CAMBRIDGE
International Education

Cambridge IGCSE™

COMPUTER SCIENCE **0265/02**

Component 2 Algorithms and Programming **For examination from 2029**

MARK SCHEME

Maximum Mark: 75

Specimen

This document has **16** pages.

© Cambridge University Press & Assessment 2026

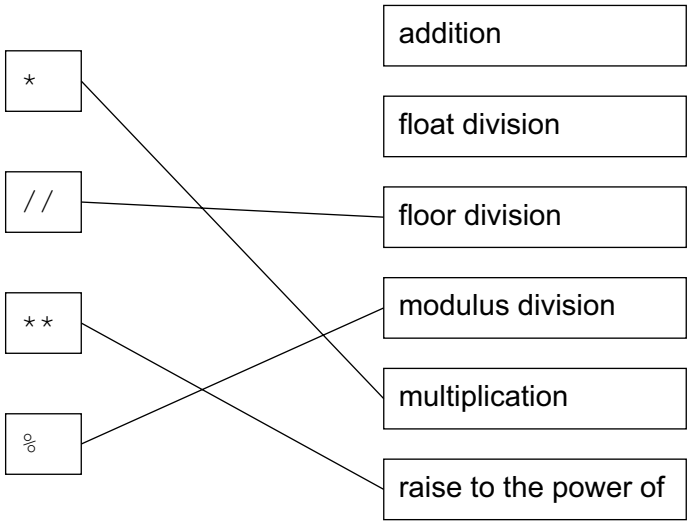
General Marking Principles

All examiners must apply these marking principles when marking candidate responses.

1	Examiners must award marks for each response in line with: • the specific **content** defined in the mark scheme • the specific **skills** defined in the mark scheme or in the level descriptions • the **standard of response** required as exemplified by the standardisation scripts.
2	Examiners must award **whole marks** (not half marks, or other fractions).
3	Examiners must award marks **positively**. This means that: • marks are not deducted for errors or omissions • marks are awarded for correct/valid answers as defined in the mark scheme and also for **other valid answers** which go beyond the scope of the syllabus and mark scheme referring to your team leader as appropriate • the quality of spelling, punctuation and grammar are judged **only** when the mark scheme indicates that these features are **specifically assessed** in the question. However, the meaning of all responses must be unambiguous.
4	Examiners must **consistently** apply the requirements of the mark scheme and any rules for what to do when candidates have not followed instructions correctly.
5	Examiners must use the **full range of marks** defined in the mark scheme, unless limited by the quality of the candidate responses seen.
6	Examiners must award marks according to the mark scheme and **must not** award marks with grade thresholds or grade descriptions in mind.

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance
1	(c) `AREA`	**1**	

Question	Answer	Marks	Guidance
2	1 mark for 2 or 3 correct lines 2 marks for 4 correct lines **operator** **use**  	**2**	

Question	Answer	Marks	Guidance
3	Cast / Casting	**1**	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

4(a)	• A suitable prompt to input an age
	• Input assigned to a variable.
	• Correct check of range.
	• Displays an error message and requests an input if not in range
	• Appropriate use of the loop to check range until the input is in range
	
	Exemplar response:
	
	`while True:`
	`  agestr = input("Please enter your age.")`
	`  age = float(agestr)`
	`  if age < 1 or age > 150:`
	`  print("Please only enter years between 1 and 150 inclusive")`
	`  else:`
	`  break`
	
	`age = int(input("enter an age"))`
	`while age <1 or age >150:`
	`  print("Error. Age must be between 1 and 150")`
	`  age = int(input("enter an age"))`
		5	

4(b)	**One** mark for: One suitable normal (between 1 and 150, inclusive) One suitable erroneous (e.g. letters, non-numeric, symbol, greater than 150)	**2**	
4(c)(i)	to check that the program accepts the data within the acceptable limits	**1**	
4(c)(ii)	to check the program rejects the data outside of the acceptable limits	**1**	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

5	• Using if / elif, correctly handling 0 and 4
	• Using if / elif, correctly calculating the surcharge for 1, 2 and 3
	• Using if / elif / else, outputting 'An invalid number was entered' for any other input
	
	Example solution:
	
	`num_bags = int(input())`
	`if num_bags == 0:`
	`  extra_cost = 0`
	`elif num_bags==1 or num_bags == 2 or num_bags ==3:`
	`  extra_cost = num_bags*25`
	`elif num_bags==4:`
	`  extra_cost =150`
	`else:`
	`  print("An invalid number was entered")`
	
	Alternative solution
	
	`num_bags = int(input())`
	`if num_bags == 0:`
	`  extra_cost = 0`
	`elif 0 < num_bags < 4:`
	`  extra_cost = num_bags*25`
	`elif num_bags == 4:`
	`  extra_cost = 150`
	`else:`
	`  print("An invalid number was entered")`
		3	

© Cambridge University Press & Assessment 2026

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>6(a)</td><td>One mark per correct column. Do not accept additional punctuation such as ‘,’ or ‘”’ in output column.

<table>
<tr><th>n</th><th>d</th><th>c</th><th>output</th></tr>
<tr><td>16.5</td><td>3</td><td></td><td></td></tr>
<tr><td>16.5</td><td>3</td><td>0</td><td></td></tr>
<tr><td>13.5</td><td></td><td>1</td><td></td></tr>
<tr><td>10.5</td><td></td><td>2</td><td></td></tr>
<tr><td>7.5</td><td></td><td>3</td><td></td></tr>
<tr><td>4.5</td><td></td><td>4</td><td></td></tr>
<tr><td>1.5</td><td></td><td>5</td><td></td></tr>
<tr><td></td><td></td><td></td><td>1.5 5</td></tr>
</table></td><td>4</td><td></td></tr>
<tr><td>6(b)</td><td>• Calculates and outputs first input DIV second input // n//d // by description e.g. integer division
• Calculates and outputs first input MOD second input// n%d // by description e.g. remainder from division

Descriptive example for 2 marks
Calculates how many times d goes into n and prints number of times and the remainder</td><td>2</td><td></td></tr>
</table>

Question	Answer	Marks	Guidance
7(a)	Abstraction – the process of creating a simplified model that represents the essential features of a problem / discarding irrelevant details	**1**	
7(b)	Decomposition – the process of breaking down problems into smaller, more manageable sub problems	**1**	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance
7(c)	Any example of decomposition relevant to the game, For example: • To check whether a piece can move on the board • To check if a space is free • How many pieces a player has left • How long a move has taken	**1**	

Question	Answer	Marks	Guidance
8(a)	To store results permanently / results are not lost when closing the program … to use at a later date … to compare results over time	**2**	

8(b)	**One** mark per box, **max. four** marks
	
	`file = open("winners.txt", "a" )`
	`file.write(winner)`
	`file.write(str(score))`
	`file.close()`
		4	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

9	**One** mark for each point
	
	• Input for weight and rounding of input to two decimal places
	• Loop to iterate through delivery and terminate loop when insertion point found
	• Method to determine number of items in list eg. (len)
	• Comparison to determine input weight in relation to weight in a specific position
	• Insertion of new weight into list in correct place
	
	Example solution:
	
	`delivery = [3.54, 6.44, 8.99, 11.03, 11.76]`
	`weight = float(input("Please enter the weight of the package: "))`
	`weight = round(weight, 2)`
	`i = 0`
	`while i < len(delivery):`
	`  if delivery[i] > weight:`
	`  break`
	`  i += 1`
	`delivery.insert(i,weight)`
		5	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

10(a)	**One** mark per mark point
	
	• Iteration correct number of times. Either `limit=len(stock)-1` or `limit-1` in range parameter in for loop
	• Correct if condition for descending order. Allow `i+1` on the left and `i` on the right.
	• Remove `[1]` from `stock[i][1] = stock[i+1]`
	• `stock[i+1]` changed to `stock[i]` (allow change of all other references if correct)
	• `limit += 1` changed to `limit -= 1 //` delete line
	
	Corrected program:
	
	`swap = True`
	`limit = len(stock)-1`
	`while swap:`
	`  swap = False`
	`  for i in range(limit):`
	`  if stock[i][1] < stock[i+1][1]:`
	`  temp = stock[i]`
	`  stock[i][1] = stock[i+1]`
	`  stock[i+1] = temp`
	`  swap = True`
	`  limit -= 1`
		5	

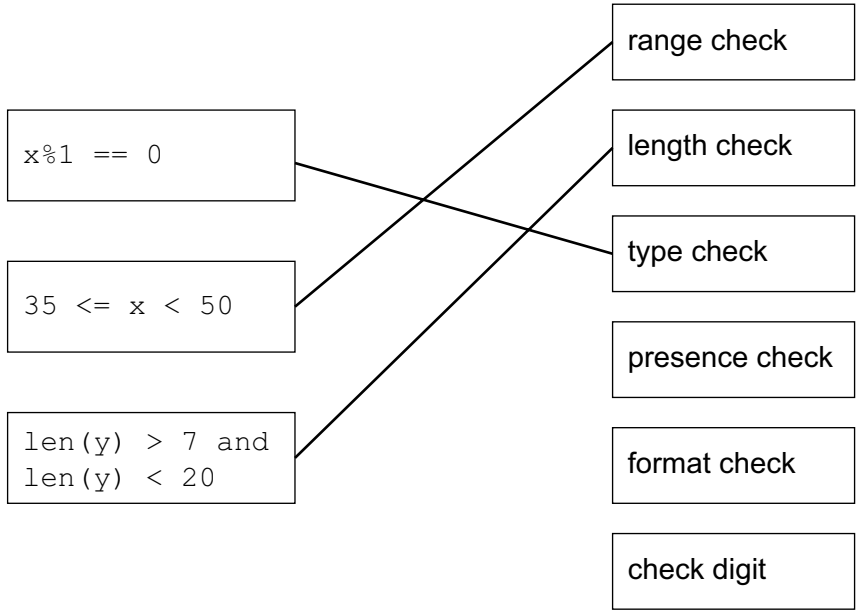
10(b)(i)	An error that allows the program to run, but causes the program to crash or stop responding when running	**1**	
10(b)(ii)	Suitable example, such as index out of range error, division by zero. Accept by example or named example.	**1**	

Question	Answer	Marks	Guidance
11(a)	**One** mark per row, **max. two** fields: 5 records: 8	**2**	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance
11(b)	**One** mark per correct line, **max. four** SELECT structure, number_moons FROM PLANET WHERE radius_km > 25 000 ORDER BY radius_km desc ;	**4**	
11(c)	`planet_name`	**1**	
11(d)	Mercury 0.4, Venus 0.72, Jupiter 5.2, Saturn 9.5 1 mark for correct fields with no additional data 1 mark for correct rows with no additional data 1 mark for correct order	**3**	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance
12	One mark per correct lines. Do not award multiple lines to/from one box. 	**3**	

Question	Answer	Marks	Guidance
13(a)	12	**1**	
13(b)	6	**1**	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance
13(c)	**One** mark for disadvantage, **one** mark for expansion e.g. • Harder to maintain/find bugs • … as not easy to find where variables are altered • More chance of errors • … as variables may be incorrectly altered in other parts of the program • Harder to test • … as modules are not standalone when using global variables	**2**	

Question	Answer	Marks	Guidance
14	Marks: • Clear instructions output to the user before the input command. • Input from the keyboard assigned to a variable. • Correct use of While (and break) to create a post condition loop. • Correct use of casting the string input to a float • Correct check of integer type • Correct check of range. • Clear output to the user when the input is not accepted (range or type). • Code is fully correct, with appropriate outputs (e.g. repeats the instructions after an invalid input and does not output invalid when it is valid).		

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

14(a)	**One** mark per mark point, **max. three**
	
	MP1 counter of rooms initialised
	MP2 loop used that iterates 30 times
	MP3 selection used to compare 'available' with each element of the list `room_occupied` …
	MP4 … if equal increment counter
	MP5 print the counter
	
	`number_available = 0`
	`for a in range(0,30):`
	`  if room_occupied[a] == "available":`
	`  number_available = number_available + 1`
	
	`print(number_available)`
		3	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

14(b)	**One** mark per mark point
	
	validates number of nights to be between 1 and 14 inclusively
	… the input variable used is appropriately named
	working selection statement for type of room
	output of the calculated room cost and booking
	program in a logical order
	inputs and outputs contain suitable messages
	
	`# takes an input of number of nights requested`
	`nights = int(input("How many nights would you like to stay?"))`
	
	`# validates the input to be within 1 and 14 nights, repeating if not`
	`while nights >14 or nights <1:`
	`  print("Can only book between 1 and 14 nights")`
	`  nights = int(input("How many nights would you like to stay?"))`
	
	`# input of room type`
	`room = input("what type of room would you like to book?")`
	
	`# checking the room type entered and outputting the cost`
	`if room == "family":`
	`  print("booked a family room for "+str(nights)+" at a cost of "+str(nights*90))`
	`elif room == "single":`
	`  print("booked a single room for "+str(nights)+" at a cost of "+str(nights*40))`
	`elif room == "twin":`
	`  print("booked a twin room for "+str(nights)+" at a cost of "+str(nights*50))`
	`elif room == "double":`
	`  print("booked a double room for "+str(nights)+" at a cost of "+str(nights*60))`
		6	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

14(b)	**Alternative using case statements**
	
	`nights=int(input("Please enter the number of nights: "))`
	`if nights > 0 and nights < 15:`
	`  print("number of nights is ok")`
	`room_type = input("Enter room type: ")`
	`match room_type:`
	`  case "double":`
	`  room_cost = nights * 60`
	`  case "family":`
	`  room_cost = nights * 90`
	`  case "single":`
	`  room_cost = nights * 40`
	`  case "twin":`
	`  room_cost = nights * 50`
	
	`print(f"nights: {nights}  room type: {room_type}  cost: ${room_cost}")`		

14(c)	**One** mark per mark point,
	
	• function `discount` defined, with a single parameter
	• return the cost with a 20% discount applied
	• any suitable comment to explain the purpose of the function
	
	`# staff discount calculated`
	`def discount(cost):`
	`  return cost*0.8`
		3	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

14(d)	**One** mark per mark point,
	
	• nested selection statement used to check both lists `room_type` and `room_occupied`
	• working loop mechanism for if there are no rooms of that type available
	• loop termination and correct output if room is available
	
	`resume = True`
	
	`while resume == True:`
	`  room = input("What type of room would you like to book? ")`
	`  room = room.lower()`
	
	`  available = False`
	
	`  for a in range(0, 30):`
	`  if room_type[a] == room:`
	`  if room_occupied[a] == "available":`
	`  available = True`
	`  room_number = a`
	`  break`
	
	`  if available == True:`
	`  print(room + " available")`
	`  print("Room number:", room_number)`
	`  else:`
	`  print(room + " of that type not available")`
	
	`  repeat = input("Would you like to try a different room type? ")`
	
	`  if repeat == "no":`
	`  resume = False`
		3	

© Cambridge University Press & Assessment 2026