1 A program stores data for a board game using Object-Oriented Programming (OOP). The game has objects that are placed on the board. Each object has a string code (for example `"A"`) and an integer value (for example, `2`).

The class `BoardObject` stores the data about the objects that can be placed on the board:

BoardObject	
`Code : String`	stores the board object's code
`Value : Integer`	stores the integer value of the board object
`Constructor()`	initialises the attributes to the parameter values
`GetCode()`	returns `Code`
`GetValue()`	returns `Value`

(a) (i) Write program code to declare the class `BoardObject` and its constructor.

Do **not** declare the other methods.

Use your programming language appropriate constructor.

If you are writing in Python, include attribute declarations using comments.

Save your program as **Question1_N25**.

Copy and paste the program code into part **1(a)(i)** in the evidence document.

[5]

(ii) The methods `GetCode()` and `GetValue()` return the appropriate attribute.

Write program code for `GetCode()` and `GetValue()`

Save your program.

Copy and paste the program code into part **1(a)(ii)** in the evidence document.

[3]

(iii) The table shows the code and value of **five** board objects. The table has the variable identifier where each of the objects are stored.

Variable identifier	Code	Value
Object1	"A"	2
Object2	"B"	3
Object3	"C"	5
Object4	"D"	2
Object5	"E"	7

Write program code for the main program to instantiate each of the **five** board objects and store them in the variables with the identifiers given.

Save your program.

Copy and paste the program code into part **1(a)(iii)** in the evidence document.

[3]

(b) The board objects are placed on a board that is represented by a 0-indexed 2D array:

- the board is a 10 x 10 grid
- each position on the board is identified by a row and column number
- rows are numbered 0 to 9
- columns are numbered 0 to 9
- board objects can be placed at a row and column position
- each board position is initialised with an empty `BoardObject`, an empty `BoardObject` has `Code = "-"` and `Value = 0`

For example, the element highlighted in the given board is in row 1 and column 3.

Row	Column									
	0	1	2	3	4	5	6	7	8	9
0										
1				■						
2										
3										
4										
5										
6										
7										
8										
9										

The class `Board` stores the data about the board and where the board objects are placed.

Board	
`TheBoard : ARRAY[0:9, 0:9] of BoardObject`	stores the board contents as a 2D array of 10 x 10 elements of type `BoardObject`
`Constructor()`	initialises each of `TheBoard` elements to an empty `BoardObject` with `Code = "-"` and `Value = 0`
`GetObject()`	takes a row and column number as parameters and returns the `BoardObject` at the row, column position
`SetObject()`	takes a `BoardObject`, row number and column number as parameters. Stores the `BoardObject` in `TheBoard` at the given row, column position
`DisplayBoard()`	outputs the code of each `BoardObject` stored in `TheBoard`, one row at a time

(i) Write program code to declare the class `Board` and its constructor.

Do **not** declare the other methods.

Use your programming language appropriate constructor.

If you are writing in Python, include attribute declarations using comments.

Save your program.

Copy and paste the program code into part **1(b)(i)** in the evidence document.

[4]

(ii) The method `GetObject()` takes a row number and column number as parameters.

The method returns the `BoardObject` stored at the parameter position.

Write program code for `GetObject()`

Save your program.

Copy and paste the program code into part **1(b)(ii)** in the evidence document.

[2]

(iii) The method `SetObject()` takes **three** parameters: a `BoardObject`, row number and column number.

The method stores the `BoardObject` parameter in the row, column position in `TheBoard`

Write program code for `SetObject()`

Save your program.

Copy and paste the program code into part **1(b)(iii)** in the evidence document.

[2]

(iv) The method `DisplayBoard()` outputs the `Code` of each `BoardObject` stored in `TheBoard` using `GetCode()`

Each row in `TheBoard` is output on one line with a space between each `Code`

For example, the following board contains these four board objects:

- one object has code `"A"` in row 0 column 7
- one object has code `"B"` in row 0 column 9
- one object has code `"C"` in row 1 column 1
- one object has code `"E"` in row 6 column 5

The other board objects are empty. The output for this board will be:

```
- - - - - - - A - B
- C - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - E - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
```

Write program code for `DisplayBoard()`

Save your program.

Copy and paste the program code into part **1(b)(iv)** in the evidence document.

[3]

(c) (i) The table gives the row and column position on the board to store each of the five objects created in part **1(a)(iii)**.

Object identifier	row position	column position
Object1	0	0
Object2	9	9
Object3	4	5
Object4	2	2
Object5	8	7

Write program code to amend the main program to:

- declare a new instance of `Board()`
- store each `BoardObject` in the position given in the table
- call `DisplayBoard()` for the new board object.

Save your program.

Copy and paste the program code into part **1(c)(i)** in the evidence document.

[3]

(ii) Test your program.

Take a screenshot of the output(s).

Save your program.

Copy and paste the screenshot(s) into part **1(c)(ii)** in the evidence document.

[1]

(d) (i) Amend the main program to:

- repeatedly take a row position as input until it is between 0 and 9 inclusive
- repeatedly take a column position as input until it is between 0 and 9 inclusive
- use the appropriate method(s) to identify if there is an object in the array position input
- output `"Miss"` if there is an empty `BoardObject` in that position
- output the `Code` and `Value` if there is a non-empty `BoardObject` in that position.

All outputs must include appropriate messages.

Save your program.

Copy and paste the program code into part **1(d)(i)** in the evidence document.

[4]

(ii) Test your program by entering this test data in the order given:

Row position first input: 10

Row position second input: 4

Column position first input: –1

Column position second input: 5

Take a screenshot of the output(s).

Save your program.

Copy and paste the screenshot(s) into part **1(d)(ii)** in the evidence document.

[1]