COMP110: Principles of Computing

Software Quality

Today's lecture

Today's lecture has **three parts**

- Software quality and quality assurance
- Pathfinding and the A* algorithm
  - Introducing the next worksheet
- Live coding: applications of OOP techniques

Software Quality and Quality Assurance

Learning Outcomes

In this section you will learn how to...

- **Explain** what ‘quality’ is
- **Explain** what ‘quality assurance’ is
- **Discuss** the role of quality assurance in software engineering

Further Reading

- Pressman, R.S. (2009) *Software Engineering: A Practitioner's Approach*. 7th Edition. McGraw-Hill.
- Kaner, C., Falk, J. And Nguyen, H.Q. (1999) *Testing Computer Software*. John Wiley and Sons.

Software Quality

"Bad software plagues nearly every organisation that uses computers, causing lost work hours during computer downtime lost or corrupted data, missed sales opportunities, high IT support, and maintenance costs, and low customer satisfaction" (ComputerWorld, 2005)

Software Quality

"The Sorry State of Software Quality—quality has gotten worse!" (InfoWorld, 2006)

Software Quality

So, what does quality look like in games?

- "A characteristic or attribute of something."
- Quality may relate to several aspects of games:
  - **the quality of the design**: the aesthetic is specified to accurately meet the desires and pleasures of the target audience; provides a distinctive experience; etc.
  - **the quality of the implementation**: the game mechanics are able to achieve the intended aesthetic; the game is well-implemented...

Software Quality

Software Quality

Software Quality

https://www.youtube.com/watch?v=VhenntkfAgU

Software Quality

Today, software quality in games remains an issue, but who is to blame?

- Players blame developers, arguing that sloppy practices lead to low-quality software.
- Investors blame developers for not understanding what players want and what is `good enough' to maximise profit.
- Developers blame the design team and their publisher, arguing that irrational delivery dates and continuous change force them to deliver software before it can be adequately tested.

Socrative `6E8NSW3IN`

So, who is responsible?

- In pairs.
- Discuss for 2-minutes whether designers, developers, or publishers are responsible for software quality.
- **Suggest** which parties are responsible **and justify** your answer.

Socrative `6E8NSW3IN`

But wait...what exactly is quality?

- In pairs.
- Discuss for 2-minutes what `software quality' means in the context of game development.
- **Give** a definition for `game software quality'.

Software Quality

"Quality...you know what it is, yet you don't know what it is. But that's self-contradictory. But some things are better than others, that is, they have more quality. But when you try to say what the quality is, apart from the things that have it, it all goes poof! There's nothing to talk about. But if you can't say what Quality is, how do you know what it is, or how do you know that it even exists? If no one knows what it is, then for all practical purposes it doesn't exist at all. But for all practical purposes it really does exist. What else are the grades based on? Why else would people pay fortunes for some things and throw others in the trash pile? Obviously some things are better than others...but what's the betterness?...So round and round you go, spinning mental wheels and nowhere finding anyplace to get traction. What the hell is Quality? What is it?"

(Robert Persid, 1974)

Software Quality

- **transcendental view**: quality is something immediately recognisable, but cannot be explicitly defined.
- **pragmatic view**: relative to utility and specific goals. If something meets our goals, it exhibits quality.
- **commercial view**: the specification is key. If the specification is sound and the product conforms to the specification, it exhibits quality.

Software Quality

- **product view**: quality is tied to inherent characteristics (e.g. functions and features) of a product.
- **value-based view**: quality is based on how much a customer is willing to pay.
- In practice, perception of quality tends to combine these different views in subtle and nuanced ways.

Software Quality

"An effective software process applied in a manner that creates a useful product that provides measurable value for those who produce it and those who use it." (Bes, 2004)

Socrative `6E8NSW3IN`

Can we now construct a better definition of software quality in games?

- In pairs.
- Discuss for 2-minutes what `software quality' means in the context of game development.
- **Give** a definition for `game software quality'.

Quality Assurance

- An **effective development process** establishes the infrastructure that supports any effort towards high quality.
- The management aspects create checks and balances to help avoid project chaos—a key contributor to poor quality.
- Software engineering practices empower developers to analyse and review their product.
- Umbrella activities, such as project management and code reviews, are key factors in determining quality and have just as an important role as any other specific source code quality assurance practice.

Quality Assurance

In order to assure quality in games, we need to know what to measure. David Garvin (1987) has some suggestions:

- **performance**: does the software deliver all content, functions, and features that are specified as part of the requirements model in a way that provides value to the end-user?
- **features**: does the software provide features that surprise and delight first-time end-users?
- **reliability**: does the software deliver all features and capability without failure? Is it available when it is needed? Does it deliver functionality that is error free?

Quality Assurance

- **conformance**: does the software conform to local and external software standards that are relevant to the application? Does it conform to de facto design and coding conventions? For example, does the user interface conform to accepted design rules for menu selection or data input?
- **durability**: Can the software be maintained (changed) or corrected (debugged) without the inadvertent generation of unintended side effects? Will changes cause the error rate or reliability to degrade with time?

Quality Assurance

- **serviceability**: Can the software be maintained (changed) or corrected (debugged) in an acceptably short time period. Can support staff acquire all information they need to make changes or correct defects?
- **look-and-feel**: Most of us would agree that an aesthetic entity has a certain elegance, a unique flow, and an obvious 'presence' that are hard to quantify but evident nonetheless.
- **socio-cultural context**: In some situations, you have a set of prejudices that will influence your perception of quality.

Quality Assurance

Other definitions to explore and factors to consider in your own time:

- McCall's Quality Factors
- ISO 9126 Quality Factors
- Targeted Factors
- IEEE 610.12 Software Assurance

Socrative `6E8NSW3IN`

Why are these factors important in games?

- In pairs.
- Discuss for 2-minutes why quality assurance is important to game development.
- **Illustrate TWO** reasons why quality assurance is important. Use examples to support your answer.

Quality Assurance

Why are these factors important?

- If you produce a software system that has terrible quality, you lose because no one will want to buy it.
- If on the other hand you spend infinite time, extremely large effort, and huge sums of money to build the absolutely perfect piece of software, then it's going to take so long to complete and it will be so expensive to produce that you'll be out of business anyway.
- Either you missed the market window, or you simply exhausted all your resources.

Quality Assurance

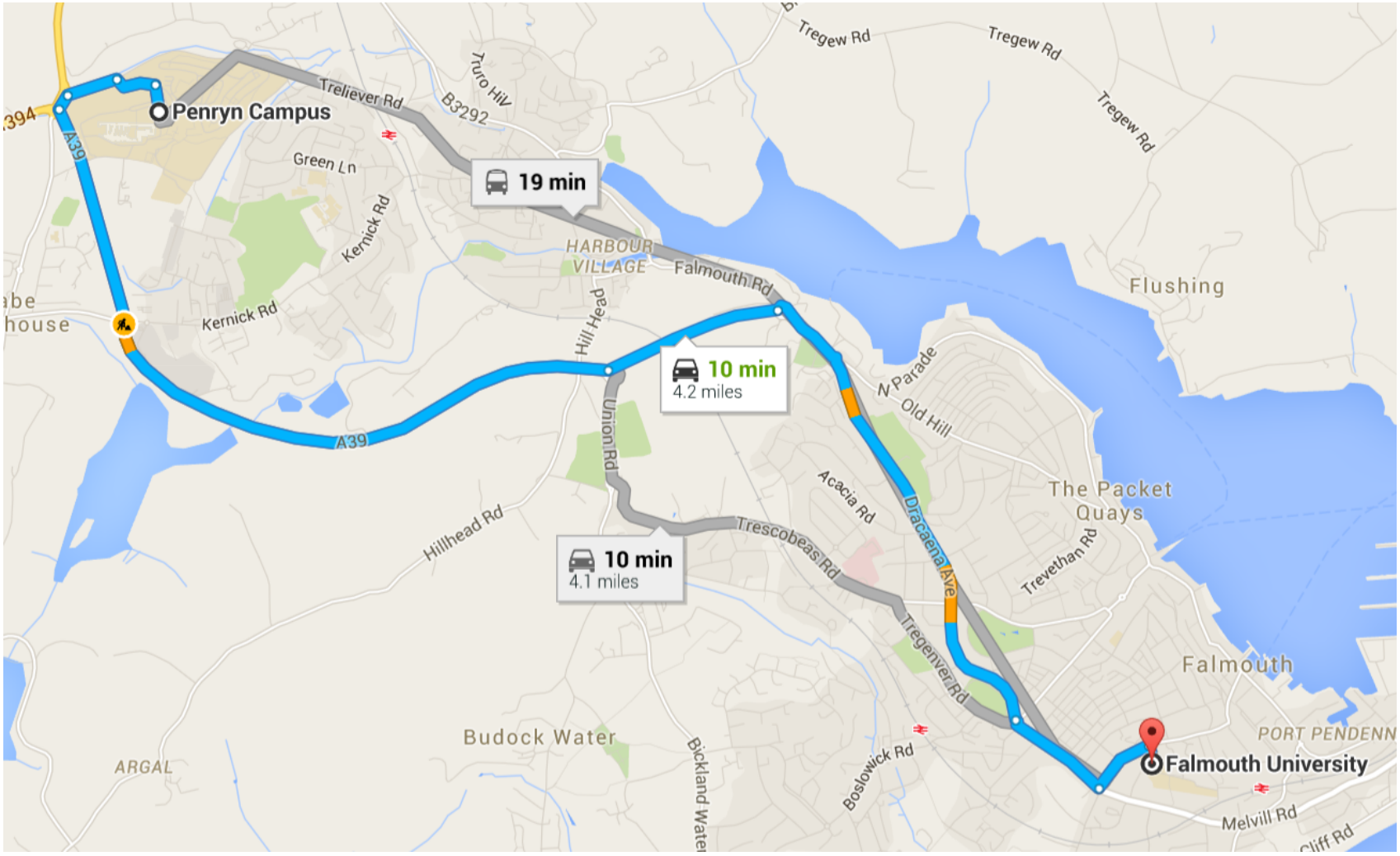
- Some aim to that magical middle ground where the product is good enough not to be rejected right away, but also not the object of so much perfectionism and so much work that it would take too long or cost too much to complete.
- So-called `good enough' software tends to deliver high quality functions and features that end-users desire, but at the same time may risk delivering other more obscure or specialized functions and features that contain bugs.
- Large companies can get away with serious bugs at launch (e.g. Activision-Blizzard and Diablo 3)—but, smaller indies risk permanent damage to their reputation.

Pathfinding

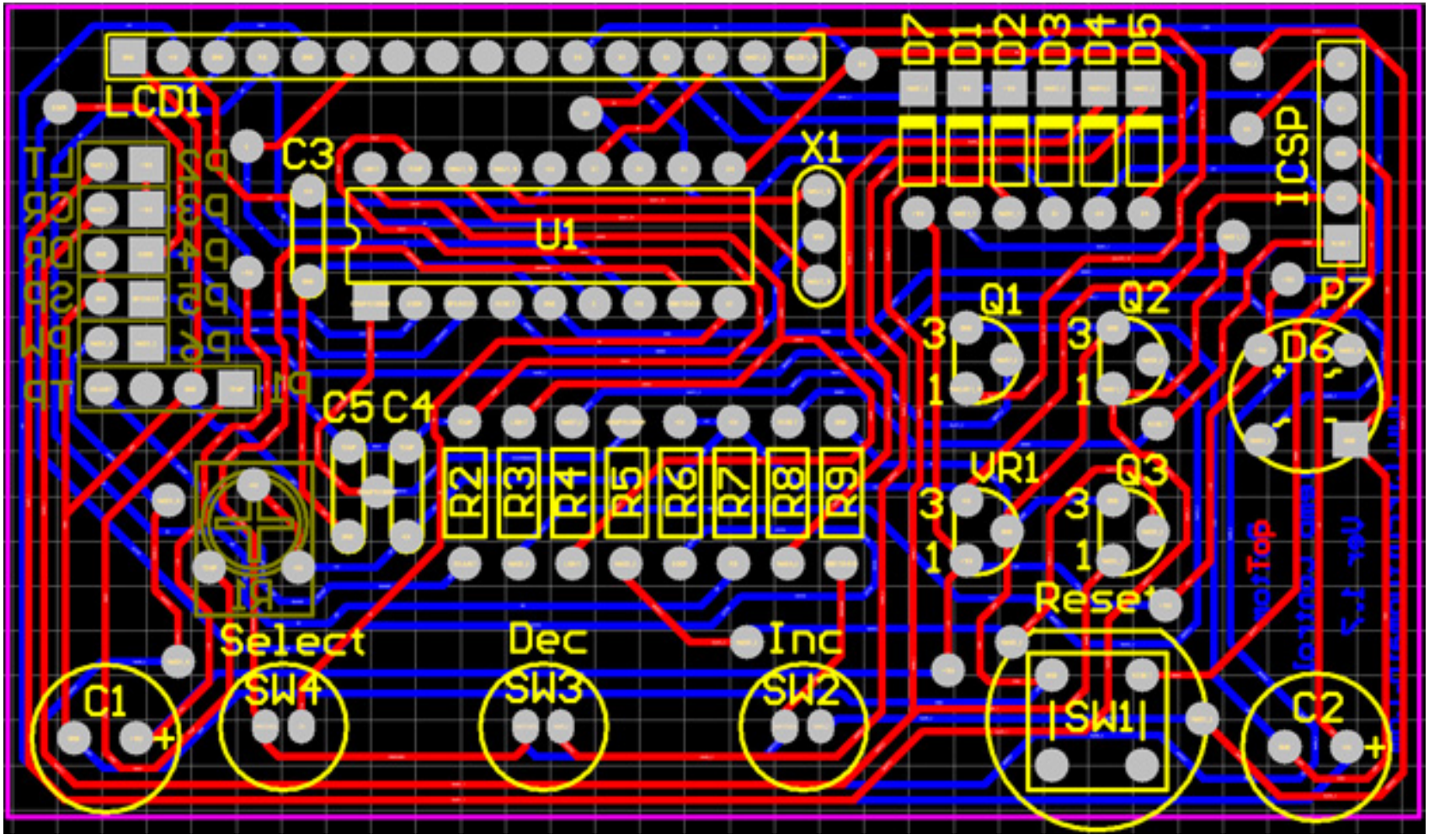
The problem

- We have a **graph**
  - **Nodes** (points)
  - **Edges** (lines between points, each with a **length**)
- E.g. a road map
  - Nodes = addresses
  - Edges = roads
- E.g. a tile-based 2D game
  - Nodes = grid squares
  - Edges = connections between adjacent squares
- Given two nodes *A* and *B*, find the **shortest path** from *A* to *B*
  - "Shortest" in terms of edge lengths — could be distance, time, fuel cost, ...

Applications of pathfinding



Applications of pathfinding



Applications of pathfinding

Many applications in game AI

- Non-player character AI
- Mouse-based movement (e.g. strategy games)
- Maze navigation
- Puzzle solving

A* search

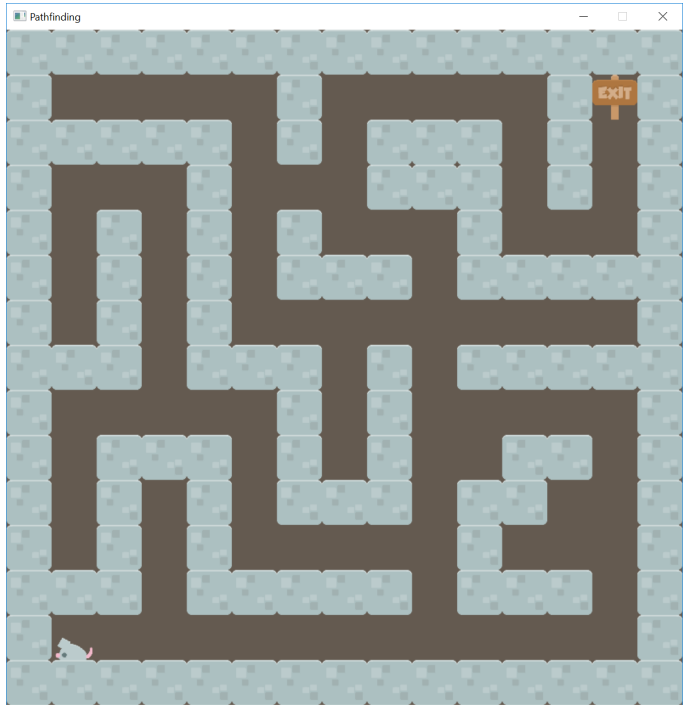
Idea:

- Expand out from the start node
- Let $g(n)$ be the length of the path currently found between the start and node n
- Let $h(n)$ be an **estimate** of the distance from n to the goal
- Prioritise nodes for which $g(n) + h(n)$ is small

Properties of A* search

- A* is **guaranteed** to find the shortest path if the distance estimate is **admissible**
- Essentially, **admissible** means it must be an **underestimate**
  - E.g. straight line Euclidean distance is clearly an underestimate for actual travel distance
- A* is an example of **heuristic** search
  - In AI, a heuristic is an estimate based on human intuition
  - Heuristics are often used to prioritise search, i.e. explore the most promising options first

Implementing A*



- Worksheet 5 on LearningSpace
- Deadline: **9th March at 6pm** (i.e. two weeks from today)

Live coding: applied OOP

Introduction

`https://github.com/Falmouth-Games-Academy/comp150-live-coding`

- Clone this repository and load it into Visual C++
  - Note the instructions in `readme.md` with regard to copying dlls
- A simple game, but implemented as one long function
- Let's improve it!

Resource Acquisition Is Initialisation (RAII)

- A common C++ idiom for handling **allocation and deallocation of resources**
- Create a class, allocate in the **constructor**, deallocate in the **destructor**
- If the instance is created on the **stack**, the destructor is called **automatically** when the instance goes out of **scope** — no need to remember to deallocate things

Accessor methods

```
private:
    int health;

public:
    int getHealth() { return health; }
    void setHealth(int h) { health = h; }
```

- A.k.a. **getters** and **setters**
- Allow finer control over access to data in a class
- E.g. could have a **public** getter and a **private** setter
- E.g. could have a setter that **validates** the new value