+

basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE/ NATIONAL SENIOR CERTIFICATE

GRADE 1

INFORMATION TECHNOLOGY P1

NOVEMBER 2020

MARKING GUIDELINES

MARKS: 150

These marking guidelines consist of 28 pages.

Copyright reserved

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 3-10) include the marking grid for each question for using either one of the two programming languages.

¶ **Annexures E, F, G and H** (pages 12-28) contain examples of solutions for Questions 1 to 4 in programming code.

¶ Copies of **Annexures A, B, C and D** (pages 3-11) 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 ï Colours]** Change the font size of cmbQ1_1 to 12pt V Add blue to cmbQ1_1 V V Set item index to 0 / set text to Green V	**4**	
1.2	**Button [1.2 ï Kite]** Create variable for area (rArea) V Test if iDiagA > iDiagB V rArea = iDiagA * iDiagB V / 2 V Display rArea on label V formatted to one decimal V Else V Display error message V	**8**	
1.3	**Button [1.3 ï Binary number]** Extract decimal number from edit box V, convert to integer V Initialise output string for binary value V Loop while Vdecimal number > 0 V Remainder = decimal number MOD 2 V (result 0 or 1) Decimal number V = decimal number DIV 2 V Add the remainder converted to a string V to beginning of output string V Display binary number on label lblQ1_3 V Alternatives: Loop while decimal number div 2 > 0 Repeat until decimal number = 0	**11**	
1.4	**Button [1.4 ï Word game]** Initialise total to 0 V Extract word from edit box V and change to uppercase V Test if V it is not a single word (has spaces) V Clear edit box V Set focus to the edit box V Show error message V Else V Loop through word using length of word V Extract character at loop variable V Test if it is a vowel V add 3 to total V Else Test for other alphabetical character V Add 2 to total V Else Add 1 to total V Display total value of word in richedit V	**17**	
	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 ï List of employees]** `SELECT * FROM tblEmployees ORDER BY JobTitle, HourlyWage DESC` **Concepts:** SELECT * V FROM tblEmployees V ORDER BY JobTitle V ,HourlyWage DESC V	**4**	
2.1.2	**Button [2.1.2 ï Engineers]** `SELECT EmployeeID, LastName, FirstName FROM tblEmployees WHERE JobTitle LIKE "%Engineer%"` **Concepts:** SELECT EmployeeID, LastName, FirstName V FROM tblEmployees V WHERE JobTitle LIKE V "%Engineer%"' V	**4**	

2.1.3	**Button [2.1.3 ï Job titles]**
	`SELECT DISTINCT JobTitle FROM tblEmployees`
	`OR`
	`SELECT JobTitle FROM tblEmployees GROUP BY JobTitle`
	Concepts:
	SELECT DISTINCT V JobTitle V FROM tblEmployees V
	OR
	SELECT JobTitle FROM tblEmployees
	GROUP BY JobTitle
		3	

2.1.4	**Button [2.1.4 ï Remove records]** `DELETE FROM tblHourLogs WHERE HoursWorked = 99` **Concepts:** DELETE FROM tblHourLogs V WHERE HoursWorked = 99 V	**2**	

QUESTION 2: MARKING GRID ï CONTINUE

2.1.5	**Button [2.1.5 ï Overtime]**	**9**	
	`SELECT LastName, FORMAT(SUM((HoursWorked - 8) * HourlyWage * 2), "CURRENCY") AS OvertimeAmt FROM tblEmployees E, tblHourLogs H WHERE E.EmployeeID = H.EmployeeID AND HoursWorked > 8 GROUP BY LastName`		
	SELECT LastName, V FORMAT(SUM V((HoursWorked - 8) V * HourlyWage * 2 V), "CURRENCY") V AS OvertimeAmt V FROM tblEmployees E, tblHourLogs H V WHERE E.EmployeeID = H.EmployeeID V AND HoursWorked > 8 GROUP BY LastName V **Concepts:** Retrieve LastName (1) Format to overtime pay as currency (1) Use SUM function correctly (1) Calculate number of hours worked overtime (1) Multiply result of above step by HourlyWage * 2 (1) Name calculated field OverTimeAmt (1) Extract data from tblEmployees and tblHourLogs (1) Test if there is a relationship between PK and FK (1) Group results by LastName (1)		
	Subtotal:	**22**	

2.2	**DATABASE MANIPULATION using Delphi code**		
2.2.1	**Button [2.2.1 ï Employees with first aid]** Display column headings // Provided Initialise counter to 0 Set tblEmployees to start reading first record V Loop while not tblEmployees.Eof V Test if tblEmployees['FirstAidTraining'] = True V Display EmployeeID, LastName and JobTitle V Increment counter by 1 V Go to next record in tblEmployees V Display counter V converted to string	**7**	

2.2.2	**Button [2.2.2 ï Add new employee]**
	
	tblEmployees.Insert V
	tblEmployees['EmployeeID'] := 'EMP986'; V
	tblEmployees['FirstName'] := 'Robert'
	tblEmployees['LastName'] := 'Laubscher' V
	tblEmployees['HourlyWage'] := 195.00 V
	tblEmployees['JobTitle'] := 'Marine Engineer'
	tblEmployees['FirstAidTraining'] := True V
	tblEmployees.Post V
	
	Alternatives
	`tblEmployees['HourlyWage'] := 'R195.00';`
	`tblEmployees['HourlyWage'] := FloatToStrF(195.00,ffCurrency,6,2);`
	
	Concepts:
	Insert /Append (1)
	String variables ï ID (1),
	FirstName and LastName (1)
	JobTitle and HourlyWage (1)
	FirstAidTraining (1)
	Post (1)
		6	

2.2.3	**Button [2.2.3 ï Update hours worked]** Get number of hours (iHours or sHours) from edtQ2_2_3 V, typecast to Integer tblHourLogs.Edit V tblHourLogs['HoursWorked'] V := iHours V tblHourLogs.Post V	**5**	
	Subtotal:	**18**	
	TOTAL SECTION B:	**40**	

ANNEXURE C

QUESTION 3: MARKING GRID - OBJECT-ORIENTED PROGRAMMING

QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
3.1.1	**Constructor method:** Header with correct parameter values (correct order, data types) V Assign customer ID parameter value to fCustomerID V Assign container size parameter value to fContainerSize V Assign storage period parameter value to fStoragePeriod Set fAmountPaid to 0 V	**4**	
3.1.2	**getAmountPaid method:** Function heading with real/double as return data type V fAmountPaid assigned to result V	**2**	
3.1.3	**updateAmountPaid method:** Procedure heading with real/double parameter value V Increment fAmountPaid V Using the parameter value V	**3**	
3.1.4	**calculateCost method:** Function declared with double/real return data type If containerSize = 'S' Set initial cost to 1000.00 Else if containerSize = 'M' Set initial cost to 1750.00 Else Set initial cost to 2500.00 Cost = storagePeriod * initial cost Percentage discount = 10% for each increment of 6 months using the floor/trunk/div function or other applicable code Test if discount > 50 Set discount = 50 Calculate discount value (Percentage x Amount) Return cost as result	**11**	

	Adapted marking approach *Allocate marks for the following concepts:* Function declared V with a return data type V *A test for the container size* V V *Assigning the correct cost* V *per size* V *Calculate cost* V using the correct formula *Determine percentage discount using months V* *Limit discount percentage to 50* V *Calculate discount value V* *Return cost as result V*		
3.1.5	**toString method:** Labels (Customer ID, Container size, Storage period, Amount paid) V Correct attributes V Correct conversions (amountPaid ï float; storagePeriod ï integer) V	**3**	
	Subtotal: Object class	**23**	

QUESTION 3: MARKING GRID (CONT.)

QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
3.2.1	**Button [3.2.1 ï Instantiate object]** Extract customer ID from edit box V Extract first character V of selected item in list box V Extract number of months from spin edit V *Instantiate the objTransaction object:* objTransaction:= V TTransaction.Create V (customerID,size,months) Disable **btnQ3_2_1** V	**7**	
3.2.2 (a)	**Button [3.2.2 (a) ï Display amount due]** Call the calculateCost method V Display the amount due V on **pnlQ3_2_2** V	**3**	
3.2.2 (b)	**Button [3.2.2 (b) ï Process payment]** Retrieve the amount from **edtQ3_2_2** V Call updateAmountPaid V with extracted amount as argument Calculate amount due using the object methods: Amount due := calculateCost - getAmountPaid Concepts: Correct method call V Correct calculation V Display amount due on **pnlQ3_2_2** V	**5**	
3.2.3	**Button [3.2.3 ï View details]** Call the toString method V Display in the richedit V	**2**	
	Subtotal: Form class	**17**	
	TOTAL SECTION C:	**40**	

ANNEXURE D

QUESTION 4: MARKING GRIDï PROBLEM SOLVING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
SECTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
4.1.1	**Button [4.1.1 ï Create harbour containers]** Loop 50 times V Generate real numbers Vbetween 1 and 99 (inclusive) V Rounded to 1 decimal Store the values in the array arrContainers V	**4**	
4.1.2	**Button [4.1.2 ï Display harbour containers]** Set index to 0 V Loop 5 times (rows) V sLine := ' ' V Loop 10 times (columns) V Increment the index with 1 V sLine := sLine + FloatToStr(arrContainers[iIndex]) V + #9; V Display sLine in the rich edit redQ4_1_2 V **Alternative**: sLine := sLine + FloatToStr(arrContainers[c+(10*(r-1)]) (3) + #9; (1) Concept: Using loop variables (1) Calculation (1) Floattostr (1) Space (1)	**8**	

4.2	**Button [4.2 ï Containers loaded to be shipped]** Set TotalTons to zero (0) V Initialise string variable used for display V Set array index (1) V Loop index < 50 V If (totalTons + weight of the containers at Index) V <= 200 V then Add weight of the containers at Index V in array to TotalTons V Add the weight of the containers at Index V in array and add #9 V to sLine V Increment the index V Outside loop: Display the weight of the containers loaded onto the ship in the richedit V Display the total weight of the containers loaded on the panel V Assign internal file name to external file name V Rewrite V Write the weight of the containers loaded onto the ship to the text file V Close the text file V Alternative For loop from 1 to 50 (3 marks)	**18**	

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

SUMMARY OF LEARNER'S MARKS:

CENTER NUMBER:		**LEARNER'S EXAM 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, ExtCtrls, Spin, pngimage, ComCtrls, Math,
Buttons;

TfrmQuestion1 = class(TForm)
    GroupBoxQ1_1: TGroupBox;
    GroupBoxQ1_4: TGroupBox;
    edtQ1_4: TEdit;
    Label9: TLabel;
    redQ1_4: TRichEdit;
    GroupBoxQ1_3: TGroupBox;
    edtQ1_3: TEdit;
    btnQ1_3: TButton;
    Label10: TLabel;
    lblQ1_3: TLabel;
    btnQ1_1: TButton;
    cmbQ1_1: TComboBox;
    btnClose: TBitBtn;
    GroupBoxQ1_2: TGroupBox;
    lblQ1_2: TLabel;
    btnQ1_2: TButton;
    btnQ1_4: TButton;
    Image2: TImage;
    procedure btnQ1_2Click(Sender: TObject);
    procedure btnQ1_4Click(Sender: TObject);
    procedure btnQ1_3Click(Sender: TObject);
    procedure btnQ1_1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmQuestion1: TfrmQuestion1;

implementation

{$R *.dfm}

//=====================================================================
//                      Question 1.1 – 4 marks
//=====================================================================
procedure TfrmQuestion1.btnQ1_1Click(Sender: TObject);
begin
// Question 1.1
  cmbQ1_1.Font.Size := 12;
  cmbQ1_1.Items.Add('Blue');
  cmbQ1_1.ItemIndex := 0;
end;
```

```
//=====================================================================
//                      Question 1.2 – 8 marks
//=====================================================================
procedure TfrmQuestion1.btnQ1_2Click(Sender: TObject);
var
  iDiagA, iDiagB: integer;
  rArea: real;
begin
//Provided code

  iDiagA := StrToInt(InputBox('Diagonal one of kite(cm)', 'Enter the
value of diagonal one (A): ', '20'));
  iDiagB := StrToInt(InputBox('Diagonal two of kite (cm)',
      'Enter the value of diagonal two (B):', ''));

// Question 1.2
  if iDiagA > iDiagB then
  begin
    rArea := (iDiagA * iDiagB) / 2;
    lblQ1_2.Caption := 'The area of the kite is ' + FloatToStrF
      (rArea, ffFixed, 10, 1) + ' square cm.';
  end
  else
  begin
    ShowMessage('The value of diagonal two (B) must be less than the
                     value of diagonal one (A)');
  end;
end;

//=====================================================================
//                      Question 1.3 – 11 marks
//=====================================================================

procedure TfrmQuestion1.btnQ1_3Click(Sender: TObject);
// Provided code
var
  iNumber, iRemainder: integer;
  sBinary: String;
begin
// Question 1.3
  iNumber := StrToInt(edtQ1_3.Text);
  sBinary := '';
  while iNumber <> 0 do
  begin
    iRemainder := iNumber MOD 2;
    iNumber := iNumber DIV 2;
    sBinary := IntToStr(iRemainder) + sBinary;
  end;
  lblQ1_3.Caption := 'Binary number: ' + sBinary;
end;
```

```
//=====================================================================
//                      Question 1.4 – 17 marks
//=====================================================================
procedure TfrmQuestion1.btnQ1_4Click(Sender: TObject);
var
  sWord: String;
  iTotal, iValue, iRand, iLen, iCount: integer;
  cChar: char;
begin
// Provided code
  redQ1_4.Clear;
// Question 1.4
  iTotal := 0;
  sWord := Uppercase(edtQ1_4.Text);

  if pos(' ', sWord) = 0 then
  begin
    iLen := Length(sWord);

    for iCount := 1 to iLen do
    begin
      cChar := sWord[iCount];

      if cChar in ['A', 'E', 'I', 'O', 'U'] then
        iTotal := iTotal + 3
      else
         if cChar in ['A'..'Z'] then
           iTotal := iTotal + 2
        else
           iTotal := iTotal + 1;
    end;

    redQ1_4.lines.Add('Total number of points: ' + IntToStr(iTotal));
  end
  else
  begin
    edtQ1_4.Clear;
    edtQ1_4.SetFocus;
    ShowMessage('Disqualified - more than one word was entered.');
  end;
end;

end.
```

ANNEXURE F: SOLUTION FOR QUESTION 2

```
unit Question2_U;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms, Dialogs, StdCtrls, Buttons, ExtCtrls, ConnectDB_U, DB, ADODB,
Grids, DBGrids, ComCtrls, DateUtils, DBCtrls;

type
  TfrmQuestion2 = class(TForm)
    pnlBtns: TPanel;
    btnRestoreDB: TBitBtn;
    grpTblHourLogs: TGroupBox;
    grpTblEmployees: TGroupBox;
    dbgEmployees: TDBGrid;
    dbgHourLogs: TDBGrid;
    tabsQ2_2ADO: TTabSheet;
    tabsQ2_1SQL: TTabSheet;
    btnQ2_2_1: TButton;
    redQ2: TRichEdit;
    grpresults: TGroupBox;
    dbgSQL: TDBGrid;
    grpOutput: TGroupBox;
    pgcTabs: TPageControl;
    pnlTables: TPanel;
    btnQ2_1_1: TButton;
    btnQ2_1_2: TButton;
    btnQ2_1_3: TButton;
    btnQ2_1_4: TButton;
    btnQ2_1_5: TButton;
    btnQ2_2_3: TButton;
    btnQ2_2_2: TButton;
    procedure btnRestoreDBClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure btnQ2_1_1Click(Sender: TObject);
    procedure btnQ2_1_2Click(Sender: TObject);
    procedure btnQ2_1_3Click(Sender: TObject);
    procedure btnQ2_1_4Click(Sender: TObject);
    procedure btnQ2_1_5Click(Sender: TObject);
    procedure btnQ2_2_1Click(Sender: TObject);
    procedure btnQ2_2_3Click(Sender: TObject);
    procedure btnQ2_2_2Click(Sender: TObject);

  private
  public
  end;

var
  frmQuestion2: TfrmQuestion2;
  dbCONN: TConnection;
```

```
// --- Global variables provided ---
  tblEmployees, tblHourLogs: TADOTable;
  qryDB: TADOQuery;
implementation
{$R *.dfm}
{$R+}
```

Question 2.1 - SQL section

```
//=====================================================================
//                     Question 2.1.1 – 4 marks
//=====================================================================
procedure TfrmQuestion2.btnQ2_1_1Click(Sender: TObject);
  var
    sSQL1: String;
begin
  sSQL1 := 'SELECT * FROM tblEmployees ' +
           'ORDER BY JobTitle, HourlyWage DESC';

  // Provided code - do not change
  dbCONN.DBExtract(sSQL1);
end;

//=====================================================================
//                     Question 2.1.2 – 4 marks
//=====================================================================
procedure TfrmQuestion2.btnQ2_1_2Click(Sender: TObject);
  var
    sSQL2: String;
begin
  sSQL2 := 'SELECT EmployeeID, LastName, FirstName FROM tblEmployees ' +
           'WHERE JobTitle LIKE "%Engineer%"';

  // Provided code - do not change
  dbCONN.DBExtract(sSQL2);
end;

//=====================================================================
//                     Question 2.1.3 – 3 marks
//=====================================================================
procedure TfrmQuestion2.btnQ2_1_3Click(Sender: TObject);
  var
    sSQL3: String;
begin
  sSQL3 := 'SELECT DISTINCT JobTitle FROM tblEmployees';

  // Provided code - do not change
  dbCONN.DBExtract(sSQL3);
end;
```

```
//=====================================================================
//                    Question 2.1.4 – 2 marks
//=====================================================================
procedure TfrmQuestion2.btnQ2_1_4Click(Sender: TObject);
  var
    sSQL4: String;
    bChange: boolean;
begin
  sSQL4 := 'DELETE FROM tblHourLogs ' +
           'WHERE HoursWorked = 99';

  // Provided code - do not change
  dbCONN.ExecuteSQL(sSQL4, dbgHourLogs);
  if bChange then
    begin
      MessageDlg('Database updated', mtInformation, [mbOK], 0);
    end;
end;

//=====================================================================
//                    Question 2.1.5 – 9 marks
//=====================================================================
procedure TfrmQuestion2.btnQ2_1_5Click(Sender: TObject);
  var
    sSQL5: String;
begin
  sSQL5 := 'SELECT LastName, FORMAT(SUM((HoursWorked - 8) * (HourlyWage
* 2)), "CURRENCY") ' +
           'AS OvertimeAmt ' +
           'FROM tblEmployees E, tblHourLogs H ' +
           'WHERE E.EmployeeID = H.EmployeeID ' +
           'GROUP BY LastName';

  // Provided code - do not change
  dbCONN.DBExtract(sSQL5);
end;
```