Y9B Design — MYP

Course Expectations • 2026–2027

Teacher	Mr. Alex Rawson • a.rawson@stleonards-fife.org
Room	PH1
We meet	Week A • Wednesday • Period 5 (14:05–15:00) Week A • Friday • Period 6 (15:05–15:55)
Course page	rawsonvault.com/myp/design

1 What this course is

MYP Design is the computing course — you will hear me call it computer science. Most of this year you will be writing **Python**: designing, building and playtesting an interactive animation or game using Pygame Zero.

This is the year the coding gets real. Not blocks — actual code, that breaks in actual ways, that you have to debug yourself. It is also the year you decide about GCSE Computer Science, so the last stretch deliberately widens out to show you what the subject looks like beyond programming.

We meet for only two periods, and both in the A week, so lessons start promptly and we try to maximize our time together.

2 What we cover

Unit	What you do	Share of year
Animations and games	Design, build and playtest a game or interactive animation in Python with Pygame Zero, through a full design cycle	Most
Computer science taster	What computer science is beyond coding, and an honest look at GCSE J277	A slice
AI and typing	What AI tools actually do, how to use them honestly, and touch-typing	End of year

3 What you need

- **Laptop** — charged. A flat battery is a lost lesson
- **Code editor** — Thonny or the Raspberry Pi code editor installed
- **Ring binder** — to keep design work, worksheets, and playtesting notes
- **Dividers** — for the ring binder, to keep units separate
- **Writing utensils** — pencils, pens (multiple colours preferred), highlighters (multiple colours preferred), a ruler, and an eraser
- Your planner

A ring binder with dividers, like this one

4 How you are assessed

You are assessed against the four MYP Design criteria, each marked out of 8. The games unit runs through all four:

	Criterion	What it looks like here
A	Inquiring and analysing	Researching your genre and audience, and working out what the game has to do
B	Developing ideas	A design specification and playable designs you can justify against it
C	Creating the solution	Writing the code, with a record of the decisions and the dead ends
D	Evaluating	Playtesting with real people, against your specification, and saying what you would change

Your grade comes from your **most recent, most consistent** work against each criterion, not an average of everything. Early attempts are for learning; they do not follow you around.

An ambitious game that is half-finished, documented honestly, will usually beat a safe one that works — provided you can explain the gap.

5 Homework

There will likely be one homework set biweekly — with both lessons in the A week, there will be work that happens on B weeks to keep you fresh. It is recommended to do the work *during* the B-week, so that if you are struggling you can come to me early in the A week to ask questions and improve your work.

6 Working with code

- **Save constantly, and save where you can find it again.** Losing a week's code to an unsaved file is the most avoidable disaster in this subject.
- **Comment as you go.** Criterion C is assessed on your record of how you built it, and comments are the cheapest record there is.
- **Debug before you ask** — read the error message, find the line, say out loud what you expected. Then ask. I will always help; I will help faster if you have done those three things.
- **Do not paste in code you cannot explain.** If you cannot walk me through it, it cannot go in your project. That applies to code from a friend, a forum, or an AI.

7 Deadlines and late work

Deadlines are given in class and posted on the course page. Work handed in late is marked and returned, but recorded as late; more than a week late is recorded as not submitted.

If something is going wrong — illness, deadlines stacking up, anything at home — **come and tell me before the deadline.** An extension asked for in advance is nearly always fine. Asking afterwards is different — by then, I usually cannot say yes.

8 Absence

Being away does not remove the work. It only delays it. **It is your job to find out what you missed** — check the course page first, then ask me.

- **Lessons.** Notes and resources are on the course page. Catch up before the next lesson.
- **Assessments.** Take the assessment within three school days of coming back. Come and book a time yourself — this is your responsibility, and I will not remind you.
- **Extended absence.** See me on your first day back and we will agree a realistic plan and a set of dates.

9 Phones, laptops, and AI

Phones go straight in your locker when you arrive. They stay there for the whole school day, in every lesson.

Laptops are open when the task needs them and closed when it does not. I will say which.

AI tools are genuinely useful and you should learn to use them well. In this course:

- **Fine:** asking it to explain something you are stuck on, generating extra practice questions, checking your own reasoning after you have done the work yourself.
- **Not fine:** getting it to produce an answer, a paragraph, a method or code that you then hand in as your thinking.
- **Always:** write one line at the end saying what you used and what for. Declaring it is never penalised. Not declaring it is a problem.

The reason is simple. The assessment has no AI in it. The moment you actually need to understand something has no AI in it either. If an AI does the work for you, you lose the time you needed to get better at computer science.

10 Academic honesty

You may work together — you learn faster when you do. But **the writing must be yours, done on your own.**

- Write down who you worked with, on the work itself.
- Cite anything you used — book, website, video, diagram, dataset. Putting it in your own words still needs a citation: you are borrowing the idea even when you change the words.

11 In the classroom

- Arrive on time and ready to start.
- Ask the question. Someone else has it too, and I would rather stop the lesson than let you fall behind.
- Wrong answers said out loud are how the lesson works. Nobody is laughed at in here.
- Leave the room ready for the next class.

12 How to do well in this course

Start smaller than you want to. Every year somebody designs an open-world epic and hands in a title screen. A simple game, finished and properly evaluated, scores far better — and you can always add to it.

Write down your thinking while you are doing it, not afterwards. And if you are wondering whether to take computer science at GCSE, ask me during the taster unit. I will give you a straight answer.