**Information Technology November Exam 2017 Grade 11 Paper 1  
Mark Sheet**

**150**

Learner’s Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 1** [40 marks]

**1.1**

|  |  |  |
| --- | --- | --- |
|  | **Max** | Learner’s Mark |
| procedure TfrmAccCustomers.btnPensionerCountClick(Sender: TObject);  var  iPensionerCount: integer;  begin  iPensionerCount := 0;  with dmAccCustomers do(correct looping used)  begin  tblAccCustomers.First;(correct use of First and Next)  while NOT tblAccCustomers.Eof do  begin  if tblAccCustomers['Pensioner'] = True then  inc(iPensionerCount);  tblAccCustomers.Next;  end; // while  end; // with  ShowMessage('Number of pensioner account customers: ' + intToStr  (iPensionerCount));  end; | **(4)** |  |

**1.2**

|  |  |  |
| --- | --- | --- |
| procedure TfrmAccCustomers.btnSortingClick(Sender: TObject);  begin  with dmAccCustomers do  begin  tblAccCustomers.Sort:='DateAccountOpened DESC,   Surname ASC, Firstname ASC';  end;  end; | (4) |  |

**1.3**

|  |  |  |
| --- | --- | --- |
| procedure TfrmAccCustomers.btnLocateByIDClick(Sender: TObject);  begin  with dmAccCustomers do  if tblAccCustomers.Locate ('CustomerID', edtCustomerID.Text, [loCaseInsensitive]) then  begin  ShowMessage('Customer ID ' + edtCustomerID.Text + ' has been found');  ShowMessage('The amount owing is ' + FloatToStr  (tblAccCustomers['AmountOwing']));  end  else  ShowMessage('No such customer ID appears in the database');  end; | **(8)** |  |

**1.4**

|  |  |  |
| --- | --- | --- |
| procedure TfrmAccCustomers.btnCreditLimitOver730Click(Sender: TObject);  begin  dmAccCustomers.tblAccCustomers.Filter := 'CreditLimit > 730'; dmAccCustomers.tblAccCustomers.Filtered := True;  end; | **(4)** |  |

**1.5**

|  |  |  |
| --- | --- | --- |
| procedure TfrmAccCustomers.btnAddRecordClick(Sender: TObject);  begin  with dmAccCustomers do  begin  tblAccCustomers.Last;  tblAccCustomers.Insert;(for .last,insert and Post)  tblAccCustomers['CustomerID'] := edtCustID.Text;  tblAccCustomers['DateAccountOpened'] := edtDateAccountOpened.Date;  tblAccCustomers['Firstname'] := edtFirstname.Text;  tblAccCustomers['Surname'] := edtSurname.Text;  tblAccCustomers['Pensioner'] := cbbPensioner.Checked;  tblAccCustomers['CreditLimit'] := StrToFloat(edtCreditLimit.Text);  tblAccCustomers['AmountOwing'] := StrToFloat(edtAmountOwing.Text);  tblAccCustomers.Post;  end;  end; | **(8)** |  |

**1.6**

|  |  |  |
| --- | --- | --- |
| procedure TfrmAccCustomers.btnTotOwingAndAveClick(Sender: TObject);  var  rTotal, rAverage: real;  begin  rTotal := 0;  dmAccCustomers.tblAccCustomers.Open;  dmAccCustomers.tblAccCustomers.First;  while NOT dmAccCustomers.tblAccCustomers.Eof do  begin  rTotal := rTotal + dmAccCustomers.tblAccCustomers['AmountOwing'];  dmAccCustomers.tblAccCustomers.Next;(including .Open and .First)  end;  rAverage := rTotal / dmAccCustomers.tblAccCustomers.RecordCount;  redDisplay.Clear;  redDisplay.Lines.Add('Total amount outstanding: ' + FloatToStrf(rTotal,  ffCurrency, 10, 2));  redDisplay.Lines.Add('Average amount owing per customer: ' + FloatToStrf  (rAverage, ffCurrency, 10, 2));(for correct formatting)  end; | **(8)** |  |

**[40]**

**Question 2** [52 marks]

**2.1**

|  |  |  |
| --- | --- | --- |
| procedure TfrmProduce.btnQ2\_1Click(Sender: TObject);  var  rKg, rPrice, rTotPrice: real;  begin  rKg := StrToFloat(edtKG.Text);  rPrice := StrToFloat(edtPrice.Text);  rTotPrice := rKg \* rPrice;  edtTotalPrice.Text := FloatToStrf(rTotPrice, ffCurrency, 6, 2);(for formatting)  end; | **(6)** |  |

**2.2**

|  |  |  |
| --- | --- | --- |
| procedure TfrmProduce.btnQ2\_2Click(Sender: TObject);  var  rTotPrice, rDiscountedPrice: real;  begin  rTotPrice := StrToFloat(edtPayPrice.Text);  rDiscountedPrice := rTotPrice - (rTotPrice \* 0.05);  edtPensionerPrice.Text := FloatToStrf(rDiscountedPrice, ffCurrency, 6, 2);(formatting)  end; | **(6)** |  |

**2.3**

|  |  |  |
| --- | --- | --- |
| procedure TfrmProduce.btnQ2\_3Click(Sender: TObject);  var  iDice1, iDice2: integer;  sPrize: String;  iRandomPrize: integer;  begin  // generate a random number for the prize  Randomize;  iRandomPrize := Random(5) + 1;   sPrize := arrPrizes[iRandomPrize];  iDice1 := Random(6) + 1;  iDice2 := Random(6) + 1;  redMainDisplay.Clear;  redMainDisplay.Lines.Add('Dice 1 = ' + IntToStr(iDice1));  redMainDisplay.Lines.Add('Dice 2 = ' + IntToStr(iDice2));  if iDice1 = iDice2 then  redMainDisplay.Lines.Add('Congratulations you have won: ' + sPrize)(correct if)  else  redMainDisplay.Lines.Add('Sorry we wish you success the next time');(correct else) end; | **(10)** |  |

**2.4**

|  |  |  |
| --- | --- | --- |
| procedure TfrmProduce.btnQ2\_4Click(Sender: TObject);  var  bBagKg1, bBagKg2, bBagKg3: Boolean;  iKgs: integer;  iNum3Bags, iNum2Bags, iNum1Bags: integer;  iNumKgLeft: integer;  sSelectedProduct: String;  begin  iNum3Bags := 0;  iNum2Bags := 0;  iNum1Bags := 0;  redMainDisplay.Clear;  sSelectedProduct := cmbProducts.Items[cmbProducts.ItemIndex];  // select number of kgs from spinEdit  iKgs := sedWeightKgs.Value;  // bag size preference  bBagKg1 := false;  bBagKg2 := false;  bBagKg3 := false;  // selection of bag sizes  if chk1kg.Checked then  bBagKg1 := true;  if chk2kg.Checked then  bBagKg2 := true;  if chk3kg.Checked then  bBagKg3 := true;(check box correctly assigned for all)  // the processing for 1,2 and 3 kg bag sizes selected  if ((chk1kg.Checked) AND (chk2kg.Checked) AND (chk3kg.Checked)) then  begin  iNum3Bags := iKgs DIV 3;  iNumKgLeft := iKgs - iNum3Bags \* 3;  iNum2Bags := iNumKgLeft DIV 2;  iNum1Bags := iKgs - (iNum2Bags \* 2 + iNum3Bags \* 3);(correct calculation)  end;  // the processing for 1 and 2 kg bag sizes selected  if ((chk1kg.Checked) AND (chk2kg.Checked)) then  begin  iNum2Bags := iKgs DIV 2;  iNumKgLeft := iKgs - iNum2Bags \* 2;  iNum1Bags := iKgs - (iNum2Bags \* 2); (correct calculation)  end;  // the processing for 1 and 3 kg bag sizes selected  if ((chk1kg.Checked) AND (chk3kg.Checked)) then  begin  iNum3Bags := iKgs DIV 3;  iNumKgLeft := iKgs - iNum3Bags \* 3;  iNum1Bags := iKgs - (iNum3Bags \* 3); (correct calculation)  end;  // the processing for 2 and 3 kg bag sizes selected  if ((chk2kg.Checked) AND (chk3kg.Checked)) then  begin  iNum3Bags := iKgs DIV 3;  iNumKgLeft := iKgs - iNum3Bags \* 3;  iNum2Bags := iNumKgLeft DIV 2;  iNum1Bags := iKgs - (iNum3Bags \* 3 + iNum2Bags \* 2); (correct calculation)  end;  // the processing for 1 kg bag sizes selected  if ((chk1kg.Checked) AND (NOT(chk2kg.Checked)) AND (NOT(chk3kg.Checked))) then  begin  iNum1Bags := iKgs;  end;  // set tabs  redMainDisplay.Paragraph.TabCount := 2;  redMainDisplay.Paragraph.Tab[0] := 10;  redMainDisplay.Paragraph.Tab[1] := 130;  // display the product and number of each bag type selected  // display product  redMainDisplay.Lines.Add('Product: ' + #9 + sSelectedProduct);  // display number of kgs product required  redMainDisplay.Lines.Add('Kgs of product required: ' + #9 + IntToStr(iKgs));  // display bag selection  redMainDisplay.Lines.Add('1 Kg bag choice: ' + #9 + IntToStr(iNum1Bags));  redMainDisplay.Lines.Add('2 Kg bag choice: ' + #9 + IntToStr(iNum2Bags));  redMainDisplay.Lines.Add('3 Kg bag choice: ' + #9 + IntToStr(iNum3Bags));(correct display)  // WRITE DATA TO TEXT FILE  // store the product type and the number of kgs sold in a text file  // create a delimited text file as follows  // name of product and total kgs sold separated by a '#'  // eg .. Potatoe#12  sSalesLine := sSelectedProduct + '#' + IntToStr(iKgs);  AssignFile(BulkSalesFile, 'bulkSale.txt');  if FileExists('bulkSale.txt') = false then  begin  Rewrite(BulkSalesFile);  end  else  begin  Append(BulkSalesFile);  Writeln(BulkSalesFile, sSalesLine);  ShowMessage('Data has been written to the bulkSale.txt file');  end;  CloseFile(BulkSalesFile);  end; | **(30)** |  |

**[52]**

**Question 3** [58 marks]

**3.1**

|  |  |  |
| --- | --- | --- |
| procedure TfrmVegFruitSales.btnStoreFileToArrayClick(Sender: TObject);  var  iPosHash: integer;  sItem: string;  iMassKg: integer; (required variables correctly declared)  begin  AssignFile(SalesFile, 'sales.txt');  Try  Reset(SalesFile);  Except  ShowMessage('File not found');(correct use of try Except and output message)  Exit;  End;  // declared globally  // iIndex: integer;  // arrPotType: array [1 .. 50] of String;  // arrKgs: array [1 .. 50] of integer;(arrays arrPotType and arrKgs declared as private or any other way giving global scope)  iIndex := 1;  while NOT EOF(SalesFile) do  begin  Readln(SalesFile, sSalesLine);  iPosHash := Pos('#', sSalesLine);  sItem := Copy(sSalesLine, 1, iPosHash - 1);  Delete(sSalesLine, 1, iPosHash);  iMassKg := StrToInt(sSalesLine); (correct separation of potato type and weight)  arrPotType[iIndex] := sItem;  arrKgs[iIndex] := iMassKg;(correct storage in arrays)  inc(iIndex);  end;  CloseFile(SalesFile);  ShowMessage('The name of the potato and weight sold stored in arrays');  end; |  | **(13)** |

**3.2**

|  |  |  |
| --- | --- | --- |
| procedure TfrmVegFruitSales.btnPotatoTypesSoldClick(Sender: TObject);  var  I: integer;  begin  // display data in arrays  // set tabs  redDisplay.Paragraph.TabCount := 2;  redDisplay.Paragraph.Tab[0] := 10;  redDisplay.Paragraph.Tab[1] := 100;  redDisplay.Lines.Add('The contents of the arrays');  redDisplay.Lines.Add('-------------------------------------');  redDisplay.Lines.Add('Potato type' + #9 + 'Weight');  for I := 1 to iIndex - 1 do  redDisplay.Lines.Add(arrPotType[I] + #9 + IntToStr(arrKgs[I]));  end; | **(8)** |  |

**3.3**

|  |  |  |
| --- | --- | --- |
| procedure TfrmVegFruitSales.btnFruitDisplayClick(Sender: TObject);  var  I: integer;  begin  redDisplay.Clear;  redDisplay.Lines.Add('The arrays containing fruit names and weight are: ');  for I := 1 to 12 do  redDisplay.Lines.Add(arrFruit[I] + #9 + IntToStr(arrWeight[I]));  end; | **(10)** |  |

**3.4**

|  |  |  |
| --- | --- | --- |
| /find fruit(Error in question regarding naming of the button – ignore the error)  procedure TfrmVegFruitSales.btnTotalSalesClick(Sender: TObject);  var sSearchFruit : string;  iFruitWeight : integer;  I,K: Integer;  bFound : boolean;  iCount : integer;  begin  redDisplay.Clear;  sSearchFruit := InputBox('Fruit search','Enter name 0f fruit to search','');  bFound := false;  iCount := 12;  K := 1;(iCount and K variables initial values)  while (Not (bFound)) AND (K<iCount) do  begin  if (arrFruit[K] = sSearchFruit) then  bFound := true  else  inc(K);  end;  if bFound then  begin  redDisplay.Lines.Add(sSearchFruit + ' was found in position ' + IntToStr(K));  redDisplay.Lines.Add('The weight of '+sSearchFruit+' is '+ intToStr(arrWeight[K]));(correct display of position and weight)  end  else  redDisplay.Lines.Add(sSearchFruit+' was not found in the array');  end; | **(8)** |  |

**3.5**

|  |  |  |
| --- | --- | --- |
| //determine fruit with highest weight  //error in question – button incorrectly named – ignore the error  procedure TfrmVegFruitSales.btnAverageSalesClick(Sender: TObject);  var  iHighest: integer;  sFruit : string; (all required variables correctly declared)  I: Integer;  begin  iHighest := arrWeight[1]; );  sFruit := arrFruit[1]; );  for I := 1 to 12 do);  begin  if arrWeight[I] > iHighest then  begin  iHighest := arrWeight[I];  sFruit := arrFruit[I]  end;  end;  redDisplay.Clear;  redDisplay.Lines.Add('The fruit with the highest weight is '+ sFruit+ ' with weight '+  intToStr(iHighest));  end; | **(7)** |  |

**3.6**

|  |  |  |
| --- | --- | --- |
| procedure TfrmVegFruitSales.btnSortKgsClick(Sender: TObject);  var  iTemp: integer;  sTemp: string;   I: integer;  J: integer;   begin  // sort the arrays based on kgs  for I := 1 to 11 do  begin  for J := I + 1 to 12 do  begin  if arrWeight[I] > arrWeight[J] then  begin  iTemp := arrWeight[I];  sTemp := arrFruit[I];   arrWeight[I] := arrWeight[J];  arrFruit[I] := arrFruit[J];   arrWeight[J] := iTemp;  arrFruit[J] := sTemp;  end;  end;  end;  ShowMessage('Arrays sorted in ascending order according to weight');  ShowMessage('Use button Question 3\_3 to see sorted arrays');  end; | **(12)** |  |

**[58]**

**TOTAL: 150**