Province of the
EASTERN CAPE
EDUCATION

NATIONAL SENIOR CERTIFICATE

GRADE 12

SEPTEMBER 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**
/49	**/55**	**/46**	**/150**

	QUESTION 1	Max. MARKS	MARKS ACHIEVED
1.1	**BUTTON: [REGISTER]** Get the name and surname as one input✓ Extract the surname✓ Get the contact number✓ Get the age✓ Get the option from cmbrep (Student/Teacher/Business)✓ Check if Contact Number is valid✓ Initialise flag variable to true✓ Loop for the length of the number✓ If a character is not a number✓ Change flag variable to false✓ Test if the first number of contact number is a '0',✓ the length is 10✓ and that all characters are numbers✓ Display a message 'Information Captured'✓ Show the OnceOffPassword button✓ Else ✓ Display a message indicating that Contact Number is invalid ✓ Clear edtContactNo✓ Place the cursor in edtContactNo (setfocus)✓	**19**	
1.2	**BUTTON: [ONCE-OFF PASSWORD]** Initialise an empty string Loop three times ✓ Randomise a number 1 – 26✓ Extract the character from arralphabet✓ and join to string✓ Add a random number (100 – 999) to the string✓ Add the first letter of the combobox to the string✓ Display the compiled string on the panel✓ Show the panel✓ If combobox = Student ✓or Teacher then ✓ Display panel3✓ else✓ Display a message✓	**25**	

	If the first option is selected (School) ✓then the filename will be School.txt✓ If the second✓ or third ✓option is selected, then the filename will be Tertiary.txt✓ Test if file exist✓ If the file does not exist, display a message✓ and exit Assign file✓ Reset✓ Loop through the file✓ Read from file✓ Display contents in the memOutput component✓ Close the file		
1.3	**BUTTON: [REGISTRATION SUMMARY]** Display the heading ('REGISTRATION INFORMATION') ✓ Display the initial and surname✓ Display the representing body✓ and the '('age')' ✓ Display a message containing the contact number✓	**5**	
		49	

QUESTION 2		Max. MARKS	MARKS ACHIEVED
2.1.1	**Constructor:** Correct method heading (two integer parameters and one string parameter) ✓ Assigning the integer parameters to attributes✓ Assign the string parameters to attributes✓	**3**	
2.1.2	**Procedure CalculateArea** Correct heading✓ Calculating the area and round up ceil✓ (TABLE * ftables + CHAIR * fchairs) ✓; Call ExhibitArea method✓	**4**	

2.1.3	fun**ction ExhibitArea: string;**
	
	Correct heading✓
	
	if (farea > 0) and (farea <= 10) then✓
	if fplug = 'No' then✓
	'Exhibition Area A'✓
	else✓
	'Exhibition Area B' ✓
	
	if (farea > 10) and (farea <= 20) then✓
	'Exhibition Area B' ✓
	
	if (farea > 20) and (farea <= 30) then✓
	'Exhibition Area C' ✓
	
	if (farea > 30) then✓
	fvenue:= 'Exhibition Area D'✓
	if fplug = 'Yes' then✓
	fvenue:= 'Exhibition Area C'✓
	Calculate the difference between area and 30✓
	while there is a difference✓
	decrease tables by 1✓
	decrease chairs by 3✓
	call the calculatearea method✓
	Calculate the difference between area and 30✓
	
	Return fvenue✓
		21	

2.1.4	**function tostring: string;** Correct method heading ✓ Compile a string containing the following: heading✓ number of tables and number of chairs✓, converted✓ if plug point is needed✓ total area required as a string✓ the venue✓	**7**	

2.1.5	**function TStalls.GetVenue: string;** Correct function heading✓ return the venue✓	**2**	
2.2	Add the class unit to uses✓ Declare the object variable✓ Get the company name✓ Get the amount of tables✓ Get the amount of chairs✓ If checkbox is selected✓ Assign 'Yes' to plug point variable✓ Else Assign 'No' to plug point variable✓ Instantiate the object with all three arguments✓ Call the CalculateArea method✓ Use the toString method to display✓ Assign the file ✓ If the file does not exists✓ Rewrite✓ Else Append✓ Write company name✓ + '-' + and venue to file✓ Close the file✓	**18**	
		55	

QUESTION 3		MAX. MARKS	MARKS ACHIEVED
3.1	**FORMCREATE** Declare ar2Donations: class scope and [1..6,1..4] ✓ Set the tab count = 5✓ Correctly set up the tabs✓ with given intervals✓ Randomly fill ar2Donations: Outer loop (1 to 6) ✓ Inner loop (1 to 4) ✓ Correctly randomise 1000 to 10000 ✓ and assign to ar2Donations✓ Initialise the total variable✓	**9**	
3.2	**DISPLAY method** Method heading✓ Clear both richedit components✓ Display the heading (DONATIONS)✓ Display the dates✓ Outer loop✓ Get the Charity✓ and assign to the string variable Inner loop ✓ Compile a string✓ making use of the values from ar2Donations✓ (currency✓) Display the compiled string✓ All information is in neat columns (#9)✓	**12**	
3.3	**DONATE button** Get the correct column✓ making use of the combobox✓ Get the correct row from the spinedit✓ Get the amount as a number✓ Add the amount✓ to the current amount in ar2Donations✓ Call the Display method✓ **Calculate the total amount for each charity:** Initialise the total variable✓ and increment it✓ **Calculate the total amount for ALL charities:** Outer loop✓ Inner loop✓ Increment the total variable✓ using ar2Donations✓ Display the total amount: label✓ and amount✓ as currency	**25**	

	Use any method to **correctly** calculate the ranking of the charities. (7 marks) *Possible Solution:* *Charities are stored in arrCharity*✓ *Totals are stored in arrTotals* ✓ *Sorting code:* *Outer Loop*✓ *Inner Loop*✓ *If arrtotal[k] < arrtotal[l]* ✓ *Swap arrtotals*✓*and arrcharity*✓ Loop 6 times✓ Display the ranking✓ as well as the charity number✓		
		46	

SAMPLE SOLUTIONS

QUESTION 1

```
unit Question1_u;

interface

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

type
  TForm1 = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    edtName: TEdit;
    edtContactNo: TEdit;
    lblName: TLabel;
    lblContactNumber: TLabel;
    edtAge: TEdit;
    lblAge: TLabel;
    cmbRep: TComboBox;
    btnRegister: TButton;
    btnOnceOffPassword: TButton;
    pnlOnceOffPassword: TPanel;
    pnlRep: TPanel;
    Panel4: TPanel;
    BitBtn1: TBitBtn;
    rgpRep: TRadioGroup;
    memOutput: TMemo;
    redSummary: TRichEdit;
    btnSummary: TButton;
    procedure btnRegisterClick(Sender: TObject);
    procedure btnOnceOffPasswordClick(Sender: TObject);
    procedure rgpRepClick(Sender: TObject);
    procedure btnSummaryClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  srep : string;
  ssurname, sinput : string;
  iage : integer;
  scontact : string;
implementation

{$R *.dfm}
```



```
procedure TForm1.btnOnceOffPasswordClick(Sender: TObject);
const
  arralphabet : array[1..26] of char =
('A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z');
var
  k, iran : Integer;
  sonceoff : string;
begin
  for k := 1 to 3 do
  begin
    iran := randomrange(1,27);
    sonceoff := sonceoff + arralphabet[iran];
  end;
  sonceoff := sonceoff + inttostr(randomrange(100,1000));
  sonceoff := sonceoff + srep[1];
  pnlOnceOffPassword.Caption := sonceoff;
  pnlOnceOffPassword.Show;

  if (cmbRep.Text[1] = 'S') or (cmbRep.Text[1] = 'T') then
    pnlRep.Show
  else
    Showmessage('Full programme available on the Facebook page');

end;

procedure TForm1.btnRegisterClick(Sender: TObject);
var

  ipos : integer;
  k: Integer;
  bcontact : boolean;
begin

  sinput := edtname.Text;
  ipos := pos(' ',sinput);
  ssurname := copy(sinput, ipos+1);
  scontact := edtContactNo.Text;
  iage := strtoint(edtAge.text);
  srep := cmbRep.text;
  bcontact := true;
  for k := 1 to length(scontact) do
  begin
    if not (scontact[k] in ['0'..'9']) then
      bcontact := false;
  end;
  if (scontact[1] = '0') and (length(scontact) = 10) and (bcontact = true) then
    begin
      Showmessage('Information Captured');
      btnOnceOffPassword.Show;
    end
  else
    begin
```

```
      Showmessage('Invalid Contact Number');
      edtContactNo.Clear;
      edtContactNo.SetFocus;
    end;

end;

procedure TForm1.btnSummaryClick(Sender: TObject);
begin
   redSummary.Lines.Add('REGISTRATION INFORMATION'+#13);
   redSummary.lines.Add(sinput[1]+ ' '+ ssurname);
   redSummary.lines.add(srep + '('+inttostr(iage)+')');
   redSummary.Lines.Add('A message will be sent to '+scontact+', please reply with your
once-off password');
end;

procedure TForm1.rgpRepClick(Sender: TObject);
var
  sfilename, soneline : string;
  myfile : textfile;
begin
  case rgpRep.itemindex of
    0 : sfilename := 'School';
    1,2 : sfilename := 'Tertiary';
  end;

  if fileexists(sfilename+'.txt') <> true then
  begin
    Showmessage('File does not exist');
    Exit;
  end;
  Assignfile(myfile, sfilename+'.txt');
  Reset(myfile);

  while not eof(myfile) do
  begin
    readln(myfile, soneline);
    memOutput.lines.add(soneline);
  end;

  Closefile(myfile);
end;

end.
```


QUESTION 2

Class Unit:

```
unit clsVenues;

interface

uses
  Math, SysUtils, Dialogs;

type
  TStalls = class
    private
      ftables : integer;
      fchairs : integer;
      fplug : string;
      farea : integer;
      fvenue : string;
    public
      constructor create (itables, ichairs : integer; splug : string);
      procedure CalculateArea;
      function tostring : string;
      function ExhibitArea : string;
      function GetVenue : string;
  end;

implementation

{ TStalls }

procedure TStalls.CalculateArea;
const
  TABLE = 1.2;
  CHAIR = 0.64;
begin
  farea := ceil(TABLE * ftables + CHAIR * fchairs);
  ExhibitArea;
end;

constructor TStalls.create(itables, ichairs: integer; splug: string);
begin
   ftables := itables;
   fchairs := ichairs;
   fplug := splug;
end;

function TStalls.ExhibitArea: string;
var
  idiff : integer;
begin
  if (farea > 0) and (farea <= 10) then
  begin
```

```
    if fplug = 'No' then
      fvenue := 'Exhibition Area A'
    else
      fvenue := 'Exhibition Area B';
  end;

  if (farea > 10) and (farea <= 20) then
    fvenue := 'Exhibition Area B';

  if (farea > 20) and (farea <= 30) then
    fvenue := 'Exhibition Area C';

  if (farea > 30) then
  begin
    fvenue := 'Exhibition Area D';
    if fplug = 'Yes' then
     begin
      fvenue := 'Exhibition Area C';
      idiff := farea - 30;
      while (idiff > 0) do
      begin
        dec(ftables);
        dec(fchairs,3);
        calculatearea;
        idiff := farea - 30;
      end;
     end;
  end;
  result := fvenue;
end;

function TStalls.GetVenue: string;
begin
  result := fvenue;
end;

function TStalls.tostring: string;
var
  soutput : string;
begin
   soutput := 'DETAILS' + #13;
   soutput := soutput + 'Number of Tables' + #9 + inttostr(ftables)+#13;
   soutput := soutput + 'Number of Chairs' + #9 + inttostr(fchairs)+#13;
   soutput := soutput + 'Plugpoint needed' + #9 + fplug+ #13;
   soutput := soutput + 'Total Area required' + #9 + inttostr(farea)+ ' square metres'+#13+#13;
   soutput := soutput + fvenue;
   result := soutput;

end;

end.
```


Main Unit:

```
unit Question2_u;

interface

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

type
  TForm1 = class(TForm)
    btnSubmit: TButton;
    Label1: TLabel;
    edtTables: TEdit;
    edtChairs: TEdit;
    cbxPlug: TCheckBox;
    lblTables: TLabel;
    lblChairs: TLabel;
    redOutput: TRichEdit;
    edtCompany: TEdit;
    lblCompany: TLabel;
    procedure btnSubmitClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  objStalls : TStalls;

implementation

{$R *.dfm}

procedure TForm1.btnSubmitClick(Sender: TObject);
var
  itables, ichairs : integer;
  splug, scompany : string;
  myfile : textfile;

begin
   redOutput.Clear;
   scompany := edtCompany.Text;
   itables := strtoint(edtTables.text);
   ichairs := strtoint(edtChairs.text);
   if cbxPlug.checked then
     splug := 'Yes'
   else
     splug := 'No';
```



```
  objStalls := TStalls.create(itables,ichairs,splug);
  objStalls.CalculateArea;
  redOutput.Lines.Add(objStalls.tostring);

  Assignfile(myfile,'Venues.txt');
  if fileexists('Venues.txt') <> true then
    Rewrite(myfile)
  else
    Append(myfile);

  writeln(myfile,scompany +'-'+objStalls.GetVenue);
  Closefile(myfile);
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  //PROVIDED CODE
  redOutput.Paragraph.TabCount:=1;
  redOutput.Paragraph.Tab[0] := 150;
end;

end.
```


QUESTION 3

```
unit Question3_u;

interface

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

type
  TfrmQuestion3 = class(TForm)
    pnlHeading: TPanel;
    cbxDate: TComboBox;
    lblDate: TLabel;
    lblCharity: TLabel;
    sedCharity: TSpinEdit;
    edtAmount: TEdit;
    lblAmount: TLabel;
    btnDonate: TButton;
    redOutput: TRichEdit;
    redStats: TRichEdit;
    procedure btnDonateClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    procedure Display;
  end;

var
  frmQuestion3: TfrmQuestion3;
  ar2Donations : array[1..6,1..4] of real;
  rtotal : real;
  arrtotal : array[1..6] of real;
  arrcharity : array[1..6] of string;

implementation

{$R *.dfm}

procedure TfrmQuestion3.btnDonateClick(Sender: TObject);
var
  irow, icol, k, l : integer;
  ramount, rtemp, rtotal : real;
  stemp : string;
begin
   case cbxDate.ItemIndex of
     0 : icol := 1;
     1 : icol := 2;
     2 : icol := 3;
     3 : icol := 4;
   end;
```



```
  irow := sedCharity.Value;

  ramount := strtofloat(edtAmount.Text);
  ar2donations[irow,icol] := ar2donations[irow,icol] + ramount;

  Display;

  for irow  := 1 to 6 do
    begin
      arrtotal[irow] := 0;
      arrcharity[irow] := 'Charity '+inttostr(irow);
      for icol := 1 to 4 do
        begin
          rtotal := rtotal + ar2donations[irow,icol];
          arrtotal[irow] := arrtotal[irow] + ar2donations[irow,icol];
        end;
    end;

  redstats.Lines.Add('Total Amount Raised'+#13+floattostrf(rtotal,ffcurrency,16,2)+#13);

  for k := 1 to 5 do
    for l := k + 1 to 6 do
      if arrtotal[k] < arrtotal[l] then
        begin
          rtemp := arrtotal[k];
          arrtotal[k] := arrtotal[l];
          arrtotal[l] := rtemp;
          stemp := arrcharity[k];
          arrcharity[k] := arrcharity[l];
          arrcharity[l] := stemp;
        end;

  for k := 1 to 6 do
    redStats.Lines.Add(inttostr(k) + ' - ' + arrcharity[k]);

end;

procedure TfrmQuestion3.Display;
var
  irow, icol : Integer;
  soutput : string;
begin

  redoutput.Clear;
  redstats.Clear;
  redoutput.lines.add('DONATIONS');
  redoutput.Lines.Add('');
  redoutput.Lines.Add(#9+'18/11/2018'+#9+'19/11/2018'+#9+'20/11/2018'+#9+'21/11/2018');
  for irow := 1 to 6 do
  begin
    soutput := 'Charity ' + inttostr(irow)+#9;
    for icol := 1 to 4 do
```

```
     begin
      soutput := soutput + floattostrf(ar2Donations[irow,icol],ffcurrency,10,2)+ #9;
      end;
    redoutput.Lines.add(soutput);
  end;
end;

procedure TfrmQuestion3.FormCreate(Sender: TObject);
var
  icol, irow : Integer;
begin
  redoutput.Paragraph.TabCount := 5;
  redoutput.Paragraph.Tab[0] := 70;
  redoutput.Paragraph.Tab[1] := 100;
  redoutput.Paragraph.Tab[2] := 130;
  redoutput.paragraph.Tab[3] := 160;
  redoutput.Paragraph.Tab[4] := 190;

  rtotal := 0;

  for irow := 1 to 6 do
    for icol := 1 to 4 do
       ar2donations[irow,icol] := randomrange(1000,10001);

end;

end.
```