HUMPHREY PERKINS SCHOOL
Lionheart Educational Trust

Humphrey Perkins School Computing Curriculum

Overarching Vision

All pupils are confident at using technology; equipped with computational thinking skills enabling them to be problem solvers, innovators and creators.

"A High-quality computing education equips pupils to use computational thinking and creativity to understand and change the world" National Curriculum 2014

Success in computing enables learners to apply their computational thinking skills independently when solving problems. To do this we aim to provide a knowledge rich curriculum that is inspiring and has a practical approach to learning. The computational knowledge and skills developed by learners will enable them to be prepared to live in a digital world but also provide them with a secure foundation to access a computing or STEM related career aspiration.

"Computing also ensures that pupils become digitally literate – able to use, and express themselves and develop their ideas through, information and communication technology – at a level suitable for the future workplace and as active participants in a digital world" National Curriculum 2014

Lionheart learners come from a diverse range of backgrounds and from across lots of different primary schools where their previous learning has been dependent on the availability of resources or Computing specialists. We want learners at the end of their learning journey to feel confident and competent when using technology. This is done by giving learners the opportunity to practice and apply their theoretical knowledge through the use of digital products.

In Lionheart secondary schools, the curriculum is seen as a 5-year or 7-year journey.

Elements of Computing Teaching

Modelling

Learners will be challenged with many activities including the use of ICT as well as tracing programs and algorithms. These are all skills that learners will need to practise. To facilitate and aid this we model those techniques so we can demonstrate the correct methods and highlight best practice. For example, when teaching programming we integrate the use of live coding within our lessons. Or if we are teaching how to use a new piece of software we will model the task to the learners first.

Scaffolding

Often, learners can become disheartened and unmotivated when presented with problems. Problems and errors are communicated as a norm within lessons and

praise is provided when learners show resilience towards solving these problems. When teaching students how to code we model the presence of errors as a positive thing because it enables students to problem solve.

Learners are also encouraged to be independent with problem solving. They will be encouraged to rely on using search engines and provided resources effectively rather than relying on teacher support. Additionally, where appropriate we would use the concept of 'pair programming' where two programmers work together. The driver controls the computing device and writes the code, while the navigator provides direction, spots errors, and thinks ahead to the next part of the project. These roles are swapped around. The paired programming technique has been proven to increase engagement and reduce cognitive load for learners.

Teachers provide explanations, model using direct instruction and provide examples of work to give pupils access to appropriate material with the aim of narrowing attainment gaps.

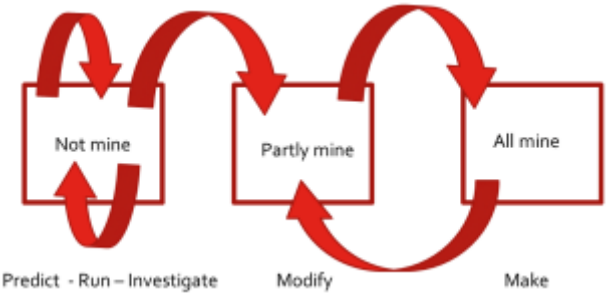
Where a pupil might struggle to learn a concept, we use concrete examples and analogies to guide them (e.g. when explaining the concept of RAM, we would compare it to keeping regularly used books from a library behind the desk rather than searching for them every time you want to use them).

Feedback

In Computing feedback can is given throughout a lesson, over a sequence of lessons and following assessments. Feedback can be verbal (individual or whole class) or written (this can be done as live marking). Learners are given opportunities to respond to feedback which can also take many forms including: making corrections (this maybe evidenced through the use of version control or resubmissions within Microsoft Teams); identifying next steps or "narrowing the gap" tasks.

PRIMM

When teaching programming we use the PRIMM approach. This is a strategy that encourages the use of reciprocal reading to question, clarify, summarise and predict how programs work by encouraging learners to read code before writing their own. It reduces cognitive load by unpacking and understanding what program code is doing. There are five stages to PRIMM:



https://primmportal.com/

Predict – learners read code and discuss what they think the code is doing.

Run – learners run the code to see what it actually does

Investigate – learners are provided with a variety of activities which include tracing, annotating and questioning the code.

Modify – learners modify the code to add new functionality

Make – learners create a new program based on the skills they have been learning

Project Based Learning

Learners are provided with opportunities to apply the skills and knowledge they have learnt through the use of project-based activities. Whereby learners are provided with a goal, audience and brief to fulfil. Allowing them to make autonomous decisions about the skills, knowledge and tools they will need.

"*This allows learners to develop their skills and understanding beyond computing, as they involve them imagining, making, and sharing their ideas*" The Big book of Computing Pedagogy, Hello World, Raspberry Pi Foundation

Cultural Capital

We provide learners with a culturally relevant curriculum whereby computing is contextualised. Teachers ensure that learners understand the relevance of the curriculum to everyday life. For example, when learning about network hardware students are provided with examples of why this knowledge is useful to them (e.g. They may want to build a network at home or for a possible business).

We do our best to make cross curricular links to other subjects or link to specific times within our School's calendar (e.g. we may provide a project that coincides with Black History Month).

We encourage our learners to have discussions that are topical and where possible link those conversations back to the curriculum (e.g. when Facebook, WhatsApp and Instagram stopped working we used this as an opportunity for students to learn about networking and the Border Gateway Protocol).

Additionally, we actively encourage learners to participate in extra curricular activities that take place within schools, across trust or externally (e.g. these could be a lunch time club or a competition such as the Bebras challenge).

Curriculum Overview

The computing curriculum has been designed to allow learners to review their knowledge and practice their skills regularly. This has been done through careful planning that ensures each topic is visited more than once over a period of time. For example, in year 7 students may be introduced to the word binary and learn that it is 0's and 1's. In year 8 they may learn how to convert between denary and binary numbers. In year 9 they would then learn how images and sound are represented

using binary numbers and in year 10 they will learn how to perform binary addition and binary shifts.

Retrieval practice is used to help learners recall knowledge and skills they have previously been taught. This may appear as starter activities where that activity may have no relevance to the current topic being taught. Or it could be mid-way through the lesson to show links to how that topic links to previous learning. For example, when we teach students about the fetch-decode-execute cycle we explain that this is an abstraction of what the CPU does. We can then link this back to when we taught computational thinking and students learnt about what the word 'Abstraction' means.

Key Stage 3 Learning

To ensure that the breadth of the Computing curriculum is covered at Key stage 3 we have adopted the CAS Computing Progression Pathways. Teachers teach from a suite of topics which are assessed against these progression pathways. We ensure that learning is both spaced and sequenced so by the end of Key stage 3 each learner will have been assessed against each pathway multiple times.

Algorithms
Programming & Development
Data & Data Representation
Hardware & Processing
Communication & Networks
Information Technology

CAS Progression Pathways

Key Stage 4 Learning

At Key stage 4 learners can choose to learn up to two of the following qualifications:

- GCSE OCR Computer Science
- OCR Cambridge National level 2 Certificate in iMedia
- BTEC Level 2 Digital Information Technology