

INTERNATIONAL GCSE **COMPUTER SCIENCE**

9210/1

Paper 1 Programming

Mark scheme

June 2024

Version: 1.0 Final



2 4 6 Y 9 2 1 0 / 1 / M S

Mark schemes are prepared by the Lead Assessment Writer and considered, together with the relevant questions, by a panel of subject teachers. This mark scheme includes any amendments made at the standardisation events which all associates participate in and is the scheme which was used by them in this examination. The standardisation process ensures that the mark scheme covers the students' responses to questions and that every associate understands and applies it in the same correct way. As preparation for standardisation each associate analyses a number of students' scripts. Alternative answers not already covered by the mark scheme are discussed and legislated for. If, after the standardisation process, associates encounter unusual answers which have not been raised they are required to refer these to the Lead Examiner.

It must be stressed that a mark scheme is a working document, in many cases further developed and expanded on the basis of students' reactions to a particular paper. Assumptions about future mark schemes on the basis of one year's document should be avoided; whilst the guiding principles of assessment remain constant, details will change, depending on the content of a particular examination paper.

Further copies of this mark scheme are available from [oxfordaqaexams.org.uk](https://www.oxfordaqaexams.org.uk)

Copyright information

OxfordAQA retains the copyright on all its publications. However, registered schools/colleges for OxfordAQA are permitted to copy material from this booklet for their own internal use, with the following important exception: OxfordAQA cannot give permission to schools/colleges to photocopy any material that is acknowledged to a third party even for internal use within the centre.

Copyright © 2024 OxfordAQA International Examinations and its licensors. All rights reserved.

How to mark

Aims

When you are marking your allocation of scripts your main aims should be to:

- recognise and identify the achievements of students
- where relevant, place students in the appropriate mark band and in the appropriate part of that mark band (high, low, middle) for **each** assessment objective
- record your judgements with brief notes, annotations and comments that are relevant to the mark scheme and make it clear to other associates how you have arrived at the numerical mark awarded for each assessment objective
- ensure comparability of assessment for all students, regardless of question or examiner.

Approach

It is important to be **open-minded** and **positive** when marking scripts.

The specification recognises the variety of experiences and knowledge that students will have. It encourages them to study computer science in a way that is relevant to them. The questions have been designed to give them opportunities to discuss what they have found out about computer science. It is important to assess the quality of **what the student offers**.

Do not mark scripts based on the answer **you** would have written. The mark schemes have been composed to assess **quality of response** and not to identify expected items of knowledge.

Assessment Objectives

This component requires students to:

AO1: Demonstrate knowledge and understanding of the key concepts and principles of computer science.

AO2: Apply knowledge and understanding of key concepts and principles of computer science.

AO3: Analyse problems in computational terms in order to develop and test programmed solutions and demonstrate an understanding of programming concepts.

The following annotation is used in the mark scheme.

; means a single mark

// means alternative response

/ means an alternative word or sub-phrase

A means acceptable creditworthy answer

R means reject answer as not creditworthy

NE means not enough

I means ignore

DPT in some questions a specific error made by a candidate, if repeated, could result in the loss of more than one mark. The **DPT** label indicates that this mistake should result in a candidate losing only one mark, on the first occasion that the error is made.

Provided that the answer remains understandable, subsequent marks should be awarded as if the error was not being repeated.

Section A

Question	Part	Marking guidance	Total marks
01	1	field // tempField; A. seedPosition in VB I: minor typos R: any additional code	1 AO3=1

Question	Part	Marking guidance	Total marks
01	2	displayMenu // displayField // main; I: minor typos R: any additional code	1 AO3=1

Question	Part	Marking guidance	Total marks
01	3	Pass values to the subroutine; Make subroutines self-contained; (Make programs) easier to test/debug; Produce code that is more reusable; Avoid using global variables; A. Pass values out of the subroutine (VB/C# only); Max 2	2 AO3=2

Question	Part	Marking guidance	Total marks
01	4	Makes code easier to read/understand/change; Changes only need to be made in one place; (Unlike a variable) cannot be changed (accidentally);	2 AO3=2

Question	Part	Marking guidance	Total marks
01	5	The number of iterations is known // The size of the field is known;	1 AO3=1

Question	Part	Marking guidance	Total marks
01	6	<code>field</code> stores the current state of the field // <code>field</code> stores the outcome of the previous week; <code>tempField</code> builds up/stores the new field cell-by-cell // <code>tempField</code> builds up/stores this week's field; With only one structure, rules would (incorrectly) be applied to a combination of the current and previous field; A. Incorrect changes would be made if only one were used	3 AO3=3

Question	Part	Marking guidance	Total marks
01	7	Converts numbers/columns to letters/characters; A. example, such as 0 → A.	1 AO3=1

Question	Part	Marking guidance	Total marks
01	8	(First character) is converted to its ASCII/Unicode/character code; 65/the ASCII/Unicode/character code of A is subtracted; A. accept a for A	2 AO3=2

Question	Part	Marking guidance	Total marks								
01	9	<table><tr><th>Type of test</th><th>Value to enter</th></tr><tr><td>Normal</td><td>any valid cell reference between A1 and O8; I. case.</td></tr><tr><td>Boundary</td><td>any cell reference comprising one of the following: • row 0, 1, 8 or 9 with a valid column (A to O) • column A, O or P with a valid row (1 to 8); A. P0 A. P9 I. case. R. if same as normal or same as erroneous.</td></tr><tr><td>Erroneous</td><td>any input that does not follow the letter-digit format, eg 8C, or input that is in valid cell reference format but outside the dimensions of the field (A1 to O8), eg Z9;</td></tr></table>	Type of test	Value to enter	Normal	any valid cell reference between A1 and O8; I. case.	Boundary	any cell reference comprising one of the following: • row 0, 1, 8 or 9 with a valid column (A to O) • column A, O or P with a valid row (1 to 8); A. P0 A. P9 I. case. R. if same as normal or same as erroneous.	Erroneous	any input that does not follow the letter-digit format, eg 8C, or input that is in valid cell reference format but outside the dimensions of the field (A1 to O8), eg Z9;	3 AO3=3
		Type of test	Value to enter								
		Normal	any valid cell reference between A1 and O8; I. case.								
		Boundary	any cell reference comprising one of the following: • row 0, 1, 8 or 9 with a valid column (A to O) • column A, O or P with a valid row (1 to 8); A. P0 A. P9 I. case. R. if same as normal or same as erroneous.								
Erroneous	any input that does not follow the letter-digit format, eg 8C, or input that is in valid cell reference format but outside the dimensions of the field (A1 to O8), eg Z9;										

Question	Part	Marking guidance	Total marks
----------	------	------------------	-------------

02	1	<table><tr><th>count</th><th>y</th><th>x</th><th>x >= 0 AND x < FIELD_WIDTH</th><th>y >= 0 AND y < FIELD_HEIGHT</th><th>NOT (x == column AND y == row)</th></tr><tr><td>0</td><td>1</td><td>-1</td><td>false</td><td>true</td><td>true</td></tr><tr><td></td><td></td><td>0</td><td>true</td><td></td><td></td></tr><tr><td>1</td><td></td><td>1</td><td></td><td></td><td></td></tr><tr><td></td><td>2</td><td>-1</td><td>false</td><td></td><td></td></tr><tr><td></td><td></td><td>0</td><td>true</td><td></td><td>false</td></tr><tr><td></td><td></td><td>1</td><td></td><td></td><td>true</td></tr><tr><td>2</td><td>3</td><td>-1</td><td>false</td><td></td><td></td></tr><tr><td></td><td></td><td>0</td><td>true</td><td></td><td></td></tr><tr><td></td><td></td><td>1</td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr></table> Initial value of count is 0; Initial value of y is 1, value of x is -1; Initial value of column 4 is false, column 5 is true, column 6 is true; 2nd values in x and column 4 are changed to 0 and true respectively; count changes to 1 and all columns contain correct values for when count is 1; count changes to 2 and all columns contain correct values for when count is 2; I. Different rows used as long as the order within columns is clear I. Duplicate values on consecutive rows within a column Max 5 marks if any errors.	count	y	x	x >= 0 AND x < FIELD_WIDTH	y >= 0 AND y < FIELD_HEIGHT	NOT (x == column AND y == row)	0	1	-1	false	true	true			0	true			1		1					2	-1	false					0	true		false			1			true	2	3	-1	false					0	true					1										6 AO3=6
count	y	x	x >= 0 AND x < FIELD_WIDTH	y >= 0 AND y < FIELD_HEIGHT	NOT (x == column AND y == row)																																																																
0	1	-1	false	true	true																																																																
		0	true																																																																		
1		1																																																																			
	2	-1	false																																																																		
		0	true		false																																																																
		1			true																																																																
2	3	-1	false																																																																		
		0	true																																																																		
		1																																																																			

Question	Part	Marking guidance	Total marks
----------	------	------------------	-------------

02	2	2; A. two	1 AO3=1
----	---	--------------	-------------------

Section B

Question	Part	Marking guidance	Total marks
03	1	1 mark program displays correct message I. minor typos and incorrect case 1 mark statement at a position so that the message only displays once per execution of the program and displays before any other output 1 mark blank line after the message Max 2 marks if any errors Python <pre>def main(): playAgain = True print("Welcome to Field Simulator") print() while playAgain: weekNo = 1 seedCount = STARTING_SEED_COUNT field = createField() done = False while done == False and weekNo < MAX_WEEK_NO: displayField(field, weekNo, seedCount) displayMenu() command = getInput().upper()</pre> C# <pre>static void Main() { bool playAgain = true; Console.WriteLine("Welcome to Field Simulator"); Console.WriteLine(); while (playAgain) { int weekNo = 1; int seedCount = STARTING_SEED_COUNT; char[,] field = createField(); bool done = false; while (done == false && weekNo < MAX_WEEK_NO) { displayField(field, weekNo, seedCount); displayMenu();</pre>	3 AO3=3

		<pre> string command = getInput().ToUpper(); Visual Basic Sub Main() Dim playAgain As Boolean = True Console.WriteLine("Welcome to Field Simulator") Console.WriteLine() While playAgain Dim weekNo As Integer = 1 Dim seedCount As Integer = STARTING_SEED_COUNT Dim field As Char(,) = createField() Dim done As Boolean = False While done = False And weekNo < MAX_WEEK_NO displayField(field, weekNo, seedCount) displayMenu() Dim Command As String = getInput().ToUpper() </pre>	
--	--	--	--

Question	Part	Marking guidance	Total marks
03	2	Screen capture(s) must match code from Q3.1 1 mark correct message displayed above all other output, with a blank line immediately below A. Incorrect message if matches Q3.1 Welcome to Field Simulator Week number: 1 6 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 4 5 6 7 8 (A)dvance the week (G)enerate a random field (J)ump to end of the simulation (L)oad pre-set seed positions (P)lant a seed at a specific position (R)eset the simulation e(X)it the program Please enter your choice:	1 A03=1

Question	Part	Marking guidance	Total marks
04	1	1 mark variable declared and initialised before any of the loops A. if variable declared globally. 1 mark comparison between <code>seedCount</code> and the new variable 1 mark 'New record' message within selection structure 1 mark variable updated under correct circumstances 1 mark display of record if a new record has not been set R. if previous record is also displayed when a record is set Max 4 marks if any errors Python <pre>def main(): playAgain = True record = 0 while playAgain: ... displayField(field, weekNo, seedCount) print("=====") print(f"Simulation ended with {seedCount} unplanted seed(s).") if seedCount > record: print("New record") record = seedCount else: print(f"Record is {record}") print("=====") print() repeat = input("Play again? (Y/N) ") C# static void Main() { bool playAgain = true; int record = 0; while (playAgain) ... displayField(field, weekNo, seedCount); Console.WriteLine("====="); Console.WriteLine(\$"Simulation ended with {seedCount} unplanted seed(s)."); if (seedCount > record) { Console.WriteLine("New record"); record = seedCount; }</pre>	5 A03=5

		<pre> } else { Console.WriteLine(\$"Record is {record}"); } Console.WriteLine("====="); Console.WriteLine(); Console.Write("Play again? (Y/N) "); string repeat = Console.ReadLine(); Visual Basic Sub Main() Dim playAgain As Boolean = True Dim record As Integer = 0 While playAgain ... displayField(field, weekNo, seedCount) Console.WriteLine("=====") Console.WriteLine(\$"Simulation ended with {seedCount} unplanted seed(s).") If seedCount > record Then Console.WriteLine("New record") record = seedCount Else Console.WriteLine(\$"Record is {record}") End If Console.WriteLine("=====") Console.WriteLine() Console.Write("Play again? (Y/N) ") </pre>	
--	--	--	--

Question	Part	Marking guidance	Total marks
04	2	Screen capture(s) must match code from Q4.1 1 mark appropriate messages displayed for record being set on the first attempt and not being set on the second <pre> Please enter your choice: J Week number: 20 35 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 4 5 6 @ 7 @ 8 @ @ @ ===== Simulation ended with 35 unplanted seed(s). New record ===== Please enter your choice: J Week number: 20 35 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 4 5 6 @ 7 @ 8 @ @ @ ===== Simulation ended with 35 unplanted seed(s). Record is 35 ===== </pre>	1 AO3=1

Question	Part	Marking guidance	Total marks
05	1	1 mark constant used to store exclamation mark A. an alternative name, as long as it clearly represents a sapling I. case Python <code>SAPLING = '!'</code> C# <code>const char SAPLING = '!';</code> Visual Basic <code>Const SAPLING = "!"</code>	1 AO3=1

Question	Part	Marking guidance	Total marks
05	2	1 mark cell containing an asterisk (*) changes to exclamation mark (!) 1 mark selection structure to check for exclamation mark (!) 1 mark cell containing an exclamation mark (!) changes to a vertical line () Max 2 marks if any errors. Python <pre>def growCrops(field, seedCount): tempField = createField() for row in range(FIELD_HEIGHT): for column in range(FIELD_WIDTH): if field[row][column] == EMPTY and countNeighbours(field, row, column) == 3: tempField[row][column] = PLANT elif field[row][column] == PLANT and countNeighbours(field, row, column) > 3: tempField[row][column] = EMPTY seedCount = seedCount + 1 elif field[row][column] == PLANT and countNeighbours(field, row, column) < 2: tempField[row][column] = EMPTY seedCount = seedCount + 1 else: tempField[row][column] = field[row][column] for row in range(FIELD_HEIGHT): for column in range(FIELD_WIDTH): if field[row][column] == SEED: field[row][column] = SAPLING elif field[row][column] == SAPLING: field[row][column] = STEM elif field[row][column] == STEM:</pre>	3 AO3=3

		<pre> field[row][column] = PLANT else: field[row][column] = tempField[row][column] return seedCount C# static int growCrops(char[,] field, int seedCount) { char[,] tempField = createField(); for (int row = 0; row < FIELD_HEIGHT; row++) { for (int column = 0; column < FIELD_WIDTH; column++) { if (field[row, column] == EMPTY && countNeighbours(field, row, column) == 3) { tempField[row, column] = PLANT; } else if (field[row, column] == PLANT && countNeighbours(field, row, column) > 3) { tempField[row, column] = EMPTY; seedCount++; } else if (field[row, column] == PLANT && countNeighbours(field, row, column) < 2) { tempField[row, column] = EMPTY; seedCount++; } else { tempField[row, column] = field[row, column]; } } } for (int row = 0; row < FIELD_HEIGHT; row++) { for (int column = 0; column < FIELD_WIDTH; column++) { if (field[row, column] == SEED) { </pre>	
--	--	--	--

		<pre> field[row, column] = SAPLING; } else if (field[row, column] == SAPLING) { field[row, column] = STEM; } else if (field[row, column] == STEM) { field[row, column] = PLANT; } else { field[row, column] = tempField[row, column]; } } return seedCount; } </pre> Visual Basic <pre> Function growCrops(field As Char(,), seedCount As Integer) As Integer Dim tempField As Char(,) = createField() For row = 0 To FIELD_HEIGHT - 1 For column = 0 To FIELD_WIDTH - 1 If field(row, column) = EMPTY And countNeighbours(field, row, column) = 3 Then tempField(row, column) = PLANT ElseIf (field(row, column) = PLANT And countNeighbours(field, row, column) > 3) Then tempField(row, column) = EMPTY seedCount += 1 ElseIf (field(row, column) = PLANT And countNeighbours(field, row, column) < 2) Then tempField(row, column) = EMPTY seedCount += 1 Else tempField(row, column) = field(row, column) End If Next Next For row = 0 To FIELD_HEIGHT - 1 For column = 0 To FIELD_WIDTH - 1 If field(row, column) = SEED Then field(row, column) = SAPLING </pre>	
--	--	--	--

		<pre> ElseIf field(row, column) = SAPLING Then field(row, column) = STEM ElseIf field(row, column) = STEM Then field(row, column) = PLANT Else field(row, column) = tempField(row, column) End If Next Next Return seedCount End Function </pre>	
--	--	--	--

Question	Part	Marking guidance	Total marks
05	3	Screen capture(s) must match code from Q5.1 1 mark advancing one week shows a cluster of five exclamation marks (!) and advancing another week shows a cluster of five vertical lines () <pre> Please enter your choice: A Week number: 2 1 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 ! 4 ! ! 5 ! ! 6 7 8 (A)dvance the week (G)enerate a random field (J)ump to end of the simulation (L)oad pre-set seed positions (P)lant a seed at a specific position (R)eset the simulation e(X)it the program Please enter your choice: A Week number: 3 1 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 4 5 6 7 8 </pre>	1 AO3=1

Question	Part	Marking guidance	Total marks
06	1	1 mark loop repeats until a seed has been planted 1 mark correctly checks for the position being empty 1 mark position changed to asterisk (*) and number of seeds decremented only if position empty 1 mark appropriate message if non-empty position given 1 mark code functions correctly if no seeds are remaining Max 4 marks if any errors Python <pre> def plantSeed(field, seedCount): if seedCount > 0: planted = False while not planted: row, column = getPosition() if field[row][column] == EMPTY: field[row][column] = SEED seedCount = seedCount - 1 planted = True else: print("Please select an empty cell") else: print("No seeds remaining") return seedCount </pre> Alternative answer <pre> def plantSeed(field, seedCount): if seedCount > 0: row, column = getPosition() while field[row][column] != EMPTY: print("Please select an empty cell") row, column = getPosition() field[row][column] = SEED seedCount = seedCount - 1 else: print("No seeds remaining") return seedCount </pre> C# <pre> static int plantSeed(char[,] field, int seedCount) { if (seedCount > 0) { bool planted = false; while (!planted) </pre>	5 AO3=5

```

        {
            int row, column;
            (row, column) = getPosition();
            if (field[row, column] == EMPTY)
            {
                field[row, column] = SEED;
                seedCount--;
                planted = true;
            } else
            {
                Console.WriteLine("Please
select an empty cell");
            }
        }
    }
else
{
    Console.WriteLine("No seeds
remaining");
}
return seedCount;
}

```

Alternative answer

```

static int plantSeed(char[,] field, int
seedCount)
{
    if (seedCount > 0)
    {
        int row, column;
        (row, column) = getPosition();
        while (field[row, column] != EMPTY)
        {
            Console.WriteLine("Please select an
empty cell");
            (row, column) = getPosition();
        }
        field[row, column] = SEED;
        seedCount--;
    }
    else
    {
        Console.WriteLine("No seeds
remaining");
    }
    return seedCount;
}

```

	Visual Basic <pre> Function plantSeed(field As Char(,), seedCount As Integer) As Integer If (seedCount > 0) Then Dim planted As Boolean = False While Not planted Dim position As (row As Integer, column As Integer) = getPosition() If field(position.row, position.column) = EMPTY Then field(position.row, position.column) = SEED seedCount -= 1 planted = True Else Console.WriteLine("Please select an empty cell") End If End While Else Console.WriteLine("No seeds remaining") End If Return seedCount End Function </pre> Alternative answer <pre> Function plantSeed(field As Char(,), seedCount As Integer) As Integer If (seedCount > 0) Then Dim position As (row As Integer, column As Integer) = getPosition() While field(position.row, position.column) <> EMPTY Console.WriteLine("Please select an empty cell") position = getPosition() End While field(position.row, position.column) = SEED seedCount -= 1 Else Console.WriteLine("No seeds remaining") End If Return seedCount End Function </pre>	
--	---	--

Question	Part	Marking guidance	Total marks
06	2	Screen capture(s) must match code from Q6.1 1 mark Message displayed in response to L3, seed planted in M3 and message not displayed A. Incorrect message if matches Q6.1 <pre> Please enter your choice: P Enter a position to plant a seed, e.g. A1: L3 Please select an empty cell Enter a position to plant a seed, e.g. A1: M3 Week number: 2 0 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 * 4 5 6 7 8 </pre>	1 AO3=1

Question	Part	Marking guidance	Total marks
07	1	1 mark value entered by user for the number of weeks is assigned to a variable R. if not within a selection structure 1 mark correct calculation for remaining number of weeks in the simulation 1 mark selection structure to determine whether there are enough weeks remaining, with correct condition DPT. if number of weeks incorrectly calculated 1 mark iterative structure used to advance simulation number of weeks entered by the user 1 mark outputs the number of remaining weeks if user attempts to advance too many weeks DPT. if number of weeks incorrectly calculated Max 4 marks if any errors Python <pre> while done == False and weekNo < MAX_WEEK_NO: displayField(field, weekNo, seedCount) displayMenu() command = getInput().upper() if command == "A": weeks = int(input("How many weeks would you like to advance: ")) if weeks <= MAX_WEEK_NO - weekNo: for weekCount in range(weeks): seedCount = growCrops(field, seedCount) weekNo = weekNo + 1 else: print(f"There are only {MAX_WEEK_NO - weekNo} weeks remaining") elif command == "G": </pre> C# <pre> while (done == false && weekNo < MAX_WEEK_NO) { displayField(field, weekNo, seedCount); displayMenu(); string command = getInput().ToUpper(); if (command == "A") { Console.WriteLine("How many weeks would you like to advance: "); int weeks = int.Parse(Console.ReadLine()); if (weeks <= MAX_WEEK_NO - weekNo) { for(int weekCount = 0; weekCount < weeks; weekCount++) </pre>	5 AO3=5

		<pre> { seedCount = growCrops(field, seedCount); weekNo++; } } else { Console.WriteLine(\$"There are only {MAX_WEEK_NO - weekNo} weeks remaining"); } } else if (command == "G") </pre> Visual Basic <pre> While done = False And weekNo < MAX_WEEK_NO displayField(field, weekNo, seedCount) displayMenu() Dim Command As String = getInput().ToUpper() If Command = "A" Then Console.WriteLine("How many weeks would you like to advance: ") Dim weeks As Integer = Console.ReadLine() If weeks <= MAX_WEEK_NO - weekNo Then For weekCount = 0 To weeks - 1 seedCount = growCrops(field, seedCount) weekNo = weekNo + 1 Next Else Console.WriteLine(\$"There are only {MAX_WEEK_NO - weekNo} weeks remaining") End If ElseIf Command = "G" Then </pre>	
--	--	--	--

Question	Part	Marking guidance	Total marks
07	2	Screen capture(s) must match code from Q7.1 1 mark Simulation advances to week 16, subsequent message indicates only 4 weeks remaining, with week number and number of seeds unchanged <pre> Please enter your choice: A How many weeks would you like to advance: 15 Week number: 16 27 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 4 5 @ 6 @ 7 @ @ @ 8 (A)dvance the week (G)enerate a random field (J)ump to end of the simulation (L)oad pre-set seed positions (P)lant a seed at a specific position (R)eset the simulation e(X)it the program Please enter your choice: A How many weeks would you like to advance: 5 There are only 4 weeks remaining Week number: 16 27 seed(s) remaining A B C D E F G H I J K L M N O 1 2 3 4 5 @ 6 @ 7 @ @ @ 8 </pre> Note to examiners: the number of remaining seeds might not be 27 and the layout of the field may be different as a result of other changes made to the code. Credit can be given as long as the number remaining is the same in both screenshots.	1 AO3=1

Section C

Question	Part	Marking guidance	Total marks
08	1	1 mark subroutine with correct parameter I. other parameters 1 mark two integer variables initialised to zero 1 mark outer loop, which can deal with rows or columns 1 mark inner loop, which must deal with the other 1 mark random number stored in variable or equivalent and used appropriately to generate a 1 in 20 chance 1 mark checking for both empty cells and weed cells 1 mark weed placed in cell based on random number 1 mark selection structure prevents a weed being added and removed in the same call 1 mark incrementing a variable within a selection structure 1 mark incrementing both variables only under correct circumstances with correct probabilities 1 mark appropriate message, using both variables, outside any loop structure Max 10 marks if any errors Python <pre>def growWeeds(field): newWeeds = 0 deadWeeds = 0 for x in range(FIELD_WIDTH): for y in range(FIELD_HEIGHT): weedChance = random.randint(1, 20) if field[y][x] == EMPTY: if weedChance == 20: field[y][x] = "#" newWeeds += 1 elif field[y][x] == "#": if weedChance > 10: field[y][x] = EMPTY deadWeeds += 1 print(f"{newWeeds} weed(s) grew and {deadWeeds} died.")</pre> C# <pre>static void growWeeds(char[,] field) { int newWeeds = 0; int deadWeeds = 0; Random r = new Random(); for (int x = 0; x < FIELD_WIDTH; x++) { for(int y = 0; y < FIELD_HEIGHT; y++) { int weedChance = r.Next(1, 21); if (field[y, x] == EMPTY)</pre>	11 AO3=11

		<pre> { if (weedChance == 20) { field[y, x] = '#'; newWeeds++; } } else if (field[y, x] == '#') { if (weedChance > 10) { field[y, x] = EMPTY; deadWeeds++; } } } } Console.WriteLine(\$"{newWeeds} weed(s) grew and {deadWeeds} died."); } </pre> Visual Basic <pre> Sub growWeeds(field As Char(,)) Dim newWeeds As Integer = 0 Dim deadWeeds As Integer = 0 Dim r As New Random() For x = 0 To FIELD_WIDTH - 1 For y = 0 To FIELD_HEIGHT - 1 Dim weedChance As Integer = r.Next(1, 21) If field(y, x) = EMPTY Then If weedChance = 20 Then field(y, x) = "#" newWeeds += 1 End If ElseIf field(y, x) = "#" Then If weedChance > 10 Then field(y, x) = EMPTY deadWeeds += 1 End If End If Next Next Console.WriteLine(\$"{newWeeds} weed(s) grew and {deadWeeds} died") End Sub </pre>	
--	--	---	--

Question	Part	Marking guidance	Total marks
08	2	1 mark syntactically valid call to <code>growWeeds</code> after <code>weekNo</code> is incremented Python <pre> elif command == 'J': while weekNo < MAX_WEEK_NO: seedCount = growCrops(field, seedCount) weekNo = weekNo + 1 growWeeds(field) elif command == 'L': </pre> C# <pre> else if (command == "J") { while (weekNo < MAX_WEEK_NO) { seedCount = growCrops(field, seedCount); weekNo++; growWeeds(field); } } else if (command == "L") </pre> Visual Basic <pre> ElseIf Command = "J" Then While weekNo < MAX_WEEK_NO seedCount = growCrops(field, seedCount) weekNo = weekNo + 1 growWeeds(field) End While ElseIf Command = "L" Then </pre>	1 AO3=1

Question	Part	Marking guidance	Total marks
08	3	Screen capture(s) must match code from Q8.1 & Q8.2 1 mark repeated instances of the message and a field containing weeds (#) A. a field containing no weeds if the total number of weeds that have appeared and disappeared are equal <pre> Please enter your choice: J 7 weed(s) grew and 0 died. 8 weed(s) grew and 5 died. 4 weed(s) grew and 5 died. 8 weed(s) grew and 6 died. 5 weed(s) grew and 3 died. 5 weed(s) grew and 9 died. 8 weed(s) grew and 5 died. 7 weed(s) grew and 6 died. 4 weed(s) grew and 7 died. 7 weed(s) grew and 4 died. 5 weed(s) grew and 6 died. 6 weed(s) grew and 3 died. 3 weed(s) grew and 10 died. 5 weed(s) grew and 3 died. 8 weed(s) grew and 3 died. 9 weed(s) grew and 8 died. 6 weed(s) grew and 10 died. 12 weed(s) grew and 9 died. 3 weed(s) grew and 10 died. Week number: 20 29 seed(s) remaining A B C D E F G H I J K L M N O 1 # # 2 3 # # 4 5 # 6 7 # # # 8 ===== Simulation ended with 29 unplanted seed(s). ===== Play again? (Y/N) </pre> Note to examiners: due to random numbers, this is an example screen capture, and may not be the same as the one shown in student answers.	1 AO3=1

Question	Part	Marking guidance	Total marks
09	1	1 mark subroutine with correct parameter I. other parameters 1 mark three values entered by user assigned to variables 1 mark converting position of start point to x and y coordinates 1 mark variable declared and initialised to track either plants gathered or seeds collected 1 mark selection structure is logically correct and uses both FIELD_WIDTH and FIELD_HEIGHT appropriately 1 mark invalid input results in an appropriate message with no further action before the return statement DPT. selection structure contains flawed logic, such as transposing FIELD_WIDTH and FIELD_HEIGHT. 1 mark valid input results in a nested loop to iterate over an area DPT. selection structure contains flawed logic 1 mark iteration over correct area 1 mark selection structure to detect a plant 1 mark plant disappears and variable incremented inside selection structure 1 mark appropriate message contains number of plants gathered and seeds collected (which is double the number of plants gathered) DPT: incorrect values 1 mark number of seeds collected is returned Max 11 marks if any errors Python <pre>def gather(field): start = input("Enter top-left position: ") width = int(input("Enter width: ")) height = int(input("Enter height: ")) xStart = ord(start[0].upper()) - 65 yStart = int(start[1]) - 1 gathered = 0 if xStart + width > FIELD_WIDTH or yStart + height > FIELD_HEIGHT: print("Invalid selection") else: for row in range(yStart, yStart + height): for column in range(xStart, xStart + width): if field[row][column] == PLANT: gathered += 1 field[row][column] = EMPTY print(f"{gathered} plants gathered, {gathered * 2} seeds collected") return gathered * 2</pre> C# <pre>static int gather(char[,] field) {</pre>	12 AO3=12

	<pre> Console.WriteLine("Enter top-left position: "); string start = Console.ReadLine(); Console.WriteLine("Enter width: "); int width = int.Parse(Console.ReadLine()); Console.WriteLine("Enter height: "); int height = int.Parse(Console.ReadLine()); int xStart = start[0] - 65; int yStart = int.Parse(start.Substring(1, 1)) - 1; int gathered = 0; if (xStart + width > FIELD_WIDTH yStart + height > FIELD_HEIGHT) { Console.WriteLine("Invalid selection"); } else { for (int row = yStart; row < yStart + height; row++) { for (int column = xStart; column < xStart + width; column++) { if (field[row, column] == PLANT) { gathered++; field[row, column] = EMPTY; } } } Console.WriteLine(\$"{gathered} plants gathered, {gathered * 2} seeds collected"); } return gathered * 2; } </pre> Visual Basic <pre> Function gather(field(,) As Char) As Integer Console.WriteLine("Enter top-left position: ") Dim start As String = Console.ReadLine() Console.WriteLine("Enter width: ") Dim width As Integer = Console.ReadLine() Console.WriteLine("Enter height: ") Dim height As Integer = Console.ReadLine() Dim xStart As Integer = Asc(start(0)) - 65 Dim yStart As Integer = Int32.Parse(start.Substring(1, 1)) - 1 Dim gathered As Integer = 0 If xStart + width > FIELD_WIDTH Or yStart + height > FIELD_HEIGHT Then Console.WriteLine("Invalid selection") End If </pre>	
--	--	--

		<pre> Else For row = yStart To yStart + height - 1 For column = xStart To xStart + width - 1 If field(row, column) = PLANT Then gathered += 1 field(row, column) = EMPTY End If Next Next Console.WriteLine(\$"{gathered} plants gathered, {gathered * 2} seeds collected") End If Return gathered * 2 End Function </pre>	
--	--	--	--

Question	Part	Marking guidance	Total marks
09	2	1 mark addition of option to indicate that C collects seeds Python <pre>def displayMenu(): print() print("(A)dvance the week") print("(C)ollect seeds") print("(G)enerate a random field") print("(J)ump to end of the simulation") print("(L)oad pre-set seed positions") print("(P)lant a seed at a specific position") print("(R)eset the simulation") print("e(X)it the program") print()</pre> C# <pre>static void displayMenu() { Console.WriteLine(); Console.WriteLine("(A)dvance the week"); Console.WriteLine("(C)ollect seeds"); Console.WriteLine("(G)enerate a random field"); Console.WriteLine("(J)ump to end of the simulation"); Console.WriteLine("(L)oad pre-set seed positions"); Console.WriteLine("(P)lant a seed at a specific position"); Console.WriteLine("(R)eset the simulation"); Console.WriteLine("e(X)it the program"); Console.WriteLine(); }</pre> Visual Basic <pre>Sub displayMenu() Console.WriteLine() Console.WriteLine("(A)dvance the week") Console.WriteLine("(C)ollect seeds") Console.WriteLine("(G)enerate a random field") Console.WriteLine("(J)ump to end of the simulation") Console.WriteLine("(L)oad pre-set seed positions") Console.WriteLine("(P)lant a seed at a specific position") Console.WriteLine("(R)eset the simulation") Console.WriteLine("e(X)it the program")</pre>	1 AO3=1

		<pre> Console.WriteLine() End Sub </pre>	
--	--	--	--

Question	Part	Marking guidance	Total marks
----------	------	------------------	-------------

09	3	1 mark selection structure detects input of C 1 mark syntactically valid call to <code>gather</code> with return being stored 1 mark adding returned value to <code>seedCount</code> Max 2 marks if any errors Python <pre> if command == "A": seedCount = growCrops(field, seedCount) weekNo = weekNo + 1 elif command == "C": collected = gather(field) seedCount = seedCount + collected elif command == "G": </pre> C# <pre> if (command == "A") { seedCount = growCrops(field, seedCount); weekNo++; } else if (command == "C") { int collected = gather(field); seedCount = seedCount + collected; } else if (command == "G") </pre> Visual Basic <pre> If Command = "A" Then seedCount = growCrops(field, seedCount) weekNo = weekNo + 1 ElseIf Command = "C" Then Dim collected As Integer = gather(field) seedCount = seedCount + collected ElseIf Command = "G" Then </pre>	3 AO3=3
----	---	---	-------------------------------------

Question	Part	Marking guidance	Total marks
09	4	Screen capture(s) must match code from Q9.1, Q9.2 & Q9.3. 1 mark entry of a 3 by 2 area results in 8 seeds recovered and the week advanced to 4 Please enter your choice: C Enter top-left position: K3 Enter width: 3 Enter height: 2 4 plants gathered, 8 seeds collected Week number: 4 11 seed(s) remaining A B C D E F G H I J K L M N O 1 2 @ 3 4 5 6 7 8 I. number of remaining seeds	1 AO3=1