more than 3)
    print(name + " is a medium long name")
else:
    # If the users name has mote than 6 characters
    print(name + " is a long name")
```

```
>>> ===== RESTART
>>>
Please enter your name: Anton
Anton is a medium long name!
>>> ===== RESTART
>>>
Please enter your name: Eva
Eva is a short name!
>>> ===== RESTART
>>>
Please enter your name: Josefin
Josefin is a long name!
```

```

0000000 0000 0001 0001 1010 0010 0001 0004 0128
0000010 0000 0016 0000 0028 0000 0010 0000 0020
0000020 0000 0001 0004 0000 0000 0000 0000 0000
0000030 0000 0000 0000 0010 0000 0000 0000 0204
0000040 0004 8384 0084 c7c8 00c8 4748 0048 e8e9
0000050 00e9 6a69 0069 a8a9 00a9 2828 0028 fdfc
0000060 00fc 1819 0019 9898 0098 d9d8 00d8 5857
0000070 0057 7b7a 007a bab9 00b9 3a3c 003c 8888
0000080 8888 8888 8888 8888 288e be88 8888 8888
0000090 3b83 5788 8888 8888 7667 778e 8828 8888
00000a0 d61f 7abd 8818 8888 467c 585f 8814 8188
00000b0 8b06 e8f7 88aa 8388 8b3b 88f3 88bd e988
00000c0 8a18 880c e841 c988 b328 6871 688e 958b
00000d0 a948 5862 5884 7e81 3788 1ab4 5a84 3eec
00000e0 3d86 dcb8 5cbb 8888 8888 8888 8888 8888
00000f0 8888 8888 8888 8888 8888 8888 8888 0000
0000100 0000 0000 0000 0000 0000 0000 0000 0000
*
0000130 0000 0000 0000 0000 0000 0000 0000
000013e

```

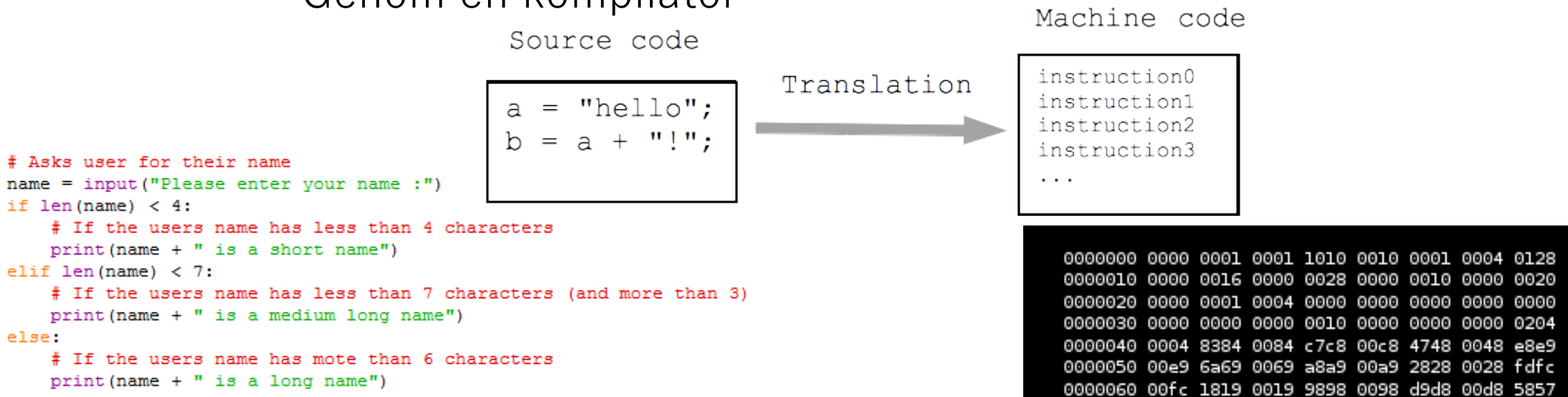
```

# Asks user for their name
name = input("Please enter your name :")
if len(name) < 4:
    # If the users name has less than 4 characters
    print(name + " is a short name")
elif len(name) < 7:
    # If the users name has less than 7 characters (and more than 3)
    print(name + " is a medium long name")
else:
    # If the users name has mote than 6 characters
    print(name + " is a long name")

```

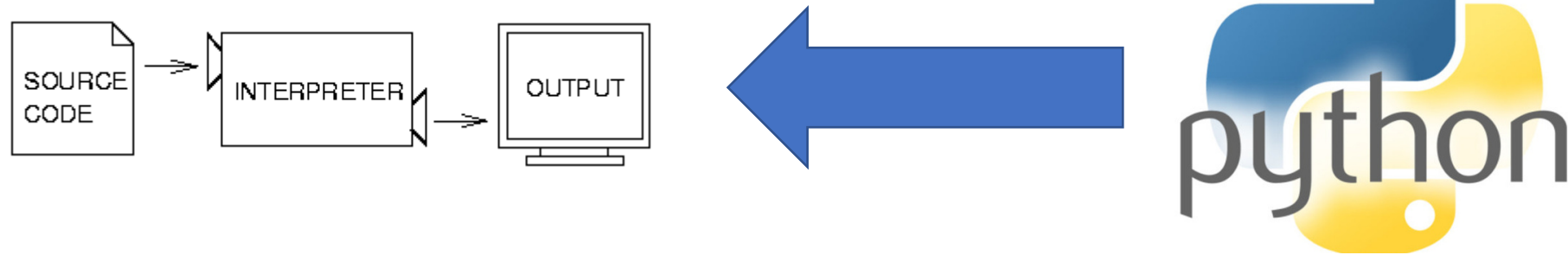
Hur fungerar högnivåspråk?

- Hur kan datorn förstå vår högnivåprogrammering, när den bara förstår binära tal?
 - Vi översätter vår högnivåkod till lågnivå
- Detta kan man göra på två olika sätt:
 - Genom en tolk
 - Genom en kompilator

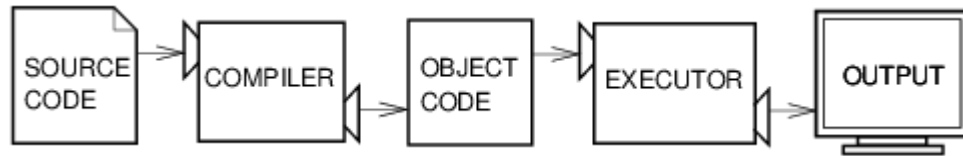


Tolkning och kompilering av kod

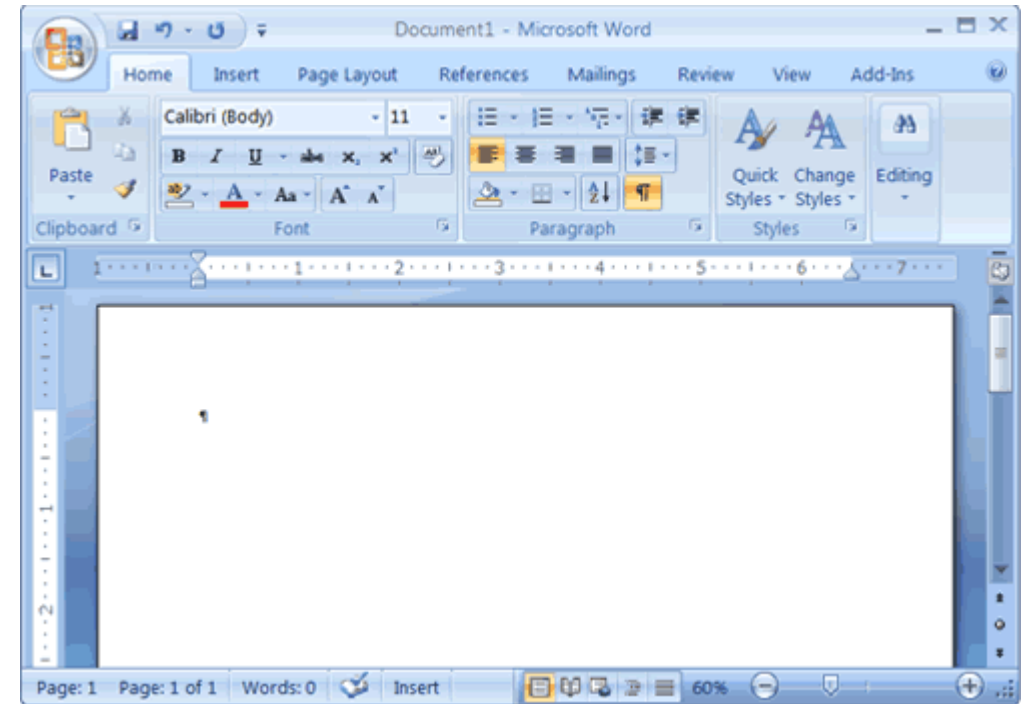
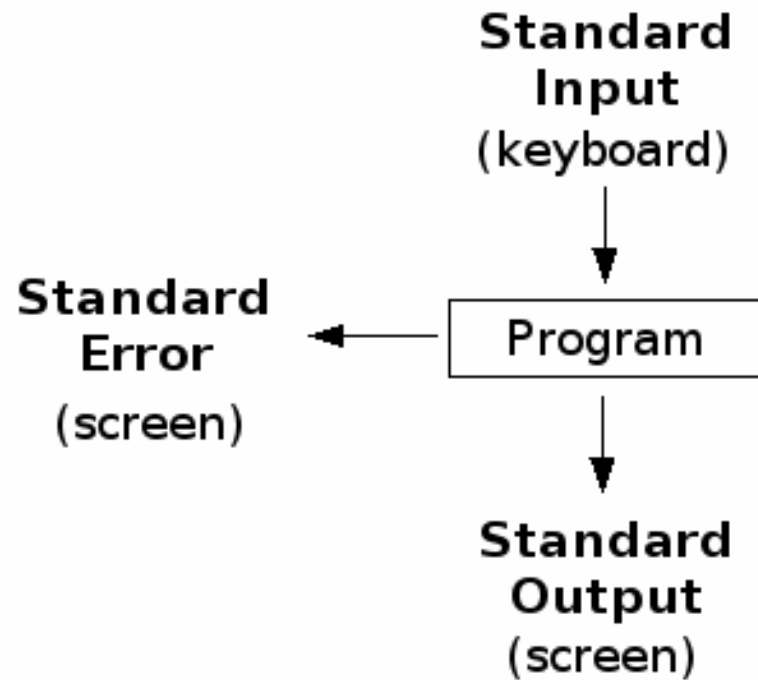
- Tolkning av kod sker "on the fly"



- Kompilering av kod sker "i förväg"



Program?



Vad gör ett program egentligen?

Input	↔	Från användare
Output	↔	Till användare
Beräkningar	↔	Beräkningar
Konditional exekvering	↔	If-satser
Repetition	↔	Iterationer

Error?

Men nä, va, hur?

ONE DOES NOT SIMPLY

**CREATE A PROGRAM THAT RUNS
ON THE FIRST TRY**

Olika typer av fel

- Syntax – Följer inte programmeringsspråket regler (grammatik)
 - `2 + "hej"`
 - `print(hej)`
- Runtime error – Fel under programmets körning

```
Traceback (most recent call last):  
  File "C:/Users/TSANTII/Desktop/name.py", line 1, in <module>  
    print name  
NameError: name 'name' is not defined
```

- Semantiska fel

```
Welcome to the addition calculator!  
-----  
Nr 1: 4  
Nr 2: 5  
Result: 20
```

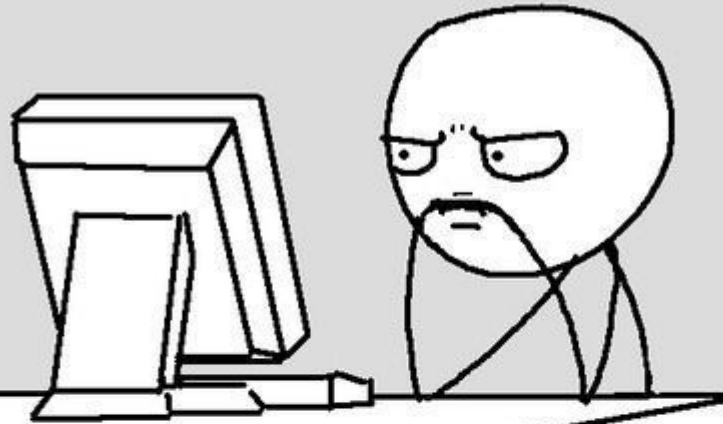
Kärlek till programmering



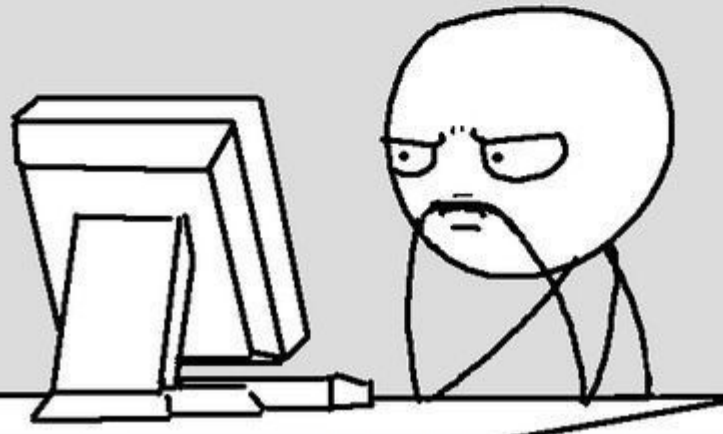
Debugging

- Att lära sig hantera dessa felmeddelande är en programmerares vardag
 - Ju mer ni programmerar, ju bättre kommer ni att bli på detta
- **Kom ihåg – alla gör fel, mest hela tiden!**
- Tänk på att programmeringsspråk är formella språk – och måste vara exakta
 - Till skillnad från naturliga språk som t.ex. svenska och engelska där vi inte behöver vara exakta för att förstå varandra – vi kan till och med använda slang – och förstå varandra! =)

It doesn't work..... why?



It works..... why?



**Varför läser vi
programmering?**

Kan man utforma bra
webbplatser utan att kunna
webbtekniker?

Kan man utforma bra
informationssystem utan att
kunna **programmera**?



Design



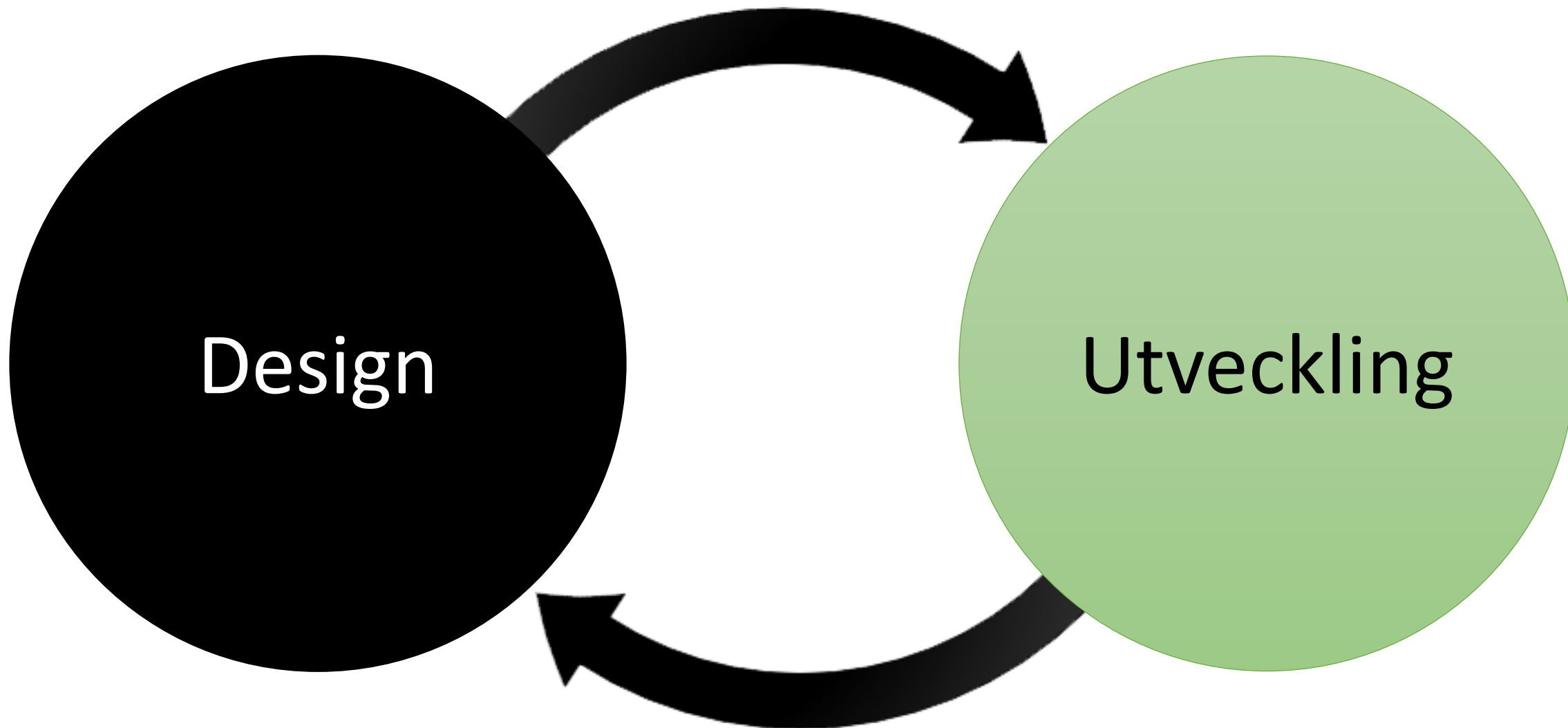
Utveckling

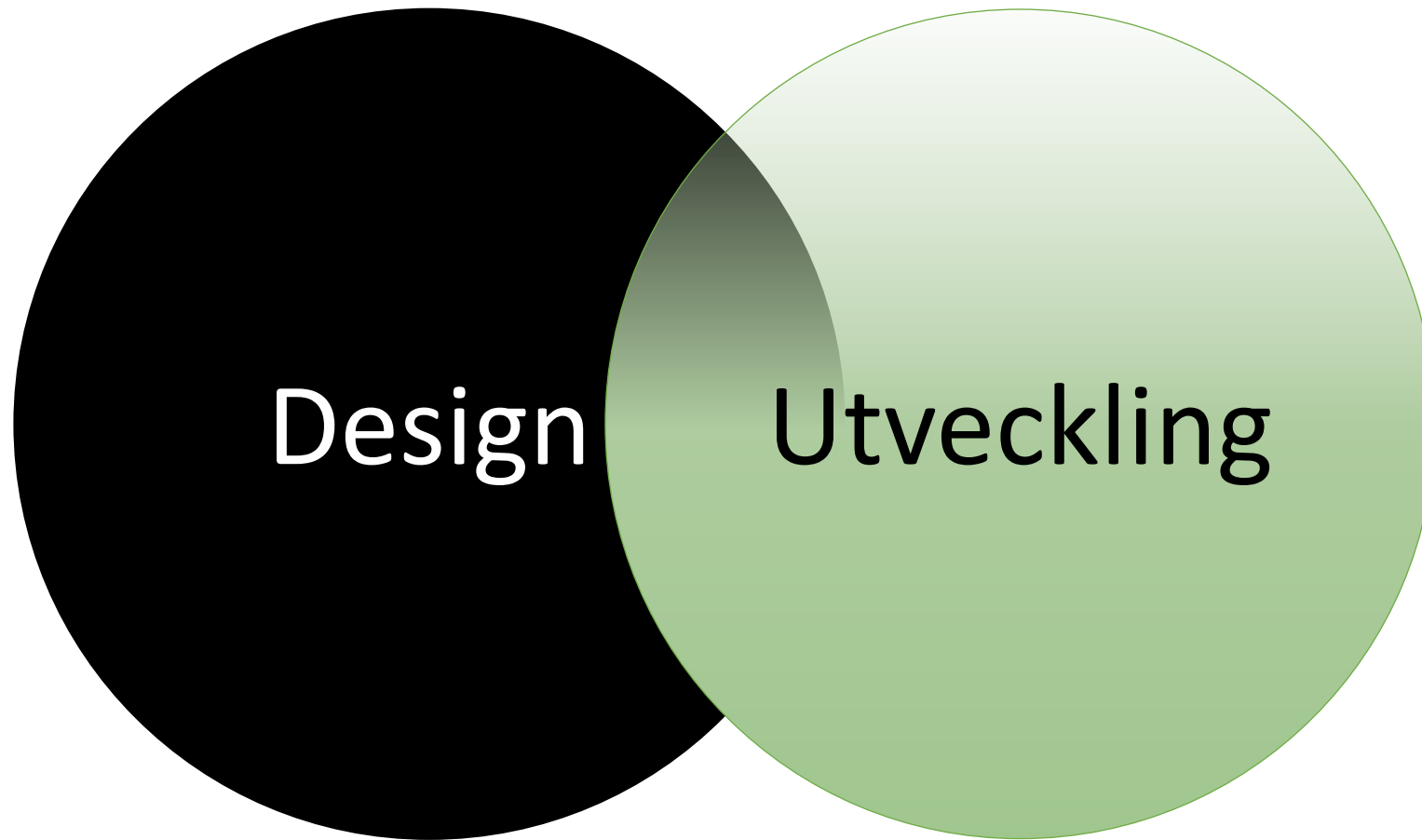


```
graph LR; Design((Design)) --> Utveckling((Utveckling))
```

Design

Utveckling







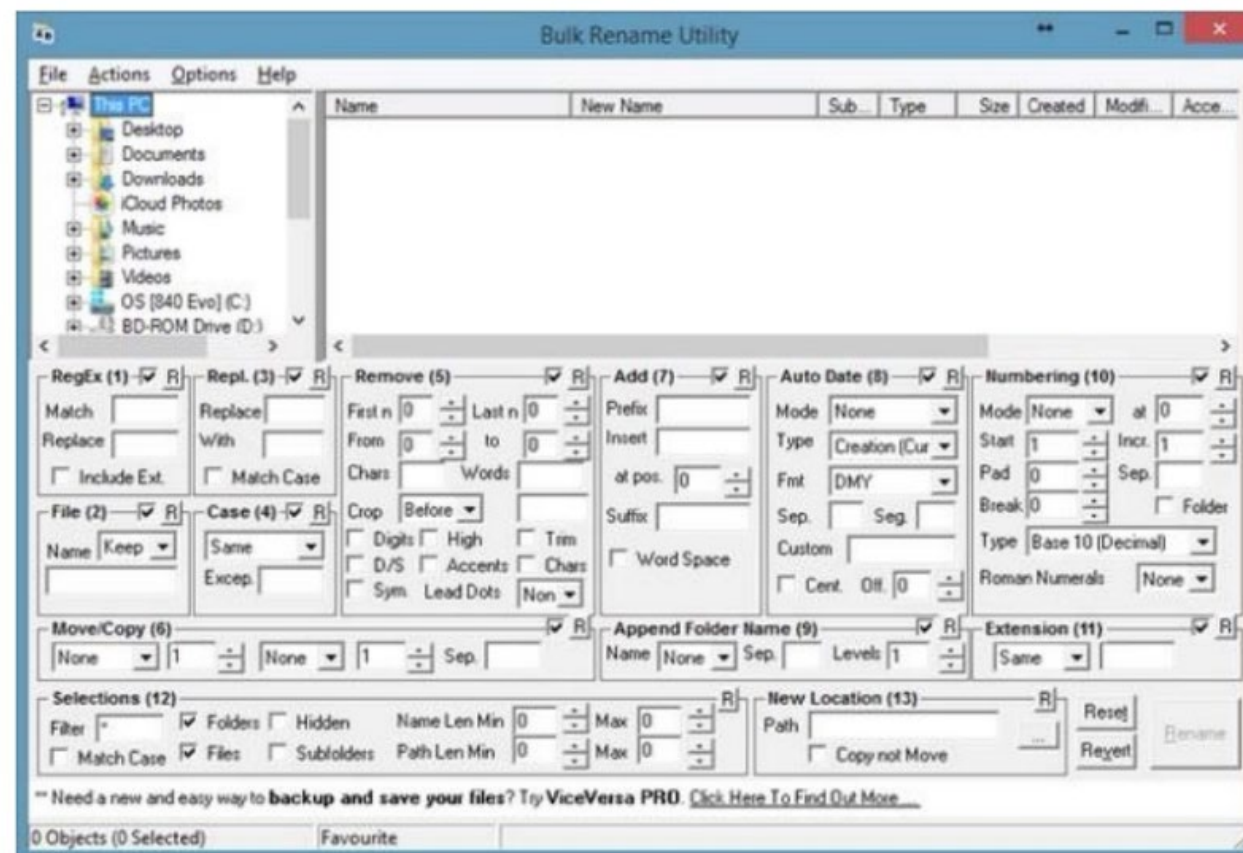
Elias Betinakis

@eliasbetinakis



Following

why designers exist



RETWEETS

46

FAVORITES

26



1:29 PM - 28 Oct 2014

Programmerare är det vanligaste yrket i Stockholm

En tredjedel av Sveriges programmerare är verksamma i Stockholms-området

WEBB / ÖVRIGT

Statistiska Centralbyrån (SCB) har publicerat en rapport som slår fast att jobb inom informationsteknologi är de vanligaste yrkena i Stockholm och var tredje systemerare samt programmerare i Sverige har sin arbetsplats i Stockholms-området.

När det gäller Stockholm så kommer yrkena företagssäljare och försäljare på plats nummer två och tre men det är dock inget som avspeglas i stora delar i resten av landet. I Sverige som helhet är den vanligaste yrkeskategorin "service-, omsorg och försäljningsarbete"

vilken inkluderar undersköterskor, sjukvårdsbiträden, hovmästare, servitörer, barerare och barmästare. För att få reda på vilka yrken som är vanligast i Stockholm kan man se rapporten "Yrkeskategorier och yrkeskategorier i Stockholm 2014".
<http://feber.se/webb/art/203640/>
[programmerare är det vanligaste](http://feber.se/webb/art/203640/)

IBM satsar i Malmö – 300 ska anställas

ARBETE TOPPDELAD IBM tänker starta ett nytt utvecklingscentrum i Malmö – och anställa minst 300 personer.



DELA

1092



TWEETA

10



MEJLA



Text: TT



Text: Marcus Svensson



Publicerad 25 augusti 2015 22.06 · Uppdaterad 26 augusti 2015 00.26

Textstorlek: + -

– Vi har utvärderat flera orter i Sverige men valde Malmö eftersom staden har en extrem mångfald, både etniskt och åldersmässigt med många unga, och att det i regionen finns närhet till universitet och högskolor, säger Johan Forssander, chef för



Kod är makt och vi är framtidens analfabeter

Publicerad 22 okt 2014 06:10



Den stereotypa 50-årige universitetsutbildade mannen har fått konkurrens i toppen. I det nya digitala samhället är kod det nya maktspråket, och borde ingå i läroplanen, skriver Siduri Poli.



Rekommendera

506



Tweeta

Vi lever i ett digitalt samhälle men ändå är det få som förstår språket det är uppbyggt på - kod. Min generation är fullt upptagen med att hinna med den digitala utvecklingen, samtidigt som arbetsmarknaden skriker efter kodningskompetens som har lett till att skolor inför kod i schemat. Men om alla vill ha, eller snart kommer att kunna, kod, vad innebär det för oss som inte kan det?

Det håller på att ske en maktförskjutning, från de med mest erfarenhet till de som precis har lärt sig det allra senaste. Den stereotypa 50-årige universitetsutbildade mannen har fått konkurrens i toppen.

<http://www.expressen.se/debatt/kod-ar-makt-och-vi-ar-framtidens-analfabeter/>

Programmeringen kommer 2016 - är skolorna redo?

Publicerad 03.09.2014 - 07:35. Uppdaterad 04.09.2014 - 10:37

Om två år ska alla barnen i hela Finland lära sig datorprogrammering från första klass. Undervisningsministeriet, skolbarnen och näringslivet jublar, medan lärarkåren fortfarande har flera frågor än svar.

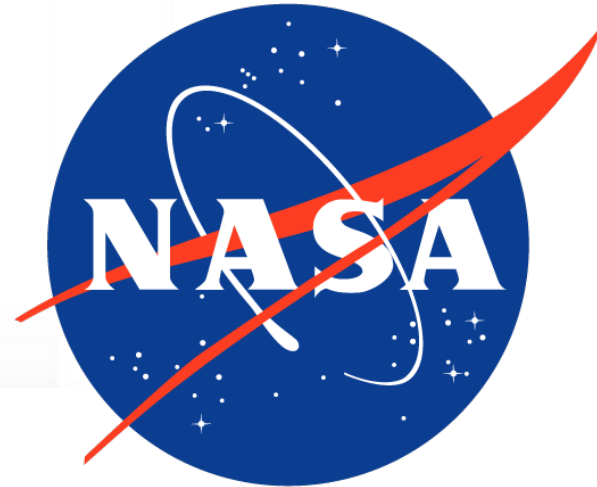
En glad katt jamar på datorskärmen och Wilmer Filéns ansikte lyser upp. Klass 5 B i Malms skola i Pargas har besök av Linda Mannila, forskare inom datavetenskapens didaktik och utbildningsteknologi vid Åbo Akademi, som vill testa hur man i framtiden kunde arbeta med programmering i våra skolor.

Hon är glad över att Finland kommer att ta in datorprogrammering som en del av

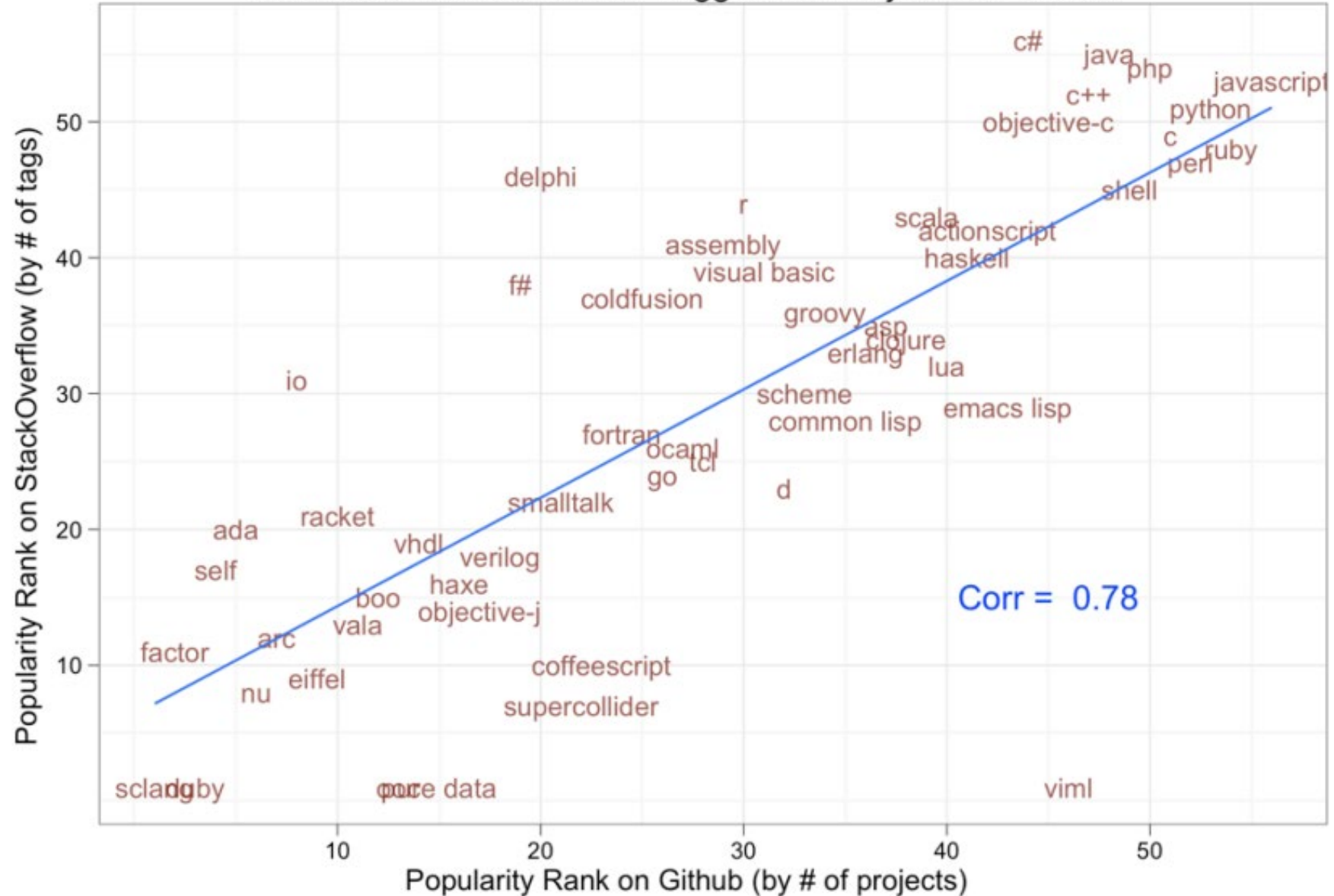
<http://svenska.yle.fi/artikel/2014/09/03/programmeringen-grundkunskap,menar-hon:kommer-2016-ar-skolorna-redo>

Google

YAHOO!



Programming Language Popularity
StackOverflow Questions Tagged vs. Projects on Github





Malmö högskola

SIGN IN



COMMUNICATIONS

OF THE

ACM

Search



HOME

CURRENT ISSUE

NEWS

BLOGS

OPINION

RESEARCH

PRACTICE

CAREERS

MAGAZINE ARCHIVE

[Home](#) / [Blogs](#) / [BLOG@CACM](#) / [Python is Now the Most Popular Introductory Teaching...](#) / [Full Text](#)

BLOG@CACM

Python is Now the Most Popular Introductory Teaching Language at Top U.S. Universities

By Philip Guo

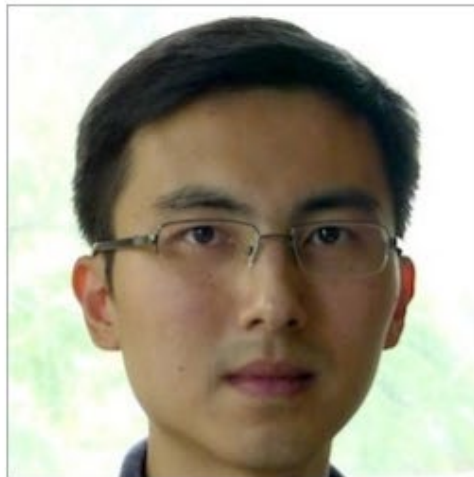
July 7, 2014

[Comments \(7\)](#)

VIEW AS:



SHARE:



Summary

At the time of writing (July 2014), [Python](#) is currently the most popular language for teaching introductory computer science courses at top-ranked U.S. departments.

Specifically, eight of the top 10 CS departments (80%), and 27 of the top 39 (69%), teach Python in introductory CS0 or CS1 courses.

Motivation

Python has been getting more popular as the first language to teach novices. Three years ago, [Mark Guzdial blogged](#) about the rise of Python as a teaching language and predictions for future teaching languages. Top-ranked CS departments at MIT and UC Berkeley recently

SIGN IN for Full Access

User Name

Password

[» Forgot Password?](#)[» Create an ACM Web Account](#)

SIGN IN

MORE NEWS & OPINIONS

[Virtual Earth Plays Out Fate of Life on the Planet](#)

New Scientist

[Python is Now the Most Popular Introductory Teaching Language at Top U.S. Universities](#)

Philip Guo

Programmera program

Att skapa ett program

- Uppgiftsformulering, vad är det för uppgift som ska lösas?
- Vilka steg behöver utföras för att lösa uppgiften?
- Vilka instruktioner kan användas för att utföra varje delsteg (algoritm)



Att leka dator
– Sortera böcker

Algoritmer

- En beskrivning över hur man löser ett problem. Algoritmen består av ett antal instruktioner och beskriver i vilken ordning instruktionerna ska utföras.

PASTA CARBONARA SKRIV UT

Gott och klassiskt recept på pasta carbonara, men på ett lite enklare sätt.

Tid: 25 minuter

4 portioner

INGREDIENSER:

- 300 g spaghetti
- 150 g bacon
- 1 st gul lök
- ev. 1 klyfta vitlök
- 2 st ägg
- 0.5 dl grädde
- 100 g riven parmesanost
- salt
- grovmalen svartpeppar

GÖR SÅ HÄR:

Koka spaghetten enligt förpackningens anvisningar, men med 1 tsk olivolja i vattnet.

Skär bacon i små bitar. Hacka den gula löken fint. Om du vill ha vitlök, hacka vitlöksklyftan fint.

Hetta upp en stekpanna på medelhög värme och stek bacon och lök var för sig tills baconet blir knaperstekt och löken genomstekt. Sänk ev. temperaturen på plattan lite så att det inte blir bränt.

Vispa under tiden ihop ägg, grädde och riven ost. Använd en vanlig visp och vispa bara lätt upp äggen. Smaksätt med salt.

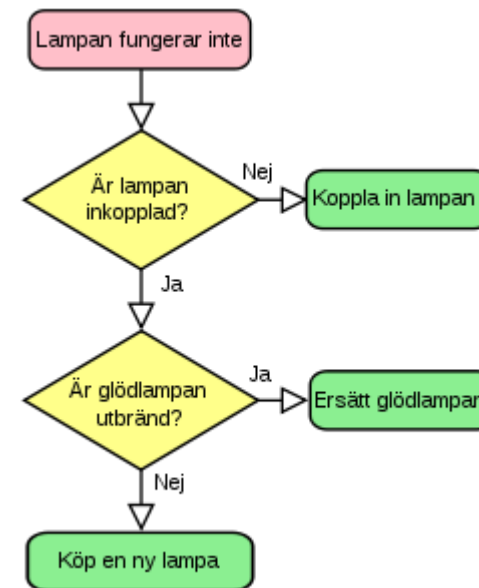
Häll av vattnet från den färdigkokta pastan och blanda med knaperstekt bacon och lök. Blanda sedan ihop med äggblandningen. Rör kraftigt med en sked. Servera på tallrik och strö över grovmalen svartpeppar och gärna lite persilja.



Spaghetti carbonara är väldigt gott. Det här är ett enkelt (utan krångel med äggulor) och mycket gott recept på denna goda rätt.

Dela: [facebook](#) [twitter](#) [Mejla](#)

http://www.recepten.se/recept/pasta_carbonara.html



<http://upload.wikimedia.org/wikipedia/commons/thumb/3/3c/LampFlowchart-sv.svg/250px-LampFlowchart-sv.svg.png>

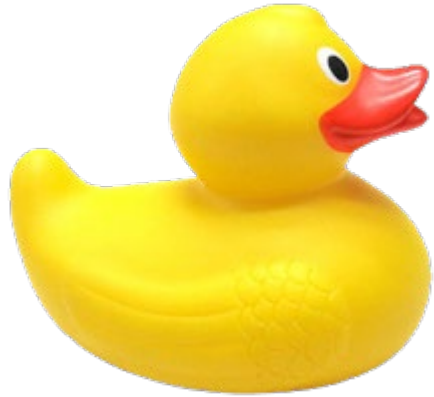
Att tänka på (1)

- **Uppgiftsformulering**, vad är det för uppgift som ska lösas? Formulera uppgiften i termer av vad en dator kan utföra. Avgränsa problemet, vad är en del av uppgiften? Vad ingår inte?
- **Algoritmkonstruktion**, vilka algoritmer är de mest lämpliga för detta problem? Konstruera strukturen på programmet och skriv ner så kallad pseudokod. Detta är kreativ problemlösning.
- **Kodning**, översätt pseudokoden till ett programmeringsspråk t.ex. JavaScript eller Python
- **Dokumentation**, beskriva din lösning både i löpande text, med hjälp av UML och som kommentarer i programmet.

Att tänka på (2)

- **Verifikation**, är programmet byggt på ett bra sätt så att det löser uppgiften utan att fel uppstår och det är lätt att underhålla.
- **Validering**, är användaren nöjd med hur programmet fungerar. Underhåll, åtgärda buggar, förbättra och lägg till funktionalitet.
- **Underhåll**, åtgärda buggar, förbättra och lägg till funktionalitet.

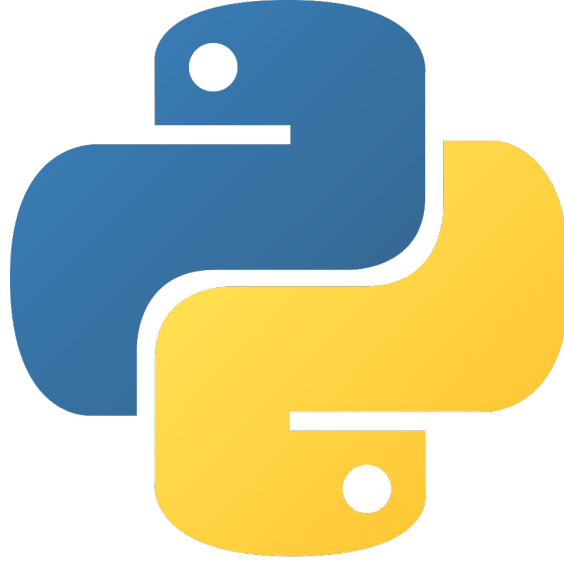
Johans tips



Johans tips



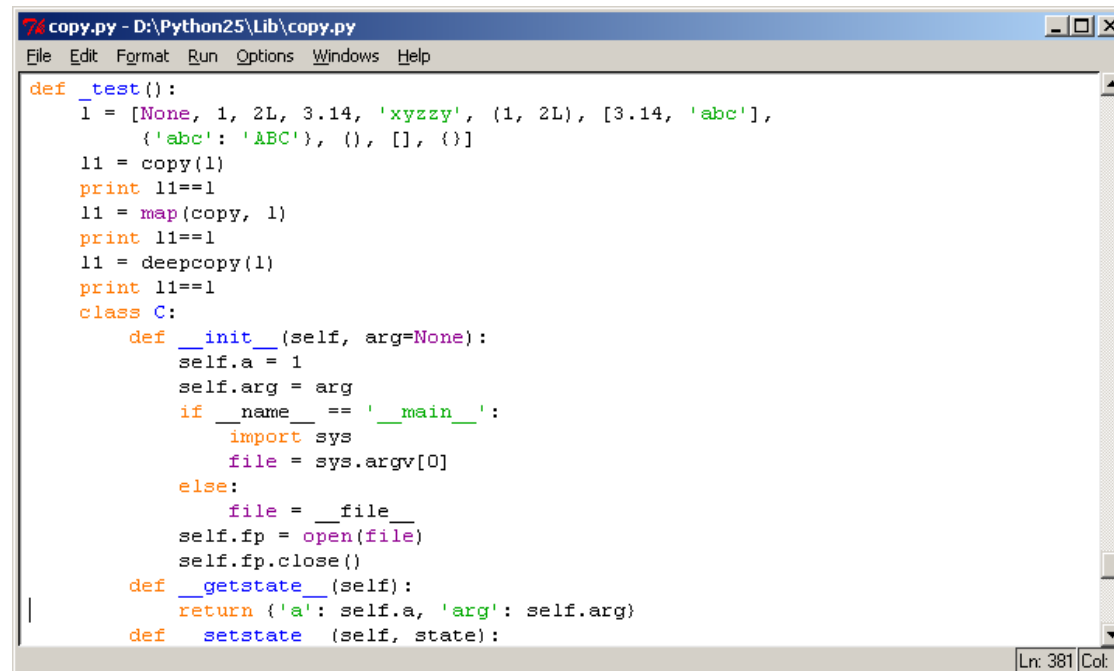
Tillbaka till Python



"Python är ett programspråk som utformades i slutet av 1980-talet av Guido van Rossum. Python har utvecklats till ett kraftfullt och samtidigt smidigt språk med stöd för flera olika programmeringsparadigmer, bland annat objektorienterad och funktionell programmering. Till språket hör ett rikt standardbibliotek."

Python - utvecklingsmiljö

- Vi kommer att arbeta i Pythons egna utvecklingsmiljö **IDLE**
 - Denna installeras automatiskt när man installerar Python
- Man kan använda valfri utvecklingsmiljö, t.ex. egna textredigerare
 - Men då får man exekvera sin kod från terminalen/konsolen istället



```
copy.py - D:\Python25\Lib\copy.py
File Edit Format Run Options Windows Help

def _test():
    l = [None, 1, 2L, 3.14, 'xyzzzy', (1, 2L), [3.14, 'abc'],
         {'abc': 'ABC'}, (), [], {}]
    l1 = copy(l)
    print l1==l
    l1 = map(copy, l)
    print l1==l
    l1 = deepcopy(l)
    print l1==l
    class C:
        def __init__(self, arg=None):
            self.a = 1
            self.arg = arg
            if __name__ == '__main__':
                import sys
                file = sys.argv[0]
            else:
                file = __file__
            self.fp = open(file)
            self.fp.close()
        def __getstate__(self):
            return {'a': self.a, 'arg': self.arg}
        def __setstate__(self, state):
            self.a, self.arg = state

if __name__ == '__main__':
    import sys
    file = sys.argv[0]
    c = C(file)
    c.__setstate__([1, file])
    print c.__getstate__()
```

Ln: 381 | Col: 0

Datatyper?

- För att Python ska veta vad för typ av data som hanteras behöver vi delge denna information till Python
- De olika datatyperna som vi kommer att titta på är:
 - **Number (int, float)**
 - **String**
 - Boolean
 - List
 - Tuple
 - Dictionary

Räkna med Python

- Python är väldigt bra på att räkna, vilket vi kan göra genom vanliga matematiska uttryck.
- När vi räknar och gör matematiska operationer använder vi oss utav datatypen **number** (numerisk datatyp). Här kan vi räkna med heltal.
- Vill vi istället räkna med decimaler använder vi datatypen **float**

```
>>> 5 + 5
10
>>> 10 * 2
20
>>> 20 / 4
5
```

Operander i Python

Namn	Operand	Beskrivning
Addition	+	Beräknar summan av två tal
Subtraktion	-	Beräknar differensen mellan två tal
Multiplikation	*	Beräknar produkten av två tal
Division	/	Beräknar kvoten av två tal
Heltalsdivision	//	Beräknar heltalskvoten av två tal
Rest	%	Beräknar resten efter en division av två heltal
Exponent	**	(x ** y) Beräknar x upphöjt till y <code>x^y</code>

Hantera text med Python

- Vill vi skriva ut text i Python så använder vi datatypen **string** (textsträng). Tänk på att textsträngar är just bara text – d.v.s. tecken uppradade efter varandra.
 - Vi kan därför t.ex. inte räkna med strängar
- Strängar skrivs m.h.a. ' eller "

```
>>> print("Hello World")
Hello World
>>> print("Tjena kexet, sitter du här och smular?")
Tjena kexet, sitter du här och smular?
```

```
>>> input("Vad heter du?")
Vad heter du?Anton
'Anton'
>>> input("Vilket är Sveriges bästa fotbollslag?")
Vilket är Sveriges bästa fotbollslag?Elfsborg
'Elfsborg'
```

Lägga ihop strängar med varandra

- Detta vill man ofta göra för att bygga upp meddelande till användaren av ens program. T.ex.

```
name = input("Vad heter du? ")  
print("Välkommen " + name)
```

```
Vad heter du? Anton  
Välkommen Anton
```

- Vi tar alltså de strängarna som står på varje sida av "+" och sammanfogar dessa.
- Skulle vi försöka med räkna mer strängar får vi alltså inte önskat resultat. T.ex.

```
>>> "5" + "7"  
'57'
```

```
>>> "5" * 4  
'5555'
```

Variabler

Spara information

Variabler

- När man bygger ett program vill man gärna spara undan värde, så att vi kan använda dem vid ett senare tillfälle.
 - Hur kul vore det om man glömde allt – hela tiden?
- Detta gör man genom att använda **variabler**
 - Variabler är ett namn som refererar till ett värde (kan vara av vilken datatyp som helst)
- Man skapar en variabel genom att
 1. Välja ett lämpligt namn
 2. Tilldela ett lämpligt värde genom "="-tecknet

Ange datatyper?

Python listar ut detta beroende värde

Kommentarer

- Det är en god idé att dokumentera sin kod
 - Man glömmer bort vad koden gör
 - Man glömmer bort hur koden gör det
 - Man glömmer bort varför koden gör det
- Man kommenterar sin kod genom #

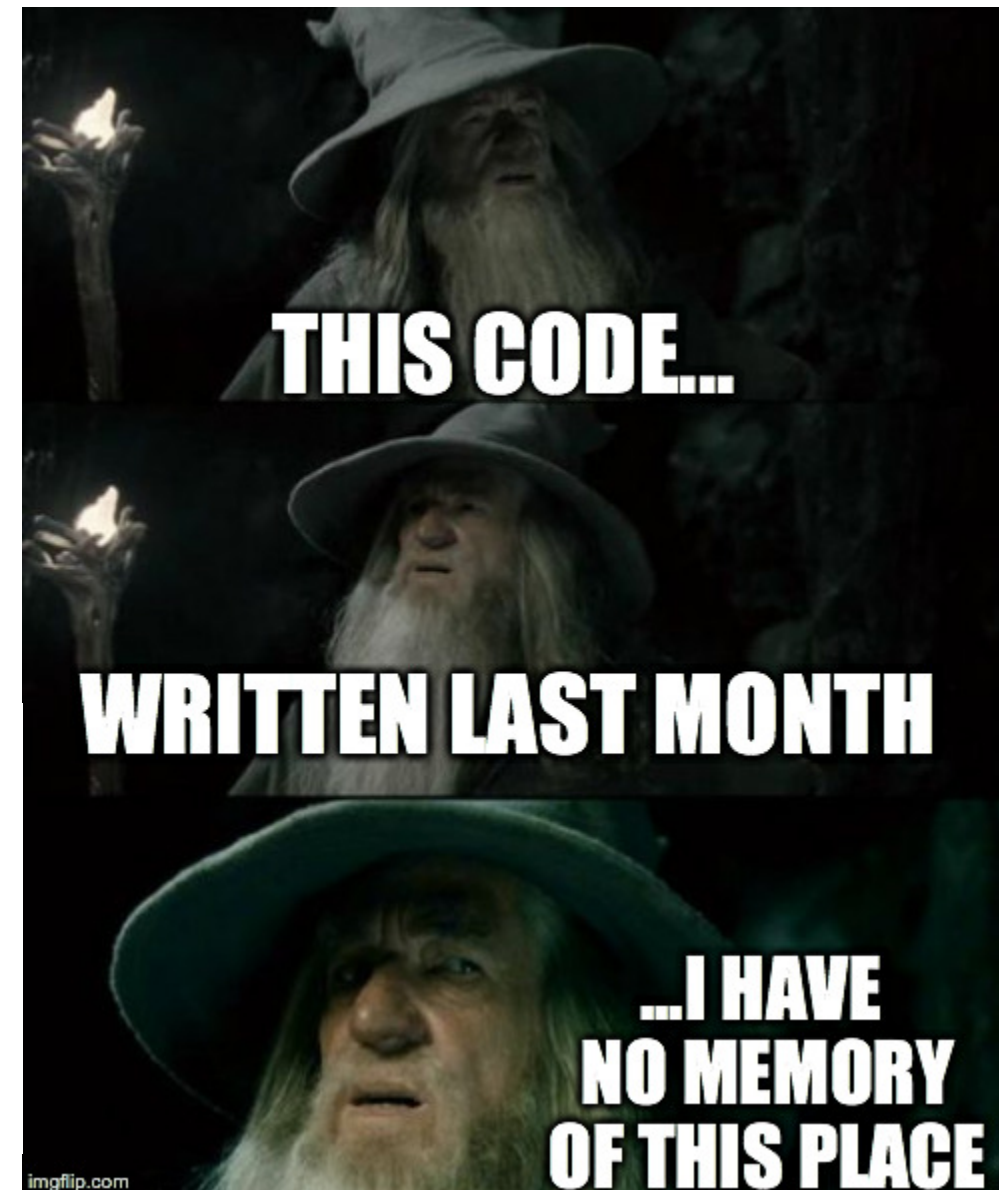
```
# Denna rad kommer inte visas, utan finns bara till för oss

# 1) Frågar efter användarens namn, och sparar det i variabeln "name"
name = input("Vad heter du? ")

# 2) Frågar efter användarens ålder, och sparar det i variabeln "age"
age = input("Hur gammal är du? ")

# 3) Skriver ut ett välkomnstmmeddelande
print("Hej " + name + " (" + age + "år)!")
```

```
Vad heter du? Anton
Hur gammal är du? 26
Hej Anton (26år)!
```



IS IT MAGIC?



DEMO TIME