Cambridge International AS & A Level

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

COMPUTER SCIENCE **9608/41**

Paper 4 Written Paper **October/November 2017**

MARK SCHEME

Maximum Mark: 75

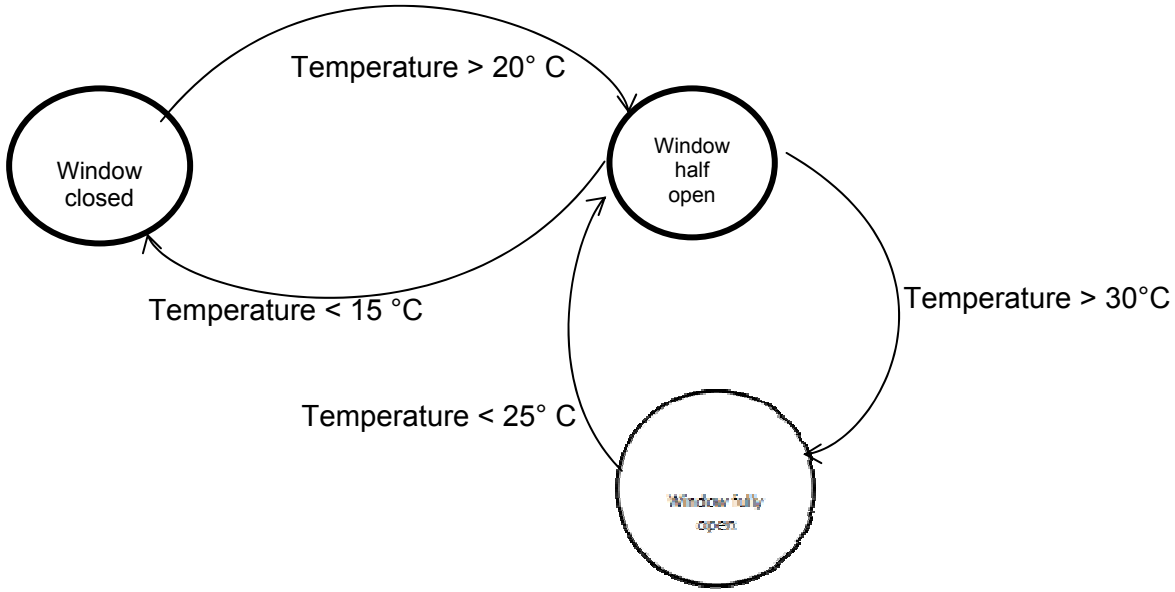
Published

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

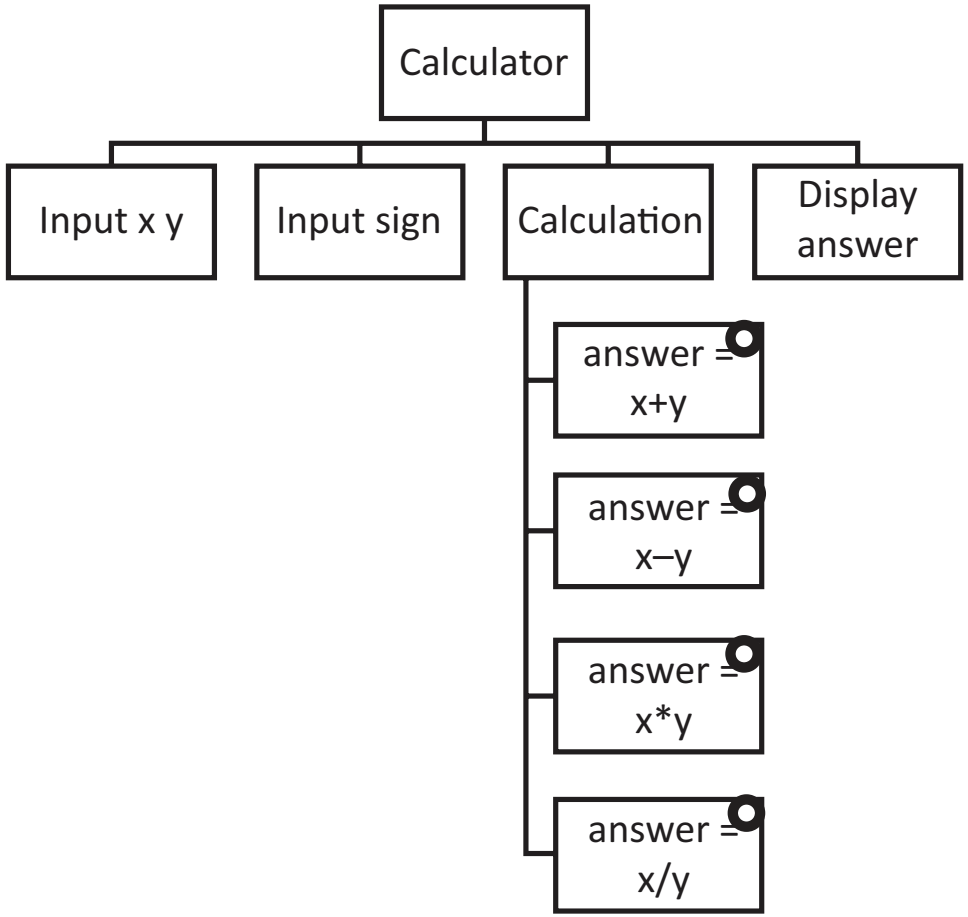
Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge International will not enter into discussions about these mark schemes.

Cambridge International is publishing the mark schemes for the October/November 2017 series for most Cambridge IGCSE®, Cambridge International A and AS Level components and some Cambridge O Level components.

Question	Answer	Marks
1	1 mark for each completed statement 	7

Question	Answer	Marks
2(a)(i)	Asterisk (*) in the corner/top of the box(es)	1
2(a)(ii)	Circle (○) in the corner/top of box(es)	1

Question	Answer	Marks
2(b)	1 mark per bullet Inputting 2 numbers, stored in x and y Inputting sign Selection used for all four calculations .. underneath an appropriate box at level 1 Displaying the answer For example:  	**5**

Question	Answer	Marks

3(a)	1 mark per clause
	`person(mimi).`
	`food(lettuce).`
	`likes(mimi, chocolate).`
	`dislikes(mimi, sushi).`
	`dislikes(mimi, lettuce).`
		5

3(b)	1 mark per answer `chocolate, pizza`	**2**

3(c)	1 mark per bullet
	`might_like(`**`B,A`**`)`
	`Person(B)`
	`Food(A)`
	`AND`
	`AND NOT`
	`Dislikes` predicate
	
	For example:
	
	`might_like(B, A).`
	
	`IF person(B) AND food(A)`
	
	`AND NOT(dislikes(B, A)).`
		6

Question	Answer	Marks
4(a)		11

Label	Op code	Operand	Comment	Marks
START:	LDM	#63	// load ASCII value for '?'	
	OUT		// OUTPUT '?'	1
	IN		// input GUESS	1
	CMP	LETTERTOGUESS	// compare with stored letter	1
	JPE	GUESSED	// if correct guess, go to GUESSED	1
	LDD	ATTEMPTS	// increment ATTEMPTS	1
	INC	ACC		1
	STO	ATTEMPTS		1
	CMP	#9	// is ATTEMPTS = 9 ?	1
	JPE	ENDP	// if out of guesses, go to ENDP	1
	JMP	START	// go back to beginning of loop	1
GUESSED:	LDM	#42	// load ASCII for '*'	
	OUT		// OUTPUT '*'	1
ENDP:	END		// end program	
ATTEMPTS:	0			
LETTERTOGUESS:	'a'			

Question	Answer	Marks
4(b)		**10**

Label	Opcode	Operand	Comment	Mark
START:	LDR	#0	// initialise the Index Register	1
LOOP:	LDX	NUMBERS	// load the value from NUMBERS	1 (LOOP) + 1(LDX NUMBERS)
	LSL	#2	// multiply by 4	1 (LSL) + 1 (#2)
	STX	NUMBERS	// store the new value in NUMBERS	1
	INC	IX	// increment the Index Register	1
	LDD	COUNT	// increment COUNT	1
	INC	ACC		
	STO	COUNT		
	CMP	#5	// is COUNT = 5 ?	1
	JPN	LOOP	// repeat for next number	1
ENDP:	END			
COUNT:	0			
NUMBERS:	22			
	13			
	5			
	46			
	12			

Question	Answer	Marks
5(a)(i)	PERT / GANTT	**1**
5(a)(ii)	1 mark per bullet to max 3 For example: Calculate total minimum time required for project Identify milestones Task dependencies Provides the critical path analysis Identify which tasks need to be prioritised Determine when to begin specific tasks/stages Identify slack time Identify when resources need allocating Identify tasks that can be completed in parallel	**3**
5(b)(i)	Integration	**1**
5(b)(ii)	Beta / acceptance	**1**

Question	Answer	Marks
6(a)	1 mark per bullet to max 6 Declaring a class with the name animal Declaring variables for across, down and score (all Integers) …as private/protected Correct constructor header and ending Randomly generating an across between 0–39 inc. in constructor Randomly generating a down between 0–39 inc. in constructor Initialising Score to zero in constructor Correct get for `Across` Correct set for `Across`	**6**

Question	Answer	Marks
6(a)	Example: VB Class Animal Private Across As Integer Private Down As Integer Private Score As Integer Function GetAcross() Return Across End Function Sub SetAcross(Value As Integer) Across = Value End Sub Sub New() Randomize() Across = randomnumber.Next(0, 40) Down = randomnumber.Next(0, 40) Score = 0 End Sub End Class	

Question	Answer	Marks
6(a)	or Class Animal Private Across As Integer Property _Across As Integer Get Return _Across End Get Set(Value As Integer) Across = Value End Set End Property Private Down As Integer Private _Score As Integer Sub New() Randomize() Across = randomnumber.Next(0, 40) Down = randomnumber.Next(0, 40) _Score = 0 End Sub End Class Example: Python class Animal : def __init__ (self) : x = random.randint(0,39) y = random.randint(0,39) self.Across = x self.Down = y self.Score = 0 def SetAcross(A) : self.Across = A def GetAcross() : return self.Across	

Question	Answer	Marks

6(a)	Example: Pascal
	type
	Animal = class
	private
	Across: integer;
	Down: integer;
	score: integer;
	public
	constructor init;
	
	procedure SetAcross(AcrossV: integer);
	function GetAcross(): integer;
	end;
	
	constructor Animal.init();
	SetAcross(random(40));
	SetDown (random(40));
	SetScore (0);
	end;
	
	procedure Animal.SetAcross(AcrossV: integer);
	begin
	Across := AcrossV;
	end;
	
	function Animal.GetAcross(): integer;
	begin
	GetAcross := Across;
	end;	

Question	Answer	Marks

6(b)	1 mark per bullet to max 5
	constructor method heading and ending
	Initialise all 40 by 40 elements of Grid as " or equivalent
	Loop 5 times…
	…Creates a new instance of animal inside loop...
	...and adds it to array `AnimalList`
	
	Call generate food **and** initialise `StepCounter` to 0
	
	Example Python
	
	`def __init__ (self) :`
	`  self.grid = [[' ' for i in range(40)] for j in range(40)]`
	`  self.AnimalList = []`
	`  self.StepCounter = 0`
	`  for i in range(5) :`
	`  newAnimal = Animal ()`
	`  self.AnimalList.append(newAnimal)`
	`  self.GenerateFood()`
	
	Example VB
	
	`Sub New()`
	`  For x = 0 To 39`
	`  For y = 0 To 39`
	`  grid(x, y) = ""`
	`  Next`
	`  Next`
	
	`  For z = 0 To 4`
	`  AnimalList(z) = New Animal`
	`  Next`
	
	`  Call GenerateFood()`
	`End Sub`
		5

Question	Answer	Marks

6(b)	Example Pascal
	
	`constructor Desert.init();`
	`  for x := 0 to 39 do`
	`  begin`
	`  for y := 0 to 39 do`
	`  begin`
	`  grid(x,y) = "";`
	`  end`
	`  end`
	
	`  for x := 0 to 4 do`
	`  begin`
	`  AnimalList(x) = object (Animal);`
	`  end`
	
	`  GenerateFood();`
	`end;`	

6(c)(i)	1 mark per bullet: Function header and ending taking one value as parameter Check if coordinate = 0 (on lower bound) …generate random number (0 or 1) Check if coordinate = 39 (on upper bound) …generate random number (–1 or 0) Generate random number (e.g. –1, 0, 1) Return the generated value	**max 4**

Question	Answer	Marks

6(c)(i)	Example VB
	
	`Function GenerateDirection(ByRef coord As Integer)`
	`  Dim lowerbound As Integer = -1`
	`  Dim upperbound As Integer = 1`
	
	`  If coord = 0 Then`
	`  lowerbound = 0`
	`  ElseIf coord = 39 Then`
	`  upperbound = 0`
	`  End If`
	
	`  GenerateDirection = randomnumber.Next(lowerbound, upperbound)`
	
	`End Function`
	
	Example Python
	
	`def GenerateDirection(Coord) :`
	`  lowerBound = -1`
	`  upperBound = 1`
	`  if Coord == 0 :`
	`  lowerBound = 0`
	`  elif Coord == 39 :`
	`  upperBound = 0`
	`  return random.randint(lowerBound, upperBound)`	

Question	Answer	Marks

6(c)(i)	Example Pascal
	
	`function GenerateDirection(coord : Integer): Integer;`
	`begin`
	`  lowerbound = -1;`
	`  upperbound = 1;`
	`  if coord = 0 then`
	`  lowerbound = 0;`
	`  else if coord = 39 then`
	`  upperbound = 0;`
	`  GenerateDirection = random(39);`
	`end;`	

6(c)(ii)	1 mark per bullet to max 4
	Procedure move header, no parameters
	Calling GenerateDirection **twice** sending across and down as separate parameters
	Add return value to Across
	Add return value to Down
	Check if the grid, at the (new) coordinates == “F”
	..if true, Call EatFood
	
	Example python
	
	`def Move(self) :`
	`  self.Across +=  GenerateChangeInCoordinate(self.Across)`
	`  self.Down +=  GenerateChangeInCoordinate(self.Down)`
	`  if grid[self.Across][self.Down] == 'F' :`
	`  self.EatFood()`
	`  return`
		4

Question	Answer	Marks

6(c)(ii)	Example VB
	
	`Sub Move(ByRef thisAnimal As Animal)`
	`  thisAnimal.across += GenerateChangeInCoordinate (thisAnimal.across)`
	`  thisAnimal.down += GenerateChangeInCoordinate (thisAnimal.down)`
	`  If thegrid._grid(thisAnimal.across, thisAnimal.down) = "F"`
	`  Then`
	`  Call EatFood()`
	`  End If`
	`End Sub`
	
	Example Pascal
	
	`procedure Move(thisAnimal : Animal);`
	`begin`
	`  thisAnimal.across = this.Animal.across + GenerateChangeInCoordinate (thisAnimal.across);`
	`  thisAnimal.down = thisAnimal.down + GenerateChangeInCoordinate (thisAnimal.down);`
	`  if (thisgrid.grid(thisAnimal.across, thisAnimal.down) = "F") then`
	`  EatFood();`
	`End;`	

6(d)	1 mark per bullet to max 3 Pre-compiled Collection of Code/modules/routines Each module performs a specific purpose/task Each module can be linked/imported into the program	**2**