LECTURE 1
PGP - COMP1039

Dr Chris Roadknight

PMB 438

Office Hours: Monday 9:00 – 11:00

CONVENERS

	Chris Roadknight	PMB 438
	Pushpendu Kar	PMB 448
	Anthony Graham Bellotti	PMB419

ONLINE TEACHING

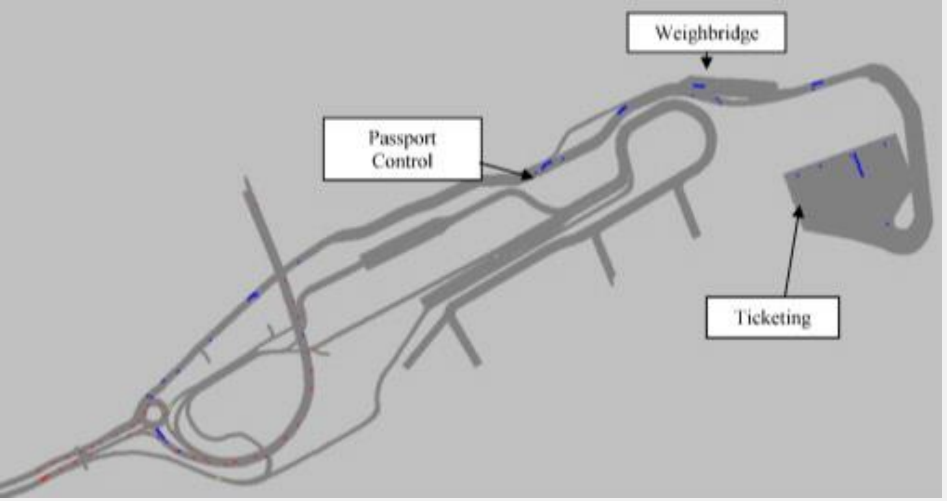
- What is online teaching?
- Means many things but typically refers to content that is delivered completely online, meaning there are no physical, face-to-face or on-campus sessions.
- Try and retain as much of the scheduling as possible.
- Watch the narrated lectures at the scheduled lecture time
- Do the lab material at the allotted time.
- Lecture pdfs will be available 3 days in advance as usual

WHAT ARE PARADIGMS

- Paradigm
  - Distinct set of concepts or thought patterns
  - A way of doing something
- Programming Paradigms
  - Process of catagorising programming languages based on their features
    - Eg. Indentation, syntax, levels of documentation, variable scopes, memory management, reserved words, exception handling, state support, non-determinism, notion of time, availability of goto……
- “No Free Lunch Theorem”
  - The Features of each programming language make it efficient at creating some programs but less efficient at creating others…

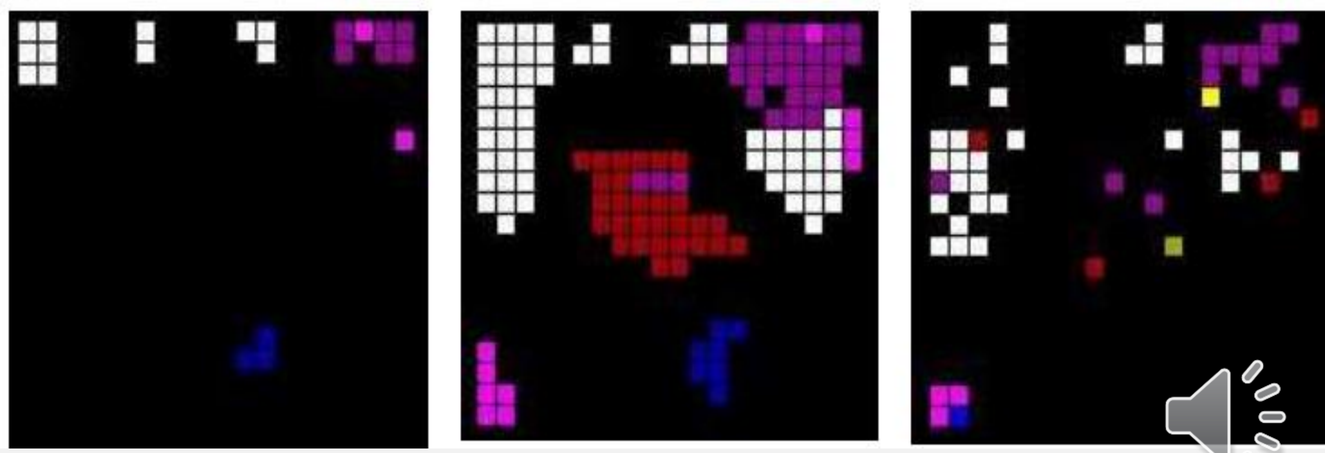
HOW MANY PROGRAMMING LANGUAGES HAVE YOU USED?

- (at least 2….depending on exact definition of unique languages)
- At your age I had used 2 (Basic and Fortran)
  - I didn’t do assembly until my MSc!
- Which of your array of languages would you use for programming a lightweight neural network?
- Which would you use for low level manipulation of memory?
  - (maybe the same one!)
- Understanding paradigms helps you make these decisions
  - (as does many other CS skills)



MY PROGRAMMING PARADIGMS

- **Basic/Fortran/Hypertext** (School and BSc)
- **Pascal** plagiarism detector (MSc)
- **C** neural networks (PhD)
- **Visual Basic/C++** Simulations – data prepared in **Bash** shell script and **MySQL** (BT)
- **MCC18** (**C** like) microcontroller programming for sensor networks (BT)
- **Matlab** Computer Vision (Lancaster Uni)
- **VISSIM/Anylogic/Java** for Simulation (Nottingham Uni)
- **R and SQL** – demographic modelling (Nottingham Uni)
- **Alteryx/Tableau** – Healthcare metric visualization (NHS)
- **R** – Machine Learning (NHS)
- **R, H2o[java], Anylogic[java] - UNNC**



HOW DO I CHOOSE THESE PARADIGMS?

- Appropriate tool [idealist view]
  - Matlab for vision, C for low level number crunching
  - SQL for big data manipulation
- Available packages [pragmatist view]
  - H2o, R, Alteryx all have packages usable in a few hours that took 6 months to program in my PhD!
- Hardware limitations
  - MCC18 required for pic chips
- F{appropriate, knowledge, support, specification….}

SOME VERY OLD (JAVA, VB) SOFTWARE!

test3.mp4

ROADKNIGHT, Chris; AICKELIN, Uwe; SHERMAN, Galina. Validation of a Microsimulation of the Port of Dover. *Journal of Computational Science*, 2012, 3.1-2: 56-66.

scroby2.exe

ROADKNIGHT, Christopher M. *Nodal policy inclusive techniques for operating an ad hoc network*. U.S. Patent No 8,031,684, 2011.

IMPORTANT POINT

- You will learn some Java in this module and some Haskell..
- ..but mainly to exemplify OOP and FP paradigms
- We assume NO knowledge of Java or Haskell
- Main difference between UK students and China students is UK students usually have some experience of Java before the start their degree
- Java comes up again and again in later modules, Haskell may not..

- Most common 'complaint' from student in later years..
- …"not enough Java in qualifying year"

- Good idea to set aside some time this summer to continue your Java learning

TIMETABLE

- Lectures:
  - Mondays: 12:00—13:00 (DB – A05)
  - Tuesdays: 16:00—18:00 (DB – A05
- Labs:
  - Thursday: 09:00-11:00 (PMB 432)
  - Fridays: 09:00—11:00 (PMB 432)
- Both Compulsory, Attendance taken at Labs only

WEEKLY LECTURE SCHEDULE

		1 hour DB-A05		2 hour DB-A05
week 23	17th feb	Intro PGP (CMR)	18th feb	Java/OOP (CMR)
week 24	24th feb	Java/OOP (CMR)	25th feb	Java/OOP (CMR)
week 25	2nd March	Java/OOP (CMR)	3rd March	Java/OOP (CMR)

Abandoned!

week 30	10th April (date change)	Haskell (AGB)	7th April	Haskell (AGB)
week 31	13th April	Haskell (AGB)	14th April	Haskell (AGB)
week 32	20th April	Haskell (AGB)	21st April	Haskell (AGB)
week 33	27th April	Haskell (AGB)	28th April	Cancelled/Spare
week 34	4th May	Haskell (AGB)	5th May	Haskell (AGB)/Exam prep (CMR/AGB)

ONLINE LEARNING [UNTIL NORMAL SERVICE RESUMED]

- Online teaching typically refers to courses that are delivered completely online, meaning there are no physical or on-campus class sessions.
- Content will be made available before the prescribed lecture times
- View at usual lecture times, lecturer will be available on moodle forum/chat for questions. Monday 12:00-13:00. Tuesday 16:00 – 18:00
  - I want to retain a sense of learning together…you are not alone!
- Additional relevant resources will be made available
- Labs will take a similar approach. Lab tasks will be made available but try and carry them out during your official lab timeslot.
  - Lab team will be available on moodle chat
- Delivery method may change each week - we have never done this before.
  - I will be asking for feedback via a moodle survey after my slot

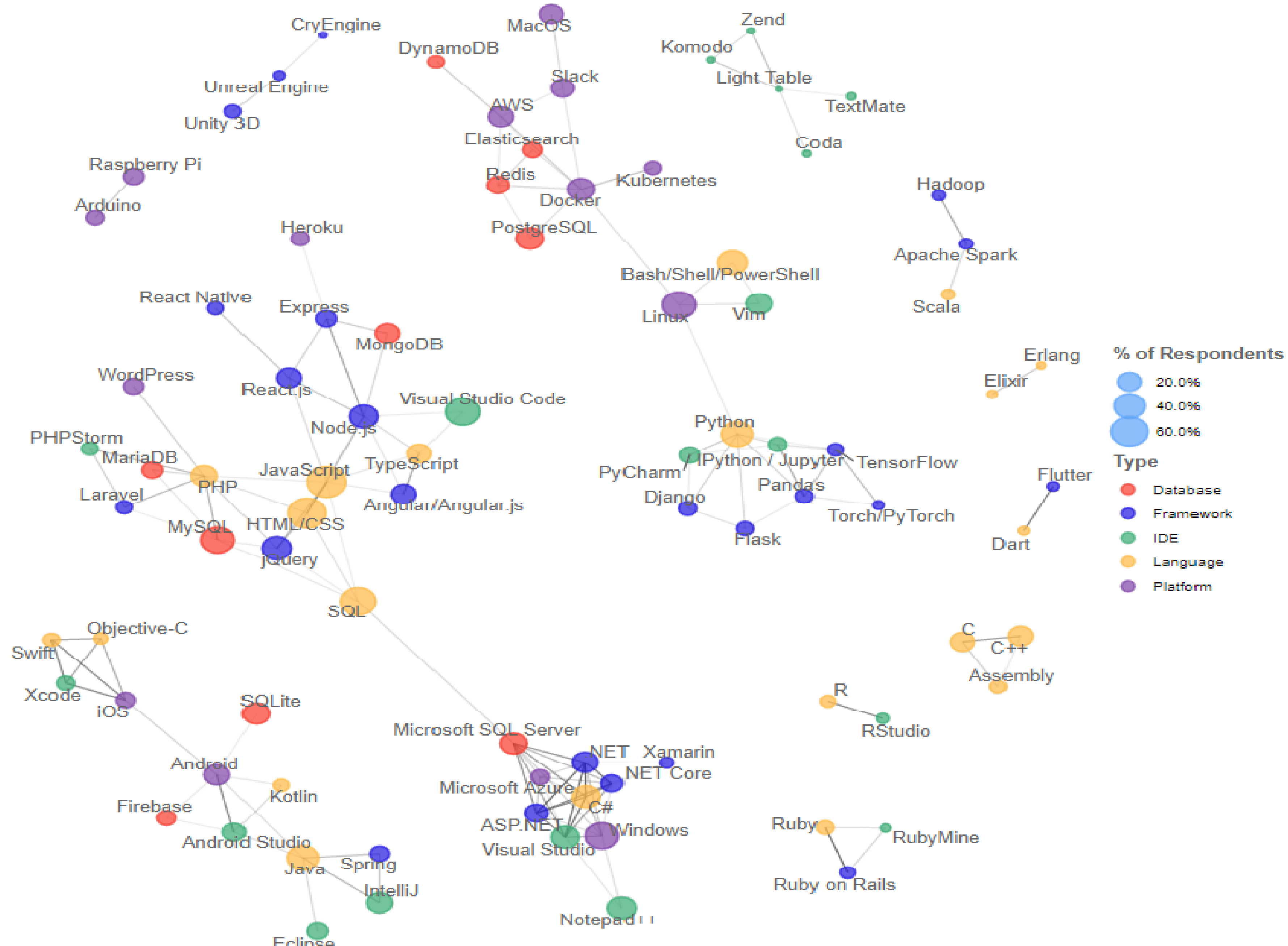
WEEK 1 LAB SESSIONS

- These are just about bringing everyone to the same start point
- Before week 2 you should be able to **write, compile and run simple code** in Java
- Hello World!

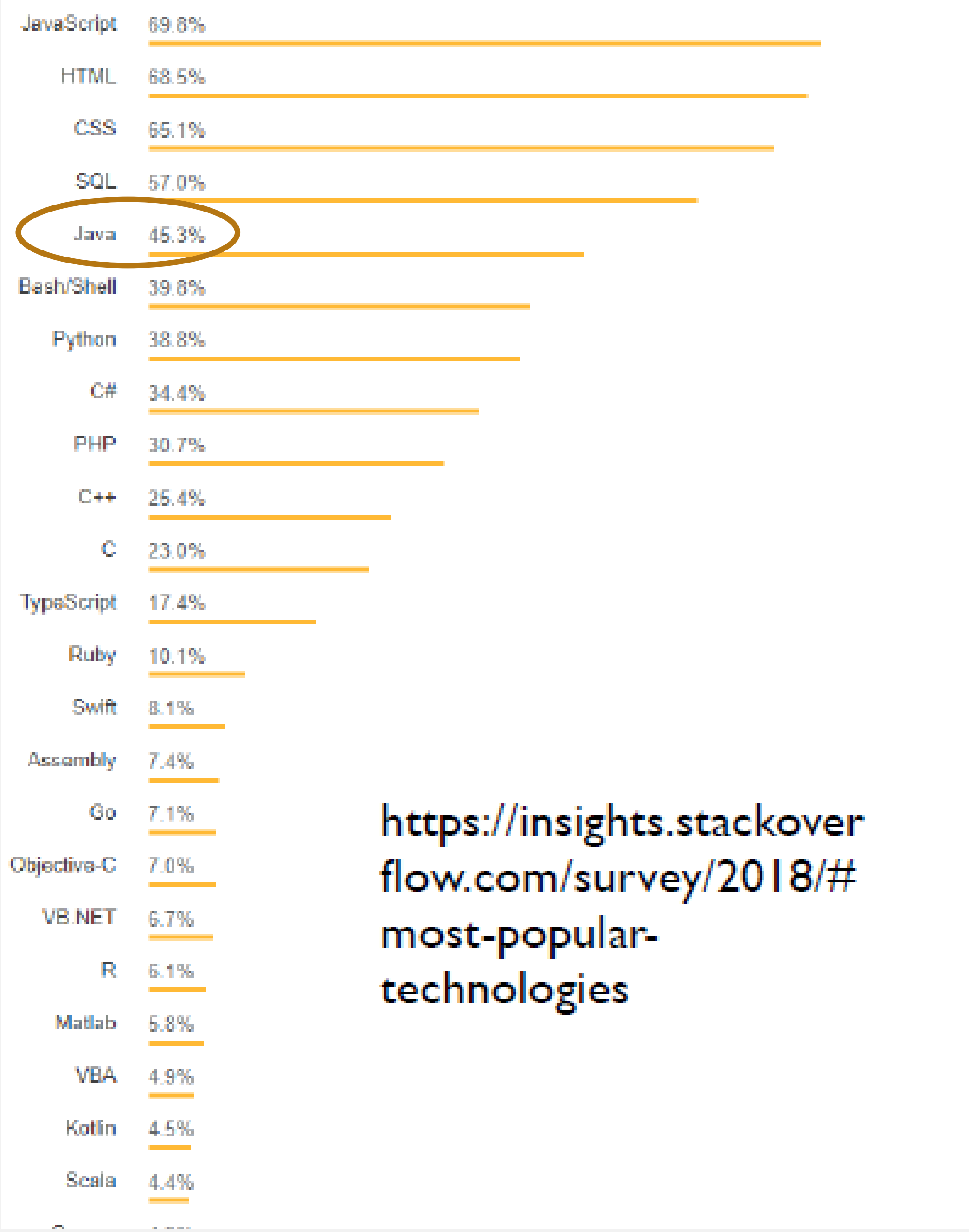
COMMON PARADIGMS AND SUB-PARADIGMS

- Unstructured Imperative (events sequentially happen based on sequential memory)
  - Machine Code
  - Assembly code (MIPS etc)
- Imperative (events have to happen with some kind of control flow)
  - Procedural (Pascal, Basic, C)
  - Object –based (Visual Basic)
  - **Object orientated (Smalltalk, C++, Java)**
- Declarative (expresses the logic without describing its control flow)
  - **Functional (Haskell, Python, R…)**
  - Logical
  - …

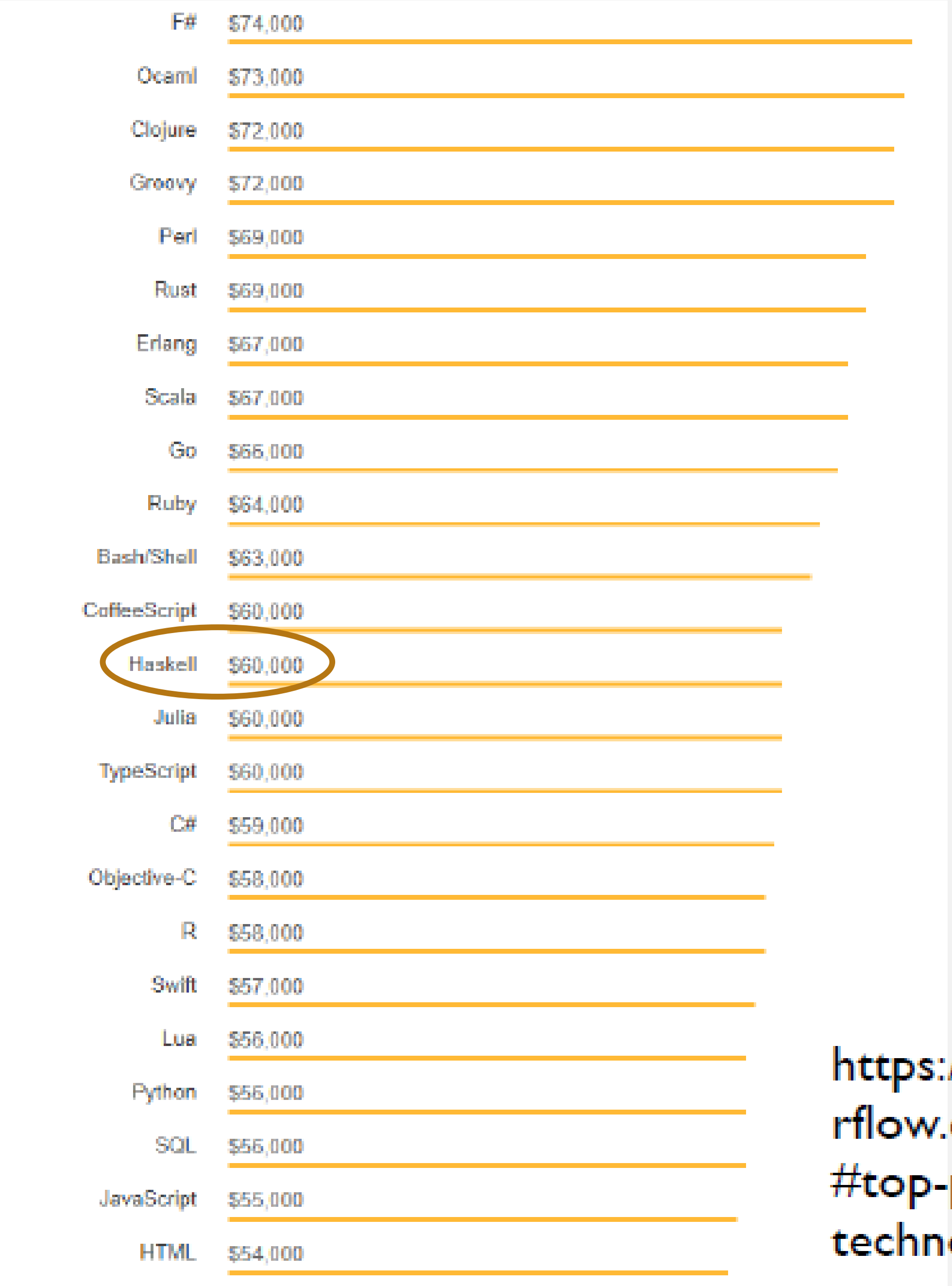
How Technologies Are Connected



https://insights.stackoverflow.com/survey/2019/#correlated-technologies



https://insights.stackoverflow.com/survey/2018/#most-popular-technologies



https://insights.stackoverflow.com/survey/2018/#top-paying-technologies

ASSESSMENT

- Coursework (25%)
  - Writing programs, possibly evaluated using moodle
- Written 210 minute exam (75%)
- **75%!!!**

COURSEWORK

- Coursework 1
  - No in-class assessment
  - 1-2 Programming Exercises (Java)
  - 15% of overall module mark
  - Short timeframe [about 48 hours]
  - “Open book” but don’t plagerise, I WILL use JPLAG
- Coursework 2
  - Programming Exercises (Haskell)
  - 10% of overall module mark

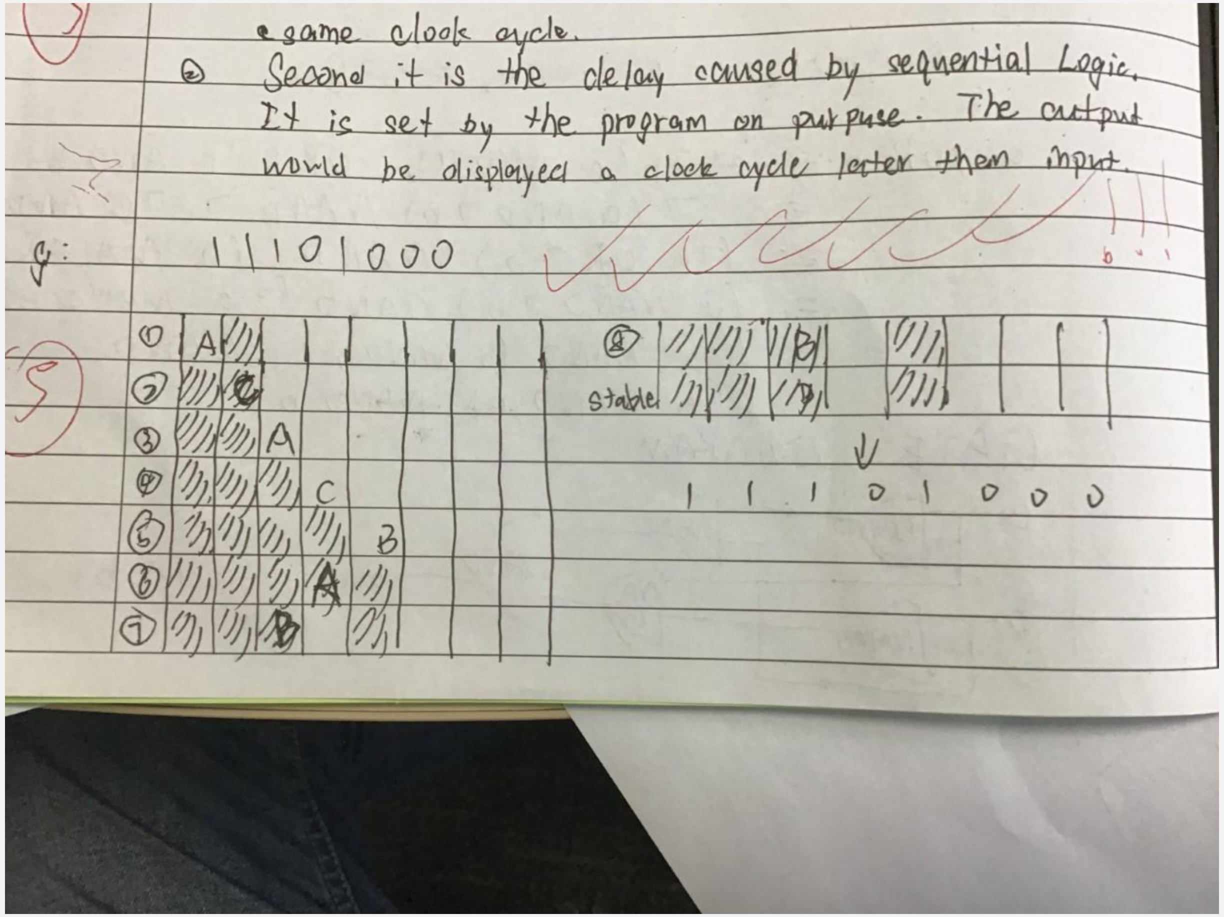
ASKING QUESTIONS IN PGP

- As much as possible…
  - If you email a sensible question, it will be pasted and answered in moodle, so everyone benefits..
  - ..so may as well ask on moodle
  - Feel free to answer someone’s question on moodle!
  - If you ask useful question in person, the question and answer may be covered in lecture/lab/moodle [not possible until normal service resumed]
  - Questions on moodle will be answered first
  - PGP Discussion Forum

MOODLE

- Resources required for labs and coursework
  - Eg. Datasets, web links, etc
- CW submission
- Slides
  - Not all content will be on slides. Some background reading required for those ‘special’ exam questions…cf. Turing Machine in CSF [sorryNotSorry]
- Announcements
- Useful additional content (papers etc)
- Questions/answers/discussion

ASIDE..TURING MACHINE



same clock cycle.

② Second it is the delay caused by sequential Logic. It is set by the program on purpuse. The output would be displayed a clock cycle later then input.

g: 11101000

5
①
②
③
④
⑤
⑥
⑦
⑧
A
C
A
C
B
A
B
B
B
stable
1 1 1 0 1 0 0 0

TOPICS COVERED IN OOP SECTION

- • We hope to discuss the following topics (roughly, up to the end of Chapter 11 of the textbook):
  - ▫ Classes, Objects, Methods
  - ▫ Inheritance
  - ▫ Abstract Classes and Interfaces
  - ▫ Exception Handling
  - ▫ Using I/O
  - ▫ Also, contrasting the OOP and FUN paradigms.
- • Depending on your/our performance, we may discuss more topics, e.g.:
  - ▫ Multithreaded Programming
  - ▫ GUI Programming
  - ▫ Generics
  - IDEs

OOP

- Before OOP we had procedural programming
  - Program divided into functions, that operate on variables
  - As programs grow, there becomes a lot of interdependencies
  - OOP combines related variables and functions into an object
- Allows for key principles of:
  - **Abstraction**
    - Handle complexity by hiding unnecessary details from the user Polymorphism
  - **Inheritance**
    - One class is allowed to inherit the features (fields and methods) of another class
  - **Encapsulation**
    - Grouping of related variables and functions that operate on them
  - **Polymorphism**
    - Ability of an object to take on many forms.

PROGRAMMING SUPPORT – A HISTORY

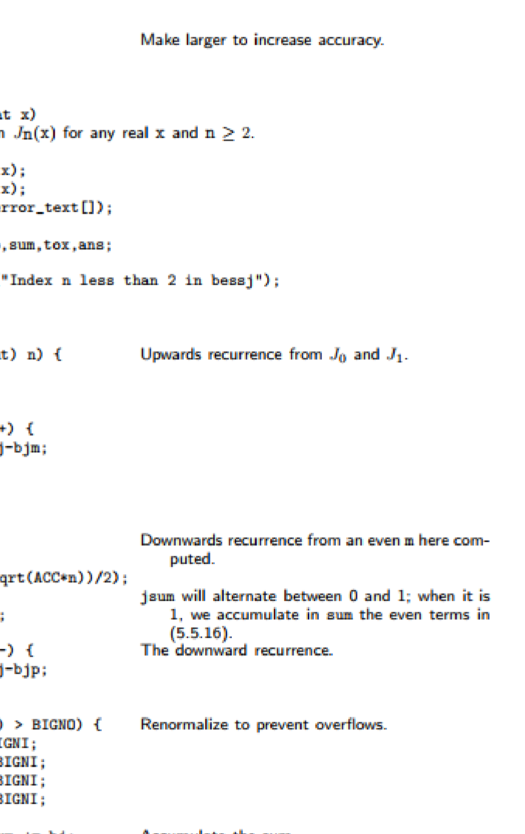
- 1995 – Books, Colleagues…

2005 – CD roms (from books), email, colleagues

2015 – Forums, reverse engineering, colleagues

2019+

- **Kaggle, Github, forking, StackOverflow, colleagues**



```
                    Make larger to increase accuracy.


t x)
n Jn(x) for any real x and n ≥ 2.

x);
x);
rror_text[]);

,sum,tox,ans;

"Index n less than 2 in bessj");



t) n) {            Upwards recurrence from J0 and J1.



+) {
j-bjm;





                    Downwards recurrence from an even m here com-
                        puted.
qrt(ACC*n))/2);
                    jsum will alternate between 0 and 1; when it is
;                       1, we accumulate in sum the even terms in
                        (5.5.16).
-) {                The downward recurrence.
j-bjp;


) > BIGNO) {        Renormalize to prevent overflows.
IGNI;
BIGNI;
BIGNI;
BIGNI;

                    Accumulate the sum.
```