# Tackling A Level Projects in Computer Science

OCR H446

Ceredig Cattanach-Chell

PG ONLINE

# Tackling A Level Projects in Computer Science

## OCR A Level Computer Science H446

Ceredig Cattanach-Chell

**Published by**
PG Online Limited
The Old Coach House
35 Main Road
Tolpuddle
Dorset
DT2 7EW
United Kingdom
sales@pgonline.co.uk
www.pgonline.co.uk
**2019**

PG ONLINE

# Contents

# Chapter 1 Starting a new project

## Objectives

- Choose a project title
- Choose a stakeholder
- Create a project outline
- Understand:
  - How to be realistic in your project scope
  - An appropriate level of difficulty for your project
  - Languages that are and are not appropriate to use
  - The design methodologies that could be used
  - Different options for IDEs
  - How your work is authenticated

## Introduction

Choosing a suitable A Level project is quite a challenge. Projects contribute to your final grade and therefore choosing the right project for you is important.

This guide will take you through the steps to create a successful project. It will not give you the answers, but instead show you tips and tools to help keep you on track, and evidence your project efficiently. Using this guide and the specification should give you confidence in being able to produce the best project you can.

Projects are not about quantity, but quality. Exam boards have specific **mark schemes**. You need to show you meet each of the mark scheme points. Clear and precise documentation makes it easier for both you, your teacher and the moderator to identify where you have met **mark criteria**.

TIP Ensure that you fully understand the mark scheme before starting your project.

# Chapter 1
**Starting a new project**

## Choosing stakeholders

A **stakeholder** is someone who may be interested in using the system but is not necessarily the person you are designing the system for.

The stakeholder you choose to use the system you create is very important.

When picking a stakeholder try to avoid:

1. Using your friends as stakeholders
2. Being your own stakeholder

Stakeholders may be used when you decide to create a program for your own needs. They are people who can help you scope the project and give you feedback on your ideas. This discussion is important to ensure that you have a range of ideas to draw on during the analysis.

**TIP** Successful projects will have other people involved to help provide ideas, feedback and testing results.

Anyone can become involved in your project, but it is better to be careful who you choose. Try to avoid choosing your Computer Science teacher. However, another member of staff could work well as a stakeholder. For instance, if you are thinking of a fitness tracker then a member of staff in the PE department could make a good choice.

Choosing classmates or friends may seem like a good idea. However, if you do, you must be sure that they are willing and able to be critical of your system. When it comes to testing and evaluation you want your stakeholders to provide sensible and critical feedback, rather than simply stating that “it all works wonderfully”.

The stakeholder must have the time to give to your project. They may say yes at first, but will they still be happy to help six months later? Let them know how much time will be required of them, and when. For instance, if you use a teacher, remember that many teachers have coursework or mock exams to mark which may be around the time that you want them to start testing your solution. This could affect your progress.

You may find that there are certain situations where you need to create your own style. This will be most likely when you wish to add programming code to your report.

When setting up a style for code, remember that it is easiest to read if a **monospace font** is used. Copying and pasting code from the IDE that you use may also keep the **syntax highlighting**. This will make it easier for your teacher and moderator to read the code.

Use just one font for your report. This will make your report look more professional. To emphasise text or titles make use of font size, bold or italics. As mentioned, your code should be in a different monospace font.

TIP Your styles should be clean and easy to read. The defaults that come with your word processor should be sufficient, so don't waste time creating your own styles other than for computer code.

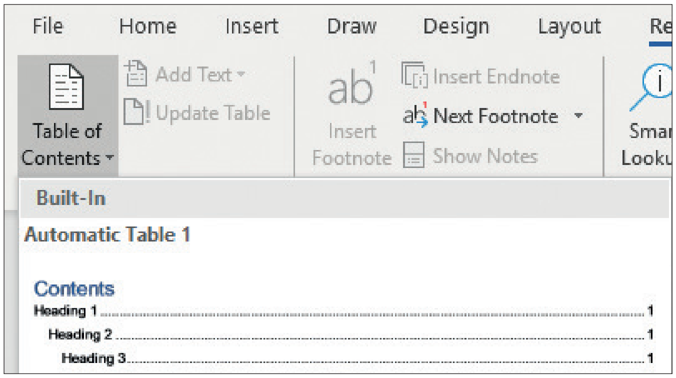
TASK Define styles for your document

## Table of contents

A **table of contents** is essential for your documentation. It helps to show the teacher and moderator where your evidence is. Your teacher will use this to help reference their marking on the form that is sent to the exam board.

If you keep your work organised, a table of contents will also help you to refer back to tests, stakeholder success criteria and other parts of your report that you need to reference.

Most word processors will automatically create a table of contents. They make use of styles to produce these. It is essential that you are using styles throughout your report for this to work. Once set up, the table of contents and page numbers can be updated with just one click.



*Word processors insert contents tables automatically if styles have been used correctly*

# Chapter 3

## Stating the problem

*People who all have the same role can be discussed as one persona*

You will need at least one stakeholder for this project. Stakeholders may also be people who can help advise on the system but may not be directly involved in the actual problem.

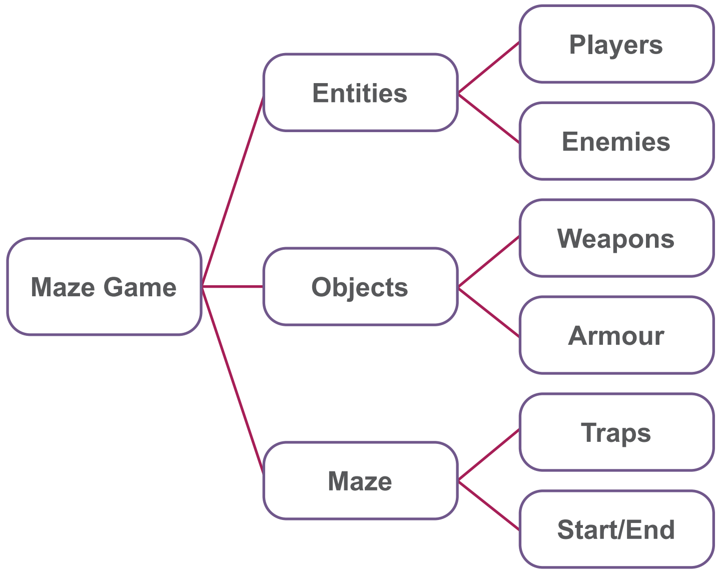
For instance, the system may need to be made suitable for people who are colour blind. Currently, no one who uses the system is colour blind, therefore, you could find someone external to act as a stakeholder and review your plans for supporting those who may have this condition.

Another example may be that the person you are designing the system for is not an IT expert. They may not know what features can be added to a new system. You may be able to find someone who could act as an advisor to help generate ideas and give you feedback on your plans.

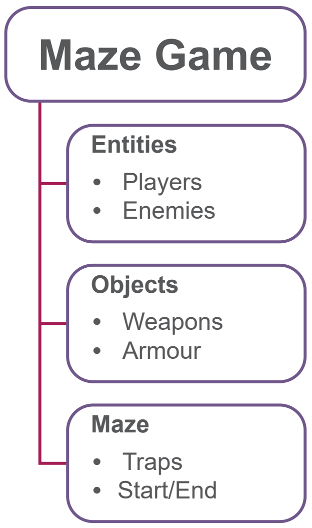
### To do list

**Have you done the following?**

- ☐ **Created a clear description of the problem**
- ☐ **Described and justified the features of the problem that are solvable by computational methods**
- ☐ **Explained why the problem lends itself well to a computational solution**
- ☐ **Identified and described the stakeholders in the solution**
- ☐ **Explained how the solution is appropriate to the needs of the stakeholders**



*Example 2 - Decomposition diagram for a maze game*



*Example 3 - Decomposition diagram for a maze game*

## Lists

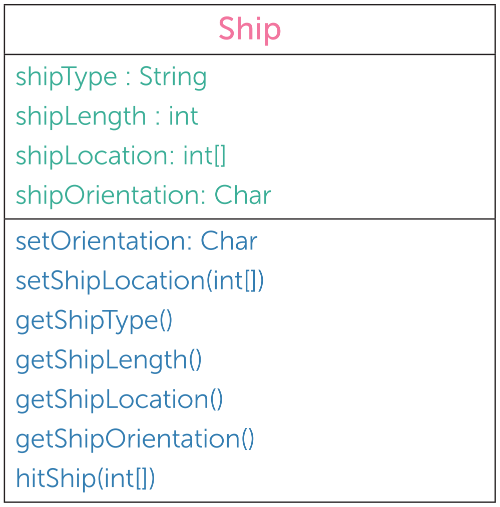
Using lists may be an easier option to set up. However, they can appear more complicated and become harder to track. If you decide to use a list, then be careful to make sure that the formatting clearly shows how the problem has been decomposed.

Maze Game

1. Entities
   a. Players
   b. Enemies
2. Objects
   a. Weapons
   b. Armour
3. Maze
   a. Traps
   b. Start/End

*Example 4 - Decomposition list for a maze game*

First, start with a class diagram for the Ship class:



*An example of one class*

**WARNING** The diagrams shown in this section are examples only and do not form a full system.

This diagram shows us that we need to create a class called Ship. It will have four attributes which store the data for the type of the ship, the length of the ship, the location of the ship and its orientation which will be either horizontal ('h') or vertical ('v').

Each attribute has a data type shown in the diagram and includes a string, an integer, an array of integers and a Boolean type.

There are two methods shown for setting the attributes. These methods are very simple and just allow the values stored in the attributes to be changed. They also show the parameters that may need to be passed to the methods. `hitShip(location)` by contrast is a method that can be called when the ship is hit. The class diagram doesn't show the algorithm that will be used for the methods, but it does show what methods the class will need to have programmed.

## Test plans

The next stage in the Design section is to think about how the system will be **tested** to ensure that it meets the requirements specification and success criteria laid out in the Analysis.

There are two sections to your test plans:

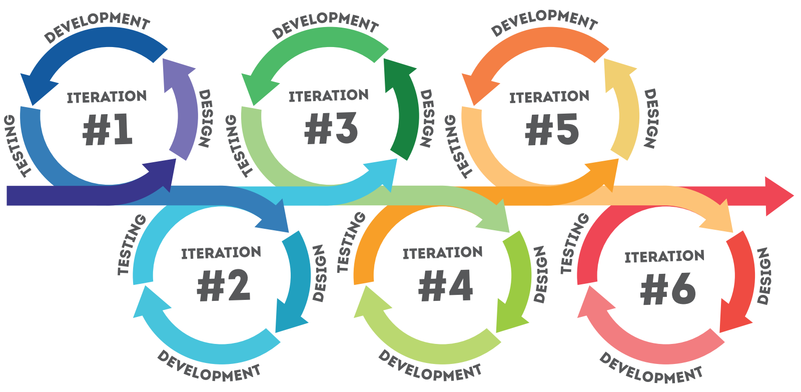
1. Iterative testing
2. Post-development testing

These tests are different to each other in nature and therefore require different styles of tests.

Testing **must** be planned before any implementation, but you do not need to plan all the testing in one go.

It is perfectly acceptable to plan tests before each iteration. The implementation of the iteration can then be implemented and tested. Once the iteration works successfully, the tests for the next iteration can be defined.

**Iterative testing** of the solution uses **white-box testing**. This means that the tests are based on the code that is written. It is useful for checking that individual algorithms or sections of the program are working correctly.



*Remember that projects are usually developed iteratively. For each iteration, a short test plan is written before development is carried out.*

**Post-development testing** is a form of **black-box testing** which is carried out once the entire solution is complete. It is possible to plan post-development testing in the design section before any implementation has been started. Your system designs could change as the project progresses, so remember before you start such testing to check that the tests are still valid and reflect any updates.

Remember that good test plans are designed to try and make the system fail. They should test that your system stands up to unintended use, and possibly malicious attack.

## Annotation

Annotation of screenshots, source code or photos can be done in many ways. You can choose to annotate by the following methods:

1. Use image editing software
2. Use presentation software and then save as a JPEG or similar
3. Paste the image into word processing software and use drawing tools.

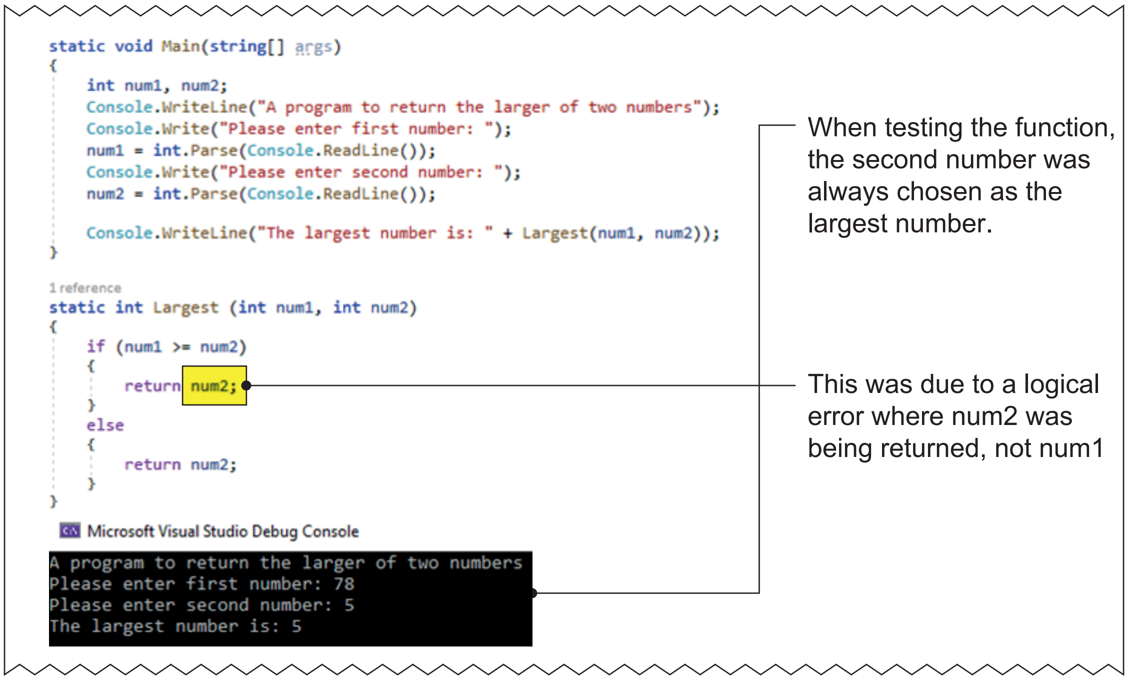
Method 1 is technically harder. Whilst the results are good, it takes time to do the annotation. It is therefore not advised unless the other methods are not appropriate.

Method 2 which uses presentation software such as Microsoft® PowerPoint® is easy to use, but it will often take longer than annotating directly in the report itself.

Method 3 tends to be the most popular and fastest. Be aware though that sometimes word processing packages can cause issues with layout. In this case, method 2 is advised.

**TIP** When annotating with a word processor, remember to group all the objects together so that they move in the report as one object.

Good use of annotation can cut down the amount of writing you need to do.



```
static void Main(string[] args)
{
    int num1, num2;
    Console.WriteLine("A program to return the larger of two numbers");
    Console.Write("Please enter first number: ");
    num1 = int.Parse(Console.ReadLine());
    Console.Write("Please enter second number: ");
    num2 = int.Parse(Console.ReadLine());

    Console.WriteLine("The largest number is: " + Largest(num1, num2));
}

1 reference
static int Largest (int num1, int num2)
{
    if (num1 >= num2)
    {
        return num2;
    }
    else
    {
        return num2;
    }
}
```

Microsoft Visual Studio Debug Console

```
A program to return the larger of two numbers
Please enter first number: 78
Please enter second number: 5
The largest number is: 5
```

When testing the function, the second number was always chosen as the largest number.

This was due to a logical error where num2 was being returned, not num1

*An example of an annotated screenshot showing the development and testing of a function. Continued on following page*

# Chapter 8
## Final checks

### Referencing

Not referencing sources that you have used is considered cheating. If you are found to have used sources, and not referenced them, then you may receive zero marks for your work or be stopped from taking your A Level completely.

Markers and moderators must be able to see the following:

- What has been copied or come from another source
- That the source has been acknowledged
- Where/how you have developed the source material into your own project or words

Your school or the exam board may use plagiarism checkers. These are sophisticated in flagging up areas of reports or projects that have been copied.

**TASK** Make sure you have included a references section and all sources have been correctly referenced.

## Submitting the project

Ask your teacher how they would like the report, project files and any other resources such as videos to be submitted. If you forget to give files to your teacher, then you may well lose marks. You may need to print the report or provide an electronic version. Be aware of any deadlines that your teacher has given and make sure you submit your work well before the deadline.

### To do list
**Have you done the following?**

- ☐ **Reviewed your project and report against the mark scheme**
- ☐ **Compared your work against the mark scheme to identify any areas that aren't complete or need improvement**
- ☐ **Completed any missing parts of the report if needed**
- ☐ **Made any improvements if needed**
- ☐ **Proofread and spell checked the report**
- ☐ **Ensured all references to other sources are included**
- ☐ **Prepared your report and other files in the correct format requested by your teacher ready for submitting**
- ☐ **Submitted your report and any other files requested before the deadline**

# Index

# Appendix
## Useful shortcuts and key combinations

### Editing shortcut key combinations

| | |
|---|---|
| **Ctrl + A** | Select all |
| **Ctrl + B** | Apply or remove bold formatting |
| **Ctrl + C** | Copy |
| **Ctrl + Shift + C** | Copy formatting |
| **Ctrl + F** | Find |
| **Ctrl + I** | Apply or remove italic formatting |
| **Ctrl + P** | Print |
| **Ctrl + S** | Save |
| **Ctrl + V** | Paste |
| **Ctrl + Shift + V** | Paste formatting |
| **Ctrl + X** | Cut |
| **Ctrl + Y** | Redo or repeat last action |
| **Ctrl + Z** | Undo an action |
| **Shift + F3** | Toggle case |
| **Alt + =** | Insert equation |
| **Shift + Enter** | Create a soft line break |
| **Ctrl + Enter** | Insert a page break |
| **Ctrl + [** | Decrease font size |
| **Ctrl + ]** | Increase font size |

### Navigation shortcuts

| | |
|---|---|
| **Ctrl + Home** | Go to beginning of document |
| **Ctrl + End** | Go to end of document |
| **Shift + F5** | Go to last place text was edited |

PG ONLINE

ISBN 978-1-910523-19-3

9 781910 523193