Province of the
EASTERN CAPE
EDUCATION

NATIONAL SENIOR CERTIFICATE

GRADE 11

NOVEMBER 2018

INFORMATION TECHNOLOGY P1
MARKING GUIDELINE

MARKS: **150**

This marking guideline consists of 17 pages.

NAME OF LEARNER:				
TOTAL QUESTION 1	**TOTAL QUESTION 2**	**TOTAL QUESTION 3**	**TOTAL QUESTION 4**	**TOTAL**
/44	**/22**	**/30**	**/54**	**/150**

QUESTION 1		MAX MARKS	MARKS ACHIEVED
1.1	**BUTTON: [QUESTION 1.1]** Get the waiter's name and surname ✓ Extract the surname ✓ making use of pos function ✓ Create the reference: first letter of name✓ and '.' ✓ and surname ✓ Randomize the code ✓ correct range ✓ Display the reference ✓ Display the code ✓	**10**	
1.2	**BUTTON: [QUESTION 1.2]** Get the ID number ✓ Get the length of the ID number ✓ If the length is equal to 13 ✓ Loop through the ID ✓ Check if characters are numbers or not ✓ if no letters in ID ✓ Get system date ✓ Calculate the age: Work out the YYYY using the ID (1900s or 2000s) ✓✓ Get the YYYY from system date ✓ Subtract ✓ If/case statement ✓ Correct ranges ✓ Display correct message ✓ If ID is incorrect ✓ Display a message ✓	**16**	
1.3	**BUTTON: [QUESTION 1.3]** Get the hours ✓ If the checkbox is selected ✓ Get the overtime hours from inputbox ✓as a number ✓ Calculate normal hours wage ✓ Calculate overtime hours wage ✓ Add together ✓ Display total wage ✓ formatted to currency and two decimal places ✓ Change panel's colour to skyblue ✓	**10**	

1.4	**BUTTON: [QUESTION 1.4]** Get the item from combobox ✓ If/Case statement ✓ Get the size ✓✓ Display the order heading ✓ item ✓ and size in brackets ✓ and Peak Cap over three lines ✓	**8**	
		44	

QUESTION 2		MAX MARKS	MARKS ACHIEVED

2.1	tblTables.first; ✓
	while not tblTables.eof do✓
	begin
	rtotal := rtotal✓ + tblTables['TableAmountPaid']; ✓
	tblTables.next; ✓
	end;
	Display total amount✓
		6	

2.2	tblTables.Sort := 'waiterid'; ✓	**1**	
2.3	Get the waiterID from inputbox✓ tblTables.First; ✓ Initialise the total variable✓ Display the WaiterID as a heading✓ Loop✓ if swaiter = tblTables['waiterID'] then✓ Display the TableID✓ Increment the total variable✓ Go to next record✓ Display total number of tables✓	**10**	
2.4	tblTables.Insert; ✓ tblTables['TableID'] := 71; ✓ tblTables['WaiterID'] := 10; ✓ tblTables['TableGuests'] := 5; ✓ tblTables.Post; ✓	**5**	
		22	

QUESTION 3		MAX MARKS	MARKS ACHIEVED
3.1	Assign file for ShoppingList.txt✓ Rewrite✓ If the file (Stocklist.txt) does not exists✓ Display a message and exit✓ Assign file for Stocklist.txt✓ Reset✓ Display the heading✓ Loop✓ read from file✓ Find the position of #✓ Copy the item ✓ Delete the item✓ Extract the current amount as an integer✓ Extract the unit✓ Extract the minimum ✓ Extract the maximum✓ If the current amount is less than the minimum✓ Calculate what is needed✓ Round the amount needed up to nearest whole number ✓ (rneed := ceil(rneed)); If unit is kg✓ or bottles then✓ Display the amount needed✓, unit✓ and item✓ Write to file✓ Else✓ Display amount needed✓ and item✓ Write to file✓ Display message that shopping list is saved ✓	**30**	
		30	

QUESTION 4		MAX MARKS	MARKS ACHIEVED
4.1	Add procedure under public✓ Procedure Sort heading✓ Outer loop✓ Inner Loop✓ If arritem[k] > arritem[l] ✓ Swop arritem ✓✓✓ Swop arrcost ✓✓✓	**11**	
4.2	Initialise food counter✓ Initialise drink counter✓ Call the Sort method✓ Loop ✓ Find the position of the #✓ If it is a Food item (F) ✓ Increase food counter✓ Assign item to an array or store for later use✓ Assign cost to an array✓ If it is a Drinks item (D) ✓ Increase drink counter✓ Assign item to an array or store for later use✓ Assign cost to an array✓ Display heading for FOOD✓ Loop✓ correct amount of times✓ Display food items✓ Display heading for DRINKS✓ Loop✓ correct amount of times Display drink items✓	**20**	
4.3	Clear the richedit✓ Set tab count to 1✓ Set tab to 150✓ Display heading✓ Loop✓ Calculate VAT✓ and markup✓ Round up (ceil) ✓ Display the price✓ formatted to currency✓	**10**	
4.4.1	function Found(sitem : string✓) : boolean; ✓ Set Boolean variable to false✓ Loop✓ If item is found ✓ Change Boolean variable to true✓ Assign Boolean variable to function name/result✓	**7**	
4.4.2	Get item from input box✓ Call the found method✓ in if statement✓ Display message (found) ✓ Else✓ Display message (not found) ✓	**6**	
		54	

SAMPLE SOLUTIONS

QUESTION 1

```
unit Question1_u;

interface

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

type
  TfrmQuestion1 = class(TForm)
    gbxQ1_1: TGroupBox;
    btnQ1_1: TButton;
    edtWaiter: TLabeledEdit;
    edtRef: TLabeledEdit;
    edtCode: TLabeledEdit;
    gbxQ1_2: TGroupBox;
    edtID: TLabeledEdit;
    btnQ1_2: TButton;
    redQ1_2: TRichEdit;
    gbxQ1_3: TGroupBox;
    gbxQ1_4: TGroupBox;
    sedHours: TSpinEdit;
    lblHours: TLabel;
    cbxOvertime: TCheckBox;
    btnQ1_3: TButton;
    pnlOutput: TPanel;
    cmbItem: TComboBox;
    btnQ1_4: TButton;
    rgpSizes: TRadioGroup;
    procedure btnQ1_1Click(Sender: TObject);
    procedure btnQ1_2Click(Sender: TObject);
    procedure btnQ1_3Click(Sender: TObject);
    procedure btnQ1_4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion1: TfrmQuestion1;

implementation

{$R *.dfm}

procedure TfrmQuestion1.btnQ1_1Click(Sender: TObject);
var
```



```
  swaiter, sname, ssurname, sref : string;
  ipos, icode : integer;
begin
  swaiter := edtwaiter.text;
  ipos := pos(' ',swaiter);
  sname := copy(swaiter, 1, ipos - 1);
  delete(swaiter, 1, ipos);
  ssurname := swaiter;
  sref := sname[1]+'. '+ssurname;
  icode := randomrange(10000,100000);
  edtref.Text := sref;
  edtcode.text := inttostr(icode);
end;

procedure TfrmQuestion1.btnQ1_2Click(Sender: TObject);
var
  sid, sdate : string;
  k, iage: Integer;
  bletter : boolean;
begin
  bletter := false;
  sid := edtid.text;
  if length(sid) = 13 then
    begin
      for k := 1 to 13 do
        begin
        if not (strtoint(sid[k]) in [0..9]) then
           bletter := true;
        end;
    if bletter = false then
    begin
      sdate := datetostr(date());
      if sid[1] = '0' then
        iage := strtoint(copy(sdate,1,4)) - (2000+(strtoint(copy(sid,1,2))))
      else
        iage := strtoint(copy(sdate,1,4)) - (1900+(strtoint(copy(sid,1,2))));
      case iage of
      0..15 : redQ1_2.Lines.Add('Too young to work');
      16..18 : redQ1_2.Lines.Add('Weekday x 1'+#13+'Weekend');
      else
      redQ1_2.lines.Add('Weekdays x 5'+#13+'Saturday OR Sunday');
      end;
    end;
    end
    else
    redQ1_2.Lines.Add('Incorrect ID');
end;

procedure TfrmQuestion1.btnQ1_3Click(Sender: TObject);
var
  ihours, iover : integer;
  rwage : real;
```

```
begin
  ihours := sedHours.Value;
  if cbxOvertime.checked then
    iover := strtoint(Inputbox('','',''))
  else
    iover := 0;
  rwage := ihours * 18.25 + 1.5*iover*18.25;
  pnlOutput.Caption := floattostrf(rwage,ffcurrency,10,2);
  pnlOutput.Color := clSkyBlue;
end;

procedure TfrmQuestion1.btnQ1_4Click(Sender: TObject);
var
  sitem, ssize : string;
begin
  sitem := cmbItem.text;
  case rgpSizes.itemindex of
    0 : ssize := 'XS';
    1 : ssize := 'S';
    2 : ssize := 'M';
    3 : ssize := 'L';
    4 : ssize := 'XL';
  end;
  Showmessage('Order:'+#13+sitem +'('+ssize+')'+#13+'Peak Cap');
  end;

end.
```


QUESTION 2

```
unit Question2_u;

interface

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

type
  TForm1 = class(TForm)
    tblTables: TADOTable;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    redoutput: TRichEdit;
    Button4: TButton;
    BitBtn1: TBitBtn;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
var
  rtotal : real;
begin
  tblTables.open;
  tblTables.first;
  while not tblTables.eof do
  begin
    rtotal := rtotal + tblTables['TableAmountPaid'];
    tblTables.next;
  end;
  redoutput.Lines.Add('Total Amount: '+floattostrf(rtotal,ffcurrency,10,2));
end;
```



```
procedure TForm1.Button2Click(Sender: TObject);
begin
  tblTables.Sort := 'waiterid';
end;

procedure TForm1.Button3Click(Sender: TObject);
var
  swaiter : string;
  itables : integer;
begin
  swaiter := inputbox('','','');
  tblTables.Open;
  tblTables.First;
  itables := 0;
  Redoutput.lines.add('WaiterID: '+swaiter+#13);
  while not tblTables.eof do
  begin
    if swaiter = tblTables['waiterID'] then
    begin
      redoutput.lines.add(tblTables['TableID']);
      itables := itables + 1;
    end;
    tblTables.Next;
  end;
  redoutput.lines.add(#13+ 'Total Number of Tables: '+inttostr(itables));
end;

procedure TForm1.Button4Click(Sender: TObject);
begin
  tblTables.Insert;
  tblTables['TableID'] := 26;
  tblTables['WaiterID'] := 10;
  tblTables['TableGuests'] := 5;
  tblTables.Post;
end;

end.
```


QUESTION 3

```
unit Question3_u;

interface

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

type
  TfrmQuestion3 = class(TForm)
    redoutput: TRichEdit;
    btnList: TButton;
    procedure btnListClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion3: TfrmQuestion3;

implementation

{$R *.dfm}

procedure TfrmQuestion3.btnListClick(Sender: TObject);
var
  icount, ipos : integer;
  myfile, myfile2 : textfile;
  rmin, rmax, rneed, rcurrent : real;
  sitem, sunit, soneline : string;
begin
  Assignfile(myfile2, 'ShoppingList.txt');
  Rewrite(myfile2);

  if fileexists('stocklist.txt') <> true then
  begin
    Showmessage('File does not exist');
    Exit;
  end;
  Assignfile(myfile,'stocklist.txt');
  Reset(myfile);
  redoutput.lines.Add('SHOPPING LIST'+#13);
  while not eof(myfile) do
  begin
    readln(myfile, soneline);
    ipos := pos('#',soneline);
    sitem := copy(soneline,1,ipos-1);
    delete(soneline,1,ipos);
```



```
    ipos := pos('#',soneline);
    rcurrent := strtofloat(copy(soneline,1,ipos-1));
    delete(soneline,1,ipos);
    ipos := pos('#',soneline);
    sunit := copy(soneline,1,ipos-1);
    delete(soneline,1,ipos);
    ipos := pos('#',soneline);
    rmin := strtofloat(copy(soneline,1,ipos-1));
    delete(soneline,1,ipos);
    rmax := strtofloat(soneline);
    if rcurrent < rmin then
      begin
        rneed := rmax - rcurrent;
        if frac(rneed) > 0 then
          rneed := ceil(rneed);
        if (sunit = 'kg') or (sunit = 'bottles') then
          begin
            redoutput.lines.add(floattostr(rneed)+' ' + sunit + ' ' + sitem);
            writeln(myfile2,floattostr(rneed)+' '+sunit+' ' +sitem);
          end
        else
          begin
          redoutput.lines.add(floattostr(rneed)+' ' + sitem);
          writeln(myfile2,floattostr(rneed)+' ' + sitem);
          end;
      end;
  end;
  redoutput.Lines.Add(#13+'Shopping list saved to file');
  Closefile(myfile);
  Closefile(myfile2);

end;

end.
```


QUESTION 4

```
unit Question4_u;

interface

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

type
  TfrmQuestion4 = class(TForm)
    btnItems: TButton;
    redoutput: TRichEdit;
    BitBtn1: TBitBtn;
    btnFoodPrice: TButton;
    btnSearch: TButton;
    procedure btnItemsClick(Sender: TObject);
    procedure btnFoodPriceClick(Sender: TObject);
    procedure btnSearchClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
      procedure Sort;
    { Public declarations }
  end;

var
  frmQuestion4: TfrmQuestion4;
    arritem : array[1..22] of string;
  arrcost : array[1..22] of real = (23.20, 24.72, 29.36, 22.23, 19.96, 20.30, 25.21, 18.55, 5.30,
4.53, 4.22, 4.72, 5.18, 3.95, 33.20, 37.35, 14.11, 17.19, 12.29, 9.53, 33.91, 3.95);
  arrfood : array[1..50] of string;
  arrdrink : array[1..50] of string;
  arrfcost : array[1..50] of real;
  arrdcost : array[1..50] of real;
  ifood, idrink : integer;
implementation

{$R *.dfm}
function Found(sitem : string) : boolean;
var
  bfound : boolean;
  k: Integer;
begin
  bfound := false;
  for k := 1 to 22 do
  begin
    if pos(uppercase(sitem), uppercase(arritem[k])) > 0 then
      bfound := true;
  end;
```



```
  result := bfound;
end;
procedure TfrmQuestion4.Sort;
var
  l,k: Integer;
  stemp : string;
  rtemp : real;
begin
  for k := 1 to 21 do
    for l := k+1 to 22 do
      if arritem[k] > arritem[l] then
      begin
        stemp := arritem[k];
        arritem[k] := arritem[l];
        arritem[l] := stemp;
        rtemp := arrcost[k];
        arrcost[k] := arrcost[l];
        arrcost[l] := rtemp;
      end;
end;

procedure TfrmQuestion4.btnFoodPriceClick(Sender: TObject);
var
  k: Integer;
  rprice : real;
begin
  redoutput.Clear;
  redoutput.Paragraph.TabCount := 1;
  redoutput.Paragraph.Tab[0] := 150;
  redoutput.Lines.Add('FOOD ITEM'+#9+'PRICE');
  for k := 1 to ifood do
  begin
    rprice := ceil(arrfcost[k] * 1.60);
    redoutput.Lines.Add(arrfood[k] + #9 + floattostrf(rprice, ffcurrency,10,0));
  end;
end;

procedure TfrmQuestion4.btnItemsClick(Sender: TObject);
var
  k, ipos : integer;
begin
   Sort;
  for k  := 1 to 22 do
    begin
      ipos := pos('#',arritem[k]);
      if arritem[k][ipos+1] = 'F' then
        begin
          inc(ifood);
          arrfood[ifood] := copy(arritem[k], 1, ipos-1);
          arrfcost[ifood] := arrcost[k];
        end
      else
```

```
        begin
          inc(idrink);
          arrdrink[idrink] := copy(arritem[k], 1, ipos-1);
          arrdcost[idrink] := arrcost[k];
        end;
    end;

    redoutput.Lines.Add('FOOD ITEMS' +#13);
    for k  := 1 to ifood do
      redoutput.Lines.Add(arrfood[k]);

    redoutput.Lines.Add(#13+'DRINK ITEMS'+#13);
    for k := 1 to idrink do
      redoutput.Lines.Add(arrdrink[k]);
end;

procedure TfrmQuestion4.btnSearchClick(Sender: TObject);
 var
  sitem : string;
begin
   sitem := inputbox('','','');
  if found(sitem) = true then
    Showmessage('Item is on the menu')
  else
    Showmessage('Item is not on the menu');
end;

procedure TfrmQuestion4.FormCreate(Sender: TObject);
begin
   ifood := 0;
  idrink := 0;

  {***GIVEN CODE***DO NOT CHANGE IT***}
  arrItem[1] := 'Hamburger and Chips#F';
  arrItem[2] := 'Cheese Burger and Chips#F';
  arrItem[3] := 'Bacon Burger and Chips#F';
  arrItem[4] := 'Veggie Burger and Chips#F';
  arrItem[5] := 'Hamburger#F';
  arrItem[6] := 'Cheese Burger#F';
  arrItem[7] := 'Bacon Burger#F';
  arrItem[8] := 'Veggie Burger#F';
  arrItem[9] := 'Chips - Large Plate#F';
  arrItem[10] := 'Soda#D';
  arrItem[11] := 'Fresh Juice - Orange#D';
  arrItem[12] := 'Fresh Juice - Apple#D';
  arrItem[13] := 'Homemade Lemonade#D';
  arrItem[14] := 'Water - Sparkling (500ml)#D';
  arrItem[15] := 'Bacon and Egg Burger with Chips#F';
  arrItem[16] := 'Giant Burger with Chips#F';
  arrItem[17] := 'Salad - Greek#F';
  arrItem[18] := 'Salad - Chicken#F';
  arrItem[19] := 'Salad - Potato#F';
```



```
  arrItem[20] := 'Salad - Pasta#F';

arrItem[21] := 'Giant Burger#F';
  arrItem[22] := 'Water - Still (500ml)#D';
end;

end.
```