basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATIONS/ NATIONAL SENIOR CERTIFICATE EXAMINATIONS

INFORMATION TECHNOLOGY P1

MAY/JUNE 2025

MARKING GUIDELINES

MARKS: 150

These marking guidelines consist of 27 pages.

GENERAL INFORMATION:

- These marking guidelines are to be used as the basis for the marking session. They were prepared for use by markers. All markers are required to attend a rigorous standardisation meeting to ensure that the guidelines are consistently interpreted and applied in the marking of candidates' work.

- Note that learners who provide an alternate correct solution to that given as example of a solution in the marking guidelines will be given full credit for the relevant solution, unless the specific instructions in the paper was not followed or the requirements of the question was not met

- **Annexures A, B, C** and **D** (pages 3 to 12) include the marking grid for each question.

- **Annexures E, F, G** and **H** (pages 13 to 27) contain examples of solutions for QUESTIONS 1 to 4 in programming code.

- Copies of **Annexures A, B, C, D** and **the summary for the marks of the learner** (pages 3 to 11) should be made for each learner and completed during the marking session.

ANNEXURE A

QUESTION 1: MARKING GRID – GENERAL PROGRAMMING SKILLS

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
1.1	**Button [1.1 - Display shape]** Check if ✓ itemIndex = 0 / item is circle ✓ Change shape property of shpQ1_1 ✓ to stCircle Else ✓ Change shape property of shpQ1_1 to stRectangle ✓ **Also accept:** • Case can be used instead of if statement **Concepts**: • Test using if / case (1) • Using itemIndex / item (1) • Changing the shape property (1) • Else (1) • Correctly assigning both shapes (1)	**5**	
1.2	**Button [1.2 - Display colour]** Correct use of the colours selected from the combo boxes ✓ Check if the colours selected in either of the combo boxes is red and blue: if ((sColor1 = 'Red') ✓ AND (sColor2 = 'Blue')) ✓ OR ✓ ((sColor1 = 'Blue') AND (sColor2 = 'Red')) ✓ then Set the colour of pnlQ1_2 to clPurple ✓ Else ✓ Set the caption of pnlQ1_2 to 'No colour' ✓ **Concepts**: • Extract colours selected (1) • Use of if-statement / case statement to test for the first colour (1) • Test for second colour in combination (1) • Correct use of AND (1), correct use of OR in both combinations (1) • Set colour to clPurple (1) • Else (1) • Set caption to No colour (1)	**8**	

1.3	**Button [1.3 - Average]** Initialise variables (sum and counter) Test if the text file does not exist (Try/FileExists) ✓ Display error message and exit program ✓ AssignFile (tFile, 'DataQ1_3.txt') ✓ Reset file ✓ Loop through the text file ✓ Read value from file into variable ✓ Add value to sum ✓ Increase counter ✓ Calculate the average ✓ and display formatted to two decimal places in memQ1_3 ✓	**10**	

1.4	**Button [1.4 - Display pattern]**
	
	Extract the number of rows selected from spnQ1_4 ✓
	Outer loop from 1 to the number of rows ✓
	Initialise string ✓
	Nested ✓ inner loop from 1 to outer loop ✓
	Add outer loop counter value ✓ to string ✓
	// End of inner loop
	Display output string in redQ1_4 ✓
	// End of outer loop
		8	

1.5	**Button [1.5 - Sum of ASCII values]** Extract word from list box ✓ Initialise sum ✓ Loop ✓ from 1 to length of word ✓ Test if character ✓ is uppercase alphabet character ✓ Add the ASCII value of the character ✓ to sum ✓ Display sum ✓	**9**	
	TOTAL SECTION A:	**40**	

ANNEXURE B

QUESTION 2: MARKING GRID – DATABASE PROGRAMMING

CENTRE NUMBER:	**EXAMINATION NUMBER:**

QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
2.1	**SQL statements**		

2.1.1	**Button [2.1.1 - USA companies]**
	`SELECT * ✓`
	`FROM tblCompanies ✓`
	`WHERE Country = "USA" ✓`
		3	

2.1.2	**Button [2.1.2 - Year released]**
	`SELECT GameTitle, Genre, DateReleased ✓`
	`FROM tblGames ✓`
	`WHERE YEAR ✓ (DateReleased) >= 2019 ✓`
	
	Also accept:
	`Left (DateReleased,4) >= 2019`
	`Mid (DateReleased,1,4) >= 2019`
	`Year (DateReleased) > 2018`
	`DateReleased >= #2019/01/01#`
		4	

2.1.3	**Button [2.1.3 - Genre]**
	`SELECT GameTitle, DateReleased`
	`FROM tblGames ✓`
	`WHERE Genre = ✓"' + sGenre + '" ✓`
	`ORDER BY DateReleased DESC ✓`
	
	Also accept:
	`WHERE Genre = ' + QuotedStr(sGenre) + '`
		4	

2.1.4	**Button [2.1.4 – Average income per year]**
	`SELECT CompanyName,`
	`Format((Sum(Income)*1000000) ✓/`
	`(Year(Date())✓ - YearFounded - 1) ✓,"Currency"✓)`
	`AS AverageIncomePerYear ✓`
	`FROM tblCompanies, tblGames ✓`
	`WHERE tblCompanies.CompanyID =`
	`tblGames.CompanyID ✓`
	`GROUP BY CompanyName, YearFounded ✓`
	
	Also accept:
	`Year(Now)`
		8	

2.1.5	**Button [2.1.5 - Remove game]**
	`DELETE FROM tblGames ✓`
	`WHERE GameTitle = "Apex Legends" ✓`
		2	

	Subtotal:	**21**	

QUESTION 2: MARKING GRID (CONT.)

2.2	**Database Manipulation**		
2.2.1	**Button [2.2.1 - Update genre]** Go to the first record in tblGames ✓ Loop through tblGames ✓ Test if Tetris ✓ is part of the field tblGames['GameTitle'] ✓ tblGames.Edit; ✓ tblGames['Genre'] := 'Puzzle' ✓ tblGames.Post; tblGames.Next; ✓	**7**	
2.2.2	**Button [2.2.2 - Show detail]** Go to the first record in tblGames ✓ Loop through tblGames ✓ Test if tblGames['GameTitle'] = sGame ✓ Go to the first record in tblCompanies ✓ Loop through tblCompanies ✓ Test if (tblGames ['CompanyID'] = ✓ tblCompanies ['CompanyID']) ✓ Use a ShowMessage dialog box ✓ to display: tblGames['GameTitle'] + ' ' + tblGames['Genre'] ✓ + #13 + 'Developed by ' + tblCompanies ['CompanyName'] + ', ' + tblCompanies ['Country'] ✓ tblCompanies.Next ✓ End loop (tblCompanies) tblGames.Next ✓ End loop (tblGames)	**12**	
	Subtotal:	**19**	
	TOTAL SECTION B:	**40**	

ANNEXURE C

QUESTION 3: MARKING GRID – OBJECT-ORIENTED PROGRAMMING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
3.1.1	**Constructor create method:** Heading with four parameter values (String, Boolean, Integer, String) ✓ Set `fName` to String parameter ✓ Set `fOnline` to Boolean parameter ✓ Set `fDownloadCount` to integer parameter ✓ Set `fPlatform` to String parameter ✓	**5**	

3.1.2	**onlineStatus function:**
	
	function heading with String return type ✓
	
	Check if fOnline ✓ = True
	  Result = 'Yes' ✓
	Else
	  Result = 'No' ✓
		4	

3.1.3	**updateDownloadCount procedure:** procedure heading with integer parameter ✓ fDownloadCount = fDownloadCount ✓ + parameter ✓	**3**	

3.1.4	**determinePopularity function:**
	
	function heading with String return type ✓
	
	Check if fOnline ✓ = True
	  if fDownloadCount < 100000 ✓
	  Result := 'Not popular' ✓
	  Else ✓
	  if fDownloadCount < 500000 ✓
	  Result := 'Popular' ✓
	  Else ✓
	  Result := 'Very popular' ✓
	Else
	  Result := 'Not an online game'; ✓
		10	

QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
3.1.5	**toString function** Result := 'Name: ' + fName + #13 + 'Internet required: ' + **onlineStatus** + #13 + 'Number of downloads: ' + IntToStr(fDownloadCount) + #13 + 'Platform: ' + fPlatform; Calling onlineStatus method ✓ at the correct position. ✓	**2**	
	Subtotal: Object class	**24**	

QUESTION 3: MARKING GRID (CONT.)

QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
3.2.1	**Button [3.2.1 - Instantiate object]** objGame := ✓ TGame.create ✓ (sName, bOnline, iNumberOfDownloads, sPlatform) ✓ Display objGame information in redQ3 ✓ using toString method. ✓	**5**	
3.2.2	**Button [3.2.2 - Game compatibility]** Check if (bOnline = objGame.getOnline) ✓ AND (sPlatform = objGame.getPlatform) ✓ ShowMessage('Game is compatible with user requirements.') ✓ Else ShowMessage('Game is NOT compatible with user requirements.'); ✓	**4**	
3.2.3	**Button [3.2.3 – Update downloads]** Extract number entered using a dialog box ✓ and typecast to integer ✓ Call the updateDownloadCount method ✓ with the extracted number as argument ✓ Display objGame information in redQ3 using toString method. ✓	**5**	
3.2.4	**Button [3.2.4 - Popularity]** Call the getName and the determinePopularity methods ✓ Display the name of the game as well as the popularity in redQ3 ✓ in the correct format	**2**	
	Subtotal Form class:	**16**	
	TOTAL SECTION C:	**40**	

ANNEXURE D

QUESTION 4: MARKING GRID – PROBLEM-SOLVING PROGRAMMING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
4.1	**Function isPangram** Function heading with string parameter ✓ with Boolean return type ✓ Loop ✓ or mechanism to check each letter of the alphabet OR build frequency/count structure Mechanism to check presence of all 26 letters: (3) • Alphabet letters are individually tracked or counted ✓ • A correct uppercase / lowercase check is applied to each letter ✓ • To ensure that the letter appears in the string ✓ Checking counter / Boolean ✓ Return TRUE if all alphabet letters are found Return FALSE if any alphabet letter is missing ✓	**8**	
4.2	**Button [4.2 - Execute]** Loop ✓ through sentences ✓ Extract the sentence from lstQ4_2 ✓ If statement – call isPangram ✓ with argument ✓ Display sentence with 'Yes' ✓ separated by #9 ✓ in redQ4_2 Else Display sentence with 'No' separated by #9 in redQ4_2✓	**8**	

4.3	**Button [4.3 - Distinct numbers]**
	
	Concepts:
	
	Create String / temporary array with unique numbers:
	Outer loop ✓ Inner loop ✓
	Determine if number is unique ✓✓✓
	Add unique number to string / temporary array ✓✓
	Display unique number ✓
	
	Determine lowest difference:
	Nested looping ✓
	Calculate difference as a positive number ✓
	Determine if current difference is the lowest ✓
	Store two numbers and the lowest difference ✓✓
	
	Display numbers with lowest difference in labels and the lowest difference correctly formatted ✓
	
	Possible solution:
	
	`Initialise index of unique array to 0`
	`Loop outer from 1 to length or arrNumbers`
	`  Initialise counter to 0`
	`   Nested loop inner from 1 to length or arrNumbers`
	`      Check if arrNumbers[outer] = arrNumbers[inner]`
	`            Increment counter`
	`   //End inner loop`
	
	`//Check if duplicate elements not found`
	`    Test if counter = 1`
	`        Increment index of unique array`
	`        arrUnique[index] = arrNumbers[outer]`
	`         Display unique number/arrUnique[index]`
	`    //End outer loop`
	
	`Initialise lowest difference variable`
	`Loop I from 1 to length of temp array - 1`
	`    Nested loop J from I + 1 to length of temp array`
	`        Current difference =`
	`                    Abs(arrTemp[I] - arrTemp[J])`
	`        Check if (rCurrDiff < rLowestDiff) AND`
	`     (rCurrDiff <> 0)`
	`            rNum1 := arrTemp[I];`
	`            rNum2 := arrTemp[J];`
	`            rLowestDiff:= rCurrDiff;`
	
	`Display numbers with lowest difference and the lowest difference on labels correctly formatted`
		14	

	TOTAL SECTION D: **GRAND TOTAL:**	**30** **150**	

SUMMARY OF LEARNER'S MARKS:

CENTER NUMBER:			**LEARNER'S EXAMINATION NUMBER:**		
	SECTION A	**SECTION B**	**SECTION C**	**SECTION D**	
	QUESTION 1	**QUESTION 2**	**QUESTION 3**	**QUESTION 4**	**GRAND TOTAL**
MAX. MARKS	**40**	**40**	**40**	**30**	**150**
LEARNER'S MARKS					

ANNEXURE E: SOLUTION FOR QUESTION 1

```
unit Question1_U;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms, Dialogs, StdCtrls, ComCtrls, Spin, ExtCtrls, Math;

type
  TfrmQuestion1 = class(TForm)
    grpQ1_1: TGroupBox;
    grpQ1_2: TGroupBox;
    grpQ1_3: TGroupBox;
    grpQ1_4: TGroupBox;
    grpQ1_5: TGroupBox;
    rgpQ1_1: TRadioGroup;
    shpQ1_1: TShape;
    btnQ1_1: TButton;
    Label1: TLabel;
    Label2: TLabel;
    cmbColour1: TComboBox;
    cmbColour2: TComboBox;
    btnQ1_2: TButton;
    pnlQ1_2: TPanel;
    Label3: TLabel;
    spnQ1_3: TSpinEdit;
    btnQ1_3: TButton;
    redQ1_3: TRichEdit;
    btnQ1_4: TButton;
    pnlQ1_4: TPanel;
    btnQ1_5: TButton;
    procedure btnQ1_1Click(Sender: TObject);
    procedure btnQ1_2Click(Sender: TObject);
    procedure btnQ1_3Click(Sender: TObject);
    procedure btnQ1_4Click(Sender: TObject);
    procedure btnQ1_5Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion1: TfrmQuestion1;

implementation

{$R *.dfm}
```

```
//=============================================================================
// Question 1.1                                                        5 marks
// =============================================================================

procedure TfrmQuestion1.btnQ1_1Click(Sender: TObject);
begin
  // Question 1.1

  if rgpQ1_1.ItemIndex = 0 then
    shpQ1_1.Shape := stCircle
  else
    shpQ1_1.Shape := stRectangle;
end;

//=============================================================================
// Question 1.2                                                        8 marks
// =============================================================================

procedure TfrmQuestion1.btnQ1_2Click(Sender: TObject);
var
  sColour1, sColour2: String;
begin
  // Provided code
  pnlQ1_2.Color := clBtnFace;
  pnlQ1_2.Caption := '';

  // Question 1.2
  sColour1 := cmbColour1.Text;
  sColour2 := cmbColour2.Text;

  if ((sColour1 = 'Red') or (sColour2 = 'Red')) AND
    ((sColour1 = 'Blue') or (sColour2 = 'Blue')) then
    pnlQ1_2.Color := clPurple
  else
    pnlQ1_2.Caption := 'No colour';

end;
```

```
//=============================================================================
// Question 1.3                                                        10 marks
// ============================================================================

procedure TfrmQuestion1.btnQ1_3Click(Sender: TObject);
var
  tFile: textFile;
  iNum, iSum, iCnt: Integer;
  rAverage: real;
begin
  AssignFile(tFile, 'DataQ1_3.txt');
  try
    reset(tFile);
  except
    ShowMessage('File not found');
    Application.Terminate;
  end;
  iSum := 0;
  iCnt := 0;
  while not eof(tFile) do
  begin
    readln(tFile, iNum);
    iSum := iSum + iNum;
    Inc(iCnt);
  end;
  rAverage := iSum / iCnt;
  memQ1_3.Lines.Add('Average: ' + FloatToStrF(rAverage, ffFixed, 8, 2));
  CloseFile(tFile);
end;

//=============================================================================
// Question 1.4                                                         8 marks
// ============================================================================

procedure TfrmQuestion1.btnQ1_4Click(Sender: TObject);
var
  iRows, I, J, K: Integer;
  sConcat: String;
begin
  // Provided code
  redQ1_4.Clear;

  // Question 1.4
  iRows := spnQ1_4.Value;

  for I := 1 to iRows do
  begin
    sConcat := '';
    for J := 1 to I do
      sConcat := sConcat + IntToStr(I);
    redQ1_4.Lines.Add(sConcat);
  end;

end;
```

```
//============================================================================
// Question 1.5                                                      9 marks
// ============================================================================

procedure TfrmQuestion1.btnQ1_5Click(Sender: TObject);
var
  sWord: String;
  iCnt, iTot: Integer;

begin
  iTot := 0;
  sWord := lstQ1_4.Items[lstQ1_4.ItemIndex];
  for iCnt := 1 to length(sWord) do
  begin
    if sWord[iCnt] IN ['A' .. 'Z'] then
    begin
      iTot := iTot + Ord(sWord[iCnt])
    end;
  end;
  lblQ1_4.Caption := IntToStr(iTot);
end;

procedure TfrmQuestion1.FormCreate(Sender: TObject);
begin
  shpQ1_1.Visible := False;
end;

end.
```

ANNEXURE F: SOLUTION FOR QUESTION 2

```
//============================================================================
// Question 2.1 - Section: SQL statements
//============================================================================

//============================================================================
//  Question 2.1.1                                                    3 marks
//============================================================================
      sSQL1 := 'SELECT * ' +
               'FROM tblCompanies ' +
               'WHERE Country = "USA"';

//============================================================================
//  Question 2.1.2                                                    4 marks
//============================================================================
      sSQL2 := 'SELECT GameTitle, Genre, DateReleased ' +
               'FROM tblGames ' +
               'WHERE YEAR (DateReleased) >= 2019';

//============================================================================
//  Question 2.1.3                                                    4 marks
//============================================================================
      sSQL3 := 'SELECT GameTitle, DateReleased ' +
               'FROM tblGames ' +
               'WHERE Genre = "' + sGenre + '" ' +
               'ORDER BY DateReleased DESC';

//============================================================================
//  Question 2.1.4                                                    8 marks
//============================================================================
      sSQL4 := 'SELECT CompanyName, format((sum(Income)*1000000)/
                (year(date()) - YearFounded - 1),"Currency") ' +
                'AS IncomePerYear ' +
                'FROM tblCompanies, tblGames ' +
                'WHERE tblCompanies. CompanyID = tblGames. CompanyID ' +
                'GROUP BY CompanyName, YearFounded';

//============================================================================
//  Question 2.1.5                                                    2 marks
//============================================================================
      sSQL5 := 'DELETE FROM tblGames ' +
               'WHERE GameTitle = "Apex Legends"';
```

```
//============================================================================
// Question 2.2 – Section Delphi code
//============================================================================

//============================================================================
//  Question 2.2.1                                                    7 marks
//============================================================================
procedure TfrmQuestion2.btnQ2_2_1Click(Sender: TObject);
begin
  tblGames.First;
  while (NOT tblGames.Eof) do
  begin
    if pos('Tetris', tblGames['GameTitle']) > 0 then
    begin
      tblGames.Edit;
      tblGames['Genre'] := 'Puzzle';
      tblGames.Post;
    end;
    tblGames.Next;
  end;
end;

//============================================================================
//  Question 2.2.2                                                   12 marks
//============================================================================
procedure TfrmQuestion2.btnQ2_2_2Click(Sender: TObject);
var
  sGame,sDetail : String;
begin
  // Provided code
  sGame := cmbQ2_2_2.Text;

  // 2.2.2 - Show detail
  tblGames.First;
   while (NOT tblGames.Eof) do
     begin
       if (sGame = tblGames['GameTitle']) then
         begin
           tblCompanies.First;
           while (NOT tblCompanies.EOF) do
             begin
               if (tblGames['CompanyID'] =
                   tblCompanies['CompanyID']) then
                 begin
                   ShowMessage(tblGames['GameTitle'] + ' ' +
                               tblGames['Genre'] + #13 +
                               'Developed by ' +
                              tblCompanies['CompanyName'] + ' ' +
                              tblCompanies['Country']);
                 end;
               tblCompanies.Next;
             end;
         end;
       tblGames.Next;
     end;
end;
```

```
// ============================================================
// {$ENDREGION}
// ============================================================
// {$REGION 'Provided code: Setup DB connections - DO NOT CHANGE!'}
// ============================================================

procedure TfrmQuestion2.bmbRestoreDBClick(Sender: TObject);
begin
  // Restores the Database
  dbCONN.RestoreDatabase;
  dbCONN.SetupGrids(dbgDevelopers, dbgGames, dbgrdSQL);
end;

procedure TfrmQuestion2.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
  // Disconnects from database and closes all open connections
  dbCONN.dbDisconnect;
end;

procedure TfrmQuestion2.FormShow(Sender: TObject);
begin
  // Sets up the connection to database and opens the tables.
  dbCONN := TConnection.Create;
  dbCONN.dbConnect;
  tblCompanies := dbCONN.tblOne;
  tblGames := dbCONN.tblMany;
  dbCONN.setupGrids(dbgDevelopers, dbgGames, dbgrdSQL);
  pgcDBAdmin.ActivePageIndex := 0;
end;
// ============================================================================
// {$ENDREGION}

end.
```

ANNEXURE G: SOLUTION FOR QUESTION 3

Object class

```
unit Game_U;

interface

Uses SysUtils;

Type
  TGame = class(TObject)
  private
    fName: String;
    fOnline: Boolean;
    fDownloadCount: Integer;
    fPlatform: String;
  public
    constructor create(sName: String; bOnline: Boolean;
                       iDownloadCount: Integer; sPlatform: String);
    function onlineStatus: String;
    procedure updateDownloadCount(iNumberDownloads: Integer);
    function determinePopularity: String;
    // Provided code
    function getName: String;
    function getPlatform : String;
    function getOnline: Boolean;
    function toString : String;
  end;

implementation

{ TGame }

// ============================================================================
// Question 3.1.1                                                    5 marks
// ============================================================================
constructor TGame.create(sName: String; bOnline: Boolean;
                         iDownloadCount: Integer; sPlatform: String);
begin
  // Question 3.1.1
  fName := sName;
  fOnline := bOnline;
  fDownloadCount := iDownloadCount;
  fPlatform := sPlatform;
end;
```

```
// ============================================================================
// Question 3.1.2                                                      4 marks
// ============================================================================
function TGame.onlineStatus: String;
begin
  // Question 3.1.2
  if fOnline then
    Result := 'Yes'
  else Result := 'No';
end;

// ============================================================================
// Question 3.1.3                                                      3 marks
// ============================================================================
procedure TGame.updateDownloadCount(iNumberDownloads: Integer);
begin
  // Question 3.1.3
  fDownloadCount := fDownloadCount + iNumberDownloads;
end;

// ============================================================================
// Question 3.1.4                                                     10 marks
// ============================================================================
function TGame.determinePopularity: String;
begin
  // Question 3.1.4
  if fOnline then
  begin
    If fDownloadCount < 100000 then
      Result := 'Not popular'
    else if fDownloadCount < 500000 then
      Result := 'Popular'
    else
      Result := 'Very popular';
  end
  else
    Result := 'Not an online game';
end;

// ============================================================================
// Question 3.1.5                                                      2 marks
// ============================================================================
function TGame.toString: String;
begin
  // Question 3.1.5
  Result := 'Name: ' + fName + #13 + 'Internet required: ' + onlineStatus +
            #13 + 'Number of downloads: ' + IntToStr(fDownloadCount) +
            #13 + 'Platform: ' + fPlatform;
end;
```

```
// ============================================================================
// Provided code - do not change
// ============================================================================
function TGame.getName: String;
begin
  Result := fName;
end;

function TGame.getOnline: Boolean;
begin
  Result := fOnline;
end;

function TGame.getPlatform: String;
begin
  Result := fPlatform;
end;{$ENDREGION}

// ============================================================================
end.
```

Main Form Unit

```
// ========================================================================
// Question 3.2.1                                                 5 marks
// ========================================================================
procedure TTfrmQuestion3.btnQ3_2_1Click(Sender: TObject);
var
  sName, sPlatform : String;
  bOnline: Boolean;
  iNumberOfDownloads: integer;
begin
  // Provided code
  redQ3.Lines.Clear;

  sName := edtQ3_2_1.Text;
  bOnline := cbxQ3_2_1.Checked;
  iNumberOfDownloads := spnQ3_2_1.Value;
  sPlatform := cmbQ3_2_1.Text;

  // End of provided code

  // 3.2.1 Instantiate object
  objGame := TGame.create(sName, bOnline,
                          iNumberOfDownloads, sPlatform);
  redQ3.Lines.Add(objGame.toString);
end;

// ========================================================================
// Question 3.2.2                                                 4 marks
// ========================================================================
procedure TTfrmQuestion3.btnQ3_2_2Click(Sender: TObject);
var
  bOnline : Boolean;
  sPlatform : String;
begin
  // Provided code
  redQ3.Lines.Clear;

  bOnline := cbxQ3_2_2.Checked;
  sPlatform := cmbQ3_2_2.Text;

  // 3.2.2 - Game compatibility
  if (bOnline = objGame.getOnline) AND (sPlatform = objGame.getPlatform)
then
    ShowMessage('Game is compatible with user requirements.')
  else
    ShowMessage('Game is NOT compatible with user requirements.');
end;
```

```
// ============================================================================
// Question 3.2.3                                                      5 marks
// ============================================================================
procedure TTfrmQuestion3.btnQ3_2_3Click(Sender: TObject);
var
  iNumberOfDownloads: integer;
begin
  // Provided code
  redQ3.Lines.Clear;

  // 3.2.3  - Update number of downloads
  iNumberOfDownloads := StrToInt(InputBox('Number of downloads', 'Please
enter number of downloads', ''));
  objGame.updateDownloadCount(iNumberOfDownloads);
  redQ3.Lines.Add(objGame.toString);
end;

// ============================================================================
// Question 3.2.4                                                      2 marks
// ============================================================================
procedure TTfrmQuestion3.btnQ3_2_4Click(Sender: TObject);
begin
  // Provided code
  redQ3.Lines.Clear;

  // 3.2.4 -  Determine popularity of a game
  redQ3.Lines.Add(objGame.getName + ': ' + objGame.determinePopularity);
end;

end.
```

ANNEXURE H: SOLUTION FOR QUESTION 4

```
unit Question4_U;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, ComCtrls, Grids, DBGrids;

type
  TfrmQuestion4 = class(TForm)
    Panel1: TPanel;
    GroupBox1: TGroupBox;
    btnQ4_2: TButton;
    redQ4_2: TRichEdit;
    lstQ4_2: TListBox;
    grpQ4_3: TGroupBox;
    redQ4_3: TRichEdit;
    btnQ4_3: TButton;
    lblNum1 : Tlabel;
    lblNum2 : Tlabel;
    lblDifference : TLabel;
    procedure btnQ4_2Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure btnQ4_3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    function isPangram(sSentence: String): boolean;
  end;

var
  frmQuestion4: TfrmQuestion4;
  arrNumbers: array [1..20] of real;

implementation

{$R *.dfm}

// ============================================================================
// Question 4.1                                                      8 marks
// ============================================================================

function TfrmQuestion4.isPangram(sSentence: String): boolean;
var
  iCount: integer;
  cChar: char;
begin
  iCount := 0;
  for cChar := 'A' to 'Z' do
  begin
    if pos(cChar, UpperCase(sSentence)) > 0 then
      Inc(iCount);
  end;
  Result :=  iCount = 26;
end;
```

```
// ============================================================================
// Question 4.2                                                       8 marks
// ============================================================================

procedure TfrmQuestion4.btnQ4_2Click(Sender: TObject);
var
  sSentence, sTemp, sWord, sOut: String;
  I : integer;
  bFlag : boolean;
begin
  // Provided code
  redQ4_2.Lines.Add('Original sentence' + #9 + 'Pangram' + #13);

  // Question 4.2
  I := 0;
  lstQ4_2.Items.Add('');

  repeat
      sSentence := lstQ4_2.Items[I];
      inc(I);
      if sSentence <> '' then
      begin
        if isPangram(sSentence) then
          sOut := sSentence + #9 + 'Yes'
        else if (isPangram(sSentence) = False) then
          sOut := sSentence + #9 + 'No';
        redQ4_2.Lines.Add(sOut);
      end;
  until (sSentence = '');
end;

// ============================================================================
// Question 4.3                                                      14 marks
// ============================================================================

procedure TfrmQuestion4.btnQ4_3Click(Sender: TObject);
var
  iLoop1, iLoop2: integer;
  rNum, rLowest, rDiff, rNum1, rNum2: real;
  iCount, iUnique: integer;
  arrUnique: array [1..20] of real;
begin
  iCount := 0;
  for iLoop1 := 1 to 20 do
    begin
      iUnique := 0;
      for iLoop2 := 1 to 20 do
        if arrNumbers[iLoop1] = arrNumbers[iLoop2] then
          inc(iUnique);

      if iUnique = 1 then
        begin
          inc(iCount);
          arrUnique [iCount] := arrNumbers[iLoop1];
          redQ4_3.Lines.Add(FloatToStr(arrUnique [iCount]));
        end;
    end;
```

```
  rLowest := ABS(arrUnique[1] - arrUnique[2]);

  for iLoop1 := 1 to iCount - 1 do
  begin
    for iLoop2 := iLoop1+1 to iCount do
    begin
      rDiff := ABS(arrUnique[iLoop1] - arrUnique[iLoop2]);
      if rDiff < rLowest then
      begin
        rLowest := rDiff;
        rNum1 := arrUnique[iLoop1];
        rNum2 := arrUnique[iLoop2];
      end;
    end;
  end;
  lblNum1.Caption := FloatToStr(rNum1);
  lblNum2.Caption := FloatToStr(rNum2);
  lblDifference.Caption := FloatToStrF(rLowest, ffFixed,8,2);
end;

// ============================================================================
// Provided code
// ============================================================================
procedure TfrmQuestion4.FormCreate(Sender: TObject);
begin
  arrNumbers[1] := 10.39;
  arrNumbers[2] := 5.33;
  arrNumbers[3] := 5.48;
  arrNumbers[4] := 9.53;
  arrNumbers[5] := 5.82;
  arrNumbers[6] := 4.21;
  arrNumbers[7] := 5.33;
  arrNumbers[8] := 2.26;
  arrNumbers[9] := 4.21;
  arrNumbers[10] := 8.48;
  arrNumbers[11] := 4.82;
  arrNumbers[12] := 9.53;
  arrNumbers[13] := 2.17;
  arrNumbers[14] := 5.33;
  arrNumbers[15] := 9.53;
  arrNumbers[16] := 8.46;
  arrNumbers[17] := 9.53;
  arrNumbers[18] := 10.26;
  arrNumbers[19] := 5.33;
  arrNumbers[20] := 9.21;
  redQ4_2.Paragraph.TabCount := 2;
  redQ4_2.Paragraph.Tab[0] := 290;
  redQ4_2.Paragraph.Tab[1] := 390;
end;

// ============================================================================
// End of provided code

end.
```