Level 4 project, part I

The Level 4 project

Mind map

- **Task**: draw a mind map of your project as it stands. Just draw out the ideas that come to you. If there are bits that are fuzzy, indicate that.

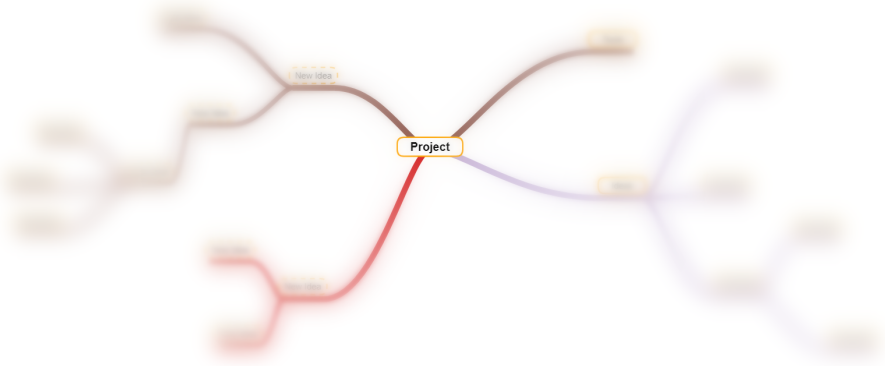

Figure 1: A mind map

Why are we here?

Why are we here?

- To make your project experiences easier.
  - Help you navigate pitfalls, set realistic expectations, lower stress and manage the process more effectively.

- To increase the quality of projects.
  - Stronger dissertations, better products, improved professional conduct, less wasted time.
- It's really not very fair to throw you in with no guidance at all.

- and so I don't have to say the same things over and over again.

- Individual project: each student works on a independent topic.
- Credits: 40 credits; 400 hours work; 33% of Level 4, 19.8% of Honours grade
- Interaction: 30 min. meetings, weekly, x 24: 12 hours per student
- Output: 40 page dissertation: 1 page per credit
- Marking: Independently double marked and reconciled based on dissertation
- 85% dissertation

- 10% professional conduct / 5% presentation

Today's objectives

After today, you should:

- understand what the individual project is about;
- know what is expected of you;
- know how projects typically run;
- know how to use supervisions effectively;
- have an idea of how the project is assessed and be able to assess a project;
- understand what we expect in terms of professional conduct and how to meet this standard;
- know how to avoid common pitfalls in individual projects;

Part I

- Learning outcomes and path of a project
- Assessing a project
- Common pitfalls

Part II

- Professional conduct: tools, techniques and good practice
- December status report
- How to research literature
- Beyond the project

December session

One further L4 project training session will be run in the first days of Week 11, during the project period. This will cover:

- How to approach the evaluation
- How to write a high-quality dissertation
- Doing a good presentation.
- Making video summaries.
- How to take your project further: portfolios, jobs, startups, PhDs, etc.

Why do an individual project?

The individual project is an important part of the degree.

This is one of the few opportunities you may have in your life to dedicate significant time to a CS project on your own.

In some cases, it is the start of a career, a PhD or a company. It is **not** a failure if this doesn't happen, but it can set you up for future success.

Professionalism

You will be a future cadre of professional engineers and scientists. In the project, you can expected to be treated as such.

The project

- gives you a chance to work on something that is **yours**. You can define what it is and how it will evolve.
- lets you work independently and take responsibility for project management;
- gives you experience in taking on board advice from a senior colleague;
- exposes you to a large (ish) project, that required planning and discipline

- lets you explore ideas, tools, algorithms and concepts that might not be covered in a CS degree
- gives you a solid, completed piece of work which you can demonstrate to employers;
- gives you a chance to develop personal connections with academic staff.

Learning outcomes

What should you be able to do after completing the project?

Students should be able to

- Manage and organise a large independent solo piece of work
- Choose among technologies, tools and approaches and apply them effectively
- Synthesise technical skills appropriately to address a single challenging problem
- Develop a substantial product which demonstrates technical achievement in computer science.
- Appropriately evaluate the outcome of their work

Students should also be able to

- Conduct themselves in a professional manner, taking responsibility for the progress of the project, recording their time and effectively using their supervisor's input
- Write a coherent, literate dissertation that documents the project, justifies decisions made and sets their work in context
- Present their work orally to a technical audience, communicating their project precisely and fluently

---

Learning outcomes [simplified]

- Think carefully (read, analyse, design)
- Build something showing technical excellence (develop, research)
- Do it professionally (plan, manage, document, coordinate)
- Show that it works (evaluate, analyse)
- Write it up (articulate, summarise)

You're responsible

The individual project is your work. Responsibility lies with you. Your supervisor provides some limited support, but you will have to manage and organise your own work.

Your supervisor can give you guidance and direction, but you must demonstrate that you are independently leading the work. Make good use of your supervisor – they will have limited time available, so make sure you prepare well for meetings and take on board guidance offered.

You have a supervisor, not a boss or a teacher, and your supervisor is not responsible for your work.

How to make the most of the experience

- Make a whole-year plan early.
- Use documentation strategies: minutes, questions, status, weekly plan, time log.
- Spread work evenly throughout the year.
- Focus on the right things: the outcome is a dissertation, not a product.
- Learn how to get the most out of your supervisor.

The School's project philosophy

There are various ways projects can be run. In Glasgow we have a distinctive philosophy. There are three principles:

- **Freedom**; we support projects on many different topics, and many different styles of supervision. You have to hand in a dissertation and do a presentation. Other than a short status report, there are no other formal requirements.
- **Fairness**; we are careful to allocate, supervise and assess projects fairly, with algorithmic allocation, double blind-marking and published marking criteria.
- **Professionalism**; we expect and require you to behave professionally: pro-actively, independently and efficiently.

Projects are individual

Everyone will have a different experience. Following the principle of **freedom** there are many styles of project and styles of supervision.

- Some projects may be pure software engineering in a classical sense (requirements, design, implement, document).
- Some might be pure research (investigate an idea, run experiments, analyse and write up results).
- Many more are hybrids of these.

What is the role of a supervisor and a student?

You will have a supervisor who will have personal meetings with you throughout the project. This can be a valuable opportunity to get to know staff personally. You should be clear about what the roles of supervisor can be.

Everyone will experience different supervision styles. There are many styles of supervision, and some supervisors will fulfil some roles more than others. Some supervisors might debug your code, while others will never look at it. Some might act as a client, setting demands for the project and letting you satisfy them, while others might expect you to set your own demands.

Supervisor roles

- **Technical expert**. The supervisor provides technical assistance, for example suggesting relevant technologies to apply.
- **Client**. The supervisor acts as a project "client", who sets out a specification and tasks the student with completing it, and who makes demands as a real client would to contractor.
- **Reflector**. The supervisor helps students reflect on issues that have arisen and organise them into tasks that can be achieved. This is a fancy way of saying "expensive rubber duck".

- **Assessor** The supervisor is also the primary marker of the dissertation, and uses observation of students during meetings to inform assessment. This is particularly relevant for professional conduct assessment.
- **Guide**. The supervisor provides guidance on how to tackle the project, common problems that are encountered and best practices for completing the project. In this "sea dog" role, the supervisor passes on their accumulated experience to the students.
- **Counsellor** The supervisor provides pastoral support for general wellbeing, checking on stress levels, progress in other courses and mental health.

Student roles

- **Developer** you will generate code, solve problems and get "work done". But simply working on the project isn't enough – you must lead the project.
- **Researcher** you will need to bring new ideas to the table and proactively research possibilities. Your supervisor might help you shape these or select them.
- **Project manager** you must organise your own time, access to resources and document/monitor your progress.

- **DevOps** you will need to organise and operate your own DevOps; setting up version control, using appropriate tools, configuring environments for your project. No-one will do this for you.
- **Slave to LaTeX** oh god why won't that figure go on this page?!?!?

Supervisor questionnaire exercise

- Take one questionnaire
- Fill it out
- Your supervisor will (hopefully) have also done so.
- Compre your results. This will help you align expectations

What is the project?

Typical path of a project

Summer

- Summer: light background research.
  - Outcome: **Understand background literature, tooling, languages, etc. that you will use**

Semester 1

- Week 1: full-time project preparation week

  - Outcome: **A concrete plan, and initial start at the project**
- Week 2-10: term weeks, steady development
  - Outcome: **An initial version of the project product**
- Week 10-12: project weeks, full-time project work
  - Outcome: **A refined, ideally near-complete product, and a status report.**

Semester 2

- Week 13-23: term weeks, steady development
  - Outcome: **An evaluation of the product, and a complete dissertation**
- Week 24, Wednesday: dissertation + code hand in

- Week 24, Friday: presentations
  - Outcome: **Free beer at the Level 4 party**

A rough timeline

Term time goes much faster than you expect.

- **January** You want to have a fairly complete "product" (for whatever that means) by the end of Semester 1. It may be refined and developed further in January.
- **February** By February you should be starting an evaluation and preparing for the write up. There may be minor tweaks and improvements to the product.
- **March** should largely be dissertation writing and presentation preparation, and finishing off evaluation analysis.

What happens at different stages of the timeline be different for different people. Don't worry if your classmates seem to be doing something very different from what you are doing. Do worry if you haven't got a working prototype yet and all your classmates are writing up.

Submission week (PROVISIONAL)

Monday/Tuesday March 23

- Presentations
- 15 minute slots: 10 talking, 3 question, 2 handover

Friday March 27

- Submit code + dissertation
- Submission on Moodle

Research versus engineering

There are two broad styles of projects. **Either would be possible for software engineers or computer scientists**. Most projects involve some elements of both.

Engineering style: product focus

Output should be a significant software artifact (strong design, programming and testing focus). The project will focus on the development of deliverable software, rather than answering a research question. **You must still be able to discuss and argue your development logically**. You must show evidence of abstract thought and reasoning.

Research style: scientific focus

Output should focus on a research contribution (strong analysis, often empirical work). The project will focus on answering a research question. **Some form of software development is almost invariably still required**. You must be able to demonstrate technical excellence.

CS versus SE

As you've already chosen your projects the topic choice is moot; but you should be aware of how your project will be judged. In practice, these distinctions between CS and SE are relatively fuzzy.

- Computer science students should do projects to do with computer science. This is broad, and could include the very practical (build a web server to improve kidney donations) to the very theoretical (prove the soundness of a subset of Rust's type system).

SE projects

- Software engineers should do projects that involve software engineering. This could either be:
- **Research style** address a research problem connected with the practice of Software Engineering. This definition can and should be interpreted very

broadly. For example, *"a new method for managing user stories across distributed teams using augmented reality technology"* or *"investigation of novel techniques for discovering security vulnerabilities in code bases"*

- **Engineering style** focused on the delivery of a production quality software product. Product-led software projects must have an identified customer with a real-world need. For example, "build a system to visualise Hunterian museum artifacts on the HTC Vive".

How is it assessed?

- **5% Oral presentation** (10 minute presentation)
- **85% Dissertation** (40 page report)
- **10% Professional Conduct** (supervisor observations)

Note that virtually all of the marks are awarded for the dissertation. **No marks at all are directly awarded for the product itself** – but you must produce a significant product to be able to write the dissertation.

Your dissertation is the evidence used to assess your work. You must both demonstrate technical mastery (by doing something impressive) *and* be able to write that up in order to complete a good project.

Assessment

It's important to understand how the project is assessed before starting work.

- You are marked (largely) on your dissertation, which is a reflection of your product.
- Great product, poor dissertation: poor mark
- Poor product, great dissertation: poor mark *[you can't polish a turd]*
- Your dissertation must be marked on the basis of the evidence presented.
- **There's no point in doing development work you will not be able to write up**

Dissertation assessment criteria

- **Analysis [15%]**
  - Clarity of thought; precise formulation of problem; understanding of background and context.
- **Product [40%]**
  - **Research** Quality of the research, innovation, rigour in the way it is conducted.

  - **Software engineering** Software design, implementation, and documentation.
- **Evaluation** [**10%**] *Thorough and fair evaluation of results, scientific analysis, effective summarisation.
- **Dissertation** [**20%**]
  - Completeness and coherence, organisation, literacy, bibliography, use of figures and diagrams
- **Professional Conduct** [**10%**]
  - Engagement with supervisor; independence; time management; tool use
- **Presentation** [**5%**]
  - Structure, clarity, fluency, question handling and video quality.

Your supervisor may choose to adjust any of these weightings by up to 5% at their discretion, *except* professional conduct and presentation, which are fixed.

Marking process

- Every project is **double marked**. The first marker is your supervisor. The second marker is another member of staff. *You will not find out who the second marker is before submission.*
- Markers assign a band for each category above, along with a written justification for the marking of each category. An overall grade is automatically computed.
- This is a blind process. Neither marker can see the others grading until they have committed the final mark.
- These marks are reconciled. If they are exactly the same this is easy. There is a formal process otherwise.

Grades

- The project is treated as an Examination and you will only ever see the final reconciled overall grade.
- You may ask your supervisor for comments and feedback after grades are published.
- Neither your supervisor nor anyone else has power to alter grades agreed by an Examination Board, and supervisors will not discuss how a particular grade was arrived at.

Reconciliation and Arbitration

[excerpt from the Assessment Criteria]

First and second markers bands may differ. Where this happens:

- If marks differ by 0-2 bands, the supervisor's overall mark will be taken as the final mark. *[The majority of projects fall into this category]*
- If marks differ by more than 2 bands, the supervisor and reader must confer to reconcile their marks. If they are unable to agree, the projects coordinator will invoke arbitration.
- If marks differ by more than 4 bands, arbitration **should normally** be invoked. For a project which receives a reconciled grade of a less than D3 (fail) arbitration **must** be invoked.

- The projects coordinator (me) has absolute discretion to invoke arbitration for any other reason, for example: where the two marks fall on either side of a critical borderline; where grades of A1 or A2 are awarded; where one of the markers requests it; or where the agreed mark seems unreasonably high or low; where there is any question of collusion of markers.
- Arbitration entails engaging a third marker, who marks the projects independently. All three markers confer to decide the final mark.
- In cases of unresolved dispute, the projects coordinator has final authority to assign a mark, which will typically be the median of the three markers results unless extraordinary circumstances prevail.
- All markers must agree on a joint text justifying the reconciled grade and enter this into the system.

Grade descriptors

A1-A5 Excellent (First-class work)

- **Analysis** The problem analysis is excellent. The survey is comprehensive. The approach is definitely feasible.
- **Product** The product is extremely well designed, implemented, and documented.
- **Evaluation** The evaluation is extremely thorough. There are excellent suggestions for further work.
- **Dissertation** The dissertation is complete, very well organised, very clear, and highly literate.
- **Overall** An excellent project. Few errors. Shows good judgement and skill in the methods used.

A1 or A2 indicates a truly outstanding and challenging project, definitely worthy of wider dissemination and/or a contender for a national prize. Very few A1s are awarded (some years none at all).

D1-D3 Adequate (Bare pass)

- **Analysis** The problem analysis is adequate. The survey is patchy. The approach is just about feasible.
- **Product** The product is adequately designed, implemented, and documented.
- **Evaluation** The evaluation is just adequate. There are unconvincing suggestions for further work.
- **Dissertation** The dissertation is partly complete, not very well organised, clear in parts, and often less than literate.
- **Overall** A fair project. There are many flaws but the overall performance is satisfactory.

F1-F3 Poor (Poor fail)

- **Analysis** The problem analysis is confused. The survey is poor. The approach is ill-conceived.
- **Product** The product is badly designed, implemented, and documented.
- **Evaluation** The evaluation is inadequate. There are scant suggestions for further work.
- **Dissertation** The dissertation is scrappy, disorganised, unclear, and less than literate.
- **Overall** A poor project. There are major problems but also signs of some work.

Mock assessment

Structure of task

- You have copies of five dissertations. In groups of five or six, quickly discuss the reports.
- Come up with a rough ranking of the reports
- Then, decide on what grades you would assign the reports.
- You have **twenty minutes** to do this. This is much shorter than a real marking exercise, but enough time to make a decent stab.

- Each report bundle has the marking scheme attached, and an ID number (1-5) printed on it.

Ground rules

- These are reports from students who have agreed that their work can be shared for academic purposes.
- Most important: **be respectful**. Please do not mock other's work.
- These reports must not leave this room. Hand them in as you leave.
- I have done my best to anonymise these reports, removing names and project titles.
  - If you know who the project belongs to, **KEEP IT TO YOURSELF**. Do not discuss or speculate on authorship.
- I will not give the real grades for these projects. I will let you know how *roughly* how your scoring matches.

Grading

- There are five projects, which range across a wide spectrum of overall grades.
- Possible grades are on the standard 21 point scale:

  A1 A2 - A3 A4 A5 | B1 B2 B3 | C1 C2 C3 | D1 D2 D3 | E1 E2 E3 | F1 F2 F3 | G1 G2 | H

classresponse.gla.ac.uk / #363

Respond via YARCS, in **PRECISELY** this format:

> C1, A2, D3, F3, A1

if you think the project 1 should be an C1, project 2 and A2 and so on.

Everyone should vote. You can just vote with your group consensus, or decide a different individual grading

---

Results

True rank against estimated rank scatterplot

Figure 2: True rank against estimated rank scatterplot

Absolute errors

_____

_____

Errors by project

_____

Typical project problems and how to avoid them

"My supervisor doesn't tell me what to do"

Getting the most out of your supervision meetings

- Your supervisor will not tell you what to do.
- They can only offer guidance. You will need to be proactive.
- Suggest ideas; explain and document what you have done; ask specific questions.
- Use the documentation tools suggested *at every meeting*

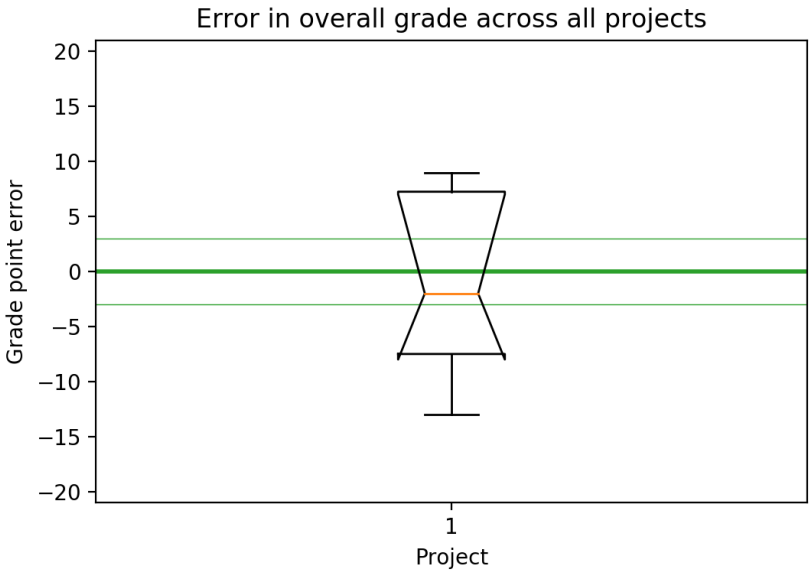

Figure 3: Signed deviation Box plot and histogram

Median error: -2.0

Max error: 9.0

Min error: -13.0

Std error: 7.8

Std err: 2.3

Figure 4: Summary statistics

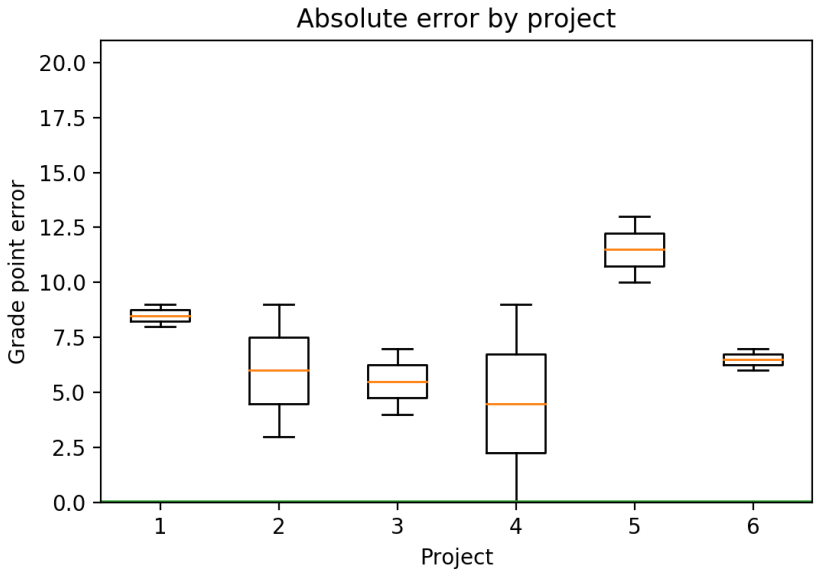

Figure 5: Absolute deviation grouped by project

- **Status, questions, minutes, plan**

Setting goals

- Set clear actionalbe, goals for each week.
- Discuss your progress with your supervisor. Be honest. Your supervisor cannot punish you; they can only give you advice.
- Measure your progress against a year-long plan.
- If you slip, or find some goals are unattainable, ask for advice for how to get back on track.
  - Your plan is flexible but that doesn't mean it's not important.

When things go wrong

- It's very rare, but sometimes supervisor-student relationships go wrong. You might feel you can't communicate effectively, or feel uncomfortable with how things are going.
- If this happens, **let the project coordinator know immediately**
- I will try and sort things out.

"The dog ate my server"

Responding to adverse events

- If you lose your code, data, dissertation or whatever: **it's your problem.**
- We **have** to assess you on what you produce, not what you could have done.
- Backup. Plan for risks.

Warning

- If you don't complete the project, **for whatever reason**, you will either fail the entire degree, or if permitted, have to resit fourth year. There is nothing else we can do. These are University regulations.

Accommodations

- If problems are genuinely out of your control, we will try and accommodate, but there may be little we can do
- The following examples are *illustrative* and do not set any precedent. Every case will judged on its merits.

Examples: no action

- Your VM environment stopped working around Christmas. *We will do nothing. You had time to work around this.*
- The collaborator you were relying on for data pulls out. *We will do nothing. You should have dealt with this early in the project and planned for this risk*
- Your dissertation was corrupted and you have no backup. *You will get an H and fail the degree.*
- You were sick for a week in January. *The deadline will not be altered.*
- You forgot to turn up for your presentation. *You will get an H for the presentation component.*

Examples: action

- GitHub goes down on the day of submission. *We will extend by one day.*
- You were sick for a week before the deadline, and you have Good Cause claim with evidence: *We will extend by a week.*

- You have serious mental health problem that develops during the year. *We will do whatever we can to support you. This might involve an extension, or other special measures. Speak to the coordinator as early as possible.*
- You absolutely have to go to an interview the day of the presentation. *We will rearrange the presentation*
- The client you are developing for stops responding. *We will allow you to adjust the topic of your project.*

When things go wrong

- We will not alter the submission deadline
  - *Unless* you have a Good Cause Claim *and* the evidence is accepted as grounds for an extension by the School's Good Cause Committee
- Even in this case, we will typically only be able to offer one week's extension to meet the Exam Board deadlines.
- If this isn't enough extra time, you will either:
  - Fail to complete the project. You *may* qualify to graduate with a Designated degree.
  - Have to resit the whole year, if permitted by the University to do so.
- Either way, you **will not graduate in the summer**.

"I didn't get much done until Christmas"

Using the first week

- **Meet your supervisor as soon as reasonable** Discuss what you have done as background, what your plan is, and how meetings will run for the rest of the year.
- **Plan the rest of your project** write a short, concise timeline of the rest of the project and discuss with your supervisor at the next meeting.
- **Set up your environment** Create a version control repository. Install and configure and software or tools you need to make progress. Set up tools like issue trackers. Check any hardware you might be using is available and that you can operate it.

- **Start work** Week 1 is an ideal time to start the real meat of the project; whether that is requirements gathering, initial prototyping or literature reviews.

Writing a plan

- **Write a *simple* plan.** This could be a week-by-week plan with rough milestones, for example.
- You don't need a fancy Gantt chart (but you could make one). You are the only person working on this project.

A mock plan

Week 1: Environment set up. List of papers to read compiled.
...
Week 10: Preparations for project sprint ready. Implementation prototyped, test cases identified.
Week 13: First fully working prototype with all features. Status report submitted.
...
Week 18: Draft of dissertation largely complete and submitted to supervisor
Week 19: Last results of evaluation written up
Week 20: Dissertation submitted

Balance against assessed exercises

- It's tempting to spend a *lot* of time on assessed exercises.
- Watch out for spending way too much time for way too few marks. Work out a % of final degree per hour.
  - An exercise that is 10% of a 10 credit Level 4 module counts for 0.6 x (0.1 x 10)/(120) = 0.5% of your degree
  - The project counts for 0.6 x (40/120) = 20% of your degree.
  - You should spend the equivalent of **40** 10% exercises on your project.
- *You may have to leave exercises incomplete to make enough time for the project.*
- The only way you'll know is if you monitor your progress and time spent.

Prepare for the end-of-term project weeks

You will have a two-week period to work on your project exclusively at the end of Semester 1. There are no exercises or other classes and you will know enough to make rapid progress.

Prepare for this period well and do not squander it. You should be completing 80 hours of work on your project [except joint Honours students], or roughly 20% of the total project progress.

You may not be physically in Glasgow, but make sure that you put the hours in.

Before the project fortnight

- Collect things you need; data, environments, user requirements
- Try and get annoying tasks out of the way
- Set clear goals for the end of the block
- Make task lists and work through them

"The project took up so much of my time I couldn't study for other subjects"

Avoiding excessive focus on developing perfect products

- Remember; you need a **good project**, not a **perfect product**.
- You are marked on the dissertation. Fixing every bug and adding every feature might not get you any additional credit.
- There is value in producing technically excellent work, particularly in developing your skills and having something to show off to employers.
- **None of that will matter if you fail the rest of your courses.**
- Spend appropriate time on the dissertation and evaluation of your work. Remember that the write-up is what gets marked.

Scoping your project well

- You will need to scope your project. You can't do everything, though it is tempting to try.
- Discuss what you can achieve with your supervisor.
- Monitor your progress against a timeline.
- It is good to be ambitious; but you must be able and willing to cut back the project if time runs out.
- Always have a plan:

  - What would I cut out and still make a good project?
  - What could I add if I had the time?
- A tightly scoped and beautifully executed project is superior to large, sprawling, incomplete one.

"I don't understand how to evaluate my project"

Making sure your product can be evaluated

- You need to show you have done a good job at solving a problem.
- Think right now: how would I convince someone of that? How would I even convince them that I knew how to tell if I had done a good job?
- You have to do something; **and** show what it is that you have done.
- This needs to be something you can measure and analyse.
- **Think about this from the start**

We will cover the evaluation in more detail in the next session in December.

Evaluations

There are many ways your evaluation might go, depending on the style of project you are doing. Your supervisor can guide you further, but it might involve any of:

- Testing for correctness (e.g. unit testing)
- Log analysis (e.g. from real-world deployment)
- Feedback from clients (e.g. interviews, questionnaires)
- Experimental results from simulations
- Empirical work with subjects (e.g. in HCI projects)

You can't just write *"the project worked okay"*. You must ask specific questions and provide evidence to answer those questions.

"I'm not used to writing dissertations like this"

Plan for the writeup from the start

- Create a skeleton document early (you will get one with the template)
  - Fill out the dull details (title, name, etc.)
- Write easy bits as you go along
  - For example, you might write up the background in the first couple of months

  - The implementation might be finished in January and you can get it written up then.
- Your supervisor may ask you to write the dissertation starting in Week 1. This is good advice.

Projects are stereotypical

Dissertations can vary, but the standard structure is:

- **Introduction**: why should I care and what are you doing?
- **Background**: what did other people do, and how is it relevant to what you want to do?
- **Analysis/Requirements** What is the problem that you want to solve, and how did you arrive at that?
- **Design** How is this problem to be approached, without reference to specific implementation details.
- **Implementation** What did you do to implement this idea, and what technical achievements did you make?
- **Evaluation** How good is your solution?
- **Conclusion** Summarise the whole project for a lazy reader who didn't read the rest (e.g. a prize-awarding committee).

Guidance

- **Introduction** *Motivate* first, then state the general problem clearly.
- **Background** Don't give a laundry list of references or other software. Tie everything you say to your problem. Present an argument. **Don't write a tutorial**; provide background and cite references for further information.
- **Analysis** Make it clear how you derived the constrained form of your problem via a clear and logical process.
- **Design** Design should cover the abstract design in such a way that someone else might be able to do what you did, but with a different language or library or tool.
- **Implementation** You can't talk about everything. Cover the high level first, then cover important, relevant or impressive details.
- **Evaluation** Ask specific questions that address the general problem and answer them with evidence. Be fair and be scientific.
- **Conclusion** Summarise briefly and fairly. Indicate what future work could be done, but remember: **you won't get credit for things you haven't done**.

Provide early drafts for your supervisor

- Supervisors will ignore you if you submit a draft a week before the deadline.

  - Provide partial drafts earlier, and discuss them in meetings; maybe a chapter a time
- Definitely get feedback on structure early on in the process
- Consider shared editing tools like `Overleaf` to make it easier for supervisors to keep up.
- **Don't be shy; share drafts even if you are unhappy with them. Feedback will improve them.**

"The project is killing me"

Student well-being and stress levels during projects

- **Look after yourself.**
- **Stress can be high.** The project is not the most important thing in the world. Take a break.
- **Remember that your other classes are important.** Don't neglect studying for adding new exciting features to your project.
- **Sleep.** Don't stay up all night working on the project, especially in the last weeks.

Be reflective and aware

- Always think: "is it worth spending more time on this now?"
- Use a time log to reflect on whether your efforts have been effective or whether you need a new strategy

---

We will support you

- Seek help if you can't manage. Let your supervisor know, at the very least.
- Speak to your adviser of studies if things are going seriously wrong (depression, for example).
- **DON'T GO OUT OF CONTACT** Every year students go "off the radar" after suffering personal problems
  - It's fine to email and say *"I haven't done anything; I'm struggling right now".*
  - Don't go out of contact because you have nothing to report.

  - Keep in touch. Email me directly if you need to.

- Many problems can be fixed, but only if they are caught early. Don't delay if you feel you cannot cope.

When to contact me

- If something goes **seriously** wrong and your supervisor can't fix it:
  - relationship with your supervisor breaks down
  - you have a serious personal problem that affects your project
  - your supervisor retires suddenly in the middle of your project
- If you have a question about the project, **and**:
  - you have read the *Student Guide* and the *Assessment Criteria* **and**
  - you have checked Moodle **and**
  - you have read all of these slides **and**
  - you still don't know the answer.

---

classresponse.gla.ac.uk/yacrs #363

- Questions so far?
- I will try and work through answers before this afternoon

Restart 1400 at McIntyre 201 Flat Hall