OXFORD
INTERNATIONAL
AQA EXAMINATIONS

INTERNATIONAL GCSE

COMPUTER SCIENCE

9210

Paper 1 Programming

Mark scheme

June 2022

Version 1.0 Final

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

Section A

Question	Part	Marking guidance	Total marks
01	1	Makes (structure of) program clearer // easier to understand; Subroutines can be easily re-used; Reduces the number of lines of code written; Subroutines can be tested independently // easier to test; Easier to debug; Easier to maintain (**A.** update) the program; Subroutines can be developed by different members of a team // use of subroutines makes it easier to work as part of a programming team; **A.** If a subroutine is called more than once, it may reduce the memory required by a program **MAX 3**	**3** **AO3=3**

Question	Part	Marking guidance	Total marks

01	2	`Count;`
		`Seeds;`
		`PitCount;`
		`Pit;`
		`DropPit;`
		`LastPit;`
		`Player;`
		`PlayerOneSeeds;`
		`PlayerTwoSeeds;`
		`TotalSeeds;`
		
		MAX 1
		
		I. case and spacing
			1
			
			AO3=1

Question	Part	Marking guidance	Total marks
01	3	`SetupBoard;` **I.** case and spacing	**1** **AO3=1**

Question	Part	Marking guidance	Total marks
01	4	(`Won` is returned) when one player has collected more than half of the seeds;	**1** **AO3=1**

Question	Part	Marking guidance	Total marks
01	**5**	A two-dimensional array would require the use of an index for row and an index for column (which is harder to program) // A one-dimensional array will only need one index (which is easier to deal with/program); **A.** clearly written answers that talk about movement instead of indexes; Wrapping around from the beginning/end to the end/beginning of the board is easier with a one-dimensional array // moving from one row to another is easier; The game board is a loop/list rather than a table/grid;	**2** **AO3=2**

Question	Part	Marking guidance	Total marks
02	1	(Indefinite iteration is) where the loop is controlled by criteria (which is tested every iteration) // iteration that loops an unknown/variable number of times // iteration that loops while a condition is true/false;	1 AO3=1

Question	Part	Marking guidance	Total marks
02	2	The loop could be definite iteration as it is known how many times to repeat the loop; We know the number of seeds to drop; The loop does use / can be controlled by a counter; The values of Seeds changes by one each time (so a for loop could be used) **MAX 1**	1 AO3=1

Question	Part	Marking guidance	Total marks
02	3	The purpose is to move forward one position in the board; (The first) MOD is used so that we move correctly from last index/11 to the next pit // pit 0 // so we wrap around from the end of the board to the beginning; (The second) MOD is used so that we skip the starting pit; **MAX 2**	2 AO3=2

Question	Part	Marking guidance	Total marks

02 | **4** | **6**

AO3=6

Board				Pit	Seeds	DropPit	PitCount
0	1	2	3				
1	4	1	2	1			
	0				**4**	**1**	**4**
		2			**3**	**2**	
			3		**2**	**3**	
2					**1**	**0**	
						1	
		3			**0**	**2**	

1 mark: Initial values for `Seeds`, `DropPit` and `PitCount` completed correctly
1 mark: `Board[1]` changes to `0`
1 mark: `Board[2]` changes to `2`
1 mark: `Seeds` column changes to `3` then `2`, `1`, `0` and has no other values
1 mark: `DropPit` column changes to `2` then `3`, `0`, `1`, `2` and has no other values
1 mark: `Board` has correct final values

NOTE: Candidate might use more rows but the value changes should happen in the same order as shown.

MAX 5 if any incorrect / extra values in table

Section B

Question	Part	Marking guidance	Total marks

03	1	**1 mark:** Program displays correct message.
		A. minor typos and incorrect case.
		I. full stop at end of message
		
		1 mark: Statements are at a position so that message is displayed whenever the menu is displayed and before the menu options;
		
		1 mark: Blank line is printed after the message is displayed;
		
		MAX 2 if any errors
		
		Python
		`def Menu():`
		`  print()`
		`  print('Capture The Seeds')`
		`  print()`
		`  print('H - Help')`
		`  print('S - Setup a basic board')`
		
		VB
		`Sub DisplayMenu()`
		`  Console.WriteLine()`
		`  Console.WriteLine("Capture The Seeds")`
		`  Console.WriteLine()`
		`  Console.WriteLine("H - Help")`
		`  Console.WriteLine("S - Setup a basic board")`
		`  Console.WriteLine("B - Play a basic game")`
		`  Console.WriteLine("T - Play the test board")`
		`  Console.WriteLine("Q - Quit")`
		`  Console.WriteLine()`
		`End Sub`
		
		C#
		`private static void DisplayMenu()`
		`{`
		`  Console.WriteLine();`
		`  Console.WriteLine("Capture The Seeds");`
		`  Console.WriteLine();`
		`  Console.WriteLine("H - Help");`
		`  Console.WriteLine("S - Setup a basic board");`
			3
			
			AO3=3

```
    Console.WriteLine("B - Play a basic
game");
    Console.WriteLine("T - Play the test
board");
    Console.WriteLine("Q - Quit");
    Console.WriteLine();
}
```

Question	Part	Marking guidance	Total marks
03	2	Screen capture(s) must match code from Q3.1 **1 mark:** Correct message displayed above menu. Capture The Seeds H - Help S - Setup a basic board B - Play a basic game T - Play the test board Q - Quit Choice: T 11 10 9 8 7 6 ---------- | 1|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 16 ---------- | 1|| 0|| 3|| 1|| 2|| 2| Player 2 holds: 20 ---------- 0 1 2 3 4 5 Player 1, which pit do you want to take seeds from?	**1** **AO3=1**

Question	Part	Marking guidance	Total marks

04	**1**	**1 mark:** Check `State[1]` / `PlayerOneSeeds` has half of the seeds
		1 mark: Check `State[2]` / `PlayerTwoSeeds` has half of the seeds
		1 mark: Correctly combine the two checks.
		
		or
		
		1 mark: check `State[1]` / `PlayerOneSeeds` / `State[2]` / `PlayerTwoSeeds` has half of the seeds
		1 mark: Check `PlayerOneSeeds` and `PlayerTwoSeeds` are the same // Check sum of `PlayerOneSeeds` and `PlayerTwoSeeds` is equal to `TotalSeeds`
		1 mark: Correctly combine the two checks.
		
		or
		
		1 mark: attempt to calculate the sum of the values in the `Board` array/list
		1 mark: correctly calculates the sum of the values in the `Board` array/list
		1 mark: compares calculated total with 0
		
		and
		
		1 mark: Return the string `Drawn`
		1 mark: Only return the string `Drawn` under correct circumstances
		
		MAX 4 if any errors
		
		Python
		`def GetGameState(Player):`
		`    global State`
		`    TotalSeeds = State[0]`
		`    PlayerOneSeeds = State[1]`
		`    PlayerTwoSeeds = State[2]`
		`    if  PlayerOneSeeds >= TotalSeeds // 2 + 1 or PlayerTwoSeeds >= TotalSeeds // 2 + 1:`
		`        return 'Won'`
		`    elif PlayerOneSeeds == TotalSeeds // 2 and PlayerTwoSeeds == TotalSeeds // 2:`
		`        return 'Drawn'`
		`    else:`
		`        return 'Play'`
			5
			
			AO3=5

Alternative answer

```
def GetGameState(Player):
    global State
    TotalSeeds = State[0]
    PlayerOneSeeds = State[1]
    PlayerTwoSeeds = State[2]
    if  PlayerOneSeeds >= TotalSeeds // 2 + 1 or
PlayerTwoSeeds >= TotalSeeds // 2 + 1:
        return 'Won'
    elif PlayerOneSeeds + PlayerTwoSeeds ==
TotalSeeds:
        return 'Drawn'
    else:
        return 'Play'
```

VB

```
Function GetGameState(Player As Integer) As
String
    Dim TotalSeeds, PlayerOneSeeds,
PlayerTwoSeeds As Integer
    TotalSeeds = State(0)
    PlayerOneSeeds = State(1)
    PlayerTwoSeeds = State(2)
    If PlayerOneSeeds >= TotalSeeds \ 2 + 1 Or
PlayerTwoSeeds >= TotalSeeds \ 2 + 1 Then
        Return "Won"
    ElseIf PlayerOneSeeds = TotalSeeds \ 2 And
PlayerTwoSeeds = TotalSeeds \ 2 Then
        Return "Drawn"
    Else
        Return "Play"
    End If
End Function
```

		C#
		`private static string GetGameState(int Player)`
		`{`
		`    int TotalSeeds, PlayerOneSeeds, PlayerTwoSeeds;`
		`    TotalSeeds = State[0];`
		`    PlayerOneSeeds = State[1];`
		`    PlayerTwoSeeds = State[2];`
		`    if (PlayerOneSeeds >= TotalSeeds / 2 + 1 || PlayerTwoSeeds >= TotalSeeds / 2 + 1)`
		`    {`
		`        return "Won";`
		`    }`
		`    **else if (PlayerOnesSeeds == TotalSeeds / 2 && PlayerTwoSeeds == TotalSeeds / 2)**`
		`    **{**`
		`        **return "Drawn";**`
		`    **}**`
		`        return "Play";`
		`}`	

Question	Part	Marking guidance	Total marks
04	**2**	Screen capture(s) must match code from Q4.1 **1 mark:** Screen capture shows the test game being played with the requested moves ending in a `Drawn` message. Capture The Seeds H - Help S - Setup a basic board B - Play a basic game T - Play the test board Q - Quit Choice: T 11 10 9 8 7 6 ---------- | 1|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 16 ---------- | 1|| 0|| 3|| 1|| 2|| 2| Player 2 holds: 20 ---------- 0 1 2 3 4 5 Player 1, which pit do you want to take seeds from? 2 Collected seeds from pit: 5 Collected seeds from pit: 4 Collected seeds from pit: 3 11 10 9 8 7 6 ---------- | 1|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 24 ---------- | 1|| 0|| 0|| 0|| 0|| 0| Player 2 holds: 20 ---------- 0 1 2 3 4 5 Player 2, which pit do you want to take seeds from? 10 Collected seeds from pit: 0 Collected seeds from pit: 11 Drawn	**1** **AO3=1**

Question	Part	Marking guidance	Total marks

05	1	**1 mark:** Use of indefinite iteration.
		1 mark: Check against `Board[Pit]` being equal to `0`.
		1 mark: The message `No seeds - pick again` is displayed after a condition has been checked.
		I. Case of output message.
		A. Minor typos in output message.
		1 mark: New value for `Pit` is input after correctly identifying a pit that contains no seeds.
		1 mark: No code inside loop that should not be there.
		
		MAX 4 if any errors
		
		Python
		`def GetMove(Player):`
		`  global Board`
		`  Pit = int(input('Player ' + str(Player) + ', which pit do you want to take seeds from? '))`
		`  while Board[Pit] == 0:`
		`  print('No seeds - pick again')`
		`  Pit = int(input('Player ' + str(Player) + ', which pit do you want to take seeds from? '))`
		`  return Pit`
		
		VB
		`Function GetMove(Player As Integer) As Integer`
		`  Console.WriteLine("Player " & Player & " which pit do you want to take seeds from?")`
		`  Dim Pit As Integer = Console.ReadLine()`
		`  While Board(Pit) = 0`
		`  Console.WriteLine("No seeds - pick again")`
		`  Console.WriteLine("Player " & Player & " which pit do you want to take seeds from?")`
		`  Pit = Console.ReadLine()`
		`  End While`
		`  Return Pit`
		`End Function`
			5
			
			AO3=5

		C#
		`private static int GetMove(int Player)`
		`{`
		`    Console.WriteLine("Player " + Player + "`
		`which pit do you want to take seeds from?");`
		`    int Pit = int.Parse(Console.ReadLine());`
		`    while (Board[Pit] == 0)`
		`    {`
		`        Console.WriteLine("No seeds – pick`
		`again");`
		`        Console.WriteLine("Player " + Player + "`
		`which pit do you want to take seeds from?");`
		`        Pit = int.Parse(Console.ReadLine());`
		`    }`
		`    return Pit;`
		`}`	

Question	Part	Marking guidance	Total marks
05	**2**	Screen capture(s) must match code from Q5.1 **1 mark:** Correct messages shown Capture The Seeds H - Help S - Setup a basic board B - Play a basic game T - Play the test board Q - Quit Choice: T 11 10 9 8 7 6 ------------------------ | 1|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 16 ------------------------ | 1|| 0|| 3|| 1|| 2|| 2| Player 2 holds: 20 ------------------------ 0 1 2 3 4 5 Player 1, which pit do you want to take seeds from? 1 No seeds - pick again Player 1, which pit do you want to take seeds from? 0 11 10 9 8 7 6 ------------------------ | 1|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 16 ------------------------ | 0|| 1|| 3|| 1|| 2|| 2| Player 2 holds: 20 ------------------------ 0 1 2 3 4 5 Player 2, which pit do you want to take seeds from?	**1** **AO3=1**

Question	Part	Marking guidance	Total marks

06	**1**	**1 mark:** Check against lower bound of `PitCount`
		1 mark: Check against upper bound of `PitCount`
		1 mark: Checks for both bounds are correct
		1 mark: Correct test that variable `PitCount` is even
		1 mark: Correctly combine the three checks
		1 mark: Appropriate message and new value for `PitCount` is input and stored correctly
		1 mark: Loop ensures that input is requested again until student's conditions are met
		
		MAX 6 if any errors
		
		Python
		`def GetInitialValues():`
		`  print()`
		`  PitCount = int(input('How many pits on the board? '))`
		`  **while PitCount < 10 or PitCount > 20 or PitCount % 2 != 0:**`
		`  **print('Invalid value')**`
		`  **PitCount = int(input('How many pits on the board? '))**`
		`  Seeds = int(input('How many seeds for the game? '))`
		`  return PitCount, Seeds`
		
		VB
		`Sub GetInitialValues(ByRef Seeds As Integer, ByRef PitCount As Integer)`
		`  Console.WriteLine()`
		`  Console.WriteLine("How many pits on the board?")`
		`  PitCount = Console.ReadLine()`
		`  **While PitCount < 10 Or PitCount > 20 Or PitCount Mod 2 <> 0**`
		`  **Console.WriteLine("Invalid value")**`
		`  **Console.WriteLine("How many pits on the board?")**`
		`  **PitCount = Console.ReadLine()**`
		`  **End While**`
		`  Console.WriteLine("How many seeds for the game?")`
		`  Seeds = Console.ReadLine()`
		`End Sub`
			7
			
			AO3=7

		C#
		`private static void GetInitialValues(ref int Seeds, ref int PitCount)`
		`{`
		`  Console.WriteLine();`
		`  Console.WriteLine("How many pits on the board?");`
		`  PitCount = int.Parse(Console.ReadLine());`
		`  while (PitCount < 10 || PitCount > 20 || PitCount % 2 != 0)`
		`  {`
		`  Console.WriteLine("Invalid value");`
		`  Console.WriteLine("How many pits on the board?");`
		`  PitCount = int.Parse(Console.ReadLine());`
		`  }`
		`  Console.WriteLine("How many seeds for the game?");`
		`  Seeds = int.Parse(Console.ReadLine());`
		`}`	

Question	Part	Marking guidance	Total marks

06	**2**	Screen capture(s) must match code from Q6.1
		
		1 mark: Screen capture shows the setup being run with values 15, 24 and then 14 being tested and the error messages.
		
		`Capture The Seeds`
		
		`H - Help`
		`S - Setup a basic board`
		`B - Play a basic game`
		`T - Play the test board`
		`Q - Quit`
		
		`Choice: S`
		
		`How many pits on the board? 15`
		`Invalid value`
		`How many pits on the board? 24`
		`Invalid value`
		`How many pits on the board? 14`
		`How many seeds for the game?`
			1
			
			AO3=1

Question	Part	Marking guidance	Total marks

07	1	**1 mark:** Use of remainder operator or equivalent
		1 mark: Correctly calculate number of remaining seeds
		1 mark: Selection statement which compares calculated integer value with 0. **A.** compare with 1 if greater than or equal to relational operator has been used
		1 mark: Appropriate prompt for remaining seeds
		1 mark: Response stored in an appropriate variable
		1 mark: Board updated appropriately
		
		MAX 5 if any errors
		
		Python
		`def CreateBoard(Seeds, PitCount):`
		`  global Board`
		`  for Count in range(PitCount):`
		`  Board.append(Seeds // PitCount)`
		`  Remainder = Seeds % PitCount`
		`  if Remainder > 0:`
		`  Pit = int(input('Which pit should get the remaining seeds? '))`
		`  Board[Pit] = Board[Pit] + Remainder`
		
		VB
		`Sub CreateBoard(Seeds As Integer, PitCount As Integer)`
		`  ReDim Board(PitCount - 1)`
		`  For Count = 0 To PitCount - 1`
		`  Board(Count) = Seeds \ PitCount`
		`  Next`
		`  Dim Remainder As Integer = Seeds Mod PitCount`
		`  If Remainder > 0 Then`
		`  Console.WriteLine("Which pit should get the remaining seed? ")`
		`  Dim Pit As Integer = Console.ReadLine()`
		`  Board(Pit) = Board(Pit) + Remainder`
		`  End If`
		`End Sub`
			6
			
			AO3=6

		C#
		`private static void CreateBoard(int Seeds, int PitCount)`
		`{`
		`    Board = new int[PitCount];`
		`    for (int Count = 0; Count < PitCount; Count++)`
		`    {`
		`        Board[Count] = Seeds / PitCount;`
		`    }`
		`    int Remainder = Seeds % PitCount;`
		`    if (Remainder > 0)`
		`    {`
		`        Console.WriteLine("Which pit should get the remaining seeds? ");`
		`        int Pit = int.Parse(Console.ReadLine());`
		`        Board[Pit] = Board[Pit] + Remainder;`
		`    }`
		`}`	

Question	Part	Marking guidance	Total marks
07	**2**	Screen capture(s) must match code from Q7.1 **1 mark:** Screen capture shows the setup being run with values 10, 25 and then 4 being entered. A basic game should be started showing the board with extra seeds in pit 4. Capture The Seeds H - Help S - Setup a basic board B - Play a basic game T - Play the test board Q - Quit Choice: S How many pits on the board? 10 How many seeds for the game? 25 Which pit should get the remaining seeds? 4 Capture The Seeds H - Help S - Setup a basic board B - Play a basic game T - Play the test board Q - Quit Choice: B 9 8 7 6 5 -------------------- | 2|| 2|| 2|| 2|| 2| Player 1 holds: 0 -------------------- | 2|| 2|| 2|| 2|| 7| Player 2 holds: 0 -------------------- 0 1 2 3 4 Player 1, which pit do you want to take seeds from?	**1** **AO3=1**

Section C

Question	Part	Marking guidance	Total marks

08	**1**	**1 mark:** Option for setup random board added into menu options
		I. Ordering of options
		A. Minor typos in output message
		
		Python
		`def Menu():`
		`  print()`
		`  print('Capture The Seeds')`
		`  print()`
		`  print('H - Help')`
		`  print('S - Setup a basic board')`
		`  print('R - Setup a random board')`
		`  print('B - Play a basic game')`
		`  print('T - Play the test board')`
		`  print('Q – Quit')`
		`  print()`
		
		VB
		`Sub DisplayMenu()`
		`  Console.WriteLine()`
		`  Console.WriteLine("Capture The Seeds")`
		`  Console.WriteLine()`
		`  Console.WriteLine("H - Help")`
		`  Console.WriteLine("S - Setup a basic board")`
		`  Console.WriteLine("R – Setup a random board")`
		`  Console.WriteLine("B - Play a basic game")`
		`  Console.WriteLine("T - Play the test board")`
		`  Console.WriteLine("Q - Quit")`
		`  Console.WriteLine()`
		`End Sub`
			1
			
			AO3=1

		C#
		`private static void DisplayMenu()`
		`{`
		`    Console.WriteLine();`
		`    Console.WriteLine("Capture The Seeds");`
		`    Console.WriteLine();`
		`    Console.WriteLine("H - Help");`
		`    Console.WriteLine("S - Setup a basic board");`
		`    Console.WriteLine("R – Setup a random board");`
		`    Console.WriteLine("B - Play a basic game");`
		`    Console.WriteLine("T - Play the test board");`
		`    Console.WriteLine("Q - Quit");`
		`    Console.WriteLine();`
		`}`	

Question	Part	Marking guidance	Total marks

08	**2**	**1 mark:** Extra case added for selection statement
		1 mark: Checks `Choice = 'R'`
		1 mark: Makes appropriate call to `GetInitialValues` or equivalent
		1 mark: makes appropriate call to `SetupBoard`
		
		MAX 3 if any errors
		
		Python
		`def Main():`
		`  Playing = True`
		`  while Playing:`
		`  DisplayMenu()`
		`  Choice = input('Choice: ')`
		`  if Choice == 'H':`
		`  DisplayHelp()`
		`  elif Choice == 'S':`
		`  PitCount, Seeds = GetInitialValues ()`
		`  SetupBoard('S', PitCount, Seeds)`
		`  **elif Choice == 'R':**`
		`  **Seeds, PitCount =** GetInitialValues **()**`
		`  **SetupBoard('R', Seeds, PitCount)**`
		`  elif Choice == 'B':`
		`  if len(Board) != 0:`
		`  Player = random.randint(1,2)`
		`  PlayGame(Player)`
		`  else:`
		`  print('You need to setup a board first')`
		`  elif Choice == 'T':`
		`  SetupBoard('T')`
		`  Player = 1`
		`  PlayGame(Player)`
		`  elif Choice == 'Q':`
		`  Playing = False`
			4
			
			AO3=4

VB

```
Sub Main()
    Dim Playing As Boolean = True
    Dim Player As Integer
    Dim PitCount As Integer
    Dim Seeds As Integer
    ReDim Board(-1)
    While Playing = True
        DisplayMenu()
        Console.Write("Choice: ")
        Dim Choice As String = Console.ReadLine()
        If Choice = "H" Then
            DisplayHelp()
        ElseIf Choice = "S" Then
            PitCount = 0
            Seeds = 0
            GetInitialValues(Seeds, PitCount)
            SetupBoard("S", Seeds, PitCount)
        ElseIf Choice = "R" Then
            GetInitialValues(Seeds, PitCount)
            SetupBoard("R", Seeds, PitCount)
        ElseIf Choice = "B" Then
            If (Board.Length <> 0) Then
                Player = RandomGenerator.Next(1, 3)
                PlayGame(Player)
            Else
                Console.WriteLine("You need to setup
a board first")
            End If
        ElseIf Choice = "T" Then
            SetupBoard("T")
            Player = 1
            PlayGame(Player)
        ElseIf Choice = "Q" Then
            Playing = False
        End If
    End While
End Sub
```

C#

```
private static void Main()
{
    bool Playing = true;
    int Player;
    int PitCount;
    int Seeds;
    Board = new int[0];
    while (Playing == true)

    {
        DisplayMenu();
        Console.Write("Choice: ");
        string Choice = Console.ReadLine();
        if (Choice == "H")
        {
            DisplayHelp();
        }
        else if (Choice == "S")
        {
            PitCount = 0;
            Seeds = 0;
            GetInitialValues(ref Seeds, ref
PitCount);
            SetupBoard('S', Seeds, PitCount);
        }
        else if (Choice == "R")
        {
             GetInitialValues(ref Seeds, ref
PitCount);
             SetupBoard('R', Seeds, PitCount);
        }
        else if (Choice == "B")
            if (Board.Length != 0)
            {
                Player = RandomGenerator.Next(1, 3);
                PlayGame(Player);
            }
            else
                Console.WriteLine("You need to setup
a board first");
        else if (Choice == "T")
        {
            SetupBoard('T');
            Player = 1;
            PlayGame(Player);
        }
        else if (Choice == "Q")
            Playing = false;
    }
}
```

Question	Part	Marking guidance	Total marks

08	**3**	**1 mark:** extension to selection statement for `TypeOfBoard` equal to `R`
		
		1 mark: Displays message `Setting up random board`
		
		1 mark: Call to `CreateBoard`
		1 mark: Correct parameters used for call to `CreateBoard`
		
		1 mark: Correctly setting up `State` variable
		
		1 mark: Generating a random number
		1 mark: Random number within the correct range
		1 mark: Correctly indexing `Board` with the random number
		1 mark: Adding 1 to current value of indexed board position
		
		1 mark: Iteration controlled correctly by number of seeds remaining
		
		MAX 9 if any errors
		
		Python
		`elif TypeOfBoard == 'S':`
		`  CreateBoard(Seeds, PitCount)`
		`  State=[Seeds,0,0]`
		`elif TypeOfBoard == 'R':`
		`  print('Setting up random board')`
		`  CreateBoard(0, PitCount)`
		`  State = [Seeds, 0, 0]`
		`  while Seeds > 0:`
		`  RandomPit = random.randint(0, PitCount - 1)`
		`  Board[RandomPit] = Board[RandomPit] + 1`
		`  Seeds = Seeds - 1`
			10
			
			AO3=10

VB

```
    If TypeOfBoard = "T" Then
        Board = {1, 0, 3, 1, 2, 2, 0, 0, 0,
0, 2, 1}
        State = {48, 16, 20}
    ElseIf TypeOfBoard = "R" Then
        Console.WriteLine("Setting up
random board")
        CreateBoard(0, PitCount)
        State = {Seeds, 0 ,0}
        While Seeds > 0
            Dim RandomPit As Integer =
RandomGenerator.Next(0, PitCount)
            Board(RandomPit) =
Board(RandomPit) + 1
            Seeds = Seeds - 1
        End While
    ElseIf TypeOfBoard = "S" Then
        CreateBoard(Seeds, PitCount)
        State = {Seeds, 0, 0}
    End If
```

C#

```
    if (TypeOfBoard == 'T')
    {
        Board = new int[12] { 1, 0, 3, 1,
2, 2, 0, 0, 0, 0, 2, 1 };
        State = new int[3] { 48, 16, 20 };
    }
    else if (TypeOfBoard == 'R')
    {
        Console.WriteLine("Setting up
random board");
        CreateBoard(0, PitCount);
        State = new int[3] { Seeds, 0, 0};
        while (Seeds > 0)
        {
            int RandomPit =
RandomGenerator.Next(0, PitCount);
            Board[RandomPit] =
Board[RandomPit] + 1
            Seeds = Seeds - 1
        }
    }
    else if (TypeOfBoard == 'S')
    {
        CreateBoard(Seeds, PitCount);
        State = new int[3] { Seeds, 0, 0 };
    }
```

Question	Part	Marking guidance	Total marks
08	**4**	Screen capture(s) must match code from Q8.1, Q8.2 and Q8.3. **1 mark:** Screen capture shows the random board setup being called with the values 10 and 20 being entered. A board should then be displayed with random distribution of the seeds. Total of the seeds should be 20. Capture The Seeds H - Help S - Setup a basic board R - Setup a random board B - Play a basic game T - Play the test board Q - Quit Choice: R How many pits on the board? 10 How many seeds for the game? 20 Capture The Seeds H - Help S - Setup a basic board R - Setup a random board B - Play a basic game T - Play the test board Q - Quit Choice: B 9 8 7 6 5 -------------------- | 4|| 2|| 2|| 3|| 4| Player 1 holds: 0 -------------------- | 1|| 0|| 0|| 0|| 4| Player 2 holds: 0 -------------------- 0 1 2 3 4 Player 1, which pit do you want to take seeds from?	**1** **AO3=1**

Question	Part	Marking guidance	Total marks

09	1	**1 mark:** Creation of new subroutine named `CollectNext`
		1 mark: Subroutine passed parameters `Pit` and `Player`
		1 mark: Appropriate call to `DropSeeds`
		1 mark: A check that `Board[LastPit]` is equal to 1
		1 mark: Working out position of next pit by adding 1
		1 mark: Including a correct usage of MOD to account for wrapping around board
		1 mark: Correct updating of `State` variable
		1 mark: Correct updating of `Board` variable
		1 mark: Displays message `Collected Seeds` when seeds are collected.
		
		MAX 8 if any errors
		
		Python
		`def CollectNext(Pit, Player):`
		`  global Board, State  # optional`
		`  LastPit = DropSeeds(Pit)`
		`  if Board[LastPit] == 1:`
		`  NextPit = (LastPit + 1) % len(Board)`
		`  State[Player] =  State[Player] + Board[NextPit]`
		`  Board[NextPit] = 0`
		`  print('Collected Seeds')`
		
		VB
		`Sub CollectNext(Pit As Integer, Player As Integer)`
		`  Dim LastPit As Integer = DropSeeds(Pit)`
		`  If Board(LastPit) = 1 Then`
		`  NextPit = (LastPit + 1) Mod Board.Length`
		`  State(Player) = State(Player) + Board(NextPit)`
		`  Board(NextPit) = 0`
		`  Console.WriteLine("Collected Seeds")`
		`  End If`
			9
			
			AO3=9

		C#
		`private static void CollectNext(int Pit, int Player)`
		`{`
		`    int LastPit = DropSeeds(Pit);`
		`    if (Board[LastPit] == 1)`
		`    {`
		`        NextPit = (LastPit + 1) % Board.Length;`
		`        State[Player] = State[Player] + Board[NextPit];`
		`        Board[NextPit] = 0;`
		`        Console.WriteLine("Collected Seeds");`
		`    }`
		`}`	

Question	Part	Marking guidance	Total marks

09	**2**	**1 mark:** Appropriate prompt to ask for kind of move placed after use of `GetMove`
		1 mark: Selection statement setup to check response
		1 mark: Call to `MakeMove` when option A has been chosen
		1 mark: Call to `CollectNext`
		1 mark: Subroutines called under correct circumstances.
		
		MAX 4 if any errors
		
		Python
		`def PlayGame(Player):`
		`  global Board, State`
		`  while GetGameState(Player) == 'Play':`
		`  DisplayBoard()`
		`  Pit = GetMove(Player)`
		`  Move = input('What kind of move (A - original, B - new)? ')`
		`  if Move == 'A':`
		`  MakeMove(Pit, Player)`
		`  else:`
		`  CollectNext(Pit, Player)`
		`  if Player == 1:`
		`  Player = 2`
		`  else:`
		`  Player = 1`
		`  print(GetGameState(Player))`
		`  Board = []`
		`  State = []`
			5
			
			AO3=5

VB

```
Sub PlayGame(Player As Integer)
    While GetGameState(Player) = "Play"
        DisplayBoard()
        Dim Pit As Integer =
GetMove(Player)
        Console.WriteLine("What kind of
move (A – original, B – new)? ")
        Dim Move As String =
Console.ReadLine()
        If Move = "A" Then
            MakeMove(Pit, Player)
        Else
            CollectNext(Pit, Player)
        End If
        If Player = 1 Then
            Player = 2
        Else
            Player = 1
        End If
    End While
    Console.WriteLine(GetGameState(Player))
    ReDim Board(-1)
    ReDim State(2)
End Sub
```

C#

```
private static void PlayGame(int Player)
{
    while (GetGameState(Player) == "Play")
    {
        DisplayBoard();
        int Pit = GetMove(Player);
        Console.WriteLine("What kind of
move (A – original, B – new)? ");
        string Move = Console.ReadLine();
        if (Move == "A")
        {
            MakeMove(Pit, Player);
        }
        else
        {
            CollectNext(Pit, Player);
        }
        if (Player == 1)
            Player = 2;
        else
            Player = 1;
    }
Console.WriteLine(GetGameState(Player));
    Board = new int[0];
    State = new int[3];
}
```

Question	Part	Marking guidance	Total marks
09	3	Screen capture(s) must match code from Q9.1 and Q9.2 **1 mark:** Screen capture shows the test being completed correctly.	**1** **AO3=1**

```
H - Help
S - Setup a basic board
B - Play a basic game
T - Play the test board
Q - Quit

Choice: T
 11  10   9   8   7   6
------------------------
| 1|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 16
------------------------
| 1|| 0|| 3|| 1|| 2|| 2| Player 2 holds: 20
------------------------
  0   1   2   3   4   5

Player 1, which pit do you want to take seeds from? 0
What kind of move (A - original, B - new)? B
Collected Seeds
 11  10   9   8   7   6
------------------------
| 1|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 19
------------------------
| 0|| 1|| 0|| 1|| 2|| 2| Player 2 holds: 20
------------------------
  0   1   2   3   4   5

Player 2, which pit do you want to take seeds from? 11
What kind of move (A - original, B - new)? B
Collected Seeds
 11  10   9   8   7   6
------------------------
| 0|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 19
------------------------
| 1|| 0|| 0|| 1|| 2|| 2| Player 2 holds: 21
------------------------
  0   1   2   3   4   5

Player 1, which pit do you want to take seeds from? 3
What kind of move (A - original, B - new)? A
Collected seeds from pit: 4
 11  10   9   8   7   6
------------------------
| 0|| 2|| 0|| 0|| 0|| 0| Player 1 holds: 22
------------------------
| 1|| 0|| 0|| 0|| 0|| 2| Player 2 holds: 21
------------------------
  0   1   2   3   4   5

Player 2, which pit do you want to take seeds from?
```