Province of the
EASTERN CAPE
EDUCATION

NATIONAL SENIOR CERTIFICATE

GRADE 12

SEPTEMBER 2017

INFORMATION TECHNOLOGY P1 MARKING GUIDELINE

MARKS: 150

This marking guideline consists of 18 pages.

QUESTION 1		MAX MARKS	MARKS ACHIEVED

1.1	**FORM CREATE: DYNAMIC COMPONENT**
	
	Imgmedical := Timage.create(frmQuestion1); ✓
	Imgmedical.parent : frmQuestion1; ✓
	Imgmedical.left := 500 AND imgmedical.top := 15; ✓
	Imgmedical.height := 90 AND imgmedical.width := 120; ✓
	Imgmedical.stretch := true; ✓
	Imgmedical.picture.loadfromfile('medical.jpg'); ✓
	Imgmedical.visible := true; ✓
		7	

1.2	**GENERAL INFO BUTTON** Get title from combobox✓ Get name and surname from edit box✓ Get ID number from edit box✓ Test if ID's length is 13 digits✓ If length is 13 digits✓ Check for only numbers✓ If only numbers ✓ Randomize a number between 100 and 999 (both included)✓ Extract the initial✓ and surname✓ Compile a file name using the first three characters✓of the surname and the random number✓ Display a welcome message including the title, initial, surname and the file name✓ Display a message if ID's length is not 13 digits✓	**14**	
1.3	**MEDICAL AID INFO BUTTON** Check if Main Member checkbox is selected✓ If the Subsidy checkbox is selected then ✓ Subsidy must be 'Yes'✓ Else Subsidy must be 'No'✓ Get the number of dependents from the radiogroup✓ Get the medical aid name from the listbox✓ Writing to file: text file name must be the file name created in Question 1.2✓ Assignfile✓ Rewrite✓ Write the file name, number of dependents, medical aid name and whether they have a subsidy to the text file✓ Must be on separate lines✓ Close the file✓ Display a message indicating that the file was written✓	**13**	

1.4	**UPDATE INFO BUTTON** Get the system date✓ Extract the year of date✓ and increase it by one year✓ Create the new date✓ Display a message including the date when details must be updated✓	**5**	
		39	

QUESTION 2		MAXIMUM MARKS	MARKS ACHIEVED
2.1.1	**CONSTRUCTOR CREATE** Constructor heading with correct paramters✓ Assign parameter values to attributes: fdoctor, fdate, ffollowup (string attributes) ✓ fmedaid (boolean attribute) ✓ initialising fmed and fpayment to 0 ✓	**4**	
2.1.2	**FUNCTION FOLLOWUPDATE** Function Heading If ffollowup attribute is 'Yes' ✓ Add 7 days to the date✓ If days more than 30✓ Change the days (-30) and add 1 to month✓ Compile the follow-up date✓ Else (if days less than 30) ✓ Compile the follow-up date ✓ Else (if followup attribute is 'No') ✓ result must be 'No Follow-Up Appointment Needed';✓	**9**	
2.1.3	**SETPAYMENT METHOD** Correct method with parameter ✓ Increase fpayment by parameter✓	**2**	
2.1.4	**SETMED METHOD** Correct method with parameter ✓ Increase fmed by parameter✓	**2**	
2.1.5	**COMPILESTRING** Correct method definition with string return type✓ If ffollowup is 'Yes'✓ Return doctor name attribute as well as followupdate method ✓ if ffollowup is 'No' return followupdate method✓ ✓	**5**	

2.2.1	**Capture Info Button** if/case statement✓ get the doctor's name✓ increase the counter for the particular doctor✓ get the system date✓ If checkbox is checked✓ Med aid is true✓ Else Med aid is false✓ Get follow up ('Yes' or 'No') from editbox✓ Create the object✓ with correct parameters(sdoctor, sdate, sfollowup, bmedaid) ✓ Call the compilestring method✓ Randomise a number between 300 and 400✓ randomrange(300,401) or random(300) + 101 If bmed is false then Call SetPayment method with parameter✓ Add amount to total ✓ Display using getpayment✓ – currency and two decimal places✓ else✓ Call SetMed method with parameter✓ Display message✓ – charged to medical aid Add amount to total✓	**20**	
2.2.2	Display total amount of cash for the day✓ Display total amount charged to medical aid✓ Amounts formatted as currency and two decimal places✓ Display number of patients each doctor has seen✓ All information on new lines✓	**5**	
		47	

QUESTION 3		MAX MARKS	MARKS ACHIEVED
3.1	**FORMCREATE** Row Headings (appointment times)✓ Column Headings (doctor names)✓ Declare 2d-array with class scope (ar2appointments) ✓ Test if file exists✓ Display a message if file does not exists✓ Assign and Reset the file✓✓ Read the first line in text file (doctors' names) ✓ Loop 10 times✓ (rows) Read next line in text file✓ Loop 4 times✓ (columns) Extract the patient's name✓ Extracting the last name correctly✓ If it is a patient's name (not -)✓ Assign to 2d array✓ correct row and column✓ Display the contents of the 2d-array by calling a display method✓ Display method: Outer loop✓ Inner loop✓ Display 2d-array[row,col]✓ in stringgrid[col,row]✓	**21**	
3.2.1	**INSERT METHOD** Method receiving name and column as parameters✓ Randomise a number 1 to 10 (both included) to represent the time slot✓ If column is 1 to 4 then refers to specific doctor✓ Conditional loop✓ If that particular doctor has an opening ✓ Assign name to opening in 2d array✓ Else✓ Randomize a new timeslot✓ Else (if column is 5 – any doctor)✓ Conditional loop✓ Randomize between 1 to 4 (both included) to select a doctor✓ If that particular doctor has an opening✓ Assign name to opening in 2d array✓ Else✓ Increase counter to select new timeslot for that doctor✓	**17**	

	Display a message if an appointment has been made✓ Else Display a message that an appointment has not been made✓		
3.2.2	**ADD APPOINTMENT BUTTON** Get the doctor's index✓ Get the patients name and surname✓ Call the Insert method✓ Call the display method✓ (Accept any alternative display code, as learners will be penalised in 4.1)	**4**	
3.3	**CHANGE APPOINTMENT BUTTON** Get the patient's name and surname✓ Outer loop✓ Inner loop✓ If patient is found✓ Call the Insert method with the patient's name and Doctor's index as parameter✓ Delete the original appointment in the 2d-array✓ Display the updated array✓	**7**	
3.4	**SEARCH BUTTON** Get the patient's name✓ Outer loop✓ Inner loop✓ If name is found in 2d array✓ Display a message including doctor and time✓	**5**	
3.5	**DOCTOR WALK-IN BUTTON** Initialise the minimum variable✓ Outer loop✓ Initialise the counter✓ Inner loop✓ If 2d-array[row,col] is empty✓ Increase counter✓ If counter < min✓ Assign counter to min✓ Assign the specific column (doctor) to a variable✓ Display a message to indicate which doctor will see patients without appointments✓	**10**	
		64	

SAMPLE SOLUTIONS

Question 1

```
unit Question1U;

interface

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

type
  TfrmQuestion1 = class(TForm)
    btnGeneralInfo: TButton;
    pnlHeading: TPanel;
    cmbTitle: TComboBox;
    lblTitle: TLabel;
    lblNameSurname: TLabel;
    edtNameSurname: TEdit;
    edtIDNumber: TEdit;
    lblIDNumber: TLabel;
    lbxMedicalAids: TListBox;
    pnlGeneral: TPanel;
    pnlMedical: TPanel;
    cbxMainMember: TCheckBox;
    rgpDependents: TRadioGroup;
    btnMedicalInfo: TButton;
    cbxSubsidy: TCheckBox;
    BitBtn1: TBitBtn;
    pnlUpdate: TPanel;
    btnUpdateInfo: TButton;
    memupdate: TMemo;
    procedure btnGeneralInfoClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure btnMedicalInfoClick(Sender: TObject);
    procedure btnUpdateInfoClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion1: TfrmQuestion1;
  imgMedical : TImage;
  sfilename : string;

implementation

{$R *.dfm}
```

```
procedure TfrmQuestion1.btnGeneralInfoClick(Sender: TObject);
var
  iran : integer;
  stitle, sfullname, sinitial, sid : string;
  k: Integer;
  bvalid : boolean;
begin
  bvalid := true;
  stitle := cmbTitle.Text;
  sfullname := edtNameSurname.Text;
  sinitial := sfullname[1];
  delete(sfullname,1,pos(' ',sfullname));
  sid := edtIDNumber.Text;

  if (length(sid) = 13) then
    begin
      for k := 1 to 13 do
        begin
        if not(sid[k] in ['0'..'9']) then
          bvalid := false;
        end;
      if bvalid = true then
      begin
      iran := (randomrange(100,1000));
      sfilename := copy(sfullname,1,3) + inttostr(iran);
      Showmessage('Welcome '+stitle+ ' '+ sinitial +' ' + sfullname+'.'+#13+'Your File
Number is: '+sfilename);
    end;
    end
  else
    Showmessage('ID incorrect');
end;

procedure TfrmQuestion1.btnMedicalInfoClick(Sender: TObject);
var
  smain, ssubsidy, smedicalaid, soneline : string;
  idependents : integer;
  myfile : textfile;
begin
  case rgpdependents.ItemIndex of
  -1 : idependents := 0;
  0 : idependents := 1;
  1 : idependents := 2;
  2 : idependents := 3;
  3 : idependents := 4;
  end;
  if cbxMainMember.Checked  then
    smain := '(Main Member)';
  if cbxSubsidy.checked then
    ssubsidy := 'Yes'
  else
    ssubsidy := 'No';
  smedicalaid := lbxMedicalAids.items[lbxmedicalaids.ItemIndex];
  Assignfile(myfile,sfilename+'.txt');
  Rewrite(myfile);
  writeln(myfile, sfilename + ':'+smain);
  writeln(myfile,'Number of dependents: '+inttostr(idependents));
  writeln(myfile,'Medical Aid: '+smedicalaid);
  writeln(myfile,'Subsidy: '+ssubsidy);
  writeln(myfile,'-------------------------------------------------------');
  Closefile(myfile);
  Showmessage('File was successfully written.');

end;

procedure TfrmQuestion1.btnUpdateInfoClick(Sender: TObject);
var
  sdatenow, syear, supdatedate : string;
begin
  sdatenow := datetostr(Date());
  syear := copy(sdatenow,1,4);
  delete(sdatenow,1,4);
  supdatedate := inttostr(strtoint(syear) + 1) + sdatenow;
  memupdate.lines.add('Please update your information at your next visit or by
'+supdatedate);

end;

procedure TfrmQuestion1.FormCreate(Sender: TObject);
begin
  imgmedical := Timage.Create(frmQuestion1);
  imgmedical.parent := frmQuestion1;
  imgmedical.Left := 500;
  imgmedical.Top := 15;
  imgmedical.Height := 90;
  imgmedical.Width := 120;
  imgmedical.Stretch := true;
  imgmedical.picture.loadfromfile('medical.jpg');
  imgmedical.Visible := true;
end;


end.
```

QUESTION 2

```
unit clsRecords;

interface

uses sysutils;

type
  Tobjmed = class
    private
      fdoctor : string;
      fdate : string;
      fmedaid : boolean;
      ffollowup : string;
      fpayment : real;
      fmed : real;
    public
      constructor create (sdoctor,sdate,sfollowup : string; bmedaid: boolean);
      function FollowUpDate: string;
      function compilestring : string;
      function getpayment : real;
      function getmed : real;
      procedure setPayment(rpayment: real);
      procedure setMed(rpayment : real);
  end;


implementation

{ Tobjrecords }

constructor Tobjmed.create(sdoctor, sdate, sfollowup: string; bmedaid : boolean);
begin
  fdoctor := sdoctor;
  fdate := sdate;
  fmedaid := bmedaid;
  ffollowup := sfollowup;
  fpayment := 0;
  fmed:= 0;
end;

function Tobjmed.FollowUpDate: string;
var
  idays, imonth : integer;
begin
  if ffollowup = 'Yes' then
   begin
    idays := strtoint(copy(fdate,9,2))+7;
    if idays > 30 then
     begin
      idays := idays - 30;
      imonth := strtoint(copy(fdate,6,2))+1;
      result := copy(fdate,1,4)+'/'+inttostr(imonth)+'/'+inttostr(idays);
     end
     else
      result := copy(fdate,1,8)+inttostr(idays);
   end
   else
      result := 'No Follow-Up Appointment Needed';
end;

function Tobjmed.getmed: real;
begin
  result := fmed;
end;

function Tobjmed.getpayment: real;
begin
  result := fpayment;
end;

procedure Tobjmed.setPayment(rpayment : real);
begin
  fpayment := rpayment;
end;

procedure Tobjmed.setMed(rpayment : real);
begin
  fmed := rpayment;
end;

function Tobjmed.compilestring: string;
begin
  if ffollowup = 'Yes' then
  result := 'Patient must please see ' +fdoctor+ ' on '+followupdate
  else
  result := followupdate;
end;

end.
```

MAIN UNIT:

```
unit Question2U;

interface

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

type
  TForm1 = class(TForm)
    btnCapture: TButton;
    edtfollowup: TEdit;
    rgpdoctors: TRadioGroup;
    cbxmed: TCheckBox;
    lblfollowup: TLabel;
    btnStats: TButton;
    procedure btnCaptureClick(Sender: TObject);
    procedure btnStatsClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  objmed : Tobjmed;
  rtotal, rtotalmed : real;
  idrsmit, idrdup, idrphillips : integer;

implementation

{$R *.dfm}

procedure TForm1.btnCaptureClick(Sender: TObject);
var
  sdoctor, sdate, sfollowup : string;
  bmedaid : boolean;
  rpayment: real;
begin
  case rgpdoctors.itemindex of
    0: begin
         sdoctor := 'Dr Smit';
         inc(idrsmit);
       end;
    1: begin
         sdoctor := 'Dr du Plessis';
         inc(idrdup);
       end;
    2: begin
         sdoctor := 'Dr Phillips';
         inc(idrphillips);
       end;
  end;

  sdate := datetostr(date());

  if cbxmed.checked then
     bmedaid := true
  else
     bmedaid := false;

  sfollowup := edtfollowup.Text;

  objmed := Tobjmed.create(sdoctor,sdate,sfollowup,bmedaid);
  showmessage(objmed.compilestring);
  rpayment := randomrange(300,401);

  if bmedaid = false then
    begin
      objmed.SetPayment(rpayment);
      rtotal := rtotal + objmed.getpayment;
      showmessage('Total money received:
'+floattostrf(objmed.getpayment,ffcurrency,10,2));
    end
    else
      begin
        objmed.SetMed(rpayment);
        Showmessage('Charged to Medical Aid');
        rtotalmed  := rtotalmed + objmed.getmed;
      end;
end;

procedure TForm1.btnStatsClick(Sender: TObject);
begin
  Showmessage('Total Amount of Cash for Day:
'+floattostrf(rtotal,ffcurrency,10,2)+#13+'Total Amount charged to Medical Aids:
'+floattostrf(rtotalmed,ffcurrency,10,2));
  Showmessage('Dr Smit: '+inttostr(idrsmit) + #13+'Dr du Plessis:
'+inttostr(idrdup)+#13+'Dr Phillips: '+inttostr(idrphillips));
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  rtotal := 0;
  rtotalmed := 0;
  idrsmit := 0;
  idrdup := 0;
  idrphillips := 0;
end;
end.
```

QUESTION 3

```
unit Question3U;
interface

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

type
  TfrmAppointments = class(TForm)
    stgAppointments: TStringGrid;
    Button1: TButton;
    BitBtn1: TBitBtn;
    Button2: TButton;
    Button4: TButton;
    Button3: TButton;
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    procedure Display;
    procedure DeleteApp(irowfound, icolfound : integer);
    procedure Insert(sname : string; icolfound: integer);
  end;

var
  frmAppointments: TfrmAppointments;
  arrdoctors : array[1..4] of string = ('Dr du Plessis','Dr Smith','Dr Wessels','Dr Tom');
  arrtimes : array[1..10] of string = ('9:00-9:30','9:30-10:00','10:00-10:30','10:30-
11:00','11:00-11:30','11:30-12:00','14:00-14:30','14:30-15:00','15:00-15:30','15:30-16:00');
  ar2appointments : array[1..10,1..4] of string;

implementation

{$R *.dfm}

procedure TfrmAppointments.Button1Click(Sender: TObject);
var
  icount, idoctor : integer;
  sname : string;
begin
  //icount := 1;
  idoctor := strtoint(inputbox('Enter Doctor','1-Dr du Plessis; 2-Dr Smith; 3-Dr Wessels; 4-Dr
Tom; 5-Any doctor',''));
  sname := inputbox('Patient','Enter Name and Surname','');
  Insert(sname,idoctor);
  Display;
```

```
end;
procedure TfrmAppointments.Insert(sname:string; icolfound:integer);
var
  bfound : boolean;
  idoctor,icount, irandoc : integer;
begin
  icount := randomrange(1,11);
  bfound := false;
  if icolfound in [1..4] then
    begin
       while (bfound = false) and (icount <= 10) do
       begin
         if ar2appointments[icount, icolfound] = '' then
         begin
           ar2appointments[icount,icolfound] := sname;
           bfound := true;
         end;
         icount := randomrange(1,11);
       end;
    end
  else
    while (icount <= 10) and (bfound = false) do
    begin
      irandoc := randomrange(1,5);
      if ar2appointments[icount,irandoc] = '' then
        begin
          ar2appointments[icount,irandoc] := sname;
          bfound := true;
        end;
        inc(icount);
    end;
  if bfound = true then
    Showmessage('Appointment made')
  else
    Showmessage('No appointment available for that day');
end;

procedure TfrmAppointments.DeleteApp(irowfound, icolfound : integer);
begin
  ar2appointments[irowfound,icolfound] := '';
end;

procedure TfrmAppointments.Button2Click(Sender: TObject);
var
  irow: Integer;
  icol, icolfound, irowfound: Integer;
  spatient : string;
begin
  spatient := inputbox('','Patient Name','');
  for irow := 1 to 10 do
    for icol := 1 to 4 do
      begin
```

```
        if spatient = ar2appointments[irow,icol] then
        begin
          irowfound := irow;
          icolfound := icol;
        end;
      end;
  Insert(spatient,icolfound);
  DeleteApp(irowfound,icolfound);;
  Display;
end;

procedure TfrmAppointments.Button4Click(Sender: TObject);
var
  irow: Integer;
  icol: Integer;
  bfound : boolean;
  sinput : string;
begin
  bfound := false;
  sinput := inputbox('Search','Enter patients name','');
  for irow := 1 to 10 do
    for icol := 1 to 4 do
      if ar2appointments[irow,icol] = sinput then
        Showmessage(sinput+' has an appointment with '+stgAppointments.cells[icol,0] + ' at
'+stgAppointments.cells[0,irow]);
end;

procedure TfrmAppointments.Button5Click(Sender: TObject);
var
  irow: Integer;
  icol, imindoc,imin, icount: Integer;
begin
  imin := 100;
  for icol := 1 to 4 do
  begin
    icount := 0;
    for irow := 1 to 10 do
      begin
        if ar2appointments[irow,icol] <> '' then
        begin
          icount := icount + 1;
        end;
      end;
    if icount < imin then
    begin
      imindoc := icol;
      imin := icount;
    end;
  end;

  Showmessage('Doctor who will see all the patients without appointments:
'+arrdoctors[imindoc]);
end;
```

```
procedure TfrmAppointments.Display;
var
  irow, icol : integer;
begin
  for irow := 1 to 10 do
    for icol := 1to 4 do
      stgAppointments.cells[icol,irow] := ar2appointments[irow,icol];
end;

procedure TfrmAppointments.FormCreate(Sender: TObject);
var
  icol, irow, ipos: Integer;
  myfile : textfile;
  soneline, spatient : string;
begin

Randomize;
if fileexists('patients.txt') <> true  then
begin
  Showmessage('File does not exist');
  Exit;
end;
Assignfile(myfile, 'patients.txt');
Reset(myfile);
readln(myfile,soneline);
for irow  := 1 to 10 do
begin
  readln(myfile,soneline);
  for icol := 1 to 4 do
    begin
      if (icol < 4) then
      begin
        ipos := pos('#',soneline);
        spatient := copy(soneline,1,ipos-1);
        delete(soneline,1,ipos);
      end
      else
      spatient := soneline;
      if spatient <> '-' then
        ar2appointments[irow,icol] := spatient;
   end;
end;
Display;
closefile(myfile);
  for irow := 1 to 10 do
    stgAppointments.Cells[0,irow] := arrtimes[irow];

  for icol := 1 to 4 do
    stgAppointments.Cells[icol,0] := arrdoctors[icol];
end;
end.
```