Singapore Examinations and Assessment Board

Cambridge Assessment International Education

Singapore–Cambridge General Certificate of Education
Ordinary Level (2023)

Computing
(Syllabus 7155)

CONTENTS

AIMS

The syllabus aims to provide students with the foundation to continue with further studies in computing and skills to participate in a rapidly changing technological environment so that the concepts and skills learnt would also be applicable in other fields that require computing.
Specifically, the two-year course at upper secondary level is to enable students to

1. Apply logical reasoning and algorithmic thinking in analysing problem situations and developing solutions
2. Develop simple program through the use of appropriate programming language(s)
3. Understand how and where information communications technology (ICT) is used in daily life
4. Understand and explain the ethical, social and economic issues associated with ICT.

Information and Communications Technology (ICT) refers broadly to technology involving computing devices, software and other hardware. The computer science concepts and skills behind ICT will also be taught where appropriate.

ASSESSMENT OBJECTIVES

The examination will assess:

(a) Knowledge and understanding of basic computing technology and systems, concepts, algorithms, techniques and tools.

(b) Application of knowledge and understanding to analyse and solve computing problems.

(c) Development, testing and refinement of solutions using appropriate software application(s) and/or programming language(s).

Students can handle and process data in computer systems, as well as the need to be ethical when dealing with data. They will demonstrate problem-solving techniques through analysing and writing programming solutions for a range of computing problems in business, education, mathematics and science. Students will be able to demonstrate computational thinking through the design and development of programming solutions.

SCHEME OF EXAMINATION

All candidates will offer Paper 1 and Paper 2. All questions are compulsory in both papers.

Paper 1 (Written examination, 2 hours)

This paper tests knowledge, understanding and application of concepts and skills in all the four modules.
The questions consist of a mixture of:

- short answer questions
- matching questions
- cloze passage
- structured questions

This paper carries 70% of the total marks, and is marked out of 80 marks.

Paper 2 (Lab-Based Examination, 2 hours 30 minutes)

This paper, taken with the use of a computer, spreadsheet and programming[1] software, tests Module 1 (Unit 1.1: Data Management) and Module 4 (Programming). Four structured questions will be set based on the following topics:

- Use of Spreadsheet functions and features
- Refinement of program
- Debugging of program
- Development of program of no more than 40 lines

Development of program will carry 20 marks. The remaining three questions average 10 marks.

A quick reference for Python will be provided for candidates.

Candidates will submit soft copies of the required work for marking. The allotted time includes time for saving the required work in the candidates' computer. This paper carries 30% of the total marks, and is marked out of 50 marks.

Summary details for each paper:

Paper	Mode	Duration	Weighting	Marks	Format	Modules Assessed
1	Written	2 h	70%	80	A mixture of • Short answer questions • Matching questions • Cloze passage • Structured questions	All the four modules
2	Lab-based	2 h 30 min	30%	50	4 compulsory structured questions • Use of Spreadsheet functions and features • Refinement of program • Debugging of program • Development of program with no more than 40 lines of code Development of program will carry 20 marks. The remaining three questions average 10 marks. A quick reference for Python will be provided for candidates.	Unit 1.1 Data Management from module 1 Module 4: Programming

[1] The centre will be required to declare the version of the programming language to be used for the examination before the centre begins teaching the course for the cohort taking the examination.

SPECIFICATION GRID

Assessment Objectives	Paper 1	Paper 2	Overall
(a) Knowledge and understanding	~35%	~5%	40%
(b) Application	~25%	~5%	30%
(c) Development, testing and refinement	~10%	~20%	30%
TOTAL	70%	30%	100%

USE OF CALCULATOR

An approved calculator may be used in Paper 1 and Paper 2.

CENTRE INFRASTRUCTURE FOR LAB-BASED EXAMINATION

The centre will ensure adequate hardware and software facilities to support the examination of its candidates for Paper 2, which will be administered over at most two shifts on the day of the examination. Each candidate should have the sole use of a personal computer for the purpose of the examination. Candidates should be able to access Spreadsheet application software and programming language software. The centre will be required to declare the name and version number of the software to be used for the cohort sitting for the examination at least two years before the cohort sits for the examination.

CONTENT OUTLINE

The syllabus consists of four modules: (1) Data and Information, (2) Systems and Communications, (3) Abstraction and Algorithms, and (4) Programming.

MODULE 1 – DATA AND INFORMATION

This module is about the handling and processing of data in computer systems, and the need to be ethical when dealing with data. Students should be able to identify different types of data, understand and explain what the data is for, and explain how the data is represented or organised for processing and output with reference to a given problem. Students will be more aware of ethical issues with respect to data, including privacy of data.

MODULE 2 – SYSTEMS AND COMMUNICATIONS

This module is about systems involving computer technology and computing devices. Students will learn the inter-relationships between parts and whole of a system; as well as the functions of parts of systems in enabling communications between human and computing device (machine), machine and machine, and within a machine.

MODULE 3 – ABSTRACTION AND ALGORITHMS

This module is about problem solving and how a problem may be solved by breaking it into smaller, manageable parts and solving all the smaller parts. An algorithm describes a solution for the problem that is independent of a programming language and may be presented in pseudo-code (where program structures will be more pronounced) or diagrammatically (flowchart). Students should be able to know the difference between pseudo-code and flowchart.

MODULE 4 – PROGRAMMING

This module is about application and development of logical thinking and reasoning, as well as problem-solving skills through the design and development of software solutions using programming language(s). An algorithm describes a solution independent of a programming language while a programming language depicts the solution that is workable on a computing device.

LEARNING OUTCOMES

The learning outcomes for each module are as follows:

MODULE 1 – DATA AND INFORMATION

Units

1.1 Data Management

1.2 Data Representation

1.3 Ethical, Social and Economic Issues

1.1 Data Management

This section emphasises logical thinking and reasoning through data analysis, data processing and visual representations of data. Students will use the Spreadsheet application software in hands-on activities to enhance learning.

<table>
<tr><th>Ref</th><th>Learning outcome</th></tr>
<tr><td></td><td>Students should be able to:</td></tr>
<tr><td>1.1.1</td><td>Tabulate data under appropriate column headings (i.e. data field names) and data types (e.g. numeric, text and date).

Exclude: plotting of charts</td></tr>
<tr><td>1.1.2</td><td>Use mathematical operators, functions and what-if data analysis (goal seeking) to prepare spreadsheets and solve real-world problems such as:
• find the total of a list of numbers, average of a list of numbers, min/max of a list of numbers, square root of a number, simple interest, remainder after division of numbers
• round values
• randomise values
• convert data from one type to another (e.g. get integer values from decimal values) and
• count number of data items.

Understand and use conditional statements (simple and nested): COUNTIF and IF with relational and Boolean operators such as AND, NOT and OR.

Include: conditional formatting

Use functions effectively to look up data in rows or columns (horizontal and vertical table lookups) in a list or table for data processing.

The list of 33 examinable functions are:
<table>
<tr><th>Area</th><th>Function</th></tr>
<tr><td>Date and Time</td><td>TODAY</td></tr>
<tr><td>Text</td><td>LEFT, LEN, MID, RIGHT</td></tr>
<tr><td>Logical</td><td>AND, IF, NOT, OR</td></tr>
<tr><td>Lookup</td><td>HLOOKUP, VLOOKUP</td></tr>
<tr><td>Mathematical</td><td>CEILING.MATH, FLOOR.MATH, MOD, POWER, QUOTIENT, RAND, RANDBETWEEN, ROUND, SQRT, SUM, SUMIF</td></tr>
<tr><td>Statistical</td><td>AVERAGE, COUNT, COUNTA, COUNTBLANK, COUNTIF, LARGE, MAX, MEDIAN, MIN, MODE.SNGL, SMALL</td></tr>
</table>
Students should be able to use absolute and relative cell referencing.</td></tr>
</table>

1.2 Data Representation

This section explains how data is represented internally as binary numbers, how the conversion of data from one number system to another is done, and where these number systems are used. The number systems covered here are binary number system, denary number system and hexadecimal number system.

Ref	Learning outcome
	Students should be able to:
1.2.1	Represent positive whole numbers in binary form.
1.2.2	Convert positive whole numbers from one number system to another – binary, denary and hexadecimal, and describe the technique used.
1.2.3	Describe situations in which the number systems are used such as ASCII codes, IP (Internet Protocol) and Media Access Control (MAC) addresses, and RGB codes.

1.3 Ethical, Social and Economic Issues

This section covers ethical use and security of data in desktop applications or applications on networks like the internet. Social and economic issues arising from the use of computing technologies are discussed.

Ref	Learning outcome
	Students should be able to:
1.3.1	Understand how data can be kept safe from accidental damage due to data corruption or human errors and malicious actions such as unauthorised viewing, deleting, copying and corrupting or malware.
1.3.2	Understand the effects of threats to privacy and security of data from spam, spyware, cookies, phishing, pharming and unauthorised access as well as defensive measures employed such as the use of appropriate hardware and software.
1.3.3	Describe ethical issues that relate to the use of computers, public/private networks, freeware, shareware and open courseware; and the sharing of information.
1.3.4	Compare the positive and negative social and economic impacts of technology in education, communication (e.g. via mobile apps and social media), finance, medicine/healthcare, transportation and entertainment; and explain plagiarism and software piracy.

MODULE 2 – SYSTEMS AND COMMUNICATIONS

Units

2.1 Computer Architecture

2.2 Data Communications

2.1 Computer Architecture

This section covers the basic computer architecture and components of a computer network. Students will understand the purpose of hardware and software required in a computer system or network but they are not required to know how these work technically.

Ref	Learning outcome
	Students should be able to:
2.1.1	Describe basic computer architecture with reference to: • Computer Processor • Memory • Data and address buses • Input (e.g. data and instructions) and output (e.g. intermediate and final results of processing) • Storage media *Exclude: fetch and execute cycle*
2.1.2	Recognise a logic gate from its truth table and evaluate Boolean statements by means of truth tables.
2.1.3	Construct the truth tables for given logic circuits (maximum 3 inputs).
2.1.4	Design and construct simple logic circuits using AND, OR, NOT, NAND and NOR.

2.2 Data Communications

This section uses networks as the context for understanding data communications. Students should have a general understanding of how data is transmitted and the need to check for data accuracy and have data security in transmissions.

Ref	Learning outcome
	Students should be able to:
2.2. 1	Identify and explain the function of different network hardware: modem, network interface controller, hub, switch and router.
2.2.2	Describe the difference between wired and wireless networks and explain the factors that will determine the use of each type of network. *Include: descriptions of LAN/MAN/WAN*
2.2.3	Describe the components for a simple home network and design a simple home network.
2.2.4	Compare and contrast client-server and peer-to-peer network strategies with emphasis on: • Purpose • Function • Organisation • Bandwidth *Include: topology principle of bus, ring and star networks*
2.2.5	Explain the use of parity and checksums in data transmission.

MODULE 3 – ABSTRACTION AND ALGORITHMS

Units

3.1 Problem Analysis

3.2 Algorithm Design

3.1 Problem Analysis

This section covers problem interpretation and analysis. Students will learn how to approach problem solving in a systematic manner.

Ref	Learning outcome
	Students should be able to:
3. 1.1	For a given problem, specify the: • inputs and the requirements for valid inputs • outputs and the requirements for correct outputs Examples • Business: produce an itemised list of items purchased, cost of each item, total cost payable (like a receipt) or calculate interest on mortgages and print instalments over a period of time. • Education: find and print the student with the top score in each subject or check user inputs against expected input (like test scoring); find the mean subject grade (MSG) for a class. • Scientific/Mathematics: determine whether an input number is odd or even and whether a number is divisible by another number. • Entertainment: a number guessing game or any game involving text manipulation.
3.1.2	Solve problems by decomposing them into smaller and manageable parts.
3.1.3	Identify common elements across similar problems and make generalisations.

3.2 Algorithm Design

This section is about interpreting and understanding algorithms; correcting and writing algorithms for given problems. Students' learning will be enhanced through written exercises and class discussions.

Ref	Learning outcome
	Students should be able to:
3.2.1	Perform a dry run of a set of steps to determine its purpose and/or output.
3.2.2	Produce trace tables to show values of variables at each stage in a set of steps.
3.2.3	Locate logic errors in an algorithm, and improve or modify an algorithm to remove the errors or for changes in task specification.
3.2.4	Produce an algorithm to solve a problem, either as a flowchart or in pseudo-code. The following pseudo-code keywords and constructs should be used: • INPUT / OUTPUT • IF… THEN… ELSEIF… ELSE… ENDIF • WHILE… ENDWHILE • FOR… NEXT

MODULE 4 – PROGRAMMING

Units

4.1 Program Development

4.2 Program Testing

4.1 Program Development

This covers the development of programming solutions (coding) for problem situations. Students will reinforce their understanding through practical work. Python is the programming language for this syllabus.

Ref	Learning outcome
	Students should be able to:
4.1.1	Understand and describe the stages in developing a program: gather requirements, plan solutions, write code, test and refine code, and deploy code.
4.1.2	Understand and use sequence, selection and iteration constructs to create a program.
4.1.3	Use and justify the use of variables, constants and simple lists in different problem contexts.
4.1.4	Understand and justify the use of different data types (integers, floating-point numbers, strings, Booleans, lists) and built-in functions.
4.1.5	Produce programming solution for a given problem to: • find min/max value in a list • find average of a list of numeric values • search for an item in a list and report the result of the search • find check digits • find the length of a string of characters • extract required characters from a string of characters
4.1.6	Write and use user-defined functions.

4.2 Program Testing

This covers the testing and refinement of programs based on test results. Students will reinforce their learning and understanding through hands-on practical work. Python is the programming language for this syllabus.

Ref	Learning outcome
	Students should be able to:
4.2.1	Identify and justify the use of data validation techniques.
4.2.2	Validate input data for acceptance by performing: • length check • range check • presence check • format check
4.2.3	Design appropriate test cases to cover normal, error and boundary conditions, and specify what is being tested for each test case.
4.2.4	Understand and describe types of program errors: syntax, logic and run-time, and explain why they occur.
4.2.5	Explain how translators are used to detect syntax errors and state the difference between interpreter and compiler translators.
4. 2.6	Understand and apply debugging techniques to isolate/identify and debug program errors: using intermittent print statements, walking through or testing a program in small chunks or by parts.

Quick Reference For Python

1. Identifiers

When naming variables, functions, and modules, the following rules must be observed:

- Names should begin with character 'a' – 'z' or 'A' – 'Z' or '_'
  and followed by alphanumeric characters or '_'.
- Reserved words should not be used.
- User-defined identifiers are case sensitive.

2. Comments and Documentation Strings

This is a comment

"""
This is a documentation string
over multiple lines
"""

3. Input/Output

print ("This is a string")

s = input ("Instructions to prompt for data entry.")

4. Import

import <module>

e.g. **import** math

5. Data Type

Data Type	Notes
int	integer
float	real number
bool	boolean
str	string (immutable)
list	series of values

6. Assignment

Assignment Statement	Notes
a = 1	integer
b = c	variable
d = "This is a string"	string
mylist = [1, 2, 3, 4, 5]	list

7. Arithmetic Operators

Operator	Notes
+ -	plus, subtract
* /	multiply, divide
%	remainder or modulus
**	exponential or power
//	quotient of the floor division

8. Relational Operators

Operator	Notes
==	equality
!=	not equal to
> >=	greater than, greater than or equal to
< <=	less than, less than or equal to

9. Boolean Expression

Boolean Expression	Notes
a and b	logical and
a or b	logical or
not a	logical not

10. Iteration

while loop	for loop
while condition(s): <statement(s)>	**for** i **in** range(n): <statement(s)>
	for record **in** records: <statement(s)>

11. Selection

Type 1
if condition(s)**:**   <statement(s)>

Type 2
if condition(s)**:**   <statement(s)> **else:**   <statement(s)>

Type 3
if condition(s)**:**   <statement(s)> **elif** condition(s)**:**   <statement(s)> **else:**   <statement(s)>

12. Functions

```
# Function definitions
def <function name> (<parameters>):
    <function body>
    return <return value>

# Function calls
<function name>(<arguments>)
```

13. Built-in Functions

(a) Basic functions

abs()	chr()	float()	input()	int()
len()	max()	min()	ord()	print()
range()	round()	str()	<str>.endswith()	<str>.find()
<str>.format()	<str>.isalnum()	<str>.isalpha()	<str>.isdigit()	<str>.islower()
<str>.isspace()	<str>.isupper()	<str>.lower()	<str>.split()	<str>.startswith()
<str>.upper()				

(b) Math module

ceil()	floor()	pow()	sqrt()	trunc()

(c) Random module

randint()

14. Reserved Words

Reserved words are part of the syntax of the language. They cannot be used as identifiers.

False	None	True	and	as
assert	break	class	continue	def
del	elif	else	except	finally
for	from	global	if	import
in	is	lambda	nonlocal	not
or	pass	raise	return	try
while	with	yield		