MORSE CODE: Programming Tasks

Suggested Solutions and Mark Scheme (VB.NET)

The following are recommended solutions, and not an exhaustive list of all possible solutions to each task. The marking guidance should be used as a guide only. Discretion should be used in awarding credit where alternative solutions are given.

Task 1 (max. 4 marks)

1 mark IF statement after `FileName` has been initialised

1 mark .txt suffix added only if missing

```
Filename = Console.ReadLine()
If Filename.IndexOf(".txt") = -1 Then
    Filename = Filename & ".txt"
End If
```

1 mark Morse code translated to 'TEA X' when suffix is not included

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: R
Enter file name: message
 - . .-   -..-
TEA X
```

1 mark Morse code translated to 'TEA X' when suffix is included

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: R
Enter file name: message.txt
 - . .-   -..-
TEA X
```


Task 2 (max. 4 marks)

1 mark Clause (probably Else) triggered by any input other than R, S or X

1 mark Correct message displayed in clause (**R**. any alternative wording, **I**. spelling/capitalisation errors)

```
If MenuOption = "R" Then
    ReceiveMorseCode(Dash, Letter, Dot)
ElseIf MenuOption = "S" Then
    SendMorseCode(MorseCode)
ElseIf MenuOption = "X" Then
    ProgramEnd = True
Else
    Console.WriteLine("Please select an option from the menu")
End If
```

1 mark Capital Q causing error followed by main menu being re-displayed

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: Q
Please select an option from the menu

Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: _
```

1 mark Capital R followed by prompt for file

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: R
Enter file name: _
```


Task 3 (max. 5 marks)

1 mark Uppercase options still function correctly

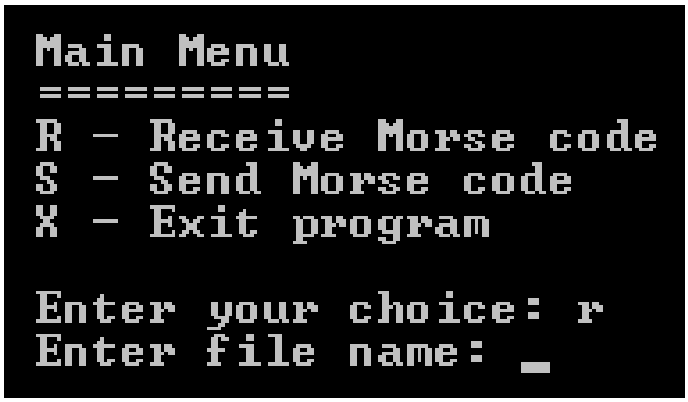
1 mark Lowercase r, s and x have the same effect as R, S and X respectively

1 mark No other inputs trigger a menu option

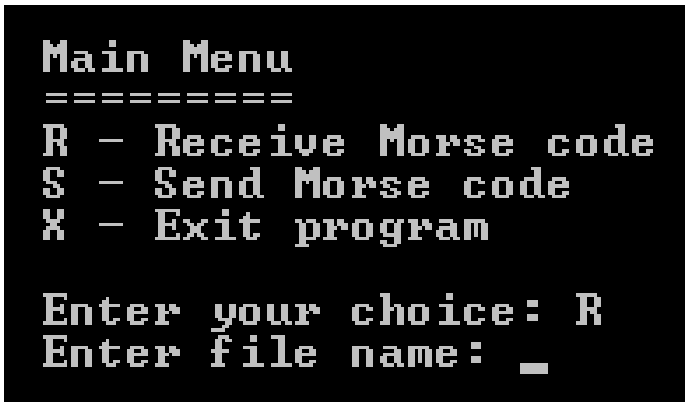
```
MenuOption = GetMenuOption()
MenuOption = MenuOption.ToUpper
If MenuOption = "R" Then
    ReceiveMorseCode(Dash, Letter, Dot)
ElseIf MenuOption = "S" Then
    SendMorseCode(MorseCode)
ElseIf MenuOption = "X" Then
    ProgramEnd = True
End If
```

Full marks can be awarded for any functioning solution; an alternative is to use OR operator to check for lowercase and uppercase letter in each If/ElseIf statement.

1 mark Lowercase r followed by prompt for file



1 mark Uppercase R followed by prompt for file



Task 4 (max. 14 marks)

1 mark Creation of a loop that begins before prompt for message

1 mark Termination condition possible (**A**. if termination condition never actually reached)

```
Dim validEntry As Boolean = False
Dim responseToYN As String

While validEntry = False

    Console.Write("Enter your message (uppercase letters and spaces only): ")
    PlainText = Console.ReadLine()
```

1 mark Every character in the input string is checked

1 mark Any lowercase characters would be detected

1 mark Integer incremented, Boolean set or complex If statement to denote detected lowercase character

```
Dim lowerCaseCout As Integer = 0
For x = 1 To PlainText.Length
    If Char.IsLower(PlainText(x - 1)) Then
        lowerCaseCout += 1
    End If
Next
```

1 mark Check for any detected lowercase character (**A**. if part of same complex If statement as above)

1 mark Correct prompt displayed (**R**. any alternative wording, **I**. spelling/capitalisation/spacing errors) only if lowercase characters have been entered

1 mark Response to prompt processed via variable or complex If statement

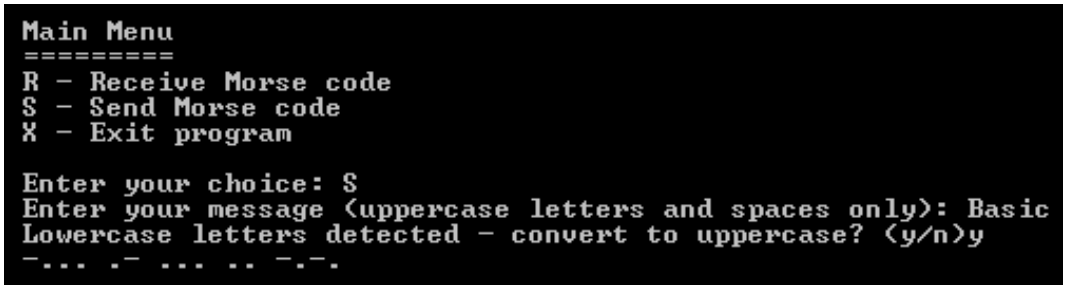
1 mark If user enters 'y' (**I**. case), convert `PlainText` to uppercase

1 mark Loop terminated if `PlainText` converted to uppercase

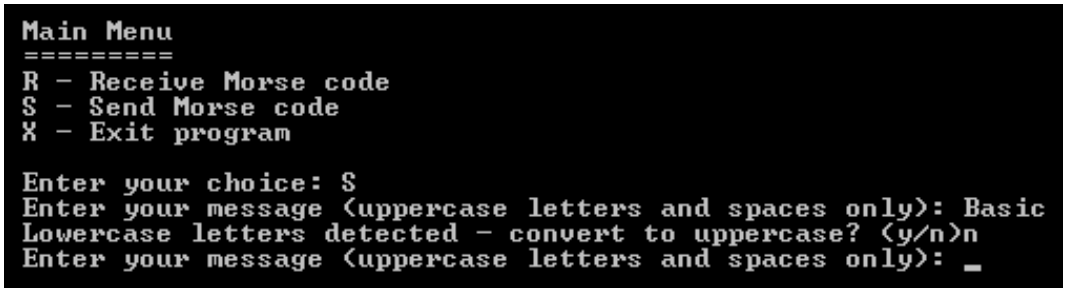
1 mark Loop terminated if there were no lowercase characters

```
    If lowerCaseCout > 0 Then
        Console.Write("Lowercase letters detected - convert to uppercase? (y/n)")
        responseToYN = Console.ReadLine
        If Not (responseToYN = "n") Then
            PlainText = PlainText.ToUpper
            validEntry = True
        End If
    Else
        validEntry = True
    End If
End While
```

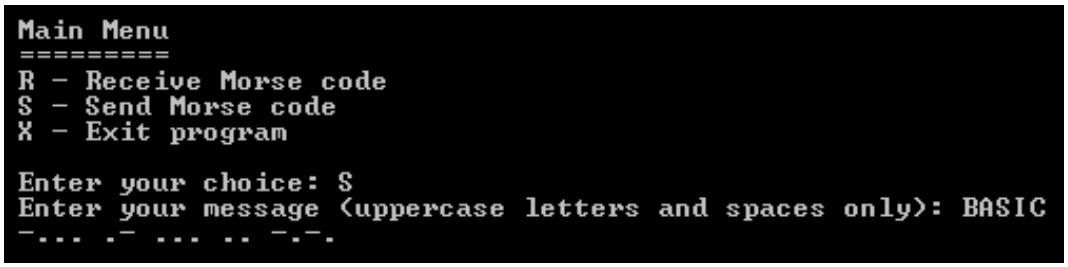

1 mark Input sequence of 'S', 'Basic', 'y' and Morse code output



1 mark Input sequence of 'S', 'Basic', 'n' and re-prompt for message



1 mark Input sequence of 'S', 'BASIC' and Morse code output



Task 5 (max. 11 marks)

1 mark New menu option added in `DisplayMenu` (**R**. any alternative wording, **I**. spelling/capitalisation errors)

```
Sub DisplayMenu()
    Console.WriteLine()
    Console.WriteLine("Main Menu")
    Console.WriteLine("=========")
    Console.WriteLine("R - Receive Morse code")
    Console.WriteLine("S - Send Morse code")
    Console.WriteLine("D - Display Morse alphabet")
    Console.WriteLine("X - Exit program")
    Console.WriteLine()
End Sub
```

1 mark Appropriate selection structure used in `SendReceiveMessages`

1 mark Call to `DisplayAlphabet` in correct place

1 mark Letter and `MorseCode` arrays passed (**R**. if any other structures or variables are passed)

```
If MenuOption = "R" Then
    ReceiveMorseCode(Dash, Letter, Dot)
ElseIf MenuOption = "S" Then
    SendMorseCode(MorseCode)
ElseIf MenuOption = "D" Then
    ShowAlphabet(Letter, MorseCode)
ElseIf MenuOption = "X" Then
    ProgramEnd = True
End If
```

1 mark Sub `DisplayAlphabet` declared

1 mark Two string arrays declared as parameters

1 mark An attempt to loop through each element of the arrays, even if unsuccessful

1 mark All plaintext letters correctly displayed

1 mark All Morse code letters correctly displayed

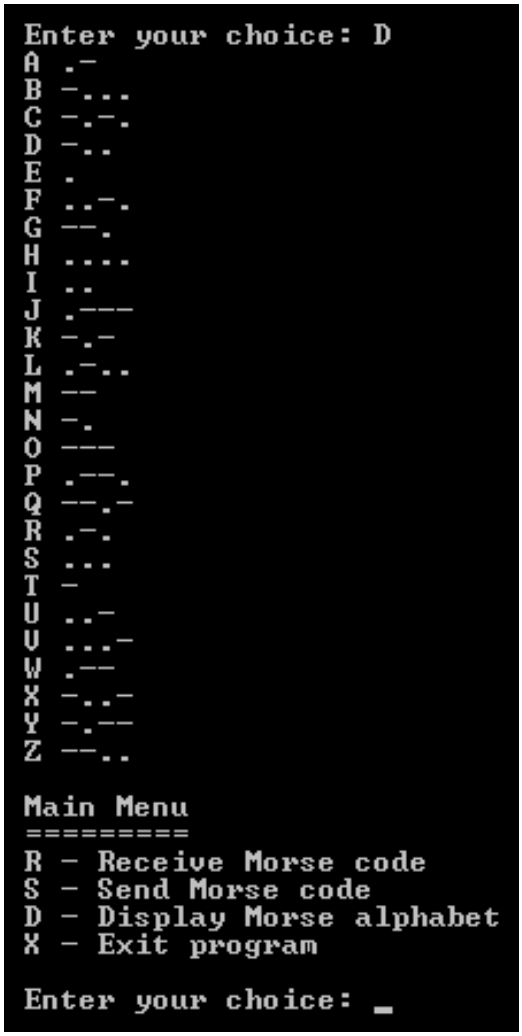
1 mark Space not displayed

```
Sub ShowAlphabet(ByVal letter() As String, ByVal MorseCode() As String)

    For x = 1 To letter.Length - 1
        Console.WriteLine(letter(x) & " " & MorseCode(x))
    Next

End Sub
```


1 mark Input of D and correct display of plaintext and Morse alphabets (**R**. if each character does not have its own line or if space is displayed)



Task 6 (max. 11 marks)

1 mark Declaration of `GetManualTransmission` as a string function

1 mark Prompting user for transmission signal (**R**. any alternative wording, **I**. spelling/capitalisation errors)

1 mark Returning the user input

```
Function GetManualTransmission() As String

    Console.WriteLine("Enter transmission signal")
    Return Console.ReadLine

End Function
```

1 mark Loop that continues until F or M entered

1 mark Menu options correctly displayed (**R**. any alternative wording, **I**. spelling/capitalisation errors)

1 mark Storage of response in string variable

```
Dim selection As String = ""

While Not (selection = "F" Or selection = "M")

    Console.WriteLine("F - read the transmission from a file")
    Console.WriteLine("M - enter the transmission manually")
    selection = Console.ReadLine

End While
```

1 mark User input of M results in no prompt for file name or attempt to access a file

1 mark User input of M results in a call to `GetManualTransmission`

1 mark User input of F results in prompt for file, followed by access to that file

1 mark Input string processed identically whether manually entered or from file

```
Try
    If selection = "F" Then
        Console.Write("Enter file name: ")
        Filename = Console.ReadLine()
        Dim Reader As New StreamReader(Filename)
        Transmission = Reader.ReadLine
        Reader.Close()
    Else
        Transmission = GetManualTransmission()
    End If
    Transmission = StripLeadingSpaces(Transmission)
    If Transmission.Length() > 0 Then
        Transmission = StripTrailingSpaces(Transmission)
        Transmission = Transmission + EOL
    End If
Catch
```

1 mark Input of 'R', 'M' and =⌂===⌂⌂⌂=⌂=⌂= results in plaintext AS

```
Enter your choice: R
F - read the transmission from a file
M - enter the transmission manually
M
Enter transmission signal
= ===   = = =
 ._ ...
AS
```


Task 7 (max. 3 marks)

1 mark Checking for seven consecutive spaces only at start of transmission

1 mark If there are seven consecutive spaces at the start, transmission is not trimmed

1 mark If there are not seven consecutive spaces at the start, transmission is trimmed as before

```
If TransmissionLength > 0 Then
    If Not (Transmission.IndexOf("       ") = 0) Then
        FirstSignal = Transmission(0)
        While FirstSignal = SPACE And TransmissionLength > 0
            TransmissionLength -= 1
            Transmission = Transmission.Substring(1)
            If TransmissionLength > 0 Then
                FirstSignal = Transmission(0)
            End If
        End While
    End If
End If
```


Task 8 (max. 9 marks)

1 mark Any attempt to detect two exceptions separately, even if unsuccessful

1 mark Try block contains code to attempt to access but not read specified file (N.B. user input of file name can be either before the Try block or within it)

1 mark Exceptions caught call `ReportError` with string 'File not found' (**R**. any alternative wording, **I**. spelling/capitalisation errors)

1 mark Exception would not be triggered by a call to `ReadLine`

```
Dim Transmission As String = EMPTYSTRING
Dim fileFound As Boolean = False

Try
    Filename = Console.ReadLine()
    reader = New StreamReader(Filename)
    fileFound = True
Catch ex As Exception
    ReportError("File not found")
End Try
```

1 mark Attempt to read file is only made if file exists

1 mark Second Try block, which must contain calls to `ReadLine`, `StripLeadingSpaces` and `StripTrailingSpaces`

1 mark Catch block calls `ReportError` with string 'File error' (**R**. any alternative wording, **I**. spelling/capitalisation errors)

1 mark No path through the code allows `Transmission` to remain uninitialised

```
If fileFound Then
    Try
        Transmission = reader.ReadLine
        reader.Close()
        Transmission = StripLeadingSpaces(Transmission)
        If Transmission.Length() > 0 Then
            Transmission = StripTrailingSpaces(Transmission)
            Transmission = Transmission + EOL
        End If
    Catch ex As Exception
        ReportError("File error")
        Transmission = EMPTYSTRING
    End Try

End If
```

1 mark Input of 'R' followed by 'mmm' resulting in 'File not found' message

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: R
Enter file name: mmm
*     File not found     *
```


Task 9 (max. 6 marks)

1 mark checking for presence of four equals

1 mark checking for presence of two equals followed by a space

1 mark checking for presence of two equals followed by EOL (**A**. "#")

1 mark 'Invalid file format' passed to `ReportError` in each case (**R**. any alternative wording, **I**. spelling/capitalisation errors)

1 mark 'Invalid file format' would only be displayed, at most, once per call to `GetTransmission`

1 mark Empty string returned in each case (**A.** "")

N.B. full credit can be given if all three invalid strings are checked for in a single line of code

```
If Transmission.Length() > 0 Then
    Transmission = StripTrailingSpaces(Transmission)
    Transmission = Transmission + EOL

    If Transmission.IndexOf("====") > -1 Then
        ReportError("Invalid file format")
        Return EMPTYSTRING
    End If

    If Transmission.IndexOf("== ") > -1 Then
        ReportError("Invalid file format")
        Return EMPTYSTRING
    End If

    If Transmission.IndexOf("==" & EOL) > -1 Then
        ReportError("Invalid file format")
        Return EMPTYSTRING
    End If

End If
```


Task 10 (max. 10 marks)

1 mark Variable to store transmission-format output

1 mark Loop to examine each character in `MorseCodeString`

1 mark Check whether character is a dot

1 mark If it is a dot, add a single equals to the transmission string (**A**. with or without space to follow)

1 mark Check whether the character is a dash

1 mark If it is a dot, add three equals to the transmission string (**A**. with or without space to follow)

1 mark Check whether the character is a space

1 mark If it is a space, either two or three spaces added to the transmission screen

1 mark Output would be correct for any valid combination of Morse code. One example is below, but candidates may use substring (or similar) to handle spaces between words.

```
Dim transmission As String = EMPTYSTRING

For x = 1 To MorseCodeString.Length

    If MorseCodeString(x - 1) = "." Then
        transmission = transmission & "= "
    ElseIf MorseCodeString(x - 1) = "-" Then
        transmission = transmission & "=== "
    ElseIf MorseCodeString(x - 1) = " " Then
        transmission = transmission & "  "
    End If

Next

Console.WriteLine(transmission)
```

1 mark Following input of 'S', then 'AQA AS', output includes correct transmission code

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: S
Enter your message (uppercase letters and spaces only): AQA AS
.- --.- .-   .- ...
= ===  === === = ===  = ===      = ===  = = =
```


Task 11 (max. 4 marks)

1 mark Check for length of `CodedLetter` being greater than four

1 mark Check for each invalid symbol given in question

1 mark Return * if invalid code detected (**A**. if errors made for previous marks)

1 mark No invalid value of `CodedLetter` would be checked using code already present in the preliminary material, e.g. only a valid `CodedLetter` value would reach the 'For i = 0...' line below

```
If CodedLetter.Length > 4 Or CodedLetter = "----" Or CodedLetter = "---." Or _
        CodedLetter = ".-.-" Or CodedLetter = "..--" Then
    Return "*"
End If

For i = 0 To CodedLetterLength - 1
```


Task 12 (max. 23 marks)

1 mark New menu item added at the correct point in the menu (**R**. any alternative wording, **I**. spelling/capitalisation errors)

```
Sub DisplayMenu()
    Console.WriteLine()
    Console.WriteLine("Main Menu")
    Console.WriteLine("==========")
    Console.WriteLine("R - Receive Morse code")
    Console.WriteLine("S - Send Morse code")
    Console.WriteLine("C - Convert Morse code")
    Console.WriteLine("X - Exit program")
    Console.WriteLine()
End Sub
```

1 mark Declaration of subroutine with two array parameters

1 mark Variable to track whether a Morse character is declared (**A**. if any attempt is made, without using such a variable, to determine whether an entered Morse character exists in the `MorseCode` array)

1 mark String variable to store the plaintext output

1 mark String variable to store Morse code input

1 mark String variable to store each Morse character in turn (mark can be awarded for alternative approaches to parsing individual letters, such as an array or list)

1 mark Any attempt to differentiate between a single space (between letters) and a triple space (between words)

1 mark Prompt the user to enter Morse code with any suitable method

1 mark Store the user response in a variable

```
Sub ConvertMorseCode(ByVal MorseCode() As String, ByVal Letter() As String)

    Dim validCharacter As Boolean
    Dim PlainText As String = ""
    Dim MorseMessage As String
    Dim MorseCharacter As String = ""
    Dim consecutiveSpaces As Integer = 0

    Console.Write("Enter Morse code message")
    MorseMessage = Console.ReadLine
```

1 mark Loop to examine each character in turn

1 mark Code checks each position for a dot

1 mark Code checks each position for a dash

```
For x = 0 To MorseMessage.Length - 1

    If MorseMessage(x) = "." Then
        MorseCharacter = MorseCharacter & "."
        consecutiveSpaces = 0
    ElseIf MorseMessage(x) = "-" Then
        MorseCharacter = MorseCharacter & "-"
        consecutiveSpaces = 0
    End If
```

1 mark Code checks each position for a space

1 mark Code checks whether it's looking at the last character in the input string

1 mark Either of the above causes the program to process the Morse code letter

1 mark Any search method that would successfully find the Morse character in the `MorseCode` array

1 mark Corresponding value in `Letter` array appended to plaintext for output

1 mark Call to `ReportError` with 'Data Entry Error' (**R**. different wordings, **I**. spelling errors) if word not found

```
If (MorseMessage(x) = " " And consecutiveSpaces = 0) Or _
    (x = MorseMessage.Length - 1 And Not (MorseMessage(x) = " ")) Then

    validCharacter = False

    For y = 0 To MorseCode.Length - 1
        If MorseCharacter = MorseCode(y) Then
            PlainText = PlainText & Letter(y)
            validCharacter = True
            MorseCharacter = ""
        End If
    Next

    If validCharacter = False Then
        ReportError("Data Entry Error")
        Exit Sub
    End If
```

1 mark Presence of three consecutive spaces adds one space to plaintext

1 mark `Plaintext` output at end of subroutine

1 mark Output would always be correct for valid Morse code entry, including any message ending in a triple space

```
        consecutiveSpaces += 1

    ElseIf MorseMessage(x) = " " And consecutiveSpaces = 2 Then
        PlainText = PlainText & " "
        consecutiveSpaces = 0
    ElseIf MorseMessage(x) = " " Then
        consecutiveSpaces += 1
    End If

Next

Console.WriteLine(PlainText)
```

1 mark Call to `ConvertMorseCode` from `SendReceiveMessages`, including `MorseCode` and `Letter` arrays

```
If MenuOption = "R" Then
    ReceiveMorseCode(Dash, Letter, Dot)
ElseIf MenuOption = "S" Then
    SendMorseCode(MorseCode)
ElseIf MenuOption = "C" Then
    ConvertMorseCode(MorseCode, Letter)
```

1 mark Output from Morse code to read 'HI THERE'

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
C - Convert Morse code
X - Exit program

Enter your choice: C
Enter Morse code message.... ..   - .... . .-. .
HI THERE
```


Task 13 (max. 7 marks)

1 mark Variable to store the indicated line number

```
Function GetTransmission() As String
    Dim Filename As String
    Dim Transmission As String
    Dim lineNumber As Integer
```

1 mark Prompt to enter line number after file name has been entered

1 mark User input in response to prompt stored in appropriate variable (the 'Try' line's position can be at any point in this part of the code)

```
Console.Write("Enter file name: ")
Try
    Filename = Console.ReadLine()
    Console.Write("Enter line number: ")
    lineNumber = Console.ReadLine
```

1 mark Use of any valid technique that could potentially reach beyond the first line

1 mark Technique would be effective, assuming integer has been input and that line exists in the file

```
Dim Reader As New StreamReader(Filename)
For x = 1 To lineNumber
    Transmission = Reader.ReadLine
Next
```

1 mark Input of 'R', 'message.txt' and '2' outputs 'No transmission found'

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: R
Enter file name: message.txt
Enter line number: 2
*     No transmission found     *
```

1 mark Input of 'R', 'message.txt' and '1' outputs 'TEA X'

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: R
Enter file name: message.txt
Enter line number: 1
 - . .-   -..-
TEA X
```


Task 14 (max. 5 marks)

1 mark Space stored with an index of 5

1 mark Other character stored with an index five higher, i.e. `Asc("A") + 6` instead of `+ 1`

1 mark Any attempt to 'wrap around' the characters V-Z, even if unsuccessful

1 mark V, W, X, Y, Z indexes altered to those of SPACE, A, B, C, D (0, 1, 2, 3, 4) respectively

```
If PlainTextLetter = SPACE Then
    Index = 5
Else
    Index = Asc(PlainTextLetter) - Asc("A") + 6
End If

If Index = 27 Then Index = 0
If Index = 28 Then Index = 1
If Index = 29 Then Index = 2
If Index = 30 Then Index = 3
If Index = 31 Then Index = 4
```

1 mark After cipher, output should be Morse equivalent of GCE AS, as below

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: S
Enter your message (uppercase letters and spaces only): BY UWN
-... -.-. .   .- ...
```


Task 15 (max. 19 marks)

1 mark Adding numerals 0-9 as strings to the `Letter` array

1 mark Adding Morse equivalents to the `MorseCode` array

7 marks One for each correct value that is highlighted below in the `Dash` array

6 marks One for each correct value that is highlighted below in the `Dot` array

```
Dim Dash = {20, 23, 0, 0, 24, 1, 0, 17, 31, 21, 28, 25, 0, 15, 11, 27, 0, 0, 0, 22, 13, 29, 30,
10, 0, 0, 0, 27, 0, 29, 0, 0, 0, 0, 0, 0, 0}
```

```
Dim Letter = {" ", "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O",
"P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z", "0", "1", "2", "3", "4", "5", "6", "7",
"8", "9"}
```

```
Dim Dot = {5, 18, 33, 0, 2, 9, 0, 26, 32, 19, 0, 3, 0, 7, 4, 35, 0, 0, 12, 8, 14, 6, 0, 16, 0, 0,
34, 36, 0, 0, 0, 0, 0, 0, 0, 35, 0}
```

```
Dim MorseCode = {" ", ".-", "-...", "-.-.", "-..", ".", "..-.", "--.", "....", "..", ".---",
"-.-", ".-..", "--", "-.", "---", ".--.", "--.-", ".-.", "...", "-", "..-", "...-", ".--",
"-..-", "-.--", "--..", ".----", "..---", "...--", "....-", ".....", "-....", "--...",
"---..", "----."}
```

1 mark Assigning Index values of 27-36 for numerals 0-9, even if unsuccessful (in `SendMorseCode`)

1 mark Successfully assigning all Index values

1 mark Not overwriting those values using the existing ASCII code, i.e. use of an ElseIf or similar

```
If IsNumeric(PlainTextLetter) Then
    Index = Integer.Parse(PlainTextLetter) + 27
ElseIf PlainTextLetter = SPACE Then
    Index = 0
Else
    Index = Asc(PlainTextLetter) - Asc("A") + 1
End If
```

1 mark Correct output for encoding a string of 123

```
Main Menu
=========
R - Receive Morse code
S - Send Morse code
X - Exit program

Enter your choice: S
Enter your message (uppercase letters and spaces only): 123
.____ ..___ ...--
```

