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 May/June 2022 series for most Cambridge IGCSE, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.
These general marking principles must be applied by all examiners when marking candidate answers. They should be applied alongside the specific content of the mark scheme or generic level descriptors for a question. Each question paper and mark scheme will also comply with these marking principles.

<table>
<thead>
<tr>
<th>GENERIC MARKING PRINCIPLE 1:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marks must be awarded in line with:</td>
</tr>
<tr>
<td>• the specific content of the mark scheme or the generic level descriptors for the question</td>
</tr>
<tr>
<td>• the specific skills defined in the mark scheme or in the generic level descriptors for the question</td>
</tr>
<tr>
<td>• the standard of response required by a candidate as exemplified by the standardisation scripts.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GENERIC MARKING PRINCIPLE 2:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marks awarded are always <strong>whole marks</strong> (not half marks, or other fractions).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GENERIC MARKING PRINCIPLE 3:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marks must be awarded <strong>positively:</strong></td>
</tr>
<tr>
<td>• marks are awarded for correct/valid answers, as defined in the mark scheme. However, credit is given for valid answers which go beyond the scope of the syllabus and mark scheme, referring to your Team Leader as appropriate</td>
</tr>
<tr>
<td>• marks are awarded when candidates clearly demonstrate what they know and can do</td>
</tr>
<tr>
<td>• marks are not deducted for errors</td>
</tr>
<tr>
<td>• marks are not deducted for omissions</td>
</tr>
<tr>
<td>• answers should only be judged on the quality of spelling, punctuation and grammar when these features are specifically assessed by the question as indicated by the mark scheme. The meaning, however, should be unambiguous.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GENERIC MARKING PRINCIPLE 4:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rules must be applied consistently, e.g. in situations where candidates have not followed instructions or in the application of generic level descriptors.</td>
</tr>
<tr>
<td>GENERIC MARKING PRINCIPLE 5:</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td>Marks should be awarded using the full range of marks defined in the mark scheme for the question (however; the use of the full mark range may be limited according to the quality of the candidate responses seen).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GENERIC MARKING PRINCIPLE 6:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marks awarded are based solely on the requirements as defined in the mark scheme. Marks should not be awarded with grade thresholds or grade descriptors in mind.</td>
</tr>
<tr>
<td>Question</td>
</tr>
<tr>
<td>----------</td>
</tr>
</tbody>
</table>
| 1(a)     | 1 mark per mark point  
• declaring array StackData and pointer StackPointer as (global data structures)  
• StackData has 10 integer elements  
• StackPointer initialised to 0  

Example program code:  
**VB.NET**  
Dim StackData(9) As Integer  
Dim StackPointer As Integer  
Sub Main()  
  StackPointer = 0  
end Sub

**Python**  
global StackData #integer  
global StackPointer  
StackData = [0,0,0,0,0,0,0,0,0,0] #integer  
StackPointer = 0

**Java**  
import java.util.Scanner;  
class Question1{  
  public static Integer[] StackData;  
  public static Integer StackPointer;  
  public static void main(String args[]){  
    StackData = new Integer[10];  
    StackPointer = 0;  
  }  
}
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 1(b)     | 1 mark per mark point  
  • procedure header with sensible identifier (and end where appropriate)  
  • outputting StackPointer  
  • outputting all 10 elements in array  

Example program code:  
**VB.NET**  
Sub PrintArray()  
    Console.WriteLine(StackPointer)  
    For x = 0 To 9  
        Console.WriteLine(StackData(x))  
    Next  
End Sub  

**Python**  
def PrintArray():  
    global StackData  
    global StackPointer  
    print(StackPointer)  
    for x in range (0, 10):  
        print(StackData[x])  

**Java**  
public static void PrintArray(){  
    System.out.println(StackPointer);  
    for(int x = 0; x < 10 ;x++){  
        System.out.println(StackData[x]);  
    }  
}  

<table>
<thead>
<tr>
<th>Marks</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question</td>
<td>Answer</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
</tr>
</tbody>
</table>
| 1(c) | 1 mark per mark point  
- function `Push()` taking an integer parameter  
- checking if stack is full ...  
- ...and returning `FALSE`  
- (if not full) storing parameter to stack at `StackPointer` ...  
- ...incrementing `StackPointer`  
- ...returning `TRUE` | 6 |
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 1(c) | **Java**  
```java
public static Boolean Push(Integer DataToPush){  
    if(StackPointer == 10){  
        return false;  
    }else{  
        StackData[StackPointer] = DataToPush;  
        StackPointer = StackPointer + 1;  
        return true;  
    }
}
``` | --- |
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 1(d)(i)  | 1 mark per mark point  
• Inputting 11 numbers ...  
• ...calling Push() with each number input as a parameter ...  
• ...outputting appropriate message if TRUE returned  
• ...outputting appropriate message if FALSE returned  
• Calling their output procedure after all 11 additions | 5     |

Example program code:

**VB.NET**

```vbnet
Sub Main()
    StackPointer = 0
    Dim TempNumber As Integer
    For x = 0 To 10
        Console.WriteLine("Enter a number")
        TempNumber = Console.ReadLine()
        If Push(TempNumber) Then
            Console.WriteLine("Stored")
        Else
            Console.WriteLine("Stack full")
        End If
    Next
    PrintArray()
End Sub
```
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1(d)(i)</td>
<td>Python</td>
</tr>
<tr>
<td></td>
<td>#main</td>
</tr>
<tr>
<td></td>
<td>StackPointer = 0</td>
</tr>
<tr>
<td></td>
<td>StackData = [0,0,0,0,0,0,0,0,0,0]</td>
</tr>
<tr>
<td></td>
<td>for x in range(0, 11):</td>
</tr>
<tr>
<td></td>
<td>TempNumber = int(input(&quot;Enter a number&quot;))</td>
</tr>
<tr>
<td></td>
<td>if Push(TempNumber) == True:</td>
</tr>
<tr>
<td></td>
<td>print(&quot;Stored&quot;)</td>
</tr>
<tr>
<td></td>
<td>else:</td>
</tr>
<tr>
<td></td>
<td>print(&quot;Stack full&quot;)</td>
</tr>
<tr>
<td></td>
<td>PrintArray()</td>
</tr>
<tr>
<td></td>
<td>Java</td>
</tr>
<tr>
<td></td>
<td>public static void main(String[] args){</td>
</tr>
<tr>
<td></td>
<td>StackData = new Integer[10];</td>
</tr>
<tr>
<td></td>
<td>StackPointer = 0;</td>
</tr>
<tr>
<td></td>
<td>Integer TempNumber = 0;</td>
</tr>
<tr>
<td></td>
<td>for(int x = 0; x &lt; 10; x++){</td>
</tr>
<tr>
<td></td>
<td>System.out.println(&quot;Enter a number&quot;);</td>
</tr>
<tr>
<td></td>
<td>Scanner scanner = new Scanner(System.in);</td>
</tr>
<tr>
<td></td>
<td>TempNumber = Integer.parseInt(scanner.nextLine());</td>
</tr>
<tr>
<td></td>
<td>if(Push(TempNumber)){</td>
</tr>
<tr>
<td></td>
<td>System.out.println(&quot;Stored&quot;);</td>
</tr>
<tr>
<td></td>
<td>}else{</td>
</tr>
<tr>
<td></td>
<td>System.out.println(&quot;Stack full&quot;);</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>PrintArray();</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1(d)(ii)</td>
<td>1 mark for inputting all 11 numbers, message for first 10 saying added (11 to 20), message stating 11th number stating stack full. Full array contents output (11 12 13 14 15 16 17 18 19 20). e.g.</td>
</tr>
</tbody>
</table>
### Question

1(e)(i) 1 mark per mark point

- Pop() function header (and close where appropriate) and returning a number in all possible situations
- checking if stack is empty (StackPointer is 0) and returning -1
- (otherwise) accessing the item at the top of the stack …
- …decrementing the stack pointer
- …returning the item removed

#### Example Program code:

**VB.NET**

```vbnet
Function Pop()
    Dim ReturnData As Integer
    If StackPointer = 0 Then
        Return -1
    Else
        ReturnData = StackData(StackPointer - 1)
        StackPointer = StackPointer - 1
        Return ReturnData
    End If
End Function
```

**Python**

```python
def Pop():
    global StackData
    global StackPointer
    if StackPointer == 0:
        return -1
    else:
        ReturnData = StackData[StackPointer - 1]
        StackPointer = StackPointer - 1
        return ReturnData
```

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1(e)(i)</td>
<td>1 mark per mark point</td>
<td>5</td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
<td>Marks</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>-------</td>
</tr>
</tbody>
</table>
| 1(e)(i)  | **Java**
  public static Integer Pop()
  {
    Integer ReturnData = 0;
    if(StackPointer == 0)
      return -1;
    else{
      ReturnData = StackData[StackPointer - 1];
      StackPointer = StackPointer - 1;
      return ReturnData;
    }
  } | |
| 1(e)(ii) | 1 mark per mark point
  - output of before removed with 11 inputs
  - output of stack (after, this could be 11–20, 11–18, or 11–18 then 'null' 'null's)
  e.g. | 2 |
<p>|         | <img src="image.png" alt="Image" /> | |</p>
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>2(a)</td>
<td>1 mark per mark point</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• in main local 2D array declared …</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• … with 10 × 10 integer elements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• initialising all array elements to a number…</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• …that is random between 1 and 100 (allow inclusive or exclusive)</td>
<td></td>
</tr>
<tr>
<td>Example program code:</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td><strong>VB.NET</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sub Main()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dim Random As New Random</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dim ArrayData(10, 10) As Integer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For x = 0 To 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For y = 0 To 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ArrayData(x, y) = Random.Next(1, 100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Console.ReadLine()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End Sub</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Python</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>import random</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#main</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ArrayData= [[0]*10 for i in range(10)] #integer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>for x in range(0, 10):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>for y in range(0,10):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ArrayData[x][y] = random.randint(1, 100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
<td>Marks</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>-------</td>
</tr>
</tbody>
</table>
| 2(a)     | **Java**  
```java
import java.util.Scanner;
import java.util.Random;

class Question2{
    public static Integer[][] ArrayData;
    public static void main(String args[]){
        Random Rand = new Random();
        ArrayData = new Integer[10][10];
        for(int x=0; x < 10; x++){
            for(int y = 0; y < 10; y++){
                ArrayData[x][y] = Rand.nextInt(100);}
        }
    }
}
``` | |
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 2(b)(i) | 1 mark per mark point  
- 1st outer loop (dimension 1)  
- 2nd loop (dimension 2)  
- inner for loop for all second dimension  
- Selection statement …  
- …swapping the numbers correctly  
Example program code:  
**VB.NET**  
Dim TempNumber As Integer  
Dim ArrayLength As Integer = 10  
For X = 0 To ArrayLength - 1  
    For Y = 0 To ArrayLength - 2  
        For Z = 0 To ArrayLength - Y - 2  
            if ArrayData(X, Z) > ArrayData(X, Z + 1) then  
                TempNumber = ArrayData(X, Z)  
                ArrayData(X, Z) = ArrayData(X, Z+1)  
                ArrayData(X, Z + 1) = TempNumber  
            end if  
        Next Z  
    Next Y  
Next X  
**Python**  
ArrayLength = 10  
for X in range(0, ArrayLength):  
    for Y in range(0, ArrayLength - 1):  
        for Z in range(0, ArrayLength - Y - 1):  
            if(ArrayData[X][Z] > ArrayData[X][Z+1]):  
                TempNumber = ArrayData[X][Z]  
                ArrayData[X][Z] = ArrayData[X][Z+1]  
                ArrayData[X][Z+1] = TempNumber  
Accept for MP5:  
ArrayData[X][Z], ArrayData[X][Z+1] = ArrayData[X][Z+1], ArrayData[X][Z] |   | 5 |
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>2(b)(i)</td>
<td><strong>Java</strong></td>
</tr>
<tr>
<td></td>
<td>Integer ArrayLength = 10;</td>
</tr>
</tbody>
</table>
|         | for(int X = 0; X < ArrayLength; X++){
|         |     for(int Y = 0; Y < ArrayLength; Y++){
|         |         for(int Z = 0; Z < ArrayLength - Y - 1; Z++){
|         |             if(ArrayData[X][Z] > ArrayData[X][Z + 1]){ |
|         |                 TempNumber = ArrayData[X][Z]; |
|         |                 ArrayData[X][Z] = ArrayData[X][Z+1]; |
|         |                 ArrayData[X][Z + 1] = TempNumber; |
|         |             }
|         |         }
|         |     }
|         | }        |
2(b)(ii) 1 mark per mark point

- procedure header (and end where appropriate)
- Outputting all $10 \times 10$ values with each 2nd dimension on a complete line
- Calling procedure before and after bubble sort

Example program code:

**VB.NET**

```
Sub Main()
    Dim random As New Random
    Dim ArrayData(10, 10) As Integer   For x = 0 To 9
        For y = 0 To 9
            ArrayData(x, y) = random.Next(1, 100)
        Next
    Next
    Console.WriteLine("before")   printarray(ArrayData)

    Dim TempNumber As Integer   Dim ArrayLength As Integer = 10
    For X = 0 To ArrayLength - 1
        For Y = 0 To ArrayLength - 2       For Z = 0 To ArrayLength - Y - 2
            If ArrayData(X, Z) > ArrayData(X, Z + 1) then
                TempNumber = ArrayData(X, Z)
                ArrayData(X, Z) = ArrayData(X, Z+1)
                ArrayData(X, Z + 1) = TempNumber
            end if       Next Z     Next Y
    Next X
    Console.WriteLine("after")   printarray(ArrayData)
Console.ReadLine() End Sub
```

3
### Question 2(b)(ii)

**Answer**

```vba
Sub Printarray(ByRef ArrayData(,) As Integer)
    For x = 0 To 9
        For y = 0 To 9
            Console.Write(ArrayData(x, y) & " ")
            Next
        Console.WriteLine()
    Next
End Sub
```

**Python**

```python
import random

def Printarray(ArrayData):
    for x in range(0, 10):
        for y in range(0, 10):
            print(ArrayData[x][y], " ", end='')
        print('')

#main
ArrayData= [[0]*10 for i in range(10)]  #integer
for x in range(0, 10):
    for y in range(0, 10):
        ArrayData[x][y] = random.randint(1, 100)

print("Before")
printarray(ArrayData)
ArrayLength = 10
for X in range(0, ArrayLength):
    for Y in range(0, ArrayLength):
        for Z in range(0, ArrayLength - Y - 1):
            if(ArrayData[X][Z] > ArrayData[X][Z+1]):
                TempNumber = ArrayData[X][Z]
                ArrayData[X][Z] = ArrayData[X][Z+1]
                ArrayData[X][Z+1] = TempNumber

print("After")
printarray(ArrayData)
```

**Marks**

---

© UCLES 2022
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
</table>
| 2(b)(ii) | **Java**

```java
import java.util.Scanner;
import java.util.Random;

class Question2{

    public static Integer[][] ArrayData;

    public static void printArray(Integer[][] theArrayData){
        for(int x = 0; x < 10; x++){
            for(int y = 0; y < 10; y++){
                System.out.printf(theArrayData[x][y] + " ");
            }
            System.out.println();
        }
    }

    public static void main(String args[]){
        Random rand = new Random();
        ArrayData = new Integer[10][10];
        for(int x=0; x < 10; x++){
            for(int y = 0; y < 10; y++){
                ArrayData[x][y] = rand.nextInt(100);
            }
        }
        Integer TempNumber = 0;

        System.out.println("Before");
        printArray(ArrayData);

        Integer ArrayLength = 10;
        for(int X = 0; X < ArrayLength; X++){
            for(int Y = 0; Y < ArrayLength; Y++){
                for(int Z = 0; Z < ArrayLength - Y - 1; Z++){
                    if(ArrayData[X][Z] > ArrayData[X][Z + 1]){
                        System.out.println(ArrayData[X][Z] + " > " + ArrayData[X][Z + 1]);
                    }
                }
            }
        }
    }
}
```

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
</table>
| 2(b)(ii) | TempNumber = ArrayData[X][Z];
           ArrayData[X][Z] = ArrayData[X][Z+1];
           ArrayData[X][Z + 1] = TempNumber;
           
           System.out.println("After");
           PrintArray(ArrayData); |

<p>| Marks |</p>
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>2(b)(iii)</td>
<td>1 mark for output showing array unsorted and then sorted on 1 of the dimensions e.g.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="before" /></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="after" /></td>
<td></td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
<td>Marks</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>-------</td>
</tr>
</tbody>
</table>
| 2(c)(i)  | 1 mark for each completed statement (6) 1 mark per mark point  • function declaration taking appropriate parameters and recursive calls  • remainder of the function is accurate including appropriate DIV operator.  

Example program code:  
**VB.NET**

```vbnet
Function BinarySearch(ByVal SearchArray(,) As Integer, Lower As Integer, Upper As Integer, SearchValue As Integer)
    Dim Mid As Integer
    If Upper >= 0 Then
        Mid = (Lower + (Upper - 1)) \ 2
        If SearchArray(0, Mid) = SearchValue Then
            Return Mid
        ElseIf SearchArray(0, Mid) > SearchValue Then
            Return BinarySearch(SearchArray, Lower, Mid - 1, SearchValue)
        Else
            Return BinarySearch(SearchArray, Mid + 1, Upper, SearchValue)
        End If
    End If
    Return -1
End Function
```

**Python**

```python
def BinarySearch(SearchArray, Lower, Upper, SearchValue):
    if Upper >= 0:
        Mid = int((Lower + (Upper - 1)) / 2)
        if SearchArray[0][Mid] == SearchValue:
            return Mid
        elif SearchArray[0][Mid] > SearchValue:
            return BinarySearch(SearchArray, Lower, Mid-1, SearchValue)
        else:
            return BinarySearch(SearchArray, Mid+1, Upper, SearchValue)
    return -1
```
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
</table>
| 2(c)(i)  | Java
public static Integer BinarySearch(Integer[][] SearchArray, Integer Lower, Integer Upper, Integer SearchValue){
    Integer Mid = 0;
    If Upper >= 0 {
        Mid = (Lower + (Upper - 1)) / 2;
        If SearchArray[0][Mid] == SearchValue {
            return Mid;
        } else if SearchArray[0][Mid] > SearchValue {
            return BinarySearch(SearchArray, Lower, Mid-1, SearchValue);
        } else{
            return BinarySearch(SearchArray, Mid+1, Upper, SearchValue);
        }
    } return -1;
} |
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 2(c)(ii) | 1 mark per mark point  
- screenshot outputting the index when Number is found  
- screenshot outputting -1 with a Number not found  
e.g. | | 2 |
### Question 3(a)

1 mark per mark point

- Card class declaration (and end where appropriate)
- Both attributes (Number and Colour) declared with suitable data types ...
- …as private
- correct constructor header (and end where appropriate) with two parameters ...
- …both parameters assigned to the attributes

**Example program code:**

**VB.NET**

```vbnet
Class Card
    Private Number As Integer
    Private Colour As String
    Sub New(Numberp, Colourp)
        Number = Numberp
        Colour = Colourp
    End Sub
End Class
```

**Python**

```python
class Card:
    #Number as Integer
    #Colour as string
    def __init__(self, Numberp, Colourp):
        self.__Number = Numberp
        self.__Colour = Colourp
```
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>3(a)</td>
<td><strong>Java</strong>&lt;br&gt;import java.util.Scanner;&lt;br&gt;import java.io.*;&lt;br&gt;class Card{&lt;br&gt;    private Integer Number;&lt;br&gt;    private String Colour;&lt;br&gt;    public Card(Integer pNumber, String pColour){&lt;br&gt;        Number = pNumber;&lt;br&gt;        Colour = pColour;&lt;br&gt;    }&lt;br&gt;    public static void main(String args[]){&lt;br&gt;    }&lt;br&gt;}</td>
<td></td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
<td>Marks</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
</tbody>
</table>
| 3(b)     | 1 mark per mark point  
<p>|          | • 1 get method header (and close where appropriate) with no parameter … | 3     |
|          | • … returning attribute                                                |       |
|          | • 2nd correct get method                                               |       |
|          | <strong>Example program code:</strong>                                              |       |
|          | <strong>VB.NET</strong>                                                             |       |
|          | Function GetNumber()                                                   |       |
|          |     Return Number                                                      |       |
|          | End Function                                                           |       |
|          | Function GetColour()                                                   |       |
|          |     Return Colour                                                      |       |
|          | End Function                                                           |       |
|          | <strong>Python</strong>                                                             |       |
|          |     def GetNumber(self):                                               |       |
|          |         return self.__Number                                           |       |
|          |     def GetColour(self):                                               |       |
|          |         return self.__Colour                                           |       |
|          | <strong>Java</strong>                                                               |       |
|          |     public Integer GetNumber(){                                       |       |
|          |         return Number;                                                 |       |
|          |     }                                                                  |       |
|          |     public String GetColour(){                                         |       |
|          |         return Colour;                                                 |       |</p>
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 3(c)     | 1 mark per mark point to max 7  
- Declaration of array with 30 elements of type Card  
- Opening the text file `CardValues.txt` for read  
- Looping until EOF/30 times  
- Reading in all sets of 2 lines (number then colour) ...  
- ...creating object of type Card...  
- ...with number and colour read in from file ...  
- ...storing in next array element  
- Try and catch for file handling...s  
- ....with appropriate outputs  
- Closing the file in a suitable place |

Example program code:  
**VB.NET**  
```vbnet
Sub Main()
    Dim CardArray(0 To 29) As Card
    Dim NumberRead As Integer
    Dim ColourRead As String

    Try
        Dim Filename As String = "CardValues.txt"
        Dim FileReader As New System.IO.StreamReader(Filename)
        For x = 0 To 29
            NumberRead = Integer.Parse(FileReader.ReadLine())
            ColourRead = FileReader.ReadLine()  
            CardArray(x) = New Card(NumberRead, ColourRead)
        Next
    Catch ex As Exception
        ' Handle exceptions here
    End Try

    FileReader.close()
End Sub
```
3(c)

```csharp
Console.WriteLine("Invalid file")
End Try
End Sub
```

**Python**

```python
CardArray = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] #integer

try:
    Filename = "CardValues.txt"
    File = open(Filename,'r')
    for x in range(0,30):
        NumberRead = int(File.readline())
        ColourRead = File.readline()
        CardArray[x] = Card(NumberRead, ColourRead)

    File.close
except IOError:
    print("Could not find file")
```

**Java**

```java
public static void main(String args[]){
    Card[] CardArray = new Card[30];
    Integer NumberRead;
    String ColourRead;
    String FileName = "CardValues.txt";
    try{
        FileReader F = new FileReader(FileName);
        BufferedReader Reader = new BufferedReader(f);
        for(Integer x = 0; x < 30; x++){
            NumberRead = Integer.parseInt(Reader.readLine());
            ColourRead = Reader.readLine();
            CardArray[x] = new Card(NumberRead, ColourRead);
        }
        Reader.close();
    }
}```
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 3(c)     | catch(FileNotFoundException ex){
              System.out.println("No file found");
          }
catch(IOException ex){
              System.out.println("No file found");
          }
|         |        |       |
### Question 3(d)

1 mark per mark point to max 6

- Implementing a suitable way of storing which card have been selected
- `Function ChooseCard() header (and close) and returning an integer (index) in all cases`
- `Reading in array index from the user ...`
  - with suitable validation **looping** until it is between 1 and 30 (inclusive)
- `Converting input to array index (e.g. −1 each time)`
- `Check if the input is already selected...`
  - ... if it is selected, loop until index input is not already selected
  - ... returning index of available card selected
- Stores the valid Card chosen as taken (using any suitable method)

**Example program code:**

**VB.NET**

```
Dim NumbersChosen(0 To 29) As Boolean
Sub Main()
    For x = 0 To 29
        NumbersChosen(x) = False
    Next
    ... End Sub

Function chooseCard()
    Dim CardSelected As Integer
    Dim flagContinue As Boolean = True
    While flagContinue = True
        Console.WriteLine("Select a Card from 1 to 30")
        CardSelected = Console.ReadLine()
        If CardSelected < 1 Or CardSelected > 30 Then
            Console.WriteLine("Number must be between 1 and 30")
        Else
            flagContinue = False
            NumbersChosen(CardSelected - 1) = True
        End If
    End While
End Function
```
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
</table>
| 3(d)     | ElseIf NumbersChosen(CardSelected - 1) = True Then  
          Console.WriteLine("Already taken")  
          Else  
          Console.WriteLine("valid")  
          flagContinue = False  
          End If  
          End While  
          NumbersChosen(CardSelected - 1) = True  
          Return CardSelected - 1  
          End Function  

**Python**  

global NumbersChosen  

...  

def chooseCard ():  
    global NumbersChosen  
    flagContinue = True  
    while flagContinue == true:  
        CardSelected = int(input("Select a Card from 1 to 30"))  
        if CardSelected < 1 or CardSelected > 30:  
            print("Number must be between 1 and 30")  
        elif NumbersChosen(CardSelected - 1) == True:  
            print("Already taken")  
        else:  
            print("Valid")  
            flagContinue = False  
            NumbersChosen[CardSelected-1] = True  
            return CardSelected-1  

...  

#main  

...  

NumbersChosen = [False for i in range(30)]
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 3(d) Java | public static Boolean[] NumbersChosen = new Boolean[30];

    public Integer chooseCard (){  
        Boolean flagContinue = true;
        Integer CardSelected = -1;
        while(flagContinue){
            System.out.println("Select a Card from 1 to 30");
            Scanner scanner = new Scanner(System.in);
            CardSelected = Integer.parseInt(scanner.nextLine());
            if(CardSelected < 1 || CardSelected > 30){
                System.out.println("Number must be between 1 and 30");
            }else if(NumbersChosen[CardSelected - 1]){  
                System.out.println("Already taken");
            }else{
                System.out.println("Valid");
                flagContinue = false;
            }
        }
        NumbersChosen[CardSelected - 1] = true;
        return CardSelected - 1;
    } | |
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Marks</th>
</tr>
</thead>
</table>
| 3(e)(i)  | 1 mark per mark point  
  • declaring array `Player1` of type Card  
  • calling the function `ChooseCard()` four times  
  • storing the card, that is in the index returned, in the array `Player1`  
  • outputting all four numbers and colours in `Player1` ...  
  • .... using the get methodss  

Example program code:

**VB.NET**

```vbnet
Dim Player1(0 To 3) As Card  
For x = 0 To 3  
    Player1(x) = CardArray(ChooseCard(NumbersChosen))  
Next  
for x = 0 to 3  
    console.writeline(Player1(x).GetColour)  
    console.writeline(Player1(x).GetNumber)  
next x
```

**Python**

```python
Player1 = [] # of type Card  
for x in range(0, 4):  
    ReturnNumber = ChooseCard()  
    Player1.append(CardArray[ReturnNumber])  
for x in range(0, 4):  
    print(Player1[x].GetColour())  
    print(Player1[x].GetNumber())
```
### Question 3(e)(i)

**Java**

```java
Card[] Player1 = new Card[5];
    for(Integer x = 0; x < 5; x++){
        Player1[x] = CardArray[ChooseCard()];
    }
    for(Integer x = 0; x < 5; x++){
        System.out.println(Player1[x].GetColour());
        System.out.println(Player1[x].GetNumber());
    }
```

### Question 3(e)(ii)

1 mark for both tests

Test 1: inputting 1, 5, 9, 10. Outputting: 1 red 9 green 9 orange 10 red
Test 2: inputting 2 2 3 4 4 5. Outputting: 2 already taken. Then 5 black 2 while 4 red 9 green

Test 1 e.g.

```
Select a card from 1 to 301
Valid
Select a card from 1 to 305
Valid
Select a card from 1 to 309
Valid
Select a card from 1 to 3010
Valid
  1
    red
  9
    green
  9
    orange
  10
    red
```
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
</table>
| 3(e)(ii) | Test 2 e.g.  
  Select a card from 1 to 302  
  Valid  
  Select a card from 1 to 302  
  Already taken  
  Select a card from 1 to 303  
  Valid  
  Select a card from 1 to 304  
  Valid  
  Select a card from 1 to 304  
  Already taken  
  Select a card from 1 to 305  
  Valid  
  5  
  black  
  2  
  white  
  4  
  red  
  9  
  green | Marks |