Summary

Unit 4 Computational thinking, problem-solving and programming

Subject	Year	Start date	Duration
Computer Science	IB1	Week 1, September	12 weeks 45 hours

Course Part

Description

The focus of this topic is how an understanding of programming languages enhances the students' understanding of computational thinking and provides them with opportunities for practical, hands-on experience of applying computational thinking to practical outcomes.

In externally assessed components questions will be presented using flow charts and/or pseudocode as outlined in the approved notation sheet. Answers will only be required in pseudocode.

Students must be given the opportunity to convert algorithms into working code that is executed and tested.

Working code will not be assessed in the externally assessed components.

Inquiry & Purpose

Inquiry / Higher Order Questions

Type	Inquiry Questions
Skills-based	Design cycle (inputs, processes, outputs, feedback, iteration); use of software such as Alice.
Content-based	Using flow charts to solve problems in real-life contexts, patterns and sequences, logic, algorithms.

Curriculum

Aims

Develop logical and critical thinking as well as experimental, investigative and problem-solving skills

Develop an appreciation of the possibilities and limitations associated with continued developments in IT systems and computer science

Encourage an understanding of the relationships between scientific disciplines and the overarching nature of the scientific method

Objectives

Know and understand

relevant facts and concepts

appropriate methods and techniques

computer science terminology

Apply and use

relevant design methods and techniques

Construct, analyse, evaluate and formulate

success criteria, solution specifications including task outlines, designs and test plans

appropriate techniques within a specified solution

Syllabus Content

Core

Topic 4 - Computational thinking, problem-solving and programming

4.1 General principles

Thinking procedurally

4.1.1 Identify the procedure appropriate to solving a problem.

4.1.2 Evaluate whether the order in which activities are undertaken will result in the required outcome.

4.1.3 Explain the role of sub-procedures in solving a problem.

Thinking logically

4.1.4 Identify when decision-making is required in a specified situation.

4.1.5 Identify the decisions required for the solution to a specified problem.

4.1.6 Identify the condition associated with a given decision in a specified problem.

4.1.7 Explain the relationship between the decisions and conditions of a system.

4.1.8 Deduce logical rules for real-world situations.

Thinking ahead

4.1.9 Identify the inputs and outputs required in a solution.

4.1.10 Identify pre-planning in a suggested problem and solution.

4.1.11 Explain the need for pre-conditions when executing an algorithm.

4.1.12 Outline the pre- and post-conditions to a specified problem.

4.1.13 Identify exceptions that need to be considered in a specified problem solution.

Thinking concurrently

4.4.14 Identify the parts of a solution that could be implemented concurrently.

4.4.15 Describe how concurrent processing can be used to solve a problem.

4.1.16 Evaluate the decision to use concurrent processing in solving a problem.

Thinking abstractly

4.4.17 Identify examples of abstraction.

4.4.18 Explain why abstraction is required in the derivation of computational solutions for a specified situation.

4.4.19 Construct an abstraction from a specified situation.

4.1.20 Distinguish between a real-world entity and its abstraction

4.2 Connecting computational thinking and program design

4.2.1 Describe the characteristics of standard algorithms on linear arrays.

4.2.2 Outline the standard operations of collections.

4.2.3 Discuss an algorithm to solve a specific problem.

4.2.4 Analyse an algorithm presented as a flow chart.

4.2.5 Analyse an algorithm presented as pseudocode.

4.2.6 Construct pseudocode to represent an algorithm.

4.2.7 Suggest suitable algorithms to solve a specific problem.

4.2.8 Deduce the efficiency of an algorithm in the context of its use.

4.2.9 Determine the number of times a step in an algorithm will be performed for given input data.

4.3 Introduction to programming

Nature of programming languages

4.3.1 State the fundamental operations of a computer.

4.3.2 Distinguish between fundamental and compound operations of a computer

4.3.3 Explain the essential features of a computer language.

4.3.4 Explain the need for higher level languages.

4.3.5 Outline the need for a translation process from a higher level language to machine executable code.

Use of programming languages

4.3.6 Define the terms: variable, constant, operator, object.

4.3.7 Define the operators =, ≠, <, <=, >, >=, mod, div.

4.3.8 Analyse the use of variables, constants and operators in algorithms.

4.3.9 Construct algorithms using loops, branching.

4.3.10 Describe the characteristics and applications of a collection.

4.3.11 Construct algorithms using the access methods of a collection.

4.3.12 Discuss the need for sub-programmes and collections within programmed solutions.

4.3.13 Construct algorithms using predefined sub-programmes, onedimensional arrays and/or collections.

ATL Skills

Approaches to Learning

Thinking

- In this unit, we will

  ask students to formulate a reasoned argument to support their opinion or conclusion

  give students time to think through their answers before asking them for a response

  reward a new personal understanding, solution or approach to an issue

  ask questions that required the use of knowledge from a different subject from the one you are teaching

  include a reflection activity

  make a link to TOK

Research

- In this unit, we will

  assign a task that required students to use the library

  require students to practise effective online search skills (for example, use of Booleans and search limiters)

  provide opportunities for students to reflect on how they determine the quality of a source, or analyse contradictory sources

  give students advice on (or provide an opportunity for students to practise) narrowing the scope of a task to make it more manageable

Developing IB Learners

Learner Profile

Inquirers

Knowledgeable

Thinkers

Open-minded

Reflective

Assessment

Assessment criteria

External Assessment

Paper 1

A: Short answer questions

B: Structured questions

Description