

Automation and programming

Ádám T. Kocsis (adam.kocsis@fau.de)

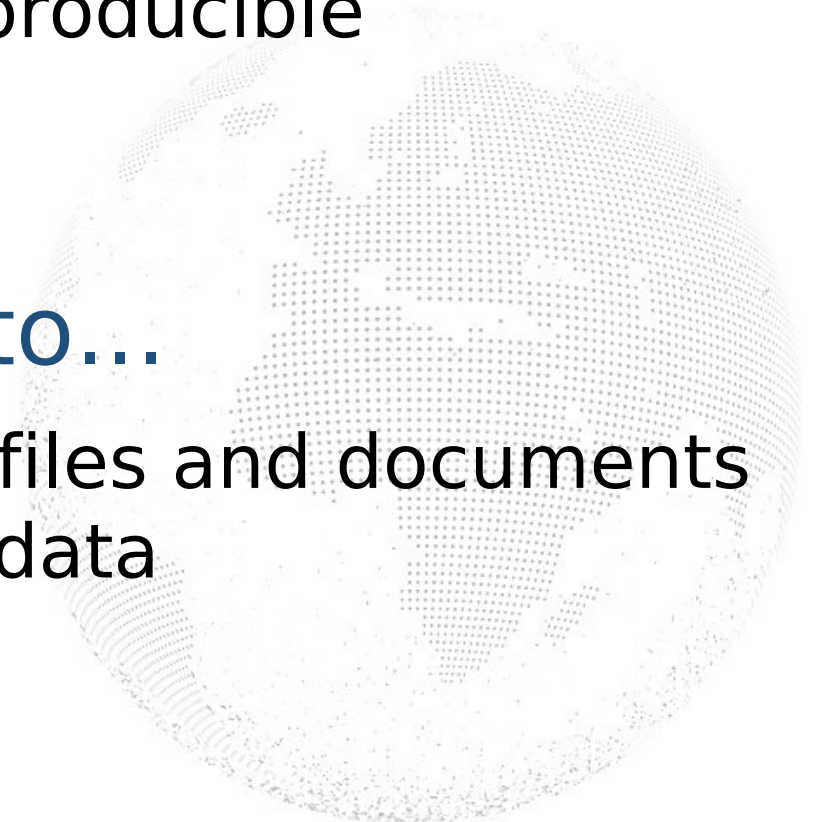


WHY? We want to ...

1. ... avoid tedious manual labor (lazy)
2. ... make sure that we work correctly
3. ... be efficient: work faster, with less energy
4. ... make our work reproducible

which applies both to...

- managing information, files and documents
- calculations, analyzing data



Console applications

UNIX philosophy: **Do one thing, but do it very well!**

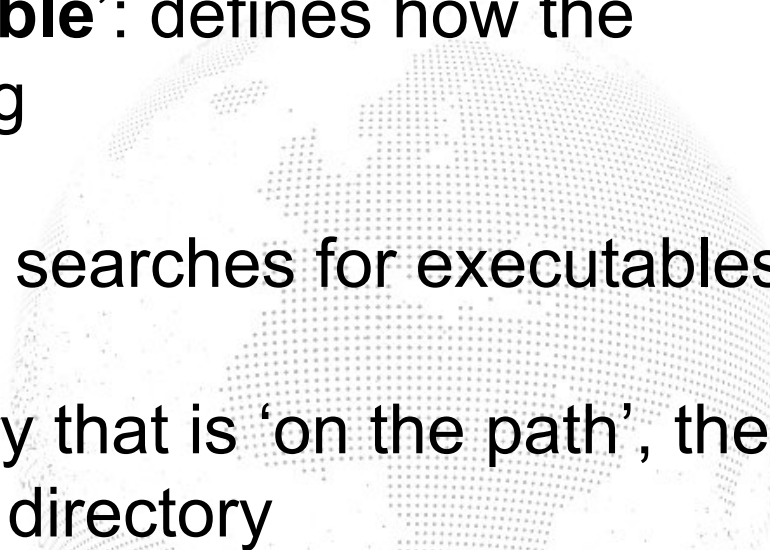
One application = one executable file = one command!

1st example: GNU ***wget***: download the targets of URLs

2nd example: ***ImageMagick*** – image manipulation



The PATH variable

- **Variable**: a named box to put information into, so you can refer to it by its name
 - This is an '**Environment variable**': defines how the working environment is working
 - List of directories that the shell searches for executables
 - If an executable is in a directory that is 'on the path', then it can be run from any working directory
- 

WHY? Exercise

1. Go to
2. Download 'first.txt' file from the webpage!
In the file there are URLs to images.
4. Download every image!

First person to finish
gets a bar of
chocolate!



Interested? Recommendation:

The screenshot shows the Coursera website interface for the course 'The Unix Workbench'. The browser address bar shows the URL <https://www.coursera.org/learn/unix>. The Coursera logo and navigation links are at the top. The course title 'The Unix Workbench' is prominently displayed, along with its rating of 4.7 stars from 1,288 ratings and a 96% approval rate. It is offered by Johns Hopkins University. A blue button 'Enroll for Free' is visible, with a note 'Starts Oct 11' and 'Financial aid available'. Below this, it states '51,516 already enrolled'. The 'About this Course' section provides a detailed description of the course's focus on Unix as a foundation for various goals. A 'SHOW ALL' link is present. A 'SKILLS YOU WILL GAIN' section lists four skills: Shell Script, Github, Bash (Unix Shell), and Cloud Computing. On the right, four key features are highlighted: 'Flexible deadlines', 'Shareable Certificate', '100% online', and 'Beginner Level'.

← → ↺ <https://www.coursera.org/learn/unix> ⌵ ☆

coursera Explore ▾ What do you want to learn? 🔍 Online Degrees ▾ Find your New Career For Enterprise For Universities Log In Join for Free

Browse > Data Science > Machine Learning

The Unix Workbench

★★★★★ 4.7 1,288 ratings | 👍 96%

Sean Kross [+3 more Instructors](#)

Enroll for Free
Starts Oct 11

Financial aid available

51,516 already enrolled

About Instructors Syllabus Reviews Enrollment Options FAQ

About this Course

50,318 recent views

Unix forms a foundation that is often very helpful for accomplishing other goals you might have for you and your computer, whether that goal is running a business, writing a book, curing disease, or creating the next great app. The means to these goals are sometimes carried out by writing software. Software can't be mined out of the ground, nor can software seeds be planted in spring to harvest by autumn. Software isn't produced in factories on an assembly line. Software is a hand-made, often bespoke good. If a software developer is an artisan, then Unix is their workbench. Unix provides an essential and simple set of tools in a distraction-free environment. Even if you're not a software developer

[SHOW ALL](#)

SKILLS YOU WILL GAIN

Shell Script Github Bash (Unix Shell) Cloud Computing

Flexible deadlines
Reset deadlines in accordance to your schedule.

Shareable Certificate
Earn a Certificate upon completion

100% online
Start instantly and learn at your own schedule.

Beginner Level

<https://www.coursera.org/learn/unix>

Instructions?

- Statements that follow each other
- Every statement does something to change the state of the computer
- Linear sequence
- How can a computer understand what we are telling it?
- Multiple levels, exact instructions combine them

Pancake Recipe

- 100g plain flour
- 2 eggs
- 300ml milk
- 1 tbsp oil
- pinch of salt



1. Put the flour and milk into a bowl.
2. Crack the eggs and add to the bowl.
3. Whisk the ingredients together.
4. Pour some of the mixture into the pan.
5. Cook until browned then flip.
6. Once the other side is brown leave to cool.
7. Enjoy eating.

Programming, again...

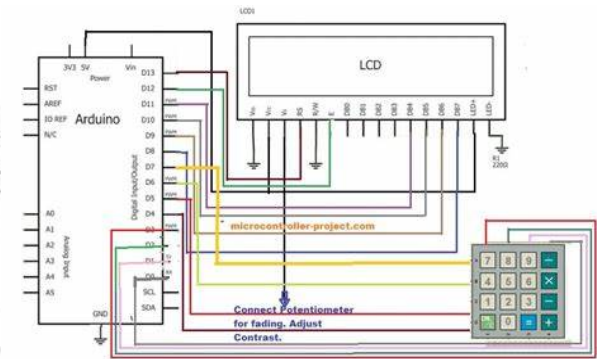


- The concept of calculation: how much is $651/7$?

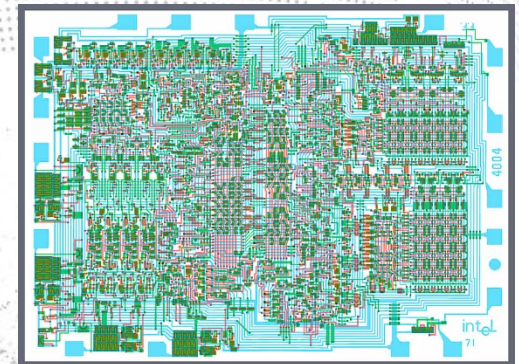
You have 651 balls.

1. You go through them one-by one.
2. You put every 7th ball in a bin.
3. After done, count the balls. (divisor)

- You can do this with electricity



- You are using a machine to define a machine that calculates numbers that represent something else (programmable computer)



What kind of languages are there?

General purpose vs. **specialized** (e.g. domain-specific) language



What kind of languages are there?

Different **levels** of programming

MONITOR FOR 6802 1.4 9-14-80 TSC ASSEMBLER PAGE 2

```
C000          ORG      RCM+$0000 BEGIN MONITOR
C000 8E 00 70  START  LDS      #STACK

*****
* FUNCTION: INITA - Initialize ACIA
* INPUT: none
* OUTPUT: none
* CALLS: none
* DESTROYS: acc A

0013          RESETA EQU    $00010011
0011          CTLREG EQU    $00010001

C003 86 13      INITA  LDA A  #RESETA  RESET ACIA
C005 B7 80 04      STA A  ACIA
C008 86 11      LDA A  #CTLREG  SET 8 BITS AND 2 STOP
C00A B7 80 04      STA A  ACIA

C00D 7E C0 F1      JMP      SIGNON  GO TO START OF MONITOR

*****
* FUNCTION: INCH - Input character
* INPUT: none
* OUTPUT: char in acc A
* DESTROYS: acc A
* CALLS: none
* DESCRIPTION: Gets 1 character from terminal

C010 B6 80 04  INCH   LDA A  ACIA      GET STATUS
C013 47          ASR A          SHIFT RDRF FLAG INTO CARRY
C014 24 FA      BCC   INCH      RECIEVE NOT READY
C016 B6 80 05      LDA A  ACIA+1  GET CHAR
C019 84 7F      AND A  #$7F      MASK PARITY
C01B 7E C0 79      JMP      OUTCH   ECHO & RTS

*****
* FUNCTION: INHEX - INPUT HEX DIGIT
* INPUT: none
* OUTPUT: Digit in acc A
* CALLS: INCH
* DESTROYS: acc A
* Returns to monitor if not HEX input

C01E 8D F0      INHEX  BSR   INCH      GET A CHAR
C020 81 30      CMP A  #'0          ZERO
C022 2B 11      BMI   HEXERR        NOT HEX
C024 81 39      CMP A  #'9          NINE
C026 2F 0A      BLE   HEXRTS        GOOD HEX
C028 81 41      CMP A  #'A          NOT HEX
C02A 2B 09      BMI   HEXERR
C02C 81 46      CMP A  #'F          NOT HEX
C02E 2E 05      BGT   HEXERR
C030 80 07      SUB A  #7          FIX A-F
C032 84 0F      HEXRTS AND A  #$0F  CONVERT ASCII TO DIGIT
C034 39          RTS

C035 7E C0 AF  HEXERR JMP      CTRL   RETURN TO CONTROL LOOP
```

```
INPUT CELSIUS_TEMP
SET FAHRENHEIT_TEMP TO CELSIUS_TEMP * 9/5 + 32
WRITE FAHRENHEIT_TEMP
```



```
makeEggs()
getEggs()
stirEggs()
cook()
serve()
```

High-Level Languages
(Java, PHP, Python, etc.)

Assembly
Language

Assembler

Machine Code

Instruction
set

Hardware

© 2015 Scriptol

Low level program



1 second



1 year



High level program



5 seconds



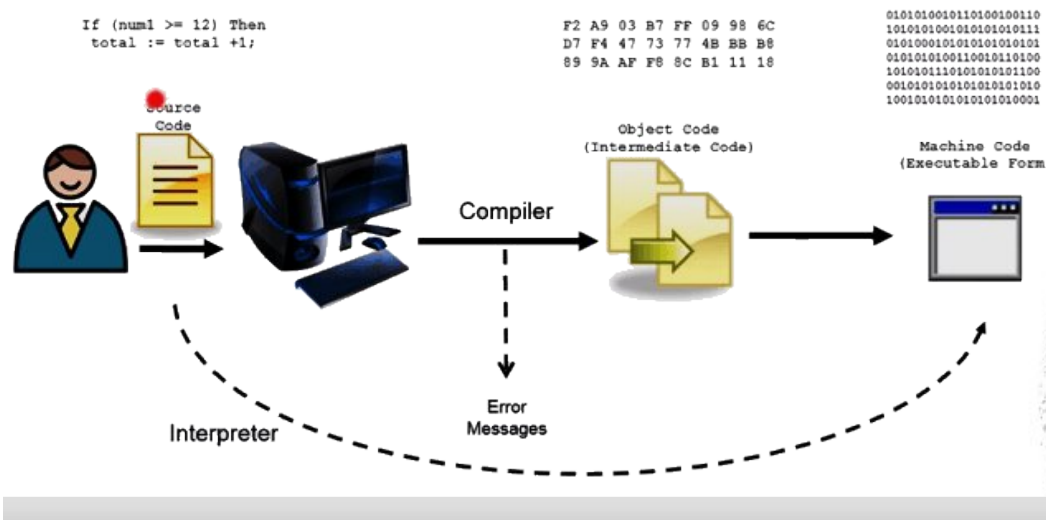
1 month



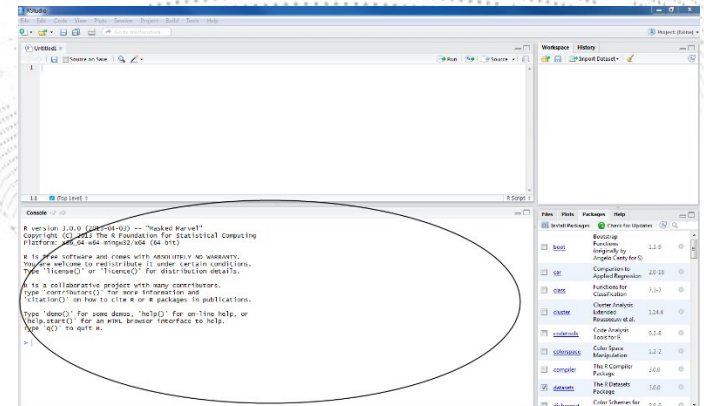
What kind of languages are there?

Interpreted vs. compiled languages

Compilers & Interpreters (high-level)



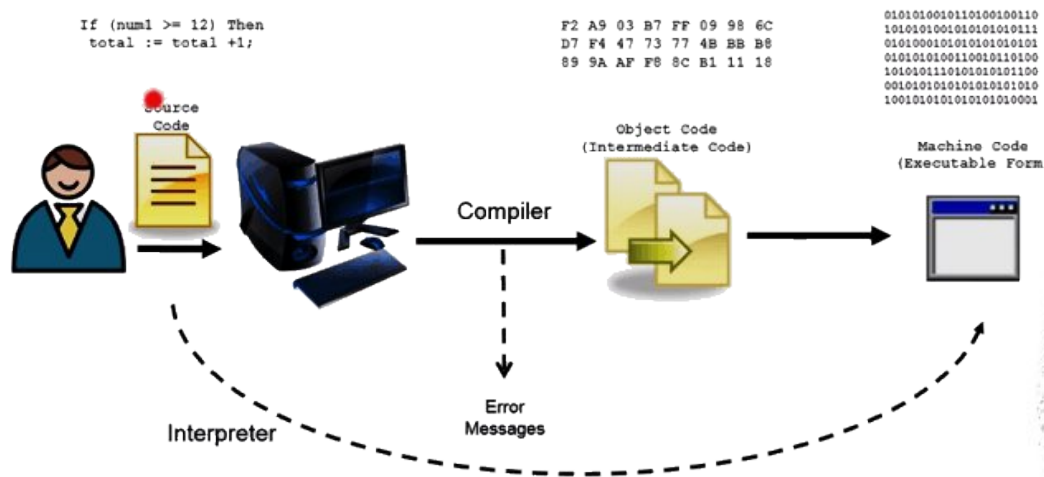
Working in a console



What kind of languages are there?

Interpreted vs. **compiled** languages

Compilers & Interpreters (high-level)



```
1 7f45 4c46 0201 0100 0000 0000 0000 0000  
2 0300 3e00 0100 0000 b007 0000 0000 0000  
3 4000 0000 0000 0000 981b 0000 0000 0000  
4 0000 0000 4000 3800 0900 4000 1d00 1c00  
5 0600 0000 0400 0000 4000 0000 0000 0000  
6 4000 0000 0000 0000 4000 0000 0000 0000  
7 f801 0000 0000 0000 f801 0000 0000 0000  
8 0800 0000 0000 0000 0300 0000 0400 0000  
9 3802 0000 0000 0000 3802 0000 0000 0000  
10 3802 0000 0000 0000 1c00 0000 0000 0000  
11 1c00 0000 0000 0000 0100 0000 0000 0000  
12 0100 0000 0500 0000 0000 0000 0000 0000  
13 0000 0000 0000 0000 0000 0000 0000 0000  
14 780b 0000 0000 0000 780b 0000 0000 0000  
15 0000 2000 0000 0000 0100 0000 0600 0000  
16 780d 0000 0000 0000 780d 2000 0000 0000  
17 780d 2000 0000 0000 9802 0000 0000 0000  
18 c003 0000 0000 0000 0000 2000 0000 0000  
19 0200 0000 0600 0000 900d 0000 0000 0000  
20 900d 2000 0000 0000 900d 2000 0000 0000  
21 0002 0000 0000 0000 0002 0000 0000 0000  
22 0800 0000 0000 0000 0400 0000 0400 0000  
23 5402 0000 0000 0000 5402 0000 0000 0000  
24 5402 0000 0000 0000 4400 0000 0000 0000  
25 4400 0000 0000 0000 0400 0000 0000 0000  
26 50e5 7464 0400 0000 e409 0000 0000 0000
```

```
adam@vulcanodon:~$ chmod +x hello_exec  
adam@vulcanodon:~$ ./hello_exec  
Hello World!  
adam@vulcanodon:~$
```

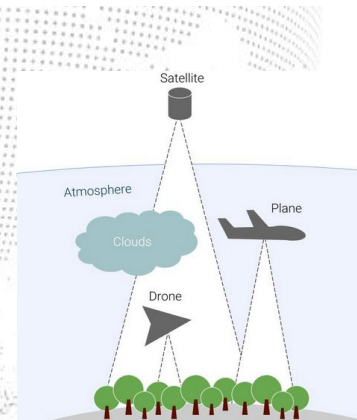
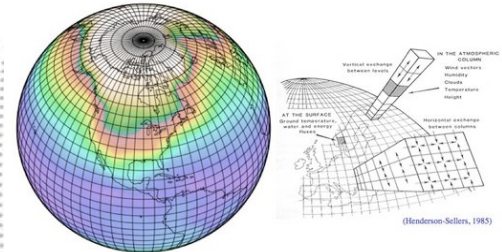
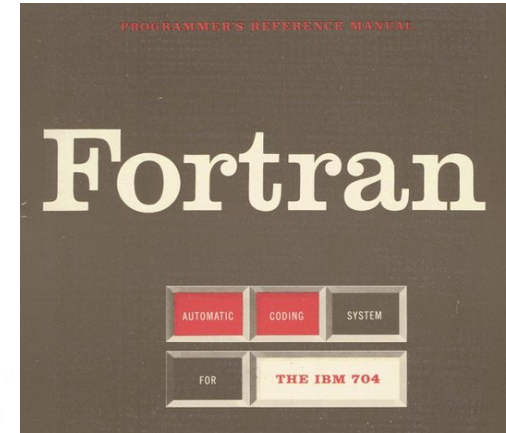
```
adam@vulcanodon:~$ g++ hello.cpp -o hello_exec
```

```
1 #include <iostream>  
2 using namespace std;  
3  
4 int main()  
5 {  
6     cout << "Hello World!\n";  
7     return 0;  
8 }  
9
```


Some programming languages...

Fortran

- IBM, since 1957, first high-level language
- For mathematical computations, compiled
- One of the fastest languages, still
- climate models, remote sensing data, crystallography



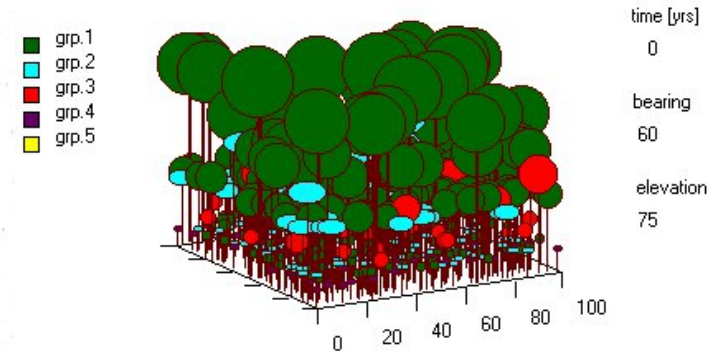
Some programming languages...

C++

- C with extended object-oriented features
- Complex data structures, yet very fast
- Used everywhere, popular desktop applications (e.g. Adobe PS, MS Office) computer games, agent-based modelling
- Very good R integration (Rcpp package)



C++ is the new C — twice the power, twice the size, works in hostile environments, and if you try to use it without care and special training you will probably crash.



Some programming languages...

Java

- Based on C too
- Compiled, runs in a virtual machine: code is very deployable
- Faster than either R or Python
- Some desktop applications, mesquite, imageJ



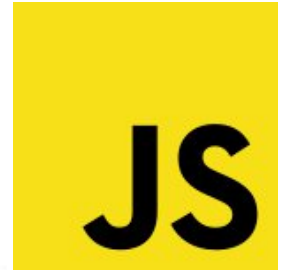
Java is another attempt to improve on C. It sort of gets the job done, but it's way slower, bulkier, spews pollution everywhere, and people will think you're a redneck.



Some programming languages

JavaScript

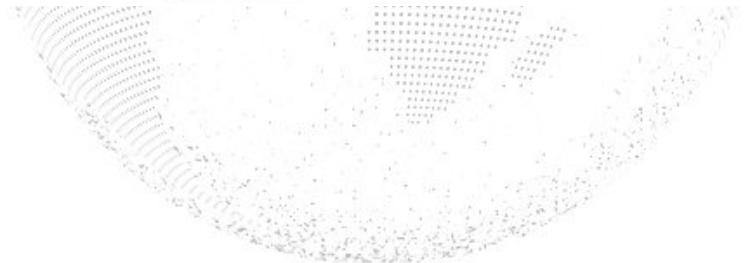
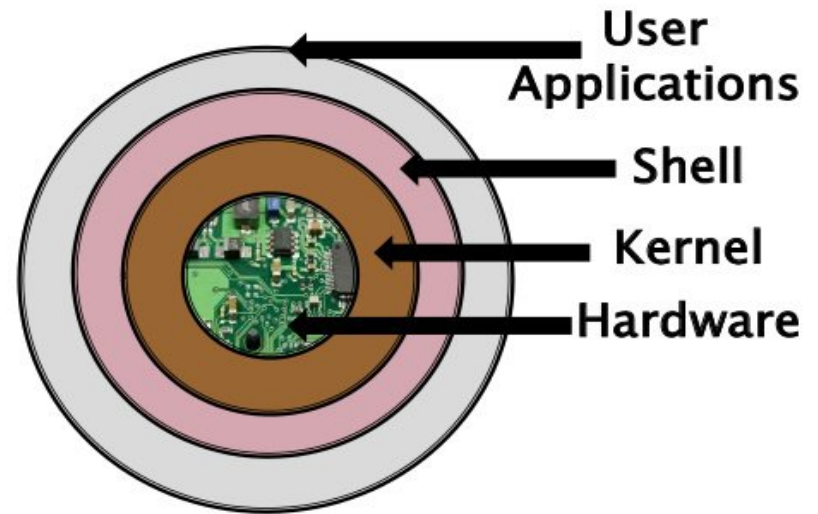
- Scripting language for the World Wide Web
- Executed by the clients (the computer visiting the website)
- Controls animations, interactive content
- Application development using node.js



Some programming languages...

bash

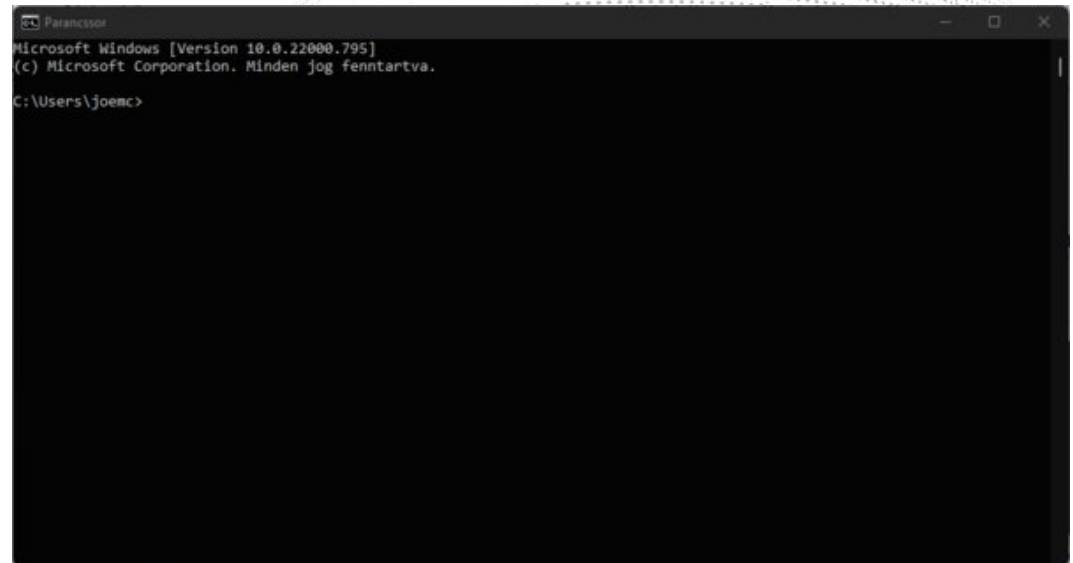
- Shell scripting language based on the Bourne shell (1976)
- Current standard on unix-like operating systems (e.g. Linux)
- Useful for file management, system administration, running console applications, raw data processing
- Other solutions: zsh, ksh, csh, fish



Some programming languages...

cmd.exe

- The command prompt
- The shell of Windows
- Very tedious to use



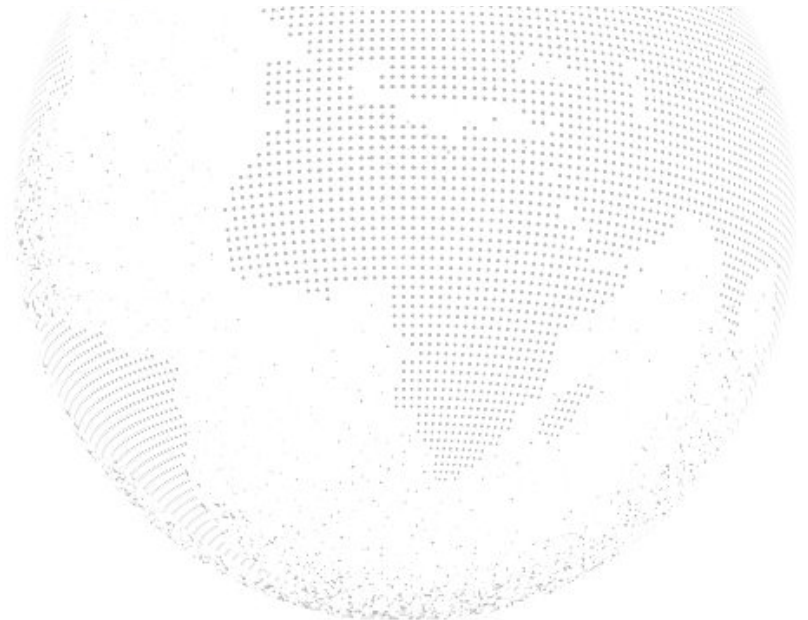
Some programming languages...

Python

- Since 1991
- Higher level than C, interpreted, general purpose
- Very popular due to the clean syntax
- Two main version still in use: Python 2 and Python 3
- Tons of scientific packages, many programs have python APIs
- Desktop applications, e.g. Gplates, debian-apt



Python is great for everyday tasks: easy to drive, versatile, comes with all the conveniences built in. It isn't fast or sexy, but neither are your errands.



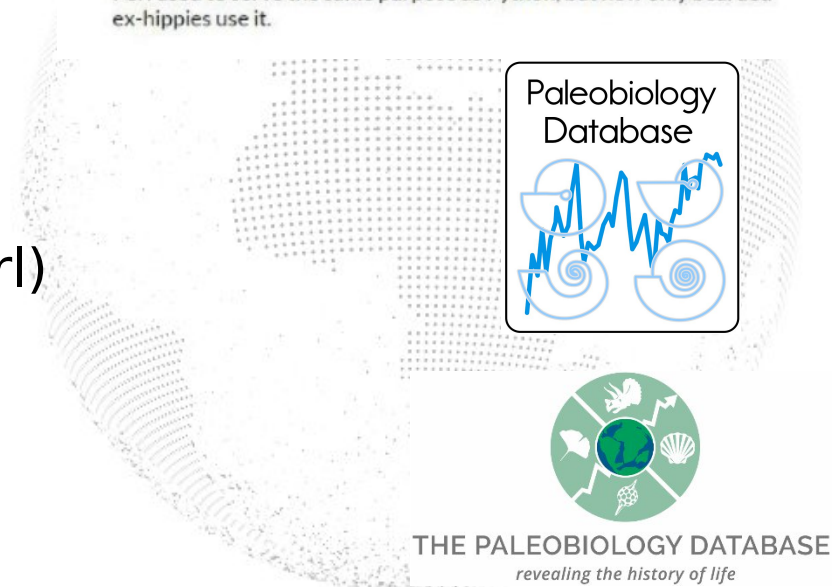
Some programming languages...

Perl

- A family of languages
- Originally for text processing, somewhat faster than python
- Used commonly in bioinformatics, e.g. DNA sequence analysis
- Sometimes for the web with databases (originally the PaleoDB website was using perl)



Perl used to serve the same purpose as Python, but now only bearded ex-hippies use it.



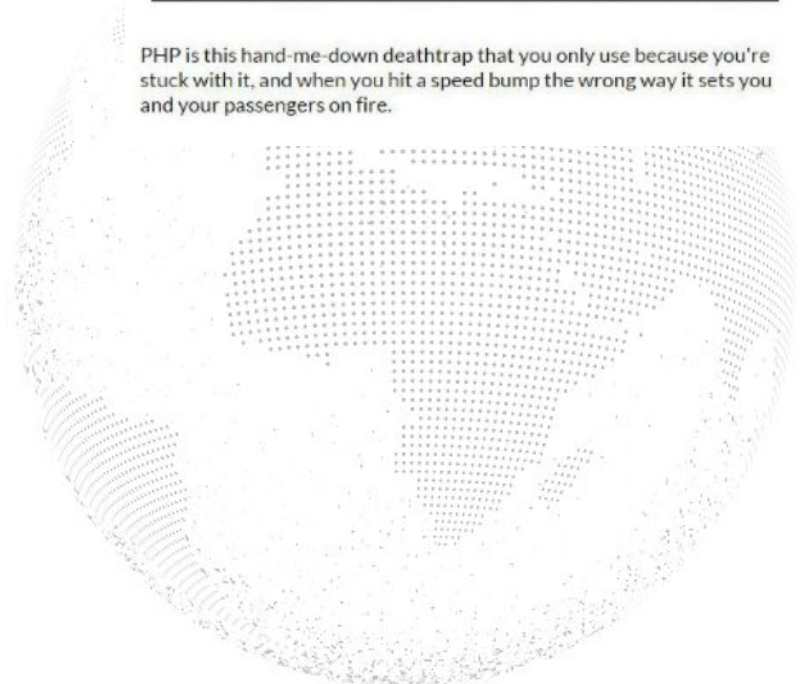
Some programming languages...

PHP

- General purpose, designed for web development, interpreted
- Server-side programming
- Very good database integration
- Web-based applications, shops, content management (e.g. Wordpress)



PHP is this hand-me-down deathtrap that you only use because you're stuck with it, and when you hit a speed bump the wrong way it sets you and your passengers on fire.



Some programming languages...

SQL

- Structured Query Language
- The language of relational databases
- Local databases: MySQL, PostgreSQL, MariaDB, Oracle Database
- Define, Manage, Query



Some programming languages...

MATLAB

- Since 1984, Mathworks
- Mathematical computations, especially linear algebra
- Proprietary – good packages
- GNU alternative: GNU Octave
- Many mathematical, engineering, scientific algorithms are only available in this language



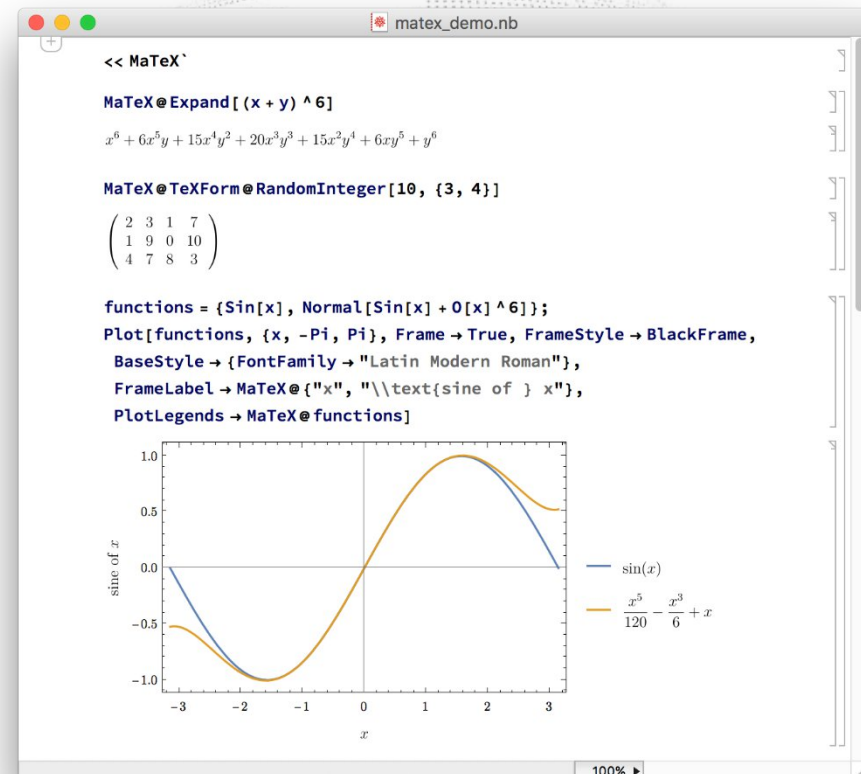
MATLAB is what scientists use to do special scientist things.



Some programming languages...

Mathematica

- Developed by Wolfram Research
- Symbolic language, as close to maths as possible
- Alternative to matlab (even more expensive)



Some more... Julia



- The next big thing, fastly growing
- Built for ease of use and performance at the same time
- Good choice for numerical simulations
- Only language besides Fortran and C++ to reach petaflops-level performance



Some even more...

- BASIC, C#, Rust, Lua, Ruby, Go, Pascal, LISP, F#, Haskell, SAS, etc..

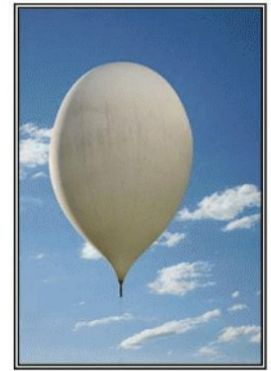


```
READY
10 FOR X=1 TO 10
20 PRINT "HELLO WIKIPEDIA"
30 NEXT X
RUN
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
HELLO WIKIPEDIA
READY
■
```

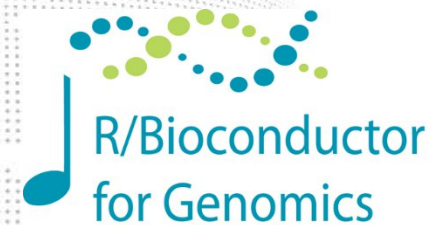


R (S)

- GNU version of **S** (1976), since 1992 (**R**oss Ihaka and **R**obert Gentleman, cf. S3, S4)
- Written mostly in C and Fortran
- Statistics-oriented
- 17th most popular language on TIOBE
- High-level language: can be very slow
- Interpreted
- **CRAN** packages (21467)
- Contributor to Debian



R is what scientists use when they can't afford MATLAB.



Package	Priority
boot	"boot" "recommended"
base	"base" "base"
boot	"boot" "recommended"
class	"class" "recommended"
cluster	"cluster" "recommended"
codetools	"codetools" "recommended"
compiler	"compiler" "base"
datasets	"datasets" "base"
foreign	"foreign" "recommended"
graphics	"graphics" "base"
grDevices	"grDevices" "base"
grid	"grid" "base"
kernSmooth	"KernSmooth" "recommended"
lattice	"lattice" "recommended"
MASS	"MASS" "recommended"
Matrix	"Matrix" "recommended"
methods	"methods" "base"
mgcv	"mgcv" "recommended"
nlme	"nlme" "recommended"
nnet	"nnet" "recommended"
parallel	"parallel" "base"
rpart	"rpart" "recommended"
spatial	"spatial" "recommended"
splines	"splines" "base"
stats	"stats" "base"
stats4	"stats4" "base"
survival	"survival" "recommended"
tcltk	"tcltk" "base"
tools	"tools" "base"
utils	"utils" "base"

Why learn / start with R?



- Isolated environment, experiment freely!
- Well-suited to statistics and scientific calculation: next step after excel
- *De facto* standard language in Ecology and Paleo
- Easy to set-up, works well on anything



R and RStudio



R: Language, tools to use it

- Terminal
- Plotting 'devices'



Rstudio: Integrated Development Environment (IDE) for R

- Runs R
- Code editor
- Document Building

