basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

NATIONAL SENIOR CERTIFICATE

GRADE 12

INFORMATION TECHNOLOGY P1

FEBRUARY/MARCH 2015

MEMORANDUM

MARKS: 150

This memorandum consists of 32 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 of whom are required to attend a rigorous standardisation meeting to ensure that the guidelines are consistently interpreted and applied in the marking of candidates' scripts.

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

- **Annexures A, B** and **C** (pages 3–7) include the marking grid for each question for using either one of the two programming languages.

- **Annexures D, E** and **F** (pages 8–17) contain examples of solutions for Java for Questions 1 to 3 in programming code.

- **Annexures G, H** and **I** (pages 18–31) contain examples of solutions for Delphi for Questions 1 to 3 in programming code.

- Copies of **Annexures A, B** and **C** (pages 3–7) should be made for each candidate and completed during the marking session.

ANNEXURE A:

SECTION A:

QUESTION 1: MARKING GRID: GENERAL PROGRAMMING SKILLS

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
1.1	**[Button – Compile advertisement]** Extract the market price ✓ Extract the selling price ✓ Extract the number of bedrooms✓ Extract the number of bathrooms✓ Use the extracted information to compile the advertisement string ✓ Check if pool is selected ✓ Join Pool# to advertisement ✓ Check if the selling price < market price ✓ Join Bargain to advertisement ✓ Display heading in the output area ✓ Display advertisement in the output area✓	**11**	
1.2.1	**Radio Button Paint:** Save selected renovation type P✓	**1**	
1.2.2	**Radio Button Tile:** Save selected renovation type as T ✓	**1**	
1.2.3 (a)	**[Button – Calculate and display renovation cost]** Initialize all counter variables to 0 ✓ Create and initialize cost variable ✓ Extract the area from text box and convert to decimal value✓ Check if type of renovation is paint ✓ Calculate the number of litres of paint Formula: numLitres = (area / 8) ✓ Round up numLitres ✓ Calculate the number of litres required If number litres required is more than 5✓ Calculate number of 5 litre drums✓ Calculate cost of 5 litre drums✓ Calculate remaining paint required✓ Repeat for: 2 litre drums✓ Assign remaining paint to one litre drums✓ Calculate cost✓ Display area and volume of litres of paint required✓ Display each drum size ✓ and number of drums required✓ Display cost to two decimal places✓	**17**	

(b)	Check if type of renovation is tile ✓ Use dialog box to enter the cost per square metre ✓ Calculate the cost of tiling using the formula below: CostOfTiling = (area + 5) ✓* costPerSquareMetre ✓ Display heading ✓ Display cost of tiling ✓	**6**	
1.3.1	**[Button – Calculate amount due]** Extract previous reading from text box as integer values✓ Extract current reading from text box as integer values ✓ If current reading < previous reading✓ Set current reading to 0✓ and display message ✓ else elecUsage = (currReading - prevReading) ✓ if elecUsage <= 600✓ elecBill = elecUsage✓ else elecBill = 600 ✓+ (elecUsage - 600) * 1.5✓ Display electricity bill in the label provided Format: currency and two decimal places ✓	**11**	
1.3.2	**[Button – List Geysers]** Extract the geyser size from the text box✓ Use loop to step through array ✓ If array entry contains geyser size entered ✓ Start if block of code ✓ Extract name of the geyser from array entry✓ Display the name of the geyser in the text area✓	**6**	
	TOTAL:	**23**	

ANNEXURE B:

SECTION B:

QUESTION 2: MARKING GRID – OBJECT-ORIENTATED PROGRAMMING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
2.1.1	Five attributes declared✓ as private ✓ String data type: applicant name Integer data type: years } ✓ Real data type: disposable income, loan amount, interest rate ✓	**4**	
2.1.2	**Constructor:** Heading ✓ with three parameters ✓ Parameters correct data types ✓ Assign parameter values to relevant three attributes ✓ Default values assigned to years and interest rate ✓	**5**	
2.1.3	**Accessor method for name of applicant:** Method heading ✓ return statement ✓	**2**	
2.1.4	**Mutator methods for years and interest rate:** setYears: parameter received ✓ and assigned to correct attribute ✓ setInterestRate: parameter received ✓ and assigned to correct attribute ✓	**4**	
2.1.5	**calculateInstalmentAmount** method: Uncomment code ✓	**1**	
2.1.6	**isApproved** method: Create boolean variable for approved set to true ✓ If loanAmount > 800000 and ✓ years < 25 ✓ change approved to false ✓ If loanAmount <= 600000 and ✓ disposableIncome < ✓ 130% of instalment ✓ change approved to false ✓ Return approved ✓	**9**	
2.1.7	**toString** method: Layout of string variable: Each part of string on a new line✓, correct labels ✓ Correct information: Applicant name, disposable income ✓, loan amount, years✓, interest rate with %-sign ✓	**5**	

2.2.1	**[Button – Find applicant]** Extract the applicant name from the combo box ✓ If a name was not selected ✓ display suitable message ✓ If a name is selected✓ locate the text file that has the same name as the applicant ✓ If the file exists ✓ open text file ✓ and read the following from text file income, expenditure ✓ loan amount✓ Calculate the disposable amount✓ Instantiate a new loan object ✓ using the correct parameters ✓✓ Display message using a dialog box✓ Enable the "Evaluate application" button✓ If the file does not exist, ✓ Display a suitable message✓ Make use of a correct validation technique for reading from a text file ✓	**18**	
2.2.2	**[Button – Evaluate application]** Extract the interest rate from the text box ✓ convert to real number ✓ Read years from combo box✓ and change to integer✓ Set the object's interest rate using the mutator method✓ Set the object's years using the mutator method✓ Check if the loan has been approved by calling the relevant method ✓ If approved ✓ display the home loan object information using toString method ✓ display label with instalment ✓ else ✓ display the name of the applicant using the accessor method and display the message 'NOT APPROVED' ✓	**12**	
	TOTAL:	**60**	

ANNEXURE C:

SECTION C:

QUESTION 3: MARKING GRID – PROBLEM-SOLVING PROGRAMMING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
3.1	**[Button – Search for agent's name]** Create required variables ✓ Use loop to step through the array named arrAgents ✓ Test if code in the array entry✓contains code entered ✓ Extract agent's name from the array entry ✓ Display the name in the label provided ✓ Terminate the search process ✓ Increment counter ✓ If the agent code is not found ✓ Display the message 'Invalid agent code' in the label ✓ Clear the text box for agent code ✓	**11**	
3.2.1	Add a suitable component like a table/grid/output area to the GUI ✓ Rename the component ✓	**2**	
3.2.2	**[Button – Agent's sales]** Create a 2d array or any structure(s) for processing✓✓ Initialise the 2d array / other structures ✓✓ Create a total sales variable ✓ Use loop to step through the array named arrSales✓ if agent code = code from array entry ✓ Extract month from the array entry✓ Set the column value using month ✓ Extract the property type from the array entry ✓ Set the row value using property type ✓✓✓ Increase the value of 2dArr[row][col] by 1 ✓ Increase the value of 2dArr[row][4] by 1 for total row ✓ Increase the total sales by the price in the array entry ✓✓ Display the content of the 2d array (or other structure(s)) in a suitable component, formatted and aligned. Display the following information: Months as headings ✓ Property type labels ✓ Row and column values✓✓✓ Row Totals ✓ Display total value of sales in a suitable component✓	**24**	
	TOTAL:	**37**	

SUMMARY OF LEARNER'S MARKS:

	SECTION A	**SECTION B**	**SECTION C**	
	QUESTION 1	**QUESTION 2**	**QUESTION 3**	**GRAND TOTAL**
MAX. MARKS	**53**	**60**	**37**	**150**
LEARNER'S MARKS				

ANNEXURE D: SOLUTION FOR QUESTION 1: JAVA

```
//    A possible solution to Question 1

//=========================================================================
//This code is given in the program
//=========================================================================


    public Question1() {
        initComponents();
        this.setLocationRelativeTo(this);
    }

//=========================================================================
// New code
//=========================================================================

DecimalFormat df = new DecimalFormat("R 0.00");
char renovationType = ' ';

//=========================================================================
//   Question 1.1
//=========================================================================
  private void btnSalesAddActionPerformed(java.awt.event.ActionEvent evt) {
        double market = Double.parseDouble(txfMarket.getText());
        double sell = Double.parseDouble(txfSellingPrice.getText());
        String bed = txfBeds.getText();
        String bath = txfBaths.getText();
        String advert = df.format(sell)+"#"+bed + "Bed#" + bath + "Bath#";
        if (chkPool.isSelected()) {
            advert += "Pool#";
        }
        if (sell < market) {
            advert += "Bargain";
        }
        txaOutput.setText("House for sale:\n");
        txaOutput.append(advert);
    }
//=========================================================================
//   Question 1.2.1
//=========================================================================

private void rbtPaintActionPerformed(java.awt.event.ActionEvent evt) {
        renovationType = 'P';
    }

//=========================================================================
//   Question 1.2.2
//=========================================================================
private void rbtTileActionPerformed(java.awt.event.ActionEvent evt) {
        renovationType = 'T';}
```

```
//=====================================================================
//   Question 1.2.3 (a)
//=====================================================================
  private void btnCalcRenovationActionPerformed
               (java.awt.event.ActionEvent evt) {

     int numOneLitreDrum = 0;
     int numTwoLitreDrum = 0;
     int numFiveLitreDrum = 0;
     double cost = 0.0;
     double area = Double.parseDouble(txfArea.getText());
     if (renovationType == 'P') {
        double numLitresOfPaint = area / 8;
        int volumeLitres = (int) (Math.round(numLitresOfPaint + 0.4));
        if (volumeLitres / 5 > 0) {
             numFiveLitreDrum += (int) (volumeLitres / 5);
             volumeLitres -= numFiveLitreDrum * 5;
             cost += numFiveLitreDrum * 199;
        }
        if (volumeLitres / 2 > 0) {
             numTwoLitreDrum += (int) (volumeLitres / 2);
             volumeLitres -= numTwoLitreDrum * 2;
             cost += numTwoLitreDrum * 92.30;
        }
        numOneLitreDrum = volumeLitres;
        cost += numOneLitreDrum * 55.50;
        txaOutputRenovation.setText("Area: " + area + " square metre");
        txaOutputRenovation.append("\nVolume of paint required: " +
              df.format(numLitresOfPaint) + " litre");
        txaOutputRenovation.append("\n1-litre drums:" +
              numOneLitreDrum);
        txaOutputRenovation.append("\n2-litre drums:" +
              numTwoLitreDrum);
        txaOutputRenovation.append("\n5-litre drums:" +
              numFiveLitreDrum);
        txaOutputRenovation.append("\n\nTotal cost: " +
              df.format(cost));
    }

//=====================================================================
//   Question 1.2.3 (b)
//=====================================================================
     if (renovationType == 'T') {
       double tileCost = Double.parseDouble(JOptionPane.showInputDialog
                         ("Enter the cost per square metre"));
       cost = (area + 5) * tileCost;
       txaOutputRenovation.setText("Tiling walls in living room:\n
                                   Total cost: " + df.format(cost));
     }
}
```

```
//=====================================================================
//   Question 1.3.1
//=====================================================================
private void btnCalcAmountActionPerformed(java.awt.event.ActionEvent evt) {
    int prevReading = 0;
    int currReading = 0;
    double elecBill = 0;
    prevReading = Integer.parseInt(txfPrevious.getText());

    currReading = Integer.parseInt(txfCurrent.getText());
    if (prevReading > currReading) {
       JOptionPane.showMessageDialog(null, "Previous reading larger than
                                    current reading, please re-type");
       txfCurrent.setText("0");
    } else {
       int elecUsage = currReading - prevReading;
       if (elecUsage <= 600) {
           elecBill = elecUsage;
       } else {
           elecBill = 600 + ((elecUsage - 600) * 1.5);
       }
    }
    lblElectricity.setText(df.format(elecBill));
 }

//=====================================================================
//   Question 1.3.2
//=====================================================================

private void btnFindGeysersActionPerformed(java.awt.event.ActionEvent evt) {
    String geyserSize = txfGeyserSize.getText().trim();
    txaOutputGeysers.setText("");
    for (int i = 0; i < arrGeysers.length; i++) {
        int psnDash = arrGeysers[i].indexOf("-");
        String size = arrGeysers[i].substring(0, psnDash);
        if (size.equals(geyserSize)) {
          txaOutputGeysers.append(arrGeysers[i].substring(psnDash + 1) +
                                  "\n");
        }
    }
 }
```

ANNEXURE E: SOLUTION FOR QUESTION 2: JAVA

Object class: HomeLoan.java

```
//   A solution to Question 2
//=========================================================================
//  Question 2.1.1
//=========================================================================
  private String applicantName;
  private double disposableIncome;
  private double loanAmount;
  private int years;
  private double interestRate;

//=========================================================================
//  Question 2.1.2
//=========================================================================
  public HomeLoan(String applicantName, double disposableIncome, double
                  loanAmount) {
        this.applicantName = applicantName;
        this.disposableIncome = disposableIncome;
        this.loanAmount = loanAmount;
        this.years = 0;
        this.interestRate = 0;
    }
//=========================================================================
//  Question 2.1.3
//=========================================================================
   public String getApplicantName() {
        return applicantName;
    }
//=========================================================================
//  Question 2.1.4
//=========================================================================
    public void setInterestRate(double interestRate) {
        this.interestRate = interestRate;
    }
    public void setYears(int years) {
        this.years = years;
    }
//=========================================================================
//  Question 2.1.5 – Given code (uncomment)
//=========================================================================
    public double calculateInstalmentAmount() {
        double payment, rate;
        int result;
        rate = interestRate / 100.0;
        payment = (rate * loanAmount)/(1 - ((Math.pow((1 + rate),(years* -
                  1)))));
        result = (int) (Math.round(payment / 12));
        return result;
    }
```

```
//============================================================================
//  Question 2.1.6
//============================================================================
    public boolean isApproved() {
        boolean approved = true;
        if (loanAmount > 800000 && years < 25) {
            approved = false;
        }
        if (loanAmount <= 600000) {
            if (disposableIncome < (calculateInstalmentAmount() * 1.3)) {
                approved = false;
            }
        }
        return approved;
    }

//============================================================================
//  Question 2.1.7
//============================================================================
    public String toString() {
       String output = "Name of applicant name: "+ applicantName
                     +"\nDisposable income: " +
                     df.format(disposableIncome) +"\nLoan amount: " +
                     df.format(loanAmount) + "\nNumber of
                     years: " + years + "\nInterest rate: " +
                     interestRate);

        return output;
    }


//============================================================================
```

GUI class: Question2.java

```
//============================================================================
//  Given code
//============================================================================
HomeLoan loanObj;

//============================================================================
//  Question 2.2.1
//============================================================================
private void btnLocateApplicantActionPerformed(java.awt.event.ActionEvent
evt) {
    String applicant = (String) (cmbApplicants.getSelectedItem());
    if (applicant.isEmpty()) {
        JOptionPane.showMessageDialog(rootPane,"Select an
                                    applicant","Warning",0);
        txaOutput.setText("");
    } else {
    try {
         Scanner sc = new Scanner(new FileReader(applicant + ".txt"));
         txaOutput.setText("A home loan application was found for: " +
                           applicant);
         String name = sc.nextLine();
         double income = Double.parseDouble(sc.nextLine());
         double expenditure = Double.parseDouble(sc.nextLine());
```

```
         double loanAmount = Double.parseDouble(sc.nextLine());
         double disposableIncome = income - expenditure;
         loanObj = new HomeLoan(applicant, disposableIncome, loanAmount);
         JOptionPane.showMessageDialog(null, "Loan object successfully
created");
         btnEvaluate.setEnabled(true);
     } catch (FileNotFoundException e) {
     txaOutput.setText("No home loan application was submitted by " +
applicant + ".");
    }
}   }

//=========================================================================
//  Question 2.2.2
//=========================================================================

private void btnEvaluateActionPerformed(java.awt.event.ActionEvent evt) {
    int years = Integer.parseInt((String) (cmbYears.getSelectedItem()));
    double interestRate = Double.parseDouble(txfInterest.getText());
    loanObj.setInterestRate(interestRate);
    loanObj.setYears(years);
    boolean appr = loanObj.isApproved();
    if (appr) {
        txaOutput.setText(loanObj.toString()
           + "%\nLoan APPROVED with a monthly instalment of " +
                      df.format(calculateInstalmentAmount());

    } else {
        txaOutput.setText("Applicant: "+loanObj.getApplicantName()+ "\n");
        txaOutput.append("Loan NOT APPROVED");
    }
}
```

ANNEXURE F: SOLUTION FOR QUESTION 3: JAVA

```
//    A possible solution to Question 3

public class Q3 extends javax.swing.JFrame {
//=============================================================================
//This code is given in the program
//=============================================================================

// Given class


public class PopulateArrays {

public String [] populateArrSales(){

String[] arrSales = {"01#C#A122;500000", "01#C#A120;1200000", "01#C#A123;850000",
"01#R#A121;75000", "02#C#A123;6200000", "02#A#A121;190000",
        "02#C#A121;500000", "02#C#A120;1200000", "02#C#A120;850000",
"02#R#A124;75000", "02#C#A123;620000", "02#A#A124;690000",
        "02#C#A121;500000", "02#C#A120;1200000", "02#C#A120;850000",
"02#R#A124;75000", "02#C#A123;620000", "02#A#A124;790000",
        "02#R#A124;1500000", "02#C#A125;200000", "02#F#A121;400000",
"03#R#A120;175000", "03#C#A123;6200000", "03#A#A123;190000",
        "03#C#A124;500000", "03#C#A125;1200000", "03#C#A123;850000",
"03#R#A122;75000", "03#C#A123;620000", "03#A#A123;890000",
        "03#R#A123;1500000", "03#C#A125;200000", "03#A#A123;400000",
"04#R#A124;175000", "04#C#A120;6200000", "04#A#A123;190000",
        "03#R#A124;1500000", "03#C#A125;200000", "03#A#A122;400000",
"04#R#A121;175000", "04#C#A123;6200000", "04#A#A123;190000",
        "04#C#A121;500000", "04#C#A120;1200000", "04#C#A120;850000",
"04#R#A124;75000", "04#C#A123;620000", "04#A#A123;590000",
        "04#C#A123;500000", "04#C#A125;120000", "04#C#A124;850000",
"04#R#A122;75000", "04#C#A123;6200000", "04#A#A120;490000",
        "04#C#A125;500000", "04#C#A125;120000", "04#C#A124;850000",
"04#R#A122;75000", "04#C#A123;6200000", "04#A#A120;390000",
        "04#R#A120;1500000", "04#C#A121;200000", "04#A#A122;400000",
"04#R#A124;175000", "04#C#A121;6200000", "04#A#A123;190000",
        "11#C#A122;500000", "07#C#A120;1200000", "10#C#A123;850000",
"12#R#A121;75000", "12#C#A123;6200000", "11#A#A121;190000",
        "12#C#A121;500000", "12#C#A120;1200000", "11#C#A120;850000",
"12#R#A124;75000", "12#C#A123;620000", "11#A#A124;690000",
        "09#C#A121;500000", "08#C#A120;1200000", "11#C#A120;850000",
"12#R#A124;75000", "12#C#A123;620000", "10#A#A124;790000",
        "08#R#A124;1500000", "08#C#A125;200000", "11#A#A121;400000",
"11#R#A120;175000", "07#C#A123;6200000", "09#A#A123;190000",
        "07#C#A124;500000", "07#C#A125;1200000", "12#C#A123;850000",
"10#R#A122;75000", "08#C#A123;620000", "08#A#A123;890000",
        "06#R#A123;1500000", "09#C#A125;200000", "09#A#A123;400000",
"09#R#A124;175000", "08#C#A120;6200000", "07#A#A123;190000",
        "08#R#A124;1500000", "06#C#A125;200000", "07#A#A122;400000",
"08#R#A121;175000", "08#C#A123;6200000", "07#A#A123;190000",
        "11#C#A121;500000", "06#C#A120;1200000", "08#C#A120;850000",
"08#R#A124;75000", "12#C#A123;620000", "07#A#A123;590000",
        "12#C#A123;500000", "07#C#A125;120000", "08#C#A124;850000",
"07#R#A122;75000", "11#C#A123;6200000", "09#A#A120;490000",
        "08#C#A125;500000", "07#C#A125;120000", "09#C#A124;850000",
"06#R#A122;75000", "10#C#A123;6200000", "09#A#A120;390000",
        "10#R#A120;1500000", "07#C#A121;200000", "09#A#A122;400000",
"06#R#A124;175000", "10#C#A121;6200000", "09#A#A123;190000"};
return arrSales};
```

```
    public String [] populateArrAgents(){
    String[] arrAgents = {"A120:Wes Seelig",
        "A121:Tandra Peters",
        "A122:Lisha Champ",
        "A123:Melda Bissonette",
        "A124:Leta Hansell",
        "A125:Sid Rebello"};
    return arrAgents;
    }
}
```

Question 3 – GUI class

```
public class Question3_Memo extends javax.swing.JFrame {

//=============================================================================
//Given code
//=============================================================================

PopulateArrays objAssign = new PopulateArrays();

    String [] arrSales  = objAssign.populateArrSales);
    String [] arrAgents = objAssign.populateArrAgents();

public Q3() {
        initComponents();
        this.setLocationRelativeTo(this);
    }

//=============================================================================
// New code
//=============================================================================
    String [] arrTypes  = objAssign.populateArrTypes();
    String [] arrMonths = objAssign.populateArrMonths();

    int [][] arrStats = new int [3][5];

    public void clearStats() {
        for (int r = 0; r < 3; r++) {
            for (int c = 0; c < 5; c++) {
                arrStats[r][c] = 0;
            }
        }
    }


=============================================================================
// Question 3.1
=============================================================================
private void btnAgentsNameActionPerformed(java.awt.event.ActionEvent evt) {
  txaOutput.setText("");
  agentCode = txfAgentCode.getText().toUpperCase();
  boolean found = false;
  int p = 0;
  while (!found && p < arrAgents.length) {
    if (arrAgents[p].substring(0, 4).equalsIgnoreCase(agentCode))
    {
      agentName=arrAgents[p].substring(arrAgents[p].indexOf(":")+1);
      lblAgentName.setText(agentName);
      found = true;
    }
    p++;
 }
 if (!found) {
    lblAgentName.setText("Invalid agent code");
    txfAgentCode.setText("");
 }
}
```

```
=============================================================================
// Question 3.2.1
=============================================================================
Add an output area to the GUI, rename the output area
The solution makes use of a textArea-component
A grid can also be used together with a label to display the total sales.

=============================================================================
// Question 3.2.2
=============================================================================
private void btnAgentsSalesActionPerformed(java.awt.event.ActionEvent evt)
{
  btnAgentsName.doClick();
  if (agentName.length() > 0) {
      clearStats();
      int totalSales = 0;
      for (int i = 0; i < arrSales.length; i++) {
          if (arrSales[i].substring(5, 9).contentEquals(agentCode)) {
             int month = Integer.parseInt(arrSales[i].substring(0, 2));
             int col = month - 1;
             char propType = arrSales[i].toUpperCase().charAt(3);
             int row = 0;
             switch (propType) {
                  case 'C':
                      row = 0;
                      break;
                  case 'R':
                      row = 1;
                      break;
                  case 'A':
                      row = 2;
                      break;
              }
              arrStats[row][col]++;
              arrStats[row][12] += 1;
              totalSales += Integer.parseInt
               (arrSales[i].substring(arrSales[i].indexOf(";") + 1));
          }
      }
      txaOutput.setText(String.format("%-13s", ""));
      for (int i = 0; i < 12; i++) {
          txaOutput.append(String.format("%-4s", arrMonths[i]));
      }
      txaOutput.append("\n\n");
      for (int r = 0; r < 3; r++) {
          txaOutput.append(String.format("%-13s", arrTypes[r]));
          for (int c = 0; c < 13; c++) {
             txaOutput.append(String.format("%-4s", arrStats[r][c]));
          }
          txaOutput.append("\n\n");
      }
      lblAgentTotalSales.setText("\nTotal value of sales: R " +
totalSales);
      } else {
        txaOutput.setText("Please enter a valid agent code");
      }
    }
```

ANNEXURE G: SOLUTION FOR QUESTION 1: DELPHI

```
unit Question1UMemo;
     //A possible solution for Question 1
interface

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

type
  TfrmQuestion1 = class(TForm)
    pgcQuestion1: TPageControl;
    tsQ1_1: TTabSheet;
    pnlButtons: TPanel;
    bmbClose: TBitBtn;
    tsQ1_3: TTabSheet;
    tsQ1_2: TTabSheet;
    lblMarketValue: TLabel;
    lblSellingPrice: TLabel;
    lblNumBedrooms: TLabel;
    lblNumBathrooms: TLabel;
    chkPool: TCheckBox;
    btnSalesAdd: TButton;
    edtMarket: TEdit;
    edtSellingPrice: TEdit;
    edtBeds: TEdit;
    edtBaths: TEdit;
    lblHeadingElecKiloW: TLabel;
    lblPrevReading: TLabel;
    lblCurrReading: TLabel;
    lblElectricity: TLabel;
    lblGeyserSizes: TLabel;
    edtPrev: TEdit;
    edtCurrent: TEdit;
    edtSize: TEdit;
    lblAreaRenovated: TLabel;
    edtArea: TEdit;
    lblUnit: TLabel;
    redRenovate: TRichEdit;
    btnCost: TButton;
    btnFindGeysers: TButton;
    grpAreaRenovated: TGroupBox;
    grpElectricityUsed: TGroupBox;
    btnCalcAmount: TButton;
    lblAmountDue: TLabel;
    grpGeyserOptions: TGroupBox;
    grpAdvertisement: TGroupBox;
    redQ1: TRichEdit;
    rbPaint: TRadioButton;
    rbTiling: TRadioButton;
    memGeysers: TMemo;
    grpMethodRenovation: TGroupBox;
    procedure btnSalesAddClick(Sender: TObject);
    procedure btnCostClick(Sender: TObject);
```

```
    procedure FormCreate(Sender: TObject);
    procedure btnCalcAmountClick(Sender: TObject);
    procedure rbPaintClick(Sender: TObject);
    procedure rbTilingClick(Sender: TObject);
    procedure btnFindGeysersClick(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion1: TfrmQuestion1;
  cTypeReno  : char;
  arrGeysers : array [1..20] of string = ('50-QuickSun50','100-
QuickSun100','150-QuickSun150','50-Solar Magic','50-InHotWater',
                                          '100-SunnyBath 100','150-
SunnyBath 150','50-WaterJoy 50','100-WaterJoy 100',
                                          '150-BigTub 150','50-Small
Wonder','100-Medium Wonder','150-Large Wonder',
                                          '100-SolarWarmth 100','150-
SolarWarmth 150','50-Sun Magic','50-Eco Wonder 50',
                                          '100-Eco Wonder 100','150-Eco
Wonder 150','150-Big Earth Saver');

implementation

{$R *.dfm}
{$R+}
=============================================================================
// Question 1.1
=============================================================================
procedure TfrmQuestion1.btnSalesAddClick(Sender: TObject);
var
   rMarket, rSell : real;
   sAdvert        : string;
   iBed, iBath    : integer;
begin

    rMarket := StrToFloat(edtMarket.Text);
    rSell := StrToFloat(edtSellingPrice.Text);
    iBed  := StrToInt(edtBeds.Text);
    iBath := StrToInt(edtBaths.Text);
    sAdvert := Format('%2.0m',[rSell]) + '#' + edtBeds.text + 'Bed#' +
               edtbaths.text +'Bath#';

    if chkPool.Checked then
         sAdvert := sAdvert + 'Pool#';

    if rSell < rMarket then
         sAdvert := sAdvert + 'Bargain';

  redQ1.Lines.Clear;
  redQ1.Lines.Add('House for sale:' + #13 + sAdvert);
end;
```

```
//============================================================================
// Question 1.2.1
//============================================================================
procedure TfrmQuestion1.rbPaintClick(Sender: TObject);
begin
    cTypeReno := 'P'
end;
//============================================================================
// Question 1.2.2
//============================================================================
procedure TfrmQuestion1.rbTilingClick(Sender: TObject);
begin
    cTypeReno := 'T';
end;
//============================================================================
// Question 1.2.3
//============================================================================
procedure TfrmQuestion1.btnCostClick(Sender: TObject);
var
    rCost, rNumLitresPaint, rDrumCost, rTileCost: real;
    iArea, iLitres, iNumOneLitreDrums, iNumTwoLitreDrums,
                iNumFiveLitreDrums, iSelected : integer;
    sOutStr: string;
begin
   // Question 1.2.3 (a)
   iArea := StrToInt(edtArea.Text);
   iNumOneLitreDrums := 0;
   iNumTwoLitreDrums := 0;
   iNumFiveLitreDrums := 0;
   rCost := 0.0;

   if cTypeReno = 'P' then
     begin
        rNumLitresPaint := iArea/8;
        iLitres := Round(rNumLitresPaint + 0.4);
        // Alternative  >> Rounding to nearest EVEN number in Delphi
        // iLitres := Round(rNumLitresPaint);
        // if (iArea mod 8 > 0)
        //   then iLitres := iLitres + 1;

        if (iLitres div 5 > 0)then
          begin
            iNumFiveLitreDrums := iLitres div 5;
            iLitres := iLitres - (iNumFiveLitreDrums * 5);
            rCost := rCost + iNumFiveLitreDrums * 199.00;
          end;
        if (iLitres div 2 > 0) then
          begin
             iNumTwoLitreDrums := (iLitres div 2);
             iLitres := iLitres - (iNumTwoLitreDrums * 2);
             rCost := rCost + iNumTwoLitreDrums * 92.30;
          end;

          iNumOneLitreDrums := iLitres;
          rCost := rCost + iNumOneLitreDrums * 55.50;
           redRenovate.Lines.Clear;
```

```
           redRenovate.Lines.Add('Area: '+ edtArea.Text + ' square metres'
                + #13 +
           'Volume of paint required: '+ FloatToStr(rNumLitresPaint) +
           ' litres' + #13 + #13 +
           'Drums of paint required: ' + #13 +
           '1-litre drums:  '+IntToStr(iNumOneLitreDrums)+#13+
           '2-litre drums:  ' + IntToStr(iNumTwoLitreDrums) + #13 +
           '5-litre drums:  ' + IntToStr(iNumFiveLitreDrums) + #13 +
                                                          #13 +
           'Total cost: ' + FloatToStrF(rCost, ffCurrency, 8,2));
  end;

//Question 1.2.3 (b)
  if cTypeReno = 'T' then
    begin
     rTileCost := StrToFloat(InputBox('Tiling cost','Enter the cost
                      per square metres','0'));
     rCost := (iArea + 5) * rTileCost;
     redRenovate.Lines.Clear;
      redRenovate.Lines.Add('Tiling: '+ edtArea.Text + ' square
      meters' + #13 + 'Total cost: ' + FloatToStrF(rCost, ffCurrency,
      8,2));
    end;
end;
=========================================================================
//   Question 1.3.1
=========================================================================
procedure TfrmQuestion1.btnCalcAmountClick(Sender: TObject);
var
  rUsed, rAmount             : real;
  iPrevReading, iCurrReading: integer;
begin
  iPrevReading :=  StrToInt(edtPrev.Text);
  iCurrReading :=  StrToInt(edtCurrent.Text);

  if(iPrevReading > iCurrReading) then
    begin
     MessageDlg('Previous reading more than current reading,
                 please re-type',mtError,[mbOK],0);
     edtCurrent.Clear;
      rAmount := 0;
    end
  else
    begin
     rUsed :=  iCurrReading - iPrevReading;
     if rUsed <= 600 then
        rAmount := rUsed
     else
        rAmount := 600 + (rUsed - 600) * 1.5;
    end;
  lblElectricity.Caption:=  FloatToStrF(rAmount, ffCurrency, 8,2);
end;
```

```
//=====================================================================
//   Question 1.3.2
//=====================================================================
procedure TfrmQuestion1.btnFindGeysersClick(Sender: TObject);
var
  K iPosDash    : integer;
  sSize, sGeyser : String;
begin
  memGeysers.Clear;
  for K := 1 to Length(arrGeysers) do
   begin
     sGeyser := arrGeysers[K];
     iPosDash := Pos('-',sGeyser);
     sSize := Copy(sGeyser,1,iPosDash - 1);
     Delete(sGeyser,1,iPosDash);
     if sSize = edtSize.Text then
        memGeysers.Lines.Add(sGeyser);
   end;
end;
//=====================================================================
procedure TfrmQuestion1.FormCreate(Sender: TObject);
begin
      CurrencyString := 'R ';
end;

end.
```

ANNEXURE H: SOLUTION FOR QUESTION 2: DELPHI

Unit class: HomeLoan_U

```
unit HomeLoan_U;
   //Possible solution for Question 2
interface

uses
  SysUtils, Math, Messages, Dialogs;
=============================================================================
// Question 2.1.1
=============================================================================
Type
  THomeLoan = class(TObject)
    private
      fApplicantName    : String;
      fDisposableIncome,
      fInterestRate,
      fLoanAmount       : real;
      fYears            : integer;
    public
      constructor Create(ApplicantName : String; DisposableIncome,
                         LoanAmount : real);
      procedure setInterestRate(InterestRate : real);
      procedure setYears(NumberOfYears : integer);
      function getApplicantName : String;
      function calculateInstalmentAmount  : real;
      function isApproved() : boolean;
      function toString : String;
  end;

implementation

{ THomeLoan }
=============================================================================
// Question 2.1.2
=============================================================================
constructor THomeLoan.Create(ApplicantName : String; DisposableIncome,
                             LoanAmount : real);
begin
   fApplicantName := ApplicantName;
   fDisposableIncome := DisposableIncome;
   fLoanAmount := LoanAmount;
   fYears := 0;
   fInterestRate := 0.0;
end;
=============================================================================
// Question 2.1.3
=============================================================================
function THomeLoan.getApplicantName: String;
begin
  Result := fApplicantName;
end;
=============================================================================
// Question 2.1.4
=============================================================================
procedure THomeLoan.setYears(NumberOfYears: integer);
```

```
begin
   fYears := NumberOfYears;
end;

//============================================================================
// Question 2.1.5 Given code (Uncomment)
//============================================================================
function THomeLoan.calculateInstalmentAmount: real;
var
  rPayment, rRate : real;
begin
  rRate := fInterestRate / 100;
  rPayment := (rRate * fLoanAmount)/(1-Power((1+rRate),-fYears));
  Result := Round(rPayment / 12);
end;
//============================================================================
// Question 2.1.6
//============================================================================
function THomeLoan.isApproved() : boolean;
var
  bApproved: boolean;

begin
   bApproved := true;
   if (fLoanAmount > 800000) and (fYears < 25) then
       bApproved := false;
   if fLoanAmount <= 600000 then
      begin
         if fDisposableIncome < (calculateInstalmentAmount * 1.3) then
           bApproved := false
      end;
   Result := bApproved;
end;
//============================================================================
// Question 2.1.7
//============================================================================
function THomeLoan.toString: String;
var
  sOutput : String;
begin
  sOutput := 'Name of applicant: ' + fApplicantName  + #13;
  sOutput := sOutput + 'Disposable income: ' +
FloatToStrF(fDisposableIncome, ffCurrency, 8, 2) + #13;
  sOutput := sOutput + 'Loan amount: ' + FloatToStrF(fLoanAmount,
ffCurrency, 8, 2) + #13;
  sOutput := sOutput + 'Number of years: ' + IntToStr(fYears) + #13;
  sOutput := sOutput + 'Interest rate: ' + FloatToStrF(fInterestRate,
ffFixed, 5,1) + '% ' + #13;
  sOutput := sOutput + 'Loan APPROVED with a monthly instalment of ' +
FloatToStrF(calculateInstalmentAmount, ffCurrency, 8, 2);
  Result := sOutput;
end;

end.
```

Form class: Question2_P

```
unit Question2_UMEMO;

interface

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

type
  TfrmQuestion2 = class(TForm)
    grpLoans: TGroupBox;
    redQ2: TRichEdit;
    cboApplicants: TComboBox;
    pnlClose: TPanel;
    bmbClose: TBitBtn;
    lblInterest: TLabel;
    edtInterest: TEdit;
    btnLocateApplicant: TButton;
    btnEvaluate: TButton;
    grpApplicationStatus: TGroupBox;
    grpPossibleApplicants: TGroupBox;
    cboYears: TComboBox;
    lblYears: TLabel;
    procedure FormCreate(Sender: TObject);
    procedure btnEvaluateClick(Sender: TObject);
    procedure btnLocateApplicantClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion2: TfrmQuestion2;

implementation

uses
  HomeLoan_U;

var
   LoanObj : THomeLoan;

{$R *.dfm}
{$R+}
```

```
================================================================================
// Question 2.2.1
================================================================================

procedure TfrmQuestion2.btnLocateApplicantClick(Sender: TObject);
var
  TFile : TextFile;
  sName : String;
  sAppName, sFileName : String;
  rIncome, rExp, rLoan, rDispIncome : real;
begin
  if cboApplicants.ItemIndex = - 1 then
    begin
      MessageDlg('Please select an applicant', mtError, [mbOK], 0);
      Exit;
    end
  else
    begin
      sAppName := cboApplicants.Items[cboApplicants.ItemIndex];
      sFileName := sAppName + '.txt';
    end;

redQ2.Lines.Clear;
if NOT FileExists(sFileName) then
   begin
      redQ2.Lines.Add('No home loan application was submitted by ' +
                              sAppName + '.');
      btnEvaluate.Enabled := false;
      Exit;
   end;
AssignFile(TFile, sFileName);
Reset(TFile);
Readln(TFile, sName);
Readln(TFile, rIncome);
Readln(TFile, rExp);
Readln(TFile, rLoan);
redQ2.Lines.Add('A home loan application was found for: ' + sName);
rDispIncome := rIncome - rExp;
LoanObj := THomeLoan.Create(sName ,rDispIncome, rLoan);
      MessageDlg('Home loan object successfully created', mtConfirmation,
                        [mbOK], 0);
btnEvaluate.Enabled := true;
closeFile(TFile);

end;

================================================================================
// Question 2.2.2
================================================================================

procedure TfrmQuestion2.btnEvaluateClick(Sender: TObject);
var
  rInterest : real;
  iYears    : integer;
begin
```

```
  iYears := StrToInt(cboYears.Items[cboYears.ItemIndex]);
    rInterest := StrToFloat(edtInterest.Text);

  LoanObj.setInterestRate(rInterest);
  LoanObj.setYears(iYears);

  redQ2.Lines.Clear;
  if LoanObj.isApproved then
    begin
      redQ2.Lines.Add(LoanObj.toString);
      redQ2.Lines.Add('Loan APPROVED with a monthly instalment of ' +
     FloatToStrF(LoanObj.calculateInstalmentAmount, ffCurrency, 8, 2));
    end
  else
    begin
      redQ2.Lines.Add('Applicant: ' + LoanObj.getApplicantName);
      redQ2.Lines.Add('Loan NOT APPROVED');
    end;
end;
//=====================================================================
procedure TfrmQuestion2.FormCreate(Sender: TObject);
begin
  CurrencyString := 'R ';
//  cboApplicants.ItemIndex := 0;
end;

end.
```

ANNEXURE I: SOLUTION FOR QUESTION 3: DELPHI

```
unit Question3_UMemo;
interface

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

type
  TfrmQ3 = class(TForm)
    grpSalesDetail: TGroupBox;
    lblAgentName: TLabel;
    edtCode: TEdit;
    btnAgentsName: TButton;
    btnAgentsSales: TButton;
    grpDisplay: TGroupBox;
    lblAgentCode: TLabel;
    bmbClose: TBitBtn;
    pcQ3: TPageControl;
    lblTotal: TLabel;
    tabREdit: TTabSheet;
    tabSGrid: TTabSheet;
    redQ3: TRichEdit;
    sgrdQ3: TStringGrid;
    procedure btnAgentsNameClick(Sender: TObject);
    procedure btnAgentsSalesClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQ3: TfrmQ3;

      arrSales:array[1..132] of string =('01#C#A122;500000',
'01#C#A120;1200000', '01#C#A123;850000', '01#R#A121;75000',
'02#C#A123;6200000', '02#A#A121;190000', '02#C#A121;500000',
'02#C#A120;1200000', '02#C#A120;850000', '02#R#A124;75000',
'02#C#A123;620000', '02#A#A124;690000',  '02#C#A121;500000',
'02#C#A120;1200000', '02#C#A120;850000', '02#R#A124;75000',
'02#C#A123;620000', '02#A#A124;790000',  '02#R#A124;1500000',
'02#C#A125;200000', '02#A#A121;400000',  '03#R#A120;175000',
'03#C#A123;6200000', '03#A#A123;190000', '03#C#A124;500000',
'03#C#A125;1200000', '03#C#A123;850000', '03#R#A122;75000',
'03#C#A123;620000', '03#A#A123;890000',  '03#R#A123;1500000',
'03#C#A125;200000', '03#A#A123;400000',  '04#R#A124;175000',
'04#C#A120;6200000', '04#A#A123;190000', '03#R#A124;1500000',
'03#C#A125;200000', '03#A#A122;400000',  '04#R#A121;175000',
'04#C#A123;6200000', '04#A#A123;190000', '04#C#A121;500000',
'04#C#A120;1200000', '04#C#A120;850000', '04#R#A124;75000',
'04#C#A123;620000', '04#A#A123;590000',  '04#C#A123;500000',
'04#C#A125;120000', '04#C#A124;850000',  '04#R#A122;75000',
'04#C#A123;6200000', '04#A#A120;490000', '04#C#A125;500000',
'04#C#A125;120000', '04#C#A124;850000', '04#R#A122;75000',
```

```
'04#C#A123;6200000', '04#A#A120;390000','04#R#A120;1500000',
'04#C#A121;200000', '04#A#A122;400000', '04#R#A124;175000',
'04#C#A121;6200000', '04#A#A123;190000','11#C#A122;500000',
'07#C#A120;1200000', '10#C#A123;850000', '12#R#A121;75000',
'12#C#A123;6200000', '11#A#A121;190000', '12#C#A121;500000',
'12#C#A120;1200000', '11#C#A120;850000', '12#R#A124;75000',
'12#C#A123;620000', '11#A#A124;690000',  '09#C#A121;500000',
'08#C#A120;1200000', '11#C#A120;850000', '12#R#A124;75000',
'12#C#A123;620000', '10#A#A124;790000',  '08#R#A124;1500000',
'08#C#A125;200000', '11#F#A121;400000', '11#R#A120;175000',
'07#C#A123;6200000', '09#A#A123;190000','07#C#A124;500000',
'07#C#A125;1200000', '12#C#A123;850000', '10#R#A122;75000',
'08#C#A123;620000', '08#A#A123;890000',  '06#R#A123;1500000',
'09#C#A125;200000', '09#A#A123;400000', '09#R#A124;175000',
'08#C#A120;6200000', '07#A#A123;190000','08#R#A124;1500000',
'06#C#A125;200000', '07#A#A122;400000', '08#R#A121;175000',
'08#C#A123;6200000', '07#A#A123;190000','11#C#A121;500000',
'06#C#A120;1200000', '08#C#A120;850000', '08#R#A124;75000',
'12#C#A123;620000', '07#A#A123;590000',  '12#C#A123;500000',
'07#C#A125;120000', '08#C#A124;850000', '07#R#A122;75000',
'11#C#A123;6200000', '09#A#A120;490000','08#C#A125;500000',
'07#C#A125;120000', '09#C#A124;850000', '06#R#A122;75000',
'10#C#A123;6200000', '09#A#A120;390000','10#R#A120;1500000',
'07#C#A121;200000', '09#A#A122;400000', '06#R#A124;175000',
'10#C#A121;6200000', '09#A#A123;190000');

      arrAgents:array[1..6] of string =('A120:Wes Seelig',
        'A121:Tandra Peters',
        'A122:Lisha Champ',
        'A123:Melda Bissonette',
        'A124:Leta Hansell',
        'A125:Sid Rebello');

implementation
{$R *.dfm}
{$R+}
var
   sAgentCode : String = '';
   sAgentName : String = '';
=============================================================================
// Question 3.1
=============================================================================
procedure TfrmQ3.btnAgentsNameClick(Sender: TObject);
var
   K : integer;
   bFound : boolean;
begin
 sAgentCode := edtCode.Text;
 bFound := false;
 K := 1;
 while  (K <= Length(arrAgents)) AND NOT bFound do
  begin
    if pos(sAgentCode,  arrAgents[K]) = 1 then
      begin
```

```
         sAgentName := copy(arrAgents[K], pos(':', arrAgents[K])+1,
                            length(arrAgents[K]));
         lblAgentName.Caption := sAgentName;
         bFound := True;
       end
      else
        inc(K, 1);
  end;//while
  if NOT bFound then
    begin
      edtCode.Clear;
      lblAgentName.Caption := 'Invalid agent code';
    end;
end;

=============================================================================
// Question 3.2.1
=============================================================================
Add an output area to the GUI, rename the output area
The solution makes use of a textArea-component
A grid can also be used together with a label to display the total sales.
=============================================================================
// Question 3.2.2
=============================================================================
procedure TfrmQ3.btnAgentsSalesClick(Sender: TObject);
var
   arrAgentSales : array[1..3, 1..13] of integer;
   iIndex, iRow, iCol, iPrice, iTotal, iRowTotal : integer;
   sTemp : String;
begin
 // Question 3.2
 //Initialise 2D array
 for iRow := 1 to 3 do
   for iCol := 1 to 13 do
     arrAgentSales[iRow, iCol] := 0;
 //analise sales array
 iTotal := 0;
 for iIndex := 1 to length(arrSales) do
  begin
    sTemp := arrSales[iIndex];
    if pos(sAgentCode, sTemp) > 0 then
      begin
         iCol := StrToInt(Copy(sTemp, 1, pos('#', sTemp)-1));
         iPrice := StrToInt(Copy(sTemp, pos(';',sTemp)+1, length(sTemp)));
         Inc(iTotal, iPrice);

         Delete(sTemp, 1, pos('#', sTemp));
         case sTemp[1] of  //type of sale
           'C' : iRow := 1;
           'R' : iRow := 2;
           'A' : iRow := 3;
         end;//case
         inc(arrAgentSales[iRow, iCol], 1);
      end; //if
  end;//for

  //determine row totals
```

```
  for iRow := 1 to 3 do
   begin
     iRowTotal := 0;
     for iCol := 1 to 12 do
      begin
         inc(iRowTotal, arrAgentSales[iRow, iCol]);
      end;
     arrAgentSales[iRow, 13] := iRowTotal;
   end;

  //Output: RichEdit
  redQ3.Lines.Clear;
  redQ3.Paragraph.TabCount := 1;
  redQ3.Paragraph.Tab[0]   := 70;
  sTemp := ' '+ #9 + 'Jan' + #9 + 'Feb' + #9 + 'Mar' + #9 + 'Apr' + #9 +
           'May' + #9+
           'Jun' + #9 + 'Jul' + #9 + 'Aug' + #9 + 'Sep' + #9 + 'Oct' + #9 +
           'Nov' + #9 + 'Dec';
  redQ3.Lines.Add(sTemp);
  redQ3.Lines.Add(' ');
  sTemp := 'Commercial' + #9;
  for iCol := 1 to 13 do
    sTemp := sTemp + IntToStr(arrAgentSales[1, iCol]) + #9 ;
  redQ3.Lines.Add(sTemp);

  redQ3.Lines.Add(' ');
  sTemp := 'Residential' + #9;
  for iCol := 1 to 13 do
    sTemp := sTemp + IntToStr(arrAgentSales[2, iCol]) + #9 ;
  redQ3.Lines.Add(sTemp);

  redQ3.Lines.Add(' ');
  sTemp := 'Agricultural' + #9;
  for iCol := 1 to 13 do
    sTemp := sTemp + IntToStr(arrAgentSales[3, iCol]) + #9 ;
  redQ3.Lines.Add(sTemp);

  redQ3.Lines.Add(' ');
  redQ3.Lines.Add('Total value of sales: R ' + IntToStr(iTotal));

  //Output: StringGrid
  //Column headings
 {sgrdQ3.Cells[1,0] := 'Jan';
  sgrdQ3.Cells[2,0] := 'Feb';
  sgrdQ3.Cells[3,0] := 'Mrt';
  sgrdQ3.Cells[4,0] := 'Apr';
  sgrdQ3.Cells[5,0] := 'May';
  sgrdQ3.Cells[6,0] := 'Jun';
  sgrdQ3.Cells[7,0] := 'Jul';
  sgrdQ3.Cells[8,0] := 'Aug';
  sgrdQ3.Cells[9,0] := 'Sep';
  sgrdQ3.Cells[10,0] := 'Oct';
  sgrdQ3.Cells[11,0] := 'Nov';
  sgrdV3.Cells[12,0] := 'Dec';
  //Row headings
  sgrdQ3.Cells[0,1] := 'Commercial';
  sgrdQ3.Cells[0,2] := 'Residensieel';
```

```
  sgrdQ3.Cells[0,3] := 'Agricultural';

  sgrdQ3.ColWidths[0] := 100;

  for iRow := 1 to 3 do
   for iRow := 1 to 13 do
     sgrdQ3.Cells[iCol, iRow] := IntToStr(arrAgentVerkope[iRow, iCol]);

  lblTotal.Caption :='Total value of sales: R' + IntToStr(iTotal); }
end;

end.
```