Highcliffe School

What will we be learning?

Unit 1 Fundamentals of programming

Why this? Why now?

Reminder of the content from the GCSE specification as well as the focus given to the more complex elements of the content, such as subroutines and files and exception handling. These may have only just been explored when covering lower down the school.

Key Words:

algorithm
structured programming
data type
variables
constants
assignment
arithmetic operations
Boolean operators
sequence
selection
definite and indefinite iteration
top down design
modular programming
subroutine
procedure
function
parameter
argument
exception handling
global and local variables
field
record
binary file
text file
data structure

What will we learn?

This unit covers the fundamentals of programming, while recognising that some students may have had limited previous experience of programming and others will already be seasoned programmers. It covers the principles of structured programming in a procedural language such as Python, arrays, subroutines, parameter passing and text and binary files. Each of the six topics may be spread over more than one lesson, especially if time is spent in the lessons coding solutions and going over homework tasks.

These theory lessons could be run in parallel with practical programming sessions, and it is recommended that students code the pseudocode solutions that they write to give extra experience in practical programming.

- Topic 1 – Programming Basics
- Topic 2 – Selection
- Topic 3 – Iteration
- Topic 4 – Arrays
- Topic 5 – Subroutines
- Topic 6 – Files and Exception Handling

What opportunities are there for wider study?

Programming is a common theme throughout the course, wherever possible we try to include opportunities throughout future units, including Unit 2 Problem Solving, Unit 7 Data Structures, Unit 8 Algorithms, Unit 12 OOP and Functional.

How will I be assessed?

End of unit assessment on paper.

Continuing worksheet and home study tasks.

Through practical programming tasks.

Highcliffe School

What will we be learning?

Unit 2 Problem solving

Why this? Why now?

Underscoring the key elements for computational thinking and problem-solving link directly with the Unit 1 Programming Fundamentals and builds on the understanding. More complex and specific terms are introduced at this point, including more clarity of definitions for abstraction.

What will we learn?

It is a theoretical unit covering the AQA AS Computer Science specification section 3.4 Theory of computation. It describes what is meant by "computational thinking" and is designed to develop this skill with the aid of many practical examples related to problem solving, abstraction and algorithm design.

- Topic 1 – Computational Thinking
- Topic 2 – Structured Programming
- Topic 3 – Writing and Interpreting Algorithms
- Topic 4 – Testing
- Topic 5 – Abstraction and Automation
- Topic 6 – Finite State Machines

What opportunities are there for wider study?

These elements are specific to the whole curriculum and, generally speaking, are the core of what makes up a computer scientist. Opportunities to apply this with a real-world setting exist in the Unit 11 Databases and Software Development, as well as in the NEA Project.

How will I be assessed?

Programming activities.

Formal end of unit assessment.

Worksheet activity

Key Words:

computational
thinking
algorithm
logic problem
problem-solving
strategy
simulation
enumeration
theoretical approach
trial and error
creative solution
brute force method
divide and conquer
decrease and conquer
structured
programming
sequence
selection
iteration
top down design
hierarchy chart
test plan
normal
boundary and invalid
data
erroneous data
trace table
abstraction
information hiding
procedural abstraction
functional abstraction
data abstraction
decomposition
composition
automation
finite state machine
automaton
finite state diagram
transition
start state
accept state
state transition table

Highcliffe School

What will we be learning?

Unit 3 Data representation

1

Why this? Why now?

Part of the jump up to A Level means that the initial understanding of the GCSE core principles (Programming/Problem Solving/Computer Hardware) need to be extended. The Data Representation unit is a logical extension of this, as links to the early programming tools and computer hardware directly. Students extend the GCSE understanding and add more complexity, such as how decimal and negative numbers are stored.

What will we learn?

This unit covers the representation of data in Section 3.5 of the specification. Six topics in this unit cover data representation of numbers, text, images and sound, with the final topic explaining and giving examples of the uses of data compression and encryption. Each of the six topics may be spread over more than one lesson, especially if time is spent in the lessons coding solutions and going over homework tasks.

- Topic 1 – Number Systems
- Topic 2 – Bits, Bytes and Binary
- Topic 3 – Binary Arithmetic
- Topic 4 – Representing Images
- Topic 5 – Representing Sound
- Topic 6 – Data Compression and Encryption Algorithms
- Appendix – Floating Point Form

What opportunities are there for wider study?

Links from this unit present themselves in the Unit 4 – Hardware and Software and Unit 5 – Computer Organisation and Architecture. These investigate how the CPU uses binary instructions to execute.

Lots more in-depth information exists but a very good place to start could be on the 'Computerphile' YouTube channel, although this can go beyond the specification for A Level.

How will I be assessed?

Formal End of Unit assessment

Classwork (including home work sheets)

Responses in lesson

Key Words:

natural
rational
irrational
hexadecimal
binary
signed and unsigned
kibi
mebi
gibi
ASCII
Unicode
parity
checksum
check digit
overflow
raster
bitmap
resolution
bit or colour depth
sample
MIDI
frequency
Hertz
lossy
lossless
compression
encryption
ciphertext
plaintext
cryptanalysis.

Highcliffe School

What will we be learning?

Unit 4 Hardware and software

Why this? Why now?

A core level of understanding is needed for the further principles of Unit 5 Computer Architecture. This unit builds on the knowledge of the GCSE Section 1 Computer Systems and provides some more detailed investigation into the physical components.

What will we learn?

It is a theoretical unit covering all of Fundamentals of Computer Systems in the AQA AS Level Specification 7516. This is the first major 'step up' through the Boolean Algebra section, as this is not really covered at all in the GCSE speciation and needs a lot of time and focus to ensure that the questions can be solved independently within exams.

- Topic 1 – Hardware and Software
- Topic 2 – Role of an Operating System (OS)
- Topic 3 – Programming Language Classification
- Topic 4 – Programming Language Translators
- Topic 5 – Logic Gates
- Topic 6 – Boolean Algebra

What opportunities are there for wider study?

Unit 5 uses this initial understanding and provides more detail in the way in which the processor receives instructions and actions this. Students could look to research the development of computer systems and the way in which different hardware function (such as input, storage and output devices) through link on YouTube.

How will I be assessed?

End of unit assessment.

Worksheet and home study task completion.

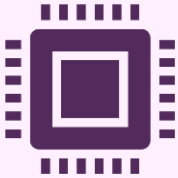
Key Words:

hardware
general-purpose/special-purpose software
operating system
utility programs
defragmenter
virus checker
library program
translator
virtual machine
processor scheduling
interrupt
embedded system
machine code
assembly language
assembler
compiler
interpreter
bytecode
logic gate
truth table
Boolean algebra

Highcliffe School

What will we be learning?

Unit 5 Computer organisation and architecture



Why this? Why now?

Builds directly on from the Unit 4 content, with a focus on how the instructions are received and processed. This builds into the completion of some assembly language programming.

What will we learn?

This unit covers the theoretical content of the internal hardware components of a computer, different architectures and the stored program concept. The fetch-execute cycle is explained in a detailed and practical way including the role of the major components and dedicated registers used by the processor. Instruction sets and addressing are covered along with basic machine code and assembly language operations.

- Topic 1 – Internal Computer Architecture
- Topic 2 – The Processor
- Topic 3 – The Processor Instruction Set
- Topic 4 – Assembly Language
- Topic 5 – Input / Output Devices
- Topic 6 – Secondary Storage Devices

What opportunities are there for wider study?

Private investigations using the lendable microbit from the library.

Challenge document from the assembly language project.

How will I be assessed?

Through end of unit test.

Including the requirement to create assembly language programs.

Online project (based on learning assembly language).

Worksheets and home study.

Key Words:

processor
main memory
address bus
data bus
control bus
I/O controller
von Neumann
Harvard
addressable memory
stored program concept
fetch
decode
execute
arithmetic logic unit
control unit
clock
register
buffer
instruction set
opcode
operand
immediate addressing
direct addressing
machine-code
branch
logical bitwise operator
logical shift
assembly language
cores
RFID
polarisation
pulse
flash
block
page
transistors
latency.

Highcliffe School

What will we be learning?

Unit 6 Communications

Why this? Why now?

What will we learn?

It is a theoretical unit covering all of the Fundamentals of Communication and Networking and Consequences of Uses of Computing from the AQA Computer Science specification. It builds on the fundamentals covered in the GCSE units on Networking and then extends this knowledge. The unit concludes with two lessons on communications and privacy and the social, legal and cultural issues presented by the use of computers and communication methods in today's world.

- Topic 1 – Communication Methods
- Topic 2 – Network Topology
- Topic 3 – Client-Server and Peer-to-Peer
- Topic 4 – Wireless Networking
- Topic 5 – Communication and Privacy
- Topic 6 – Social, Legal and Cultural Issues

What opportunities are there for wider study?

How will I be assessed?

Key Words:

serial transmission
parallel transmission
USB (Universal Serial Bus)
synchronous transmission
asynchronous transmission
start bit
stop bit
baud rate
bit rate
bandwidth
latency
protocol
star topology
bus topology
network
client-server networking
peer-to-peer networking
Wi-Fi
wireless access point
WPA
WPA2
SSID
MAC
CSMA/CA
RTS/CTS
privacy

Highcliffe School

What will we be learning?

Unit 7 Data structures

Why this? Why now?

Builds on the understanding from Unit 1-3 and creates more complex scenarios using the core building blocks of elementary and composite data types.

What will we learn?

It covers all of Section 4.2 of the AQA A-Level specification 7517. (Arrays, records and files are covered in AS Unit 1.) The unit gives practical and worked examples of each of the different abstract data structures including queues, stacks, lists, graphs, trees, hash tables and dictionaries. The function and practical application of each data type is discussed, with pseudocode and coded program solutions for relevant algorithms in Python.

- Topic 1 – Queues
- Topic 2 – Lists
- Topic 3 – Stacks
- Topic 4 – Hash Tables
- Topic 5 – Graphs
- Topic 6 – Trees
- Topic 7 - Vectors

What opportunities are there for wider study?

Continued focus on elements within this in Unit 8 Algorithms as well as more principles of these elements (lists/stacks/queues) within the Object Orientated Programming of Unit 12.

Students will likely utilise these data structures in their preliminary materials (Prelim Code) as well as in their own NEA Projects.

How will I be assessed?

Through practial programming activities.

Whilst completing theory based worksheets, including tracing how these principles are applied.

Formal end of unit assessment

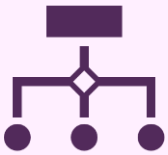
Key Words:

elementary data type
composite data type
abstract data type
encapsulation
information hiding
static data structure
dynamic data structure
heap
overflow / underflow
queue
circular queue
priority queue
First In First Out (FIFO)
Enqueue / dequeue
Append / push
pop
stack
Last In First Out (LIFO)
call stack
stack frame
parameter
return address
Hashing
hash table
collision
mid-square method
folding method
dictionary
graph
edge / arc
vertex / node
directed graph / digraph
undirected graph
weighted edge
adjacency matrix
adjacency list
Page Rank algorithm
Tree
root
child
parent
subtree
leaf node
binary search tree
pre-order
in-order and post-order
traversal

Highcliffe School

What will we be learning?

Unit 8 Algorithms



Why this? Why now?

Use of the principles from Unit 7 throughout the new ground that this unit explores. Links directly back to the main programming logic on Unit 1 and Unit 2 but expands the use in more complex methods.

What will we learn?

This is a theoretical unit covering all of Section 4.3 Fundamentals of algorithms (except Reverse Polish which is covered in Unit 9). Searching and sorting algorithms are covered in an interactive and practical way, with reference to Big-O notation in terms of time and space complexity. It also covers on the role of stack frames in subroutine calls, and recursive techniques, putting these into practice with tree traversals and a depth-first graph traversal. Optimisation algorithms, such as Dijkstra's shortest path algorithm are covered along with a complete topic on the limits of computation, including intractable algorithms and the Halting problem.

- Topic 1 – Recursive Algorithms
- Topic 2 – Big-O Notation
- Topic 3 – Searching and Sorting
- Topic 4 – Graph Traversal Algorithms
- Topic 5 – Optimisation Algorithms
- Topic 6 – Limits of Computation

What opportunities are there for wider study?

How will I be assessed?

Through practial programming activities.

Whilst completing theory based worksheets, including tracing how these principles are applied.

Formal end of unit assessment

Key Words:

recursion
recursive subroutine
call stack
tree traversal
pre-order
in-order
post-order traversal
Big-O notation
linear
polynomial
exponential
logarithmic
functions
permutation
time complexity
linear search
binary search
binary tree search
bubble sort
merge sort
depth-first traversal
breadth-first
traversal
optimisation
problem
Dijkstra's shortest
path algorithm
limits of
computation
travelling salesman
problem (TSP)
computational
problem
tractable and
intractable problems
heuristic solution
computable and
non-computable
problems
Halting problem

Highcliffe School

What will we be learning?

Unit 9 Regular languages

Why this? Why now?

What will we learn?

After covering the structure listed below the students being given plenty of opportunity to practise skills and techniques throughout each lesson.

- Topic 1 – Mealy Machines
- Topic 2 – Sets
- Topic 3 – Regular Expressions
- Topic 4 – Turing Machine
- Topic 5 – Backus-Naur Form
- Topic 6 – Reverse Polish Notation

What opportunities are there for wider study?

How will I be assessed?

Formal End of Unit assessment

In class through engagement with instruction

In worksheet and home work tasks.

Key Words:

Finite state machine
Mealy machine
transition
transition condition
state
state transition table
set
member
element
ordered / unordered
common sets
set comprehension
compact representation
membership
union
intersection
difference
subset
proper subset
Cartesian product
Infinite / finite
countably infinite
cardinality
Regular Expressions
precedence
regular language
decompose
| * + ? ()
Turing machine
state transition diagram
tape
read-write head
halting
δ
Universal Turing machine
computable
Backus-Naur Form
Terminal / Non-terminal
pipe
syntax diagram
parsing
parse tree
Infix
prefix
postfix
Polish Notation
Reverse Polish Notation
order of precedence

Highcliffe School

What will we be learning?

Unit 10 The Internet

Why this? Why now?

Taught alongside Unit 6 – Networks, due to the direct links between how these are used to create the Wide Area Network of the internet.

What will we learn?

It is a theoretical unit covering all of section 4.9.3 and 4.9.4 of the AQA 7517 specification. Internet functions including packet switching, DNS and the role of the router are covered in the first two topics of this unit. Symmetric and asymmetric encryption, and the use of digital signatures are covered in the following topic. Standard Application Layer protocols such as SSH are covered with reference to the TCP/IP protocol stack. Subnetting, DCHP and Network Address Translation are covered in the penultimate topic, rounded off with a final topic on web CRUD and RESTful applications in relation to the client server model.

- Topic 1 – Structure of the Internet
- Topic 2 – Packet Switching and Routers
- Topic 3 – Internet Security
- Topic 4 – TCP IP Standard Application Layer Protocols
- Topic 5 – IP Addresses
- Topic 6 – Client-server Model

What opportunities are there for wider study?

How will I be assessed?

Formal End of Unit assessment

In class through engagement with instruction

In worksheet and home work tasks.

Key Words:

Internet
World Wide Web
URL
Internet registry
registrar
DNS
FDQN
Internet Protocol
packet
packet switching
router
gateway
hop
header
NIC
firewall
filter
proxy server
port
stateful inspection
encryption
symmetric / asymmetric
public / private key
digital signature
hash
digital certificate
worm
Trojan
malware
virus
TCP
stack
protocol
MAC address
FTP
SSH
POP / SMTP / IMAP
server
browser
subnet mask
DHCP
Routable / Non-routable
NAT
port forwarding
client server model
API
CRUD
JSON / XML
REST
thick client / thin client

Highcliffe School

What will we be learning?

Unit 11 Databases and software development

Why this? Why now?

This has importance to be covered alongside the start up to the independent NEA project, as a lot of the elements cross over with the work that this unit demands.

What will we learn?

It is a theoretical unit the production of a data model, entity definitions and entity relationship diagram, and normalisation to Third Normal Form. Further we investigate the use of SQL to retrieve, update, insert and delete data from multiple tables in a database, and the creation of new tables. Practical exercises using MS Access and MySQL will enable students to see how relational databases may be set up to maintain referential integrity, and how a Query By Example translates into an SQL statement. Client server database and problems of concurrent databases are also covered.

- Topic 1 – Entity Relationship Modelling
- Topic 2 – Relational Database and Normalisation
- Topic 3 – Introduction to SQL
- Topic 4 – Defining and Updating tables using SQL
- Topic 5 – Systematic Approach to Problem Solving

What opportunities are there for wider study?

Massive cross over with a lot of the programming and broad understanding of the systems development life cycle in the NEA Project. Students get the opportunity to investigate creating a coded solution to the given project.

How will I be assessed?

Practical programming, with the help of the XAMMP system and PHPMySQL.

Worksheets and activities.

Formal end of unit assessment.

Key Words:

entity
attribute
identifier
primary key
composite primary key
foreign key
relationship
entity relationship (E-R) diagram
normalisation
relation
relational database
First Normal Form (1NF)
Second Normal Form (2NF)
Third Normal Form (3NF)
partial dependency
non-key dependence
data integrity
SQL
client-server database
record locking
serialisation
timestamp ordering
commitment ordering
agile modelling
prototyping

Highcliffe School

What will we be learning?

Unit 12 OOP and Functional Programming

Why this? Why now?

The Preliminary materials generally use some elements of OOP therefore this unit is timed to occur alongside the exploration of the pre-release.

Key Words:

object
class
attribute
method
encapsulation
information hiding
constructor
instantiation
inheritance
subclass
superclass
polymorphism
overriding
modifier
public
private
protected
class diagram
aggregation
composition
association
abstract method
virtual method
static method
interface

What will we learn?

It is a theoretical unit covering all of Object-oriented programming, Fundamentals of functional programming and Big Data. The first two lessons cover the basics of object-oriented programming and object-oriented design principles, with practical examples in Python. The three following lessons take the form of a tutorial in Haskell, a functional programming language, accompanied by theory to enable students to answer exam questions on this topic. The final lesson describes examples of Big Data, its application and benefits in areas such as healthcare and medicine, business, communication and many other fields.

- Topic 1 – Basic Concepts of OOP
- Topic 2 – OOP Design Principles
- Topic 3 – Functional Programming
- Topic 4 – Functional Application
- Topic 5 – Lists in Functional Programming
- Topic 6 – Big Data

What opportunities are there for wider study?

Multiple online courses that can use OOP and Functional programming in their methods. Will likely use within the preparation for the Prelim (paper 1) as well as possibly within their NEA Project, depending on student choice.

How will I be assessed?

Formal end of unit assessment

Worksheet and home study.

Highcliffe School

What will we be learning?

NEA Introduction

Why this? Why now?

Initial introduction to the NEA timed to allow for the initial project ideas to be formed, approved and even designed.

Key Words:

What will we learn?

An awareness for process and protocols that exist in many industry situations, requiring approval from a manager role before moving on to the next stages in the work. Students need to find their own client to provide the project and manage the process independently. Students will experience the full process of analysing, planning, designing, creating, testing, and evaluating their coded solution.

Students will develop strong logical thinking, programming, and design skills. They will also can develop real world communication skills, having to interact with their client in a formal interview, collect data from users (via a survey or questionnaire) and discover the current practises of their problem.

What opportunities are there for wider study?

Students are provided with a companion book and example work to help them on this journey. The work from Unit 11 is relied upon heavily, even with the basis of their project being formed from the same structures.

Depending on the choice of project this can be used to investigate the online systems life cycle, through a website and 'backend' design or to improve their knowledge and experience of using Object Orientated Programming. The skills that each student develops are unique to their demands in their selected project.

How will I be assessed?

Students will generate a project approval form, which will be submitted to staff before moving forwards with their project.

The whole NEA is assessed through a portfolio of evidence, including the stages listed above. This document is normally created online and shared with staff so that the progress can be seen.

Highcliffe School

What will we be learning?

NEA Project Write Up

Why this? Why now?

After the stages of initial approval and analysis the design stage leads on to the creation of the coded solution. This is where the majority of the practical process is applied, but it is important to also ensure that it is effectively documented.

Key Words:

What will we learn?

Students will investigate techniques of planning and documentation of their programming project, including the creation of the project itself. Students will have the support in lesson to help with the programmed solution but will also be required to spend time outside of lessons working on their solution.

Support is also given for the way in which the project needs to be documented. This includes adding code, planning, and completing testing, providing evaluation from both a programmer and user standpoint.

What opportunities are there for wider study?

This is one of the best opportunities to engage with a real-world project but there are also opportunities to take this even further. Students are able to use these skills to develop coded solutions for other individuals and therefore have a practical way of engaging with the technical landscape. Students have previously benefited from their project through the support that this solution needs and the skills that this demonstrates to their clients.

How will I be assessed?

There is a formal process for marking the pojects, including a break down of the marks for each section. This is shared with the students and discussed when completing each section.

More information can be found - https://www.aqa.org.uk/subjects/computer-science-and-it/as-and-a-level/computer-science-7516-7517/subject-content-a-level/non-exam-assessment-the-computing-practical-project

Highcliffe School

What will we be learning?

Prelim Code

Why this? Why now?

Paper 1 of the specification is broken down into four part and the Preliminary Code is featured in two of these. Investigating this process is important as it gives students a thorough understanding before entering the exam.

Key Words:

What will we learn?

Students will explore the different parts of the code and begin to understand how this has been constructed. There is an opportunity for collaborative work here, as they are able to discover how each section works and describe this to one another.

Paper 1 is split into four section:

A. General programming theory, covering units 7-12
B. Programming task, unrelated to the preliminary materials
C. Theory based on the prelim, from units 7-12
D. Programming tasks based on the preliminary materials

What opportunities are there for wider study?

These tasks can be explored more thoroughly independently as there is ample time before the formal assessment process. Students will have a chance to experience this kind of assessment in their Year 13 mock.

How will I be assessed?

On screen exam based questions and submission via electronic answer document (in a similar manner to the exam).

Highcliffe School