Table of Contents

LECTURES

INFO1111 BREAKDOWN

- **Future as an IT professional** – rapidly changing environment requires continuous learning
- **Professional Development Plan** – tool to plan for the future
- How **professionalism and skills** fit into the requirements to become a capable IT professional
- **Professional bodies, societies and frameworks** that apply to IT professionals e.g. SFIA, ACS etc
- **Teams and teamwork:**
  - what is a team?
  - how diversity affects team?
  - what makes a team successful?
  - teams in student life and in professional IT world
- **Communication:**
  - models of communication
  - context of communication
  - forms and related tools
  - challenges and common mistakes
  - LaTeX and Markup
- **Collaboration:**
  - how this works in professional practice
  - project management and task allocation
  - working with Git
  - Git commands
- **Finding and using information**
  - accessing, assessing and using sources of information
- **Problem Solving**
  - the different types of problems IT professionals are expected to solve
  - how do you approach a problem?
    - Understanding the problem
    - identifying solutions using different methods
  - Algorithms
    - how we represent them
    - how we develop them
  - computational thinking
- **Intellectual Property and Commercialisation:**
  - different forms of protection for intellectual property
  - forms of commercialisation
- **Ethics:**
  - how do you know right from wrong
  - how do you judge other people/companies/groups ethics
  - theories/scenarios/case studies
  - A broader look at ethics in Computer Science
  - Professional conduct
  - scenarios / debates
- **Systems Thinking:**
  - Predicting a systems behaviour
    - break down to simple components
    - evaluate within context
    - take into account feedback loops and time delays
    - complexity and emergence
  - Difference with other forms of thinking
  - Application in IT
    - Designing systems components
    - Designing pocesses and workflows using components
    - User-centric thinking
- **Majors:** Computer Science, Software Development, Computational Data Science, Information Systems

WEEK 2: ICT PROFESSIONALS & TEAMWORK

QUESTIONS	ANSWER
	ICT PROFESSIONALS

Define "professionalism"	Consists of skills:
	• professionalism
	  ◦ (e.g. internal processes, legislation, processes, products, services, technology)
	• behavioural
	  ◦ (varies, as valued by organisation)
	• qualifications
	  ◦ (e.g. business process improvement, database design

What is SFIA?	Skills Framework for the Information Age • Globally accepted common language to measure workforce skills and competencies needed in the digital world • Creates a common goal for ICT field (is constantly revised) • Assesses future requirements and ability of business to meet them • Enables development of targeted training Model made of professional skills on one axis & 7 levels of responsibilites on the other describing behaviour/values/knowledge/characteristics required to be competent: 1. follow: complete tasks under supervision 2. assist: range of tasks, discretion 3. apply: complete work packages w/ milestone reviews 4. enable: work under general direction 5. ensure + advise 6. initiate + influence 7. set strategy, inspire + mobilise: highest
Give examples of professional associations	ACM = Association for Computing Machinery • world's largest computing society supporting professional growth, lifelong learning, career development and professional networking with motto of "advancing computing as a science + profession" • umbrella organisation bringing together educators, researchers, professionals to inspire dialogue, share resources, address field's challenges

Give examples of professional associations	ACS = Australian Computing Society • Promote development of Australia information, commerical technical resources Professionals Australia • formerly APESMA (Associate of Professional Engineers, Scientists, Managers Australia) • union for technology professionals, gives advice (salary + industrial)
	TEAMWORK
Define "teamwork"	A group of people with a full set of complementary skills and shared purpose, working together to achieve a common goal through coordinated effort which allows each member to maximise strength + minimise weaknesses willingness to work collaboratively within a group to achieve a goal, using their individual skills, provide constructive feedback, build on strengths, create positive working atmosphere, support each other and enhance team performance
Give examples of some common teamwork issues	• Logistics • Scheduling conflicts • Task delegation • Coordinating member commitments/contributions (especially when team product is lower than individual product = high level of disatisfaction and stress)
Some improvement strategies	• Positive organisational strategies (e.g. team constitution, open discussion to improve team dynamic) • Dealing with differences (tolerance) + negative behaviour (e.g. discounting, aggressiveness, controlling, freeloading, blocking)
What are 3 factors for success?	• Sufficient contribution • full discussion of issues • member support (high quality result + member satisfaction)

What are Hackman recommendatons for successful teams?	• Satisfy internal + external clients • develop capabilities to sustain future performance • members find meaning + satisfaction
What are 5 factors that increase factors for success?	• ‘real’ team (clear membership, stability, shared goals) • compelling direction (SMART goals) • enabling internal structure + team size • supportive context (rewards, learning, development) • expert mentoring, coaching, evaluation
Differences between IT professional teams + student teams	IT professionals: • multidisciplinary + multifaceted, traditional plan + doc structures (e.g. Agile, scrum, xp), Students: • lack of fully shared fate (differing schedules + goals) • limited consequences
What are some common beliefs?	• Harmonious teams outperform teams with some conflict (F) • Larger teams outperform smaller ◦ (F – too large can become dysfunctional, too small to be ambitious) • teams with intact membership can gradually deteriorate • team dynamics are largely determined by leadership style (e.g. authoritarian vs dynamic)
Qualities of a good team member	• Honest • Sharing the load • Positve attitude • Respect • Reliable (meets deadlines, punctual) • Fair • Complements (provides diverse + unique skill/knowledge that moves the team forward) • Good communication

Effective teams are	• Cohesive (open, trust, respectful, organised) • clear direction + purpose (measurable objectives that unifies goals + desired outcomes) • Collaborative spirit • Defined roles + rules (on track, eliminates ambiguity, skill sets,thinking styles) • Encourage differences in opinion • Mutual accountability • Team trust • Efficient use of ideas • Decision-making (hierarchy helps quick reraction + effective. Each member respected for various areas of expertise, leader obtains opinion to form group response)

WK 13 REVIEW Qs	• **Teams and teamwork:**
	  ◦ What is a team?
	  ◦ How does diversity affect a team?
	  ◦ What makes a team successful?
	  ◦ Teams in student life and in professional IT world

WEEK 3: COMMUNICATION

QUESTIONS	ANSWER
	COMMUNICATION
Define "communication"	A dynamic process of information exchange between two people using common system of symbols/behaviour/signs with a shared purpose + method intent of creating a shared understanding + connection, and a goal of creating a common conception, change behaviour or acquiring/transferring information
What is a communication model in general?	A conceptual model used to explain the human communication process
What is the communication process?	1. sender (intend to convey) 2. encoding (requires knowing audience) 3. message [noise] 4. channel [noise] 5. receiver [noise] 6. decoding (active listening) 7. feedback Sender conceptualises ideas, transmits msg through channel to R who gives feedback in form of some msg/signal
3 models of communication (strengths + weaknesses)	1. TRANSMISSION • Linear, one-way • Focus is on the sender to deliver effective communication • Weaknesses: ◦ 3 issues creators recognised: ▪ technical (how *accurately* can msg be transmitted) ▪ semantic (how *precisely* can meaning be conveyed) ▪ effectiveness (how *effectively* does received meaning affect behaviour) ◦ suscpetible to noise ▪ physical e.g. environmental, deaf, blind ▪ semantic / problems with encoding/decoding e.g. different language ▪ cultural ◦ assumes isolated communicators – prevents differing purposes/interpretations/unequal power relations/situational contexts

3 models of communication (strengths + weaknesses)	2. INTERACTION
	• Two-way – focus on participant interaction
	• like 2 linear models stacked on top of each other + 'field of experience' - cultural background, ethnicity, location, personal experience accumulated
	• Context of communicators:
	  ◦ physical (env where interaction takes place)
	  ◦ psychological (emotional / mental state of each communicators)
	• Weaknesses:
	  ◦ multiple msgs pass between (sometimes unintentional)
	  ◦ ineffective if only based on one msg
	  ◦ has feedback, but not instant
	• Example:
	  ◦ IM (S must wait for R to reply)
	
	3. TRANSACTION
	• two-way
	• takes into account contextual
	• characteristics
	  ◦ both communicators are S-R simultaneously (even if unconscious)
	  ◦ communication affects all parties involved, thus fluid + simultaneous
	  ◦ contextual (social, physical, psychol, relational, cultural)
	• Example:
	  ◦ conversation with a friend

Some forms of interpersonal communication	• Media (e.g. written, oral)
	• Scales (e.g. 50 page report / 1 pg summary)
	• Target
	  ◦ type (boss / subordinate / colleague)
	  ◦ number (broadcast, 1-to-1)
	• Interaction pattern (frequently alternating, one-way)
	• Purpose (convey facts, inform, guide actions)

Define "effective communication"	• Information has been received as intended (underlying emotions and intentions) • all participants are aware, feel heard and understood • assertive communication • ability to understand own emotions and others

What are 3 steps to effective communication?	1. KNOW YOUR PURPOSE • what change/reality do you want to make e.g. learn a tool, feel a certain way, do certain actions 2. CONTEXTUALISE AUDIENCE • start from what they... ◦ ...know ▪ explain unknown in relation to known ◦ ...expect ▪ in familiar terms ▪ follow their style (dress, speaking) ◦ ...value • will be able to refine actions to produce outcome 3. RESPECT AUDIENCE • make use of their domains ◦ avoid repetition ◦ don’t do what they can do better • respect their time ◦ be well-prepared - focus on meaning of what you want to communicate ◦ provide an easy exit ◦ connect/align with their goals and beliefs
7 Cs to communication	Clear, courteous, considerate, concise, concrete, complete, correct
What are some effective communication skills?	• Active listening (clarify + summarise + ask Qs + give feedback) • Non-verbal • Clear + succinct • Empathy (develop trust + rapport + being present)
Benefits of effective communication	• Reduces misunderstanding / overlooking important information – saves time due to accurate assessment of info • Creates non-threatening environment – share thoughts/ feelings/ideas, feel respected + able to work together + solve problems + make decisions • Builds trust, promotes desire to work towards common goal • Can communicate important messages without conflict

Barriers to effective communication	• Judgement (criticism, diagnosing, insulting) • Inattentive (distraction, looking away, irrelevance) • Technical language (over-complex, unfamiliar jargon) • Giving solutions / unwanted advice • Avoiding concerns / not addressing problem (diverting convo, reassurance, discounting) • Emotional barriers / taboos • Difference in perception + viewpoints

WK 13 REVIEW Qs	• **Communication:**
	  ◦ models of communication
	  ◦ context of communication
	  ◦ forms and related tools
	  ◦ challenges and common mistakes
	  ◦ LaTeX and Markup

WEEK 4: INFORMATION RETRIEVAL & COLLABORATION

	INFORMATION RETRIEVAL
Define "sources"	Materials from which information/ideas are gathered - print (books , journals) - non-print (radio , music)
Define "evaluation"	• Authority – holds author accountable for ideas • Objectivity – how does info appear in relation to other reliable sources of information, can make appropriate decision on what to do with • Coverage – adequately addresses target audience • Peer-reviewed – check-and-balance system mitigates release of unsound information • Suitability of material • Scholarly journals • Written by/for faculty, researchers • Uses citations and technical language • Longer in length
What is one method of evaluation?	CRAAP Test • Currency ◦ timeliness ◦ recency of date – some info is time-sensitive • Reliability ◦ importance ◦ consistent + complete ◦ is content biased/fact/opinion ◦ references for quotations/data • Authority ◦ source ◦ who can you contact ◦ credentials / reputable publishing • Accuracy ◦ usefulness ◦ reliability, truthfulness, correctness ◦ evidence-supported, verifiable ◦ spelling/typographical errors (inaccurate info) • Purpose ◦ intent, domain, ads, audience good info sources will present in a calm reasonable tone avoid writing that's overly critical, attacking or spiteful unique – keeps company competitive, adds value and is distinctive

Define "sufficiency"	Wide range of sources that support / oppose
Define "referencing"	Distinguishing between work that is your own and another's
What are the purposes of referencing?	• Promotes academic integrity - acknowledges + respects other intellectual rights • lends credibility to argument – you know the field in which you operate • allows others to use/critique/research/compare/analyse • comparing and contrasting thoughts = fact-checking tool that enables better verification of work • acknowledges previous work in field + positions new research in relation to previous • directs others to original sources of work so they can independently determine whether attributes support your argument
How would you reference?	• Direct quotations, paraphrasing, cite own work • Don't fabricate data or knowingly assist others in acts of dishonesty
	COLLABORATION
Define "collaboration"	• Process of working jointly with others to create/achieve/satisfy an outcome • 2/+ people [team] working together [processes] towards a shared/purpose [without it, people are just cooperating, need a clear and narrow focus] • knowledge management and transfer in a sustainable way over time period • relies on openness + sharing + focus + accountability negotations, conflict resolution + strategy, assertiveness + cooperation • seeks to advantage to all parties, successful when participant goals are compatible, interaction among them important to its achievement

Define “collaboration”	conceptual requirements:
	• awareness
	  ◦ part of working identity / shared purpose
	• motivation
	  ◦ gain consensus in problem-solving
	• self-sync
	  ◦ decide as individuals when things will happen
	• participation
	  ◦ collab and expect others to do the same
	• mediation
	  ◦ negotiate
	• reciprocity
	  ◦ share and expect others to share
	• reflection
	  ◦ think and consider alternatives
	• engagement
	  ◦ proactive vs active
	
	Can be:
	• focuses:
	  ◦ simple
	  a lot of dialogue, instant response but distracting
	  ◦ document
	  deliverables
	  ◦ structured
	  required format of workflow process
	• time:
	  ◦ synchronous: real-time
	  e.g. face to face, skype
	  ◦ asynchronous: time-shifted interaction

Are there any issues to collaboration?	An upper limit to how many people can collaborate can lead to noise and distraction
How would you project manage?	• To-do lists • multiple methodologies

How do IT professionals and students differ?	IT professionals
	• adhere to industry standards
	  ◦ traceability, security, accountability
	  ◦ online tool use (discussion forums, trello, shared repos)
	
	Students:
	• max output with limited resources
	• group work, diversity, attribution

What is version control?	Component of software configuration management – manages changes in documents, each revision associated with a timestamp and person making change • Must use to contribute to the team / small experimental projects e.g. bitbucket, github

What is version control?	• Simultaneous work & data history
	  ◦ Common: multiple revisions of same software displayed in different sites and developers to work simultaneously on updates
	  ◦ Bugs/features of software only present on some versions and not others due to fixing problems/introduction of others as the program develops
	  ◦ To locate and fix bugs need to be able to return to a previous version where the problem occurred. Two versions of program often needs to be developed concurrently (version wehre bugs are fixed and no new features introduced = branch, version where new features are added = trunk)

Define version control?	Allows correct mistakes and make revisions by reverting a document to a previous version, track edits to source code over time and defend against vandalism

- what is the process	• Unstaged + untracked
	  ◦ gitadd
	• Staged
	  ◦ gitcommit
	• Local repo
	  ◦ gitpush
	• Remote repo

- what is the distributed model	Peer-to-peer system instead of a centralised server Each node in distributed network is a full repo. Users make changes in working copy then commit changes to local repo which can exchange info (push/pull) with other repos

- branches	• Internal clone of a repo
	• when working on one will choose a branch to work on otherwise will use the default
	• branch will be made for a specific release and allows you to
	  ◦ maintain dev ver of repo separate to production code
	  ◦ test features before merging back to trunk
	  ◦ branch by task/individual/team

- tags	• label a particular revision • associate revisions with specific releases/milestones • identify code releases in history
- forks/clones	• branch alternative as an external copy • completely separate repo related to original repo and can be merged between • flexible, easily disposable or archived to another folder
What are the benefits of VC?	• Effective for distributed/outsourced teams + remote workers • east to fork (clone) a repo for backup + archiving • fearless experimentation • enables safe workflow • rollback to previous version • isolate problematic revisions • manage multiple versions via tagging, branching, forking • coordinate teamwork • back up • understand project history (who created + last edited) • continuous integration + delivery

What are the issues of VC?	• Difficult to undertand
	• performance degrades as projects increase in size
	• multiple people working on the same artefact
	  ◦ security
	  ◦ clashes in version history
	  ▪ parallel edits to different versions of the same doc set
	  ◦ traceability
	  ▪ bug added but difficult to return to exact point in which it was added
	  ▪ crossfile dependencies

WK 13 REVIEW Qs	• **Finding and using information**
	  ◦ accessing, assessing and using sources of information
	• **Collaboration:**
	  ◦ how this works in professional practice
	  ◦ project management and task allocation
	  ◦ working with Git
	  ◦ Git commands

WEEK 5: PROBLEM-SOLVING

	PROBLEM-SOLVING
Define "problem solving"	Problem = state of desire for reaching a definite goal from present condition Solving = management of problem in a way that successfully meets goals set for treating it Even though goal might not completely solve problem, does reduce it. you'll move back and forth between steps as you continue to work on since implementation is iterative with a focus on short cycles with planning + execution, testing + feedback (reviewing what worked and what didn't, what impact the solution did have) problems happen all the time. They present opportunities to improve the system and provides info for us to fix it
What is the process of problem solving?	Have a good process when approaching a problem 1. identify issues and define 2. understand different interests make assumptions clear, establish clear connection between business cases, project goals + outcomes, expand reqs → features specify requirements ◦ good ▪ don't diverge much from actual ▪ time to adjust project execution + still finish ◦ okay ▪ conflicting/missing core requirements ▪ not enough people consulted ▪ scope adjustment is a requirement not scope creep ◦ bad ▪ incorrect/incomplete/overoptimistic/unrealistic ▪ vague/solution-focused ▪ leads to scope creep + expensive projects ▪ 3. list possible solutions/options, brainstorming ideas 4. evaluate options (-ve / +ve) 5. select solution + implement 6. document agreements 7. agree on contingencies, monitoring + eval (conditions can change, how will you monitor compliance / followthrough, create opportunities to evaluate agreements, review)

How do you identify solutions?	Solutions are: • effective (meets goals) • efficient (affordable) • fewest side effects as possible (limited consequences from implementation) • make sure that you're identifying the solution, not just the symptoms Decompose to components (coupling/cohesion) based on • function • procedure • domain • role
What is the difference between COUPLING & COHESION?	Ideal programs should have high cohesion, low coupling • COUPLING ◦ measure of interdependence between independent modules ◦ BAD: high coupling ▪ inflexible, difficult to change ▪ would need to revamp whole system • COHESION ◦ internal elements of module functionally directed towards performing a single task – partitioned functionality ◦ BAD: low cohesion ▪ low reusability ▪ difficult to understand ▪ broad and unfocused leads to monolithic classes ▪ given module performs tasks not really related to each other
Differences between solutions at system and unit level	System = architect Unit = algorithm
Differences between algorithms + heuristics	Computer Science is the study of algorithms depends on if you need accuracy or speed • Algorithm ◦ step-by-step procedure that will always lead to the correct/best possible answ ◦ formal methods + proofs ◦ finite + predictable ◦ accurate/precise but time-consuming

Differences between algorithms + heuristics	• Heuristic ◦ general rule of thumb used for everyday situations based on past experiences ◦ quick but correct soln not guaranteed every time ◦ explores most likely option; tells you what to look for, not what to find ◦ utilises learning + discovery to reach a solution ◦ E.g. ▪ physician diagnosis ▪ looking for key attributes of virus
How would you develop an algorithm?	1. describe problem 2. analyse 3. develop high level solution 4. refine – add/remove detail 5. review
What are some ways to problem solve **- creative**	Word/visual association zero draft – what is known, what isn't/why superheros/superpowers/alteregos (impractical → realistic) forced connections reverse brainstorm (thinking of problems)
- visual	mindmaps (problem in centre, needs layer, solutions layer) storyboarding, groupsketching, flowcharts (seeing how ideas connect)
- idea-sorting	S.W.O.T S.C.A.M.P.E.R word bank (big groups of words grouped in themes, retroactively forming connections) 6 Hats = logic, emotion, creativity, devil's advocate, optimism, management
Are all problems solvable?	• P easy to solve • NP (algorithm) NP-hard / NP-complete truly difficult that need truly bad algorithms, unlikely to be efficient hybrid: ◦ easy + fast to check answer if given solution ◦ otherwise difficult to solve unless brute force

WEEK 13 REVIEW Qs	• **Problem Solving**
	  ◦ the different types of problems IT professionals are expected to solve
	  ◦ how do you approach a problem?
	  ▪ Understanding the problem
	  ▪ identifying solutions using different methods
	  ◦ Algorithms
	  ▪ how we represent them
	  ▪ how we develop them
	  ◦ Computational thinking

WEEK 6: INTELLECTUAL PROPERTY & COMMERCIALISATION

	INTELLECTUAL PROPERTY
Define IP **- why do we need to protect it?**	Ownership of ideas (how you can benefit) and control (choose distribution) the tangible/virtual representations of ideas using another person's IP may/may not involve royalty payments/permission but must always include proper credit to source • stimulates human mind for benefit of all by ensuring advantages derived from exploiting a creation (invention, literary, artistic works, ideas, info, knowledge, results/outcomes of research) that benefits creator • encourages publication, distribution, disclosure of creation to public • differentiates companies from competition with original concepts + ideas developed under contract becomes a company asset • extract value from IP + prevent others from deriving value = important responsibility since intagible + indivisible + can suffer from appropriation – unlimited # of people can consume without depletion
What are the general principles of IP Law **- Differences in ownership depending on who wrote code?**	Software Copyright Ownership protected as a "literary work" – sequence of instructions written to perform a specific task usage + ownership governed by any licencing agreements in place but doesn't cover titles/images • Belongs to employer if developer is hired • otherwise to original developer of code
In what ways can you protect IP?	Through IPR (Intellectual Property Rights) • confers certain rights / priviledges e.g. to exclude others from use (limits by scope, geography, duration) therefore has direct + substantial impact on industry + trade since can prevent manufacturer use/sale of product • encourages creation of wide variety of intellectual goods, economic incentives • Tradesecrets • Copyright • Patents • Trademarks
How can you protect IP via T.S.	Secrecy

- what can you protect	What can you protect • confidential information ○ e.g. secret formulas, processes, methods, what's used in production • any knowledge that gives businesses an economic advantage over competitors + customers
- caveats	Caveats: • doesn't stop anyone from independently creating same product + exploiting commercially • doesn't give exclusive rights + vulnerable when employees with knowledge leave your company • difficult to maintain over a long time • can't be registered like a trademark, proprietary responsibility to protect knowledge
- solutions for	Solutions: • when in public domain (patented) only release executables, not source code • keep out of public domain if long creation process • take appropriate steps to ensure secrecy (physical/electronic security, confidentiality agreements + policies, inventories of identifiable materials, written undertakings that will ensure employee will not compete with the business after leaving)
- example	- Coca-cola never applied for patent protection, thus not required to disclose formula
What is Copyright (C)?	Describes rights that creators have over their literary + artistic works, technical drawings, maps, films
How can you protect IP via C?	Whoever owns code has automatic copyright (NO NEED TO APPLY – the moment an idea/creative concept is documented on paper/electronically – original work of authorship) for 70 years after owner death / from year of first publication after death
- what rights does ownership of C entitle owner to?	Whoever has copyright is automatically entitled to certain rights others don't have • publish/adopt/communicate software to public • reproduce physically/with a computer (usual way of protecting software)

- what is a caveat of C	• varies from country
	  ◦ America/Canada: works must be fixed to a tanglible medium
	  ◦ Australia/France: more relaxed
	• owner's original expression of idea protected but not idea itself, still allows independent creation of original work

- define Free Use Exception
 	
- what are some examples of free use exception
	Copyright Act 1968:
	• exclusive rights to licence others in regard to copying work / performing in public / broadcasting / publishing / making an adaption
	• includes exceptions to infringements that allows uses of copyright material without permission
	can be done without permission of copyright owner, for a limited and "transformative" process
	
	• Dealing
	  ◦ Fair : commentary/criticism & parody
	  ◦ Flexible : used in teaching
	• Backups
	• Educational exceptions: used in classrooms

What do APIs define?	Application Programming Interface defines the way 3rd party programs interact with a system
How can you protect IP via Patent? **Types of patents** **- what does it protect / not protect** **- what are some requirements of P?**	Allows owner to have exclusive right/monopoly over invention, can decide if allow/licence/prevent others' usage + benefits . lasts 20 years Types of patents: utility, design, plant Protects: • technical solution/inventions (process/product that is new, inventive/not obvious, useful/industrially applicable) inventions = solution to specific technical problem • NB: if software is part of a process can be patented Doesn't protect • algorithms (sometimes) • abstract ideas Technical info needs to be described to the patent office (IP Australia) and distributed to public to enrich body of knowledge + stimulate innovation

- who is IP Aus	IP Aus • federal government agency responsible for granting rights in patents, Tmarks, designs, plant breeder rights
How can you protect IP via T? sign/symbol	Protects specific/distinctive brand that represents business' market position • colour, smell, logo, shape, word, phrase, letter, picture, sound, packaging, or a combination
- what are some caveats of T use	• distinct from name of company / web domain i.e. =/= IP just because you own one, doesn't automatically protect the other. But still important to protect because forms part of identity/reputation/brand. • must be actively used, or can be rescinded
- do you need to register?	Not necessarily (like copyright) but does help Registering a business, company/domain name doesn't give you exclusive rights like a registered IP does. If you register a business, company/ domain name don't automatically have the right to use that name as a trademark
What are licencing agreements?	• Legal written contract between 2 parties (licensor, licensee) by which property owner permits another party to use property under specific set of parameters + conditions e.g. geographical, time period, • delineates terms under which licensee may use property of licensor (allows brand protection) • widely used for commercialisatation of technologies
- what does it allow?	Allows others to access / use / benefit
- how does it vary?	In terms of: • rights granted ◦ use, copy, distribute, sublicence for set period of time + on certain conditions • liabilities accepted
- caveats	- IP e.g. patents, TM, copyright for written/visual art - post-IP ownership established (not software)
- name some examples?	• EULA (End User Licence Agreement) ◦ proprietary software ◦ contract between licensor + purchaser e.g. attempt to hold harmless the software licensor in the event that software causes damage to user's comp/data ◦ establishes purchaser's right to use software

- name some examples?
 	
- realistic examples
	• FOSS (Free and Open Source Software)
	  ◦ opposite to proprietary aims
	  ◦ free software (movement → software user's civil liberties) + open-source software (developmental methodology)
	  ◦ anyone freely licensed to use, copy, study, change software in any way
	  ◦ source code openly shares to encourage voluntary improvement of software design
	  ◦ creator still retains IP rights
	
	Benefits of FOSS
	• lower software costs
	• higher security + stability
	• high level of control of software functions
	
	e.g. companies that use computer software must typically enter into licencing agreements with copyright holders of software. Bargaining power of 2 parties depends on product

Why do licencing models matter? **What are the cons of models in general?**	• Allows original owner to receive benefits • easy entry into foreign markets • potential large return on investment • may lose intellectual property (licence period usually limited) • licensee can become a competitor (weakened control over tech because transferred to unaffiliated company)

What are licencing models?
 	
What are they and what are their caveats of the models?
	Application that needs a license + license file containing the info
	
	• Copyright (C)
	  ◦ basic rights
	  ◦ no distribution – proprietary developers take away freedoms
	  ◦ used by proprietary companies prevent individual use
	
	• Copyleft
	  ◦ protective, program needs to be free (freedom)
	  ◦ can distribute
	  ◦ need to retain rights owner specified
	  ◦ must place any changes made to software under copyleft)
	  ◦ add distribution terms (gives everyone rights to use, modify + redistrib program code)

What are they and what are their caveats of the models?	• Copycentre
	  ◦ permissive
	  ◦ can distribute for free/charge
	  ◦ can change rights but credit source
	  ◦ allows commercial organisations to use copycentered code, possible alter (add value) and sell it back to you
	
	• Creative Commons (CC)
	  ◦ enables free distribution (share, use, build on) of otherwise copyrighted work
	  ◦ provides author flexibility + protects people who distribute/use from concerns of copyright infringement as long as they abide by conditions specified in license by which author distributed work

	COMMERCIALISATION
Define commercialisation	Stage-wise process of managing / running something primarily for financial gain making available on the mass market launch – when, where, how (legal measures taken before product brought to market) 3 key aspects: • Ideation phase (funnel) : look at many sides to get 1 or 2 longterm sustainable ideas (product, price, place, promotion) • Business process stage: each stage has own key goals + milestones • Stakeholder stage = vital to include key stakeholders early e.g. customers

What is the language of commercialisation?	• Software, accelerators, incubators, R&D, tax incentives
	• business plans + models, beta products, value proposition, pitches
	• (seed / crowd) funding, angels, investors, sourcing
	• licencing, policy hack, confidentiality, outsourcing agreements
	due diligence
	  ◦ check validity of data
	  ◦ have in writing they have no claim over data
	• markets, launch, adoption
	• NDA, IPO, technology transfer

What are some business models and their examples?	• Brokerage arranges transactions between buyers + seller for commission when deal is executed ◦ marketplace ▪ Freelancer ◦ virtual marketplace ▪ Amazon ◦ auction ▪ eBag ◦ transaction broker provides limited representation to both buyer/seller ▪ Paypal • Advertising ◦ portal ▪ Yahoo ◦ classifieds ▪ Gumtree ◦ user registration ▪ SMH • Infomediary internet company that gathers + links on particular subjects on behalf of commercial organisations + potential customers ◦ advertising network ▪ Doubleclick ▪ Google ◦ audience management ▪ Nielsen ◦ incentive marketing ▪ Scoopon • Merchant person/company involved in wholesale trade – supply goods i.e. particular trade ◦ click-and-mortar online + offline operationes (website + phys store) ▪ Woolies ◦ virtual ▪ Amazon ◦ bit vendor e-tailor only dealing in digital goods + services ▪ iTunes • Manufacturer person/company involved in wholesale trade, supplies good for a particular trade ◦ any computer company ▪ Dell

What are some business models and their examples?	• Affiliate officially attach and connect (a subsidiary group/person) to an organisation ◦ pay-per-click ◦ revenue sharing • Community ◦ open content ▪ Wiki ◦ social networks ▪ Flickr • Subscription arrangement to receive something, typically a publication regularly by paying in adance ◦ content services ▪ Netflix ◦ networking ▪ classmates • Utility state of being useful, profitable/beneficial ◦ metered subscription ◦ metered usage ▪ ISPs : pay per use internet providers consumers have access to potentially unlimited resources but only pays for what they actually use
WEEK 13 REVIEW Qs	• **Intellectual Property and Commercialisation:** ◦ different forms of protection for intellectual property ◦ forms of commercialisation

WEEK 7 : ETHICS

	ETHICS *"A man's destiny is the sum total of all the decisions that they take over the course of his life"*
Define ethics ETHOS = custom/habit	Branch of philosophy, a system of moral principles concerned with decisions, actions, judgements (right/wrong, good/evil, virtuous/non-virtuous), defines what's good for individuals/society, provides guidance + common standards to promote consistency what life is worth living based on values, principles, purposes
How do ethics differ from morals?	Ethics: • rules of conduct accepted within a social context, put forward by organisations to unite many people with different morals • map of how to make choices • developed purposefully over time Morals: • individual principles of right/wrong that guides personal behaviour • inherent truth ; felt intuitively
Why are ethics important?	- satisifies basic human need of fairness, honesty - creates credibility - unites people + leadership - long term gains of securing society - improves decision-making when driven by values
Define "ethical pluralism"	versus "value monism" Different ways of deciding what is/isn't ethical may be incompatible and/or incommensurable with own / conflict with each other but all equally fundamental + correct
Define "teleological ethics" **- egoism**	"state of the world" E.g. a person should be killed if it saves 2 other innocent lives Egoism • "what's best for me" • individual situation • but consider flow-on effects – how others will react, how you will respond

- utilitariansim	A consequentialist philosophy • "greatest good for the greatest number" • good specified as outcome • total situation of total people (good of others + self) • how is the situation weighted, who is included

Define "deontological ethics"
 	
- W.D. Ross' 7 Prima Facie
	"Science of duty"
	"do what is right though the world may perish"
	Actions themselves will decide as opposed to the consequences or the characters of the people themselves. One action will always outweigh the other
	Ethical behaviour follows/guided by system of rules
	commitment to moral principles without regards for ends
	E.g. if killing is wrong, it remains wrong even if it could save lives
	
	Deontological ethics separated into
	• universalism
	  ◦ rules apply to everyone, under all circumstances
	• relativism
	  ◦ rules apply to people under certain circumstances, within certain traditions
	
	• fidelity
	• reparation
	• gratitude
	• justice
	• beneficence
	• self-development
	• non-injury/maleficence

Define "contractarianism"	People are primarily self-interested, and actions will be to maximise self-interest which will lead to moral actions since moral norms are determined by maximisation of joint interest basic rules = non-violence implies in turn that each is to have as much liberty as is possible for all with no more for any individual / subset. We simply do better to live under mutually advantageous laws
Rules of professional conduct of ACS	1. primacy of public interest 2. enhancement of quality of life 3. honesty 4. competence 5. professional development 6. professionalism

Rules of professional conduct of ACM/IEEE-CS	1. public 2. client/employer 3. product 4. judgement 5. management 6. profession 7. colleague 8. self
Define "corporate ethics"	Basic level of trust exists between consumers + various forms of market participants with businesses <u>Corporate Social Responsibility (CSR)</u>: when law fails, ethics may stop organisations from harming. Corporates responsible for the impacts of their actions/decisions on: • society • environment ◦ ecological ▪ waste + pollution reduction processes ◦ social ▪ contributing educational + social programs ▪ earning adequate returns on employed resources
What are the "bottom lines"	• Profit [what] • People [how] • Planet [how] • Purpose (progress, principles) [why] Fourth bottom line = return to one's spiritual self improving lives can be a factor valuable enough to rival other business objectives due to being a key motivating factor for any business to continue
WEEK 13 REVIEW Qs	• **Ethics:** ◦ how do you know right from wrong ◦ how do you judge other people/companies/groups ethics ◦ theories/scenarios/case studies ◦ A broader look at ethics in Computer Science ◦ Professional conduct ◦ scenarios / debates

WEEK 8: SYSTEMS THINKING & APIs

	SYSTEMS THINKING
Define systems thinking	How parts of a structure interrelate to form an overall system see things as a whole holistic approach to analysis that focuses on the way that a system's constituent parts interrelate and how systems work over time & within context of larger systems + recurring patterns in the relationships between subsystems systems theory greatly influences how we understand + change organisations – requires merging of multiple perspectives + sources of info + deals with complex systems in tech, society and science an iterative learning process in which one tales a broad holistic long-term perspective of the world + examines the linkages + interactions among its elements strategic value analysis based on • factors – int / ext • trends • causal effects help deliver broader benefits + move organisation to higher levels of thinking + performance e.g. biological cell – how complexity arises out of simple multiplicity
Define the different levels of systems thinking	- specific: system components and how they interact with each other - broad: how systems interact with the wider society
Why is predicting a system's behaviour difficult?	Unforeseen consequences can arise from unknown interactions how dysfunctional behaviours result from interactions among parts of system over time. Examine potential consequences of proposed interventions + recognise impact of time delays + feedback. Leads to better assessment + more effective actions than linear thinking Parts can only be managed by understanding how they interact
Give examples of other forms of systems thinking	Design, linear, critical, process, traditional, strategic, reductionism

What is traditional thinking?	Breaking the system down into its individual components, optimising each one and putting it back together but doesn't necessarily lead to an improved system causality is an ongoing process, not a one-time event with "effect" feeding back to influence the causes + the causes affecting each other
Define Tragedy of Commons	If an individual begins to reap the resources, will eventually cause irreversible damage to all what happens when individuals act in their own best self interest + ignore what's best for the whole group when using a shared group resource. Results in collapse of the reseource over whioch they're competing. e.g. even if herder receives benefit from additional cow, damage to the commons shared by all herders involved. Neglecting the wellbeing of society for personal gain. Therein is the tragedy – each man locked into a system that compels him to increase his own without limit in a world that is limited, and in a society that believes in the freedom of the commons overconsumption, underinvestment + ultimately depletion of the resouce. resource must be • scarce • rivalrous in consumption • nonexcludable
Solutions to Tragedy of Commons	• Provide more info to lower uncertainty about future • ensure people's needs for strong social identity + sense of community • able to trust institutions we put in charge of our commons • incentives to improve selves and others + punish overuse
What do systems behaviours stem from? **- define them**	*The world reacts to our interventions* *Our actions alter the environment,* *and therefore, as an unanticipated side effect,* *the possibility of actions we can take tomorrow.* • Feedback loops positive (self-reinforcing) negative (self-correcting)

- what are the issues of these characteristics? **- what are examples?**	○ issues: ▪ time delays • short + long-run impacts are different • cycles take a long time, slow accumulation of evidence • identified problem might be part of unseen interconnected network of other issues. They're not in isolation. Causes instability • Emergent behaviour ○ differing uses to what designer intended because it is adapted to another use. Therefore fails original purpose ○ unpredictable, system behaviour depends on interrelationships between individual parts ▪ health issues of reusing disposable plastic bottles ▪ stockmarket ping system meant to aid companies hijacked by hackers to cause DoS
Why is systems thinking so important? **- examples**	Enables organisations + individuals to take full advantage of any element within their system + identify solutions that address as many problems as possible positive effects of solutions leverage improvement throughout system since focused on wider context + appreciates interactions between different components Actions have unintended consequences + complex systems can fail from reinforcing effects • wolves vs sheep • farmers + cattle • Borneo mosquitoes • man-made El Nino effect • Therac-25 (race conditions)
What is the purpose of systems thinking?	• Maintain larger picture pespective ○ who is the system made for ○ how is used, how can it be used ○ implications of usage: ▪ problems ▪ emergent behaviours • study susceptibilities + failure points to identify positive / different solutions to effect change • simplify seemingly complex things into various simple components (simpler solutions) • understand that different components might fail

	APIs
Define APIs	Application Programming Interface • clearly defined set of methods describing how various components interact and communicate • design processes + workflows using components • set of functions + procedures allowing creation of applications that access features/data of an OS/application/other service • Allow different technologies to work together regardless of language/platform ◦ API separation from implementation can allow programs written in one language to use libraries of another (usage varies depending on type of programming language involved) ◦ can specify interface between application & OS remote APIs allow devs to manipulate remote resources through protocols, specific standards, for communication. E.g. like electrical sockets • standard interface – any compatible consumer can easily outsource requirements ◦ devices can easily move frrom one portal to another • layer of abstraction – hides specifics to underlying sevice • similarly transparency works both ways – to the service all consumers look the same on the other side
What do developers use APIs for? **- example and its use** **- caveats**	As building blocks to create new systems via API integration to improve business efficiencies simplifies programming by abstracting the underlying implementation + only exposing objects / actions developer needs describes + prescribes “expected behaviour” of set rules. Single API can have multiple implementations (or none, as an abstraction) in form of different libraries sharing same programming interface If This Then That (IFTTT) chains of simple conditional statements that occur within multiple webservices need to take care of 3rd party systems that interact with your own (security / IP)

What's involved in the process of an API?	1) You share an asset on an API 2) API provides universal access to other developers who use to create Mobile + Web apps 3) data + software (assets + brand) become more valuable as leveraged by ◦ 3rd party access services ◦ partners ◦ developers 4) enriched experience for end users leveraging data + services of other apps
How should companies approach consumers in terms of systems thinking?	• Keep decision-making user-centric ◦ what are users aiming to achieve ◦ how do users want to engage with sevices ◦ how are users actually engaging with services (in conjunction with other services) ◦ how can we best serve their needs + make things as easy as possible • what are the implications of hardware + software merging
WEEK 13 REVIEW Qs	• **Systems Thinking:** ◦ Predicting a systems behaviour ▪ break down to simple components ▪ evaluate within context ▪ take into account feedback loops and time delays ▪ complexity and emergence ◦ Difference with other forms of thinking ◦ Application in IT ▪ Designing systems components ▪ Designing processes and workflows using components ▪ User-centric thinking

2019 EXAM STRUCTURE:

- 4 questions / 100
  short answ = 3 – 4 lines
  long answ = 15 lines
  - General knowledge : 10 x short answ (4 marks each = 40 marks)
  - Nature of Majors : 2 x long answ (10 marks each = 20 marks)
  - Tools and techniques : 2 x long answ (10 marks each = 20 marks)
  - Professional skills : 2 x long answ (10 marks each = 20 marks)

THINGS TO STUDY:

- professional associations and their role
- types of tools
- information and its reliability
- requirements
- communications
- ethics frameworks
- intellectual property
- patents, copyright etc
- problem solving, algorithms, heuristics, solvability
- systems concepts
- different majors and their characteristics
- Latex
- Git commands

the internet is reliable because there's a large group of people to check (F)
algorithms can only be protected by tradesecrets | algorithm vs software | liability – week 7

computational thinking

- algorithmic thinking = getting to soln through a series of steps
- evaluation = process that ensure the algorithmic thinking is good
- decomposition = breaking down probs into smaller ones to make it easier
- abstraction = hiding detail / removing unecessary complexities
- generalisation = a quick way to solve new problems based on the successful solution to previous problems

trademark doesnt always have to be registered to be valid. Like copyright, is automatic only patents need to be registered

MAJORS

MAJORS	CHARACTERISTIC
COMPUTER SCIENCE **- general overview**	GENERAL: • Learns about algorithms and solutions to problems with computation • programming in different coding languages, such as Python, Java etc. to solve and show those problems • major with the least amount of interaction, as opposed to Information Systems. However, it is still necessary to have those communicative skills • A computer programmer will be hired to create a program and/or software using code, and then test it and renew to improve or develop applications for companies and users • underlying theories + algorithms • computation, procedural knowledge about how we can methodically calculate and operate on information • broad approach to the study of principles and use of computers that covers both theory + application. Involves understanding and application of both abstract + concrete knowledge FUTURE CAREERS: • computer programmer, applications developer, game designer/developer, multimedia programmer, computer systems administrator, systems analyst, data scientist etc. • quite flexible, as in people with a degree in mathematics, or even those in a different major can take on jobs of a computer programmer etc., where these skills can be learned on the job. • start off in a junior position before moving on to a senior role or even manager/leader of a team. SKILLS: • good problem-solving abilities – process of coding will produce errors and as such, one must be able to find and solve these errors through intense concentration for potentially many hours, thus requiring a strong will and the ability to find and solve issues with the code • flexible to change and others. The world of technology is constantly evolving, and to stay on top of it, one must evolve with it – willingness and desire to expand knowledge, learning new coding languages, using new technology and taking in colleagues' and users' opinions and perspectives.
- CD vs DS **work with lots of data sets**	FUTURE:

- CD vs IS **Information Systems focuses more on management, and communicating with clients and colleagues as does Software Development.**	• New and improving technologies e.g. augmented reality, smarter cars and AI implying new equipment that one must get used to, and also seeking ideas to improve them
COMP. DATA SCIENCE **- general overview** **- DS vs IS** **- DS vs SD**	GENERAL • Develops skills in data science and computing • analyse and deal with large or complex data sets • the ability to harness these data sets, building systems that assist in data-driven decision making • major focuses more on roles related to data science and analytics vs pure programming like other three. These roles, require not only the programming skills all the majors develop, but also the statistical thinking skills this major develops • analysis + management of data FUTURE CAREERS: • data engineer and data scientist • heavily teamwork based, having a focus in collaboration between individuals with different and distinct expertise to work towards the company's goals • individuals spend their time communicating with stakeholders, researching and cleaning data, and also analysing and creating data visualisations of data. • Expected career progression is starting from a junior data analytic or engineer role to a senior data analytic or engineering data manager role • It is expected to continue this way, due to the field's interdisciplinary nature and the way data is becoming more entrenched in every industry from finance to science FUTURE: • new and improved cloud and AI services emerge in the next few years, causing automation of the data collection and cleaning process, graduates can expect to spend less of their time cleaning and sifting through data and more time analysing and using data for innovative purposes • greater need for graduates with strong analytic skills appropriate for what field they work in, meaning graduates will thus need to gain more education in the field they are working in for example finance to be able to effectively develop real world solutions in their field

INFORMATION SYSTEMS **- general overview** **- IS vs SD**	GENERAL: • Develops an understanding of both people and systems within a business context • more focus on the client and knowledge of how computers can work within the broader socio-technical lens in a way that best serves the organisation's needs • use tools to collect and quantitatively analyse data. • understanding of system development and implementation, strategic planning, as well as end-user needs, and how individuals use and are impacted by computers. • Design + management of business solutions • wide range of industries as their skills are relevant to many different professions • capabilities in testing, debugging and programming in various languages. They may also choose to pursue additional education such as a Masters or diplomas to gain more experience within a specific domain • diverse areas as actuarials, computer or information research scientists, computer network architects and database administrators • Background may be technical, and require abilities such as logical thinking and problem-solving • role will characteristically be more client-facing and interpersonal. Much like all IT professionals in future, graduates must be competent in both hard and soft skills as they communicate with clients internal and external to the company. • system analysts - individuals who evaluate an organisation's needs and develop cost-effective computer systems that are tailored to increase business efficiency • work independently as freelance consultants with other IT professionals on a contractual or project basis, or be employed within a consulting firm. At times they may be required to travel for short periods for projects or clients. • obtained one to five years of prior professional experience in IT service or analysis in the public domain. entry-level roles like a system administrator/ maintaining a professional portfolio or taking certified courses from companies such as Microsoft and IBM. Regardless, key skills can include system administration, project management, and SQL • likely be handling the system as a finished product as opposed to its backend development or architectural

	creation such as those with a Computer Science or Software Development background • will not be as deeply as involved with the data as a person with Computational Data Science knowledge • characteristically more client-facing, analyse commercial needs to tailor an efficient system best suited to those needs
SOFTWARE DEVELOPMENT **- general overview**	Builds software systems process of designing solutions application of engineering processes to the creation, maintenance, and design of software for a variety of different purposes • good at professional skills like software project management, software designing, testing and maintenance • designing software to meet client requirements • Developers have to communicate with clients to know their needs to design software that meets those needs and explain to customers how the software works and answer any questions that arise instruct others • programming, designing, testing and software building, are the basic skills for a professional software developer • Analytical skills are also needed as software developers must analyze users' needs • Software developer must work well with colleagues, as a part of a team with interpersonal skills • self-study ability, every requirement from client is different, software developers always need to study new coding language to help them to complete these requirements • creativity, developers are the great minds behind new computer software

GITHUB	

What is Github **- how does it fit into the professional workplace?**	Method for collaboration + version control maintenance
Github commands	gitclone gitinit gitdiff gitlog = gitstatus gitcommit
LATEX	
What is Latex?	Document markup system specifically for technical and scientific writing
MARKUP	
What is Markup?	A computer language that uses tags/annotations to define elements within a document. Distinguishable from the text itself annotates text so that the computer can manipulate it - human-readable (standard words vs typical programming syntax) - e.g. HTML and XML these are saved in a plain text format, viewable in a standard text editor
	OVERLEAF provides a WYSIWYG (what you see is what you get) overlay, browser-based editor that allows direct code editing and templates - supports math and science students with discipline-specific writing needs - supports collaborative research teams who needs to write together
Begin document	\documentclass{article} \begin{document} text \end{document}
Insert images	File upload a file from computer dropbox \begin{figure}

GIT CHEAT SHEET

GETTING & CREATING PROJECTS

`git init` Initialize a local Git repository

`git clone ssh://git@github.com/[username]/[repository-name].git`
Create a local copy of a remote repository

BASIC SNAPSHOTTING

`git status`	Check status
`git add [file-name.txt]`	Add a file to the staging area
`git add -A`	Add all new and changed files to the staging area
`git commit -m "[commit message]"`	Commit changes
`git rm -r [file-name.txt]`	Remove a file (or folder)

BRANCHING & MERGING

`git branch`	List branches (the asterisk denotes the current branch)
`git branch -a`	List all branches (local and remote)
`git branch [branch name]`	Create a new branch
`git branch -d [branch name]`	Delete a branch
`git push origin --delete [branch name]`	Delete a remote branch
`git checkout -b [branch name]`	Create a new branch and switch to it
`git checkout -b [branch name] origin/[branch name]`	Clone a remote branch and switch to it
`git checkout [branch name]`	Switch to a branch
`git checkout -`	Switch to the branch last checked out
`git checkout -- [file-name.txt]`	Discard changes to a file
`git merge [branch name]`	Merge a branch into the active branch
`git merge [source branch] [target branch]`	Merge a branch into a target branch
`git stash`	Stash changes in a dirty working directory
`git stash clear`	Remove all stashed entries

A good version control system will include a complete change history of every file

CVC – centralised version control
DVC – distributed version control