NATIONAL SENIOR CERTIFICATE EXAMINATION
MAY 2023

INFORMATION TECHNOLOGY: PAPER I

MARKING GUIDELINES

Time: 3 hours 150 marks

These marking guidelines are prepared for use by examiners and sub-examiners, all of whom are required to attend a standardisation meeting to ensure that the guidelines are consistently interpreted and applied in the marking of candidates' scripts.

The IEB will not enter into any discussions or correspondence about any marking guidelines. It is acknowledged that there may be different views about some matters of emphasis or detail in the guidelines. It is also recognised that, without the benefit of attendance at a standardisation meeting, there may be different interpretations of the application of the marking guidelines.

SECTION A SQL

QUESTION 1.1 [4]

```
SELECT *
FROM tblHouseLeaders
WHERE HouseCaptain = TRUE
ORDER BY House
```

QUESTION 1.2 [5]

```
SELECT *
FROM tblFundraisers
WHERE Title LIKE '*show*' OR Title LIKE '*sale*'
```
Accept % for MySQL/JavaDB

QUESTION 1.3 [6]

```
SELECT Title, '#' & LEFT( Title, 1) & RIGHT( Title , 1) &
MID(Organiser, INT(Len(Organiser) / 2 ) +1, 2) AS Hashtag
FROM tblFundraisers
```

JavaDB

```
SELECT Title, '#' || SUBSTR( Title, 1,1) || SUBSTR( Title ,
LENGTH(title) , 1) || SUBSTR(Organiser, (Length(Organiser) / 2 )
+1, 2) AS Hashtag
FROM tblFundraisers
```

MySQL

```
SELECT Title, CONCAT ( "#", LEFT(Title,1),
                  RIGHT(Title,1),SUBSTRING(Organiser , (
LENGTH(Organiser) DIV 2 ) + 1 , 2) ) as Hashtag
FROM tblFundRaisers
```

QUESTION 1.4 [5]

```
SELECT ROUND (SUM(Amount) * 0.035,2) As BankCharges
FROM tblDeposits
WHERE DepositType <> 'cash'
```

JavaDB

```
SELECT CAST ( SUM(Amount)*0.035 AS Decimal(10,2)) As BankCharges
FROM tblDeposits
WHERE DepositType <> 'cash'
```

QUESTION 1.5 [5]

```
SELECT Title, GoalAmount
FROM tblFundraisers
WHERE GoalAmount < (SELECT AVG(GoalAmount) FROM tblFundraisers)
```

QUESTION 1.6 [4]

```
SELECT *
FROM tblHouseLeaders
WHERE tblHouseLeaderS.LeaderID NOT IN (SELECT LeaderID from
tblDeposits WHERE FundRaiserID =13)
```

Alternative using LEFT join

```
SELECT *

FROM tblHouseLeaders

WHERE LeaderID NOT IN (

SELECT tblHouseLeaders.LeaderID

FROM tblDeposits LEFT JOIN tblHouseLeaders

ON tblDeposits.LeaderID = tblHouseLeaders.LeaderID

WHERE FundRaiserID = 13

)
```

QUESTION 1.7 [9]

```
SELECT Title, SUM(Amount) AS TotalAmount
FROM tblFundraisers, tblDeposits
WHERE tblFundraisers.FundraiserID = tblDeposits.FundRaiserID
GROUP BY Title
HAVING SUM(AMOUNT) > 2500
```

MYSQL Alternative using HAVING with Aliase

```
SELECT Title, SUM(Amount) AS TotalAmount
FROM tblFundraisers, tblDeposits
WHERE tblFundraisers.FundraiserID = tblDeposits.FundRaiserID
GROUP BY Title
HAVING TotalAmount  > 2500
```

QUESTION 1.8 [7]

```
INSERT INTO tblDeposits (FundraiserID, Amount, DepositType,
DepositDate, LeaderID )
SELECT 15, Amount, 'eft', NOW ,LeaderID
FROM tblDeposits
WHERE FundraiserID = 15
```

JAVADB

```
INSERT INTO tblDeposits (FundraiserID, Amount, DepositType,
DepositDate, LeaderID )
SELECT 15, Amount, 'eft', CURRENT_DATE,LeaderID
FROM tblDeposits
WHERE FundraiserID = 15
```

MySQL

```
INSERT INTO tblDonations (FundraiserID, Amount, DepositType,
DepositDate, LeaderID )
SELECT 15, Amount, 'eft', CURDATE(),LeaderID
FROM tblDeposits
WHERE FundraiserID = 15
```

QUESTION 1.9 [5]

```
UPDATE tblDeposits
SET DepositType = DepositType + "-pm"
WHERE DepositType = "cash" AND depositDate > #12:00:00#;
```

SECTION B OBJECT ORIENTATED PROGRAMMING

JAVA SOLUTION

QUESTION 2 Award.java

```
//Q2.1 - 4
Class header correct
public class Award
{

    fields private
    typed correctly
    named correctly
    private String fullname;
    private String description;
    private int hoursLogged;

    //Q2.2 - 1
    field declared as private name and typed correctly
    private static int minHourFactor = 10;

    //Q2.3 - 3
    header correct
    parameters correct
    public Award(String inFN, String inDN, int inHD)
    {
        fields assigned correctly
        fullname = inFN;
        description = inDN;
        hoursLogged = inHD;
    }

    //Q2.4 - 3
    All getter methods named correctly
    All return correct type
    All returning correct field
    public String getFullname()
    {
        return fullname;
    }


    public String getDescription()
    {
        return description;
    }


    public int getHoursLogged()
    {
        return hoursLogged;
    }
```

```
    //Q2.5 - 2
    Both methods static named and typed correctly/correct
parameters
    public static int getMinHourFactor()
    {
        return minHourFactor;
    }

    public static void setMinHourFactor(int inMHF)
    {
        minHourFactor = inMHF;
    }

    //Q2.6 - 5
    Header correct
    public String getAwardColour()
    {

        checking the hoursLogged
        using minHourFactor
        correct colour for multiplier
        if (hoursLogged >= minHourFactor * 3)
        {
            return "Gold";
        } else if (hoursLogged >= minHourFactor * 2)
        {
            return "Silver";
        } else
        {
            return "Bronze";
        }
        returning award colour
    }

//Q2.7 - 4

    public String toString()
    {
        contains all fields
        formatting correct
        includes method call for awardColour
        returns string
        return fullname + " " + description + " " + hoursLogged +
" hrs (" + getAwardColour() + ")";
    }
}
```

QUESTION 3 SpecialAward.java

```
//Q3.1 - 2
Class named correctly
Extends Award
public class SpecialAward extends Award
{
    //Q3.2 - 3
    fields private
    correctly named and typed
    dateStarted uses a suitable date object
    private String project;
    private LocalDate dateStarted;


    //Q3.3 - 1
    field declared as public name and typed correctly
    public static int numSpecialAwards = 0;

    //Q3.4 - 5
    Header correct
    public SpecialAward(String inFN, String inDN , int inHD,
String inPT, LocalDate inDS)
    {
        Parent constructor called
        Correct parameters given
        super(inFN, inDN , inHD);

        child class fields assigned using parameters
        project = inPT;
        dateStarted = inDS;
        increment static field
        numSpecialAwards++;
    }

    //Q3.5 - 2
    getter methods correct
    returning correct type

    public String getProject()
    {
        return project;
    }
    public LocalDate getDateStarted() {
        return dateStarted;
    }

    //Q3.6 - 5
    header correct
    public int getNumYears()
    {
        Using appropriate class
        Using the custom date 18/5/2023
```

```
        determining the difference in the dates
        Period diff = Period.between(dateStarted,
LocalDate.of(2023, 5, 18));

        return the number of years
        return diff.getYears();

    }

    //Q3.7 - 4
    Method header correct
    public String toString()
    {
        contains relevant fields
        formating correct
        returning string
        return super.toString() + " " + project + " project " +
getNumYears() + " Years";
    }


}
```

QUESTION 4 & 6.1 ServiceAwardManager.java

```
//Q4.1 - 1
Class header correct
public class AwardManager
{

    //Q4.2 - 4
    fields private
    fields named correctly
    array of 50 ServicesAwards (Type of parent class)
    size field correct type
    private Award awArr[] = new Award[50];
    private int size = 0;

    //Q4.3 - 11
    Constructor method header correct
    public AwardManager()
    {
        try
        {
            open file
            Scanner sc = new Scanner(new File("awards.txt"));

            loop through file
            while (sc.hasNextLine())
            {
                read line
                String line = sc.nextLine();
```

```
                extract first 3 fields
                String tokens[] = line.split("#");

                String name = tokens[0];
                String description = tokens[1];
                int hoursLogged = Integer.parseInt(tokens[2]);

                check for special service award tokens
                if (tokens.length > 3)
                {
                    get fields for special service award
                    String project = tokens[3];
                    parsing date (must be same as class type)
                    LocalDate dateStarted =
LocalDate.parse(tokens[4], DateTimeFormatter.ofPattern("yyyy-MM-
dd"));

                    create special service award object
                    awArr[size] = new SpecialAward(name,
description, hoursLogged, project, dateStarted);
                } else
                {
                    create service award object
                    awArr[size] = new Award(name, desciption,
hoursLogged);
                }
                increment size
                size++;
            }
        } catch (FileNotFoundException e)
        {
            System.out.println("File Missing " + e);
        }
    }

    //Q4.4 - 5
    method header correct
    public String toString()
    {
        String r = "";
        loop through array
        for (int i = 0; i < size; i++)
        {
            appending to string
            add new line
            r += awArr[i] + "\n";
        }
        return appended string
        return r;
    }

    //Q4.5 - 7
    method header correct
```

```
    public void sort()
    {
        outside for loop correct
        for (int i = 0; i < size - 1; i++)
        {
            inside for loop correct
            for (int j = i + 1; j < size; j++)
            {

                compare correct elements
                using getFullname
                if (awArr[i].getFullname().compareTo
                   (awArr[j].getFullname()) > 0)
                {

                    correct temp type
                    using correct value from loops
                    Award temp = awArr[i];
                    awArr[i] = awArr[j];
                    awArr[j] = temp;
                }
            }
        }
    }

    //Q4.6 - 5
    Method header correct
    private void deleteAward(int pos)
    {
        looping through array
        starting at deletion position
        for (int i = pos; i < size-1; i++)
        {
            shift correct
            awArr[i] = awArr[i+1];
        }
        decreament size
        size--;
    }

  //Q6.1 - 16
    Method header correct
    public String changeRequirements()
    {
        list for student who do not get a special award
        String specialStr = "Students who do not achieve a Special
Award\n";
        list for deleted students
        String delString = "Deleted students\n";
        setting the static property to 12 using the accessor
method
        Award.setMinHourFactor(12);
```

```
        resetting the number of special awards to 0 using the
static property
        SpecialAward.numSpecialAwards = 0;
        loop through array
        int i = 0;
        while (i < size)
        {
            check minimum hours logged
           if (awArr[i].getHoursLogged()<
Award.getMinHourFactor())
            {
                appending deleted student name
                delString += awArr[i].getFullname() + "\n";
                delete element using method created in Q4.6
                deleteAward(i);
                reduce the loop to account for the shift
                i--;
            }
           check for instance of special awards
           else if (awArr[i] instanceof SpecialAward)
            {
                casting to instance of speciaAward
                SpecialAward temp = (SpecialAward) awArr[i];
                check if award is not gold
                if (!temp.getAwardColour().equals("Gold"))
                {
                    append to list

                    specialStr += temp.getFullname() + "\n";
                    replace special award with normal award using
the data from the special award
                    awArr[i] = new Award( temp.getFullname(),
temp.getDescription() , temp.getHoursLogged());
                }
            }

           i++;
        }

        return lists
        return specialStr + "\n" + delString;
    }
}
```

QUESTION 5 & 6.2,6.3 AwardUI.java

```
//Q5.1 - 1
Class header correct
public class AwardUI {

    public static void main(String[] args) {
        //Q5.2 - 1
        creating AwardManager object in appropriate place
        AwardManager am = new AwardManager();

        //Q5.3 - 2
        Calling the sort method
        am.sort();
        redisplay sorted list
        System.out.println(am);

        //Q5.4 - 1
        Calling static class property with label
        System.out.println("Number of special award Recipients: "
+ SpecialAward.numSpecialAwards);

        //Q6.2 - 2
        Calling changeRequirements method
        as a typed method
        System.out.println(am.changeRequirements());
    }
}
```

DELPHI SOLUTION

QUESTION 2 uAward.pas

```
unit uAward;

interface
uses SysUtils;
//Q2.1 - 4
Class header correct
type TAward = class
  private
    fields private
    typed correctly
    named correctly
    fullname : string;
    description : string;
    hoursLogged : integer;

    //Q2.2 - 1
    field declared as private name and typed correctly
    class var minHourFactor : integer;
    class constructor Create;

  public

    constructor Create (inFN: string; inDN: string; inHD :
integer);
    function getFullname() : string;
    function getDescription() : string;
    function getHoursLogged() : integer;
    function getAwardColour() : string;
    class function getMinHourFactor() : integer; static;
    class procedure setMinHourFactor( inMHF:integer); static;
    function toString() : string; virtual;

end;

implementation


{ TServiceAward }

//Q2.3 - 3
header correct
parameters correct
constructor TAward.Create(inFN: string; inDN: string; inHD :
integer);
begin
fields assigned correctly
  fullName := inFN;
  description := inDN;
  hoursLogged := inHD;
```

```
end;


class constructor TAward.Create;
begin
      minHourFactor := 10;
end;




//Q2.4 - 3
All getter methods named correctly
All return correct type
All returning correct field

function TAward.getDescription: string;
begin
  Result := description;
end;

function TAward.getFullname: string;
begin
  Result := fullName;
end;

function TAward.getHoursLogged: integer;
begin
  Result := hoursLogged;
end;

//Q2.5 - 2
//Both methods named and typed correctly
class function TAward.getMinHourFactor: integer;
begin
  Result := minHourFactor;
end;

class procedure TAward.setMinHourFactor(inMHF: integer);
begin
  minHourFactor := inMHF;
end;

//Q2.6 - 5
Header correct
function TAward.getAwardColour: string;
var
  c : string;
begin
     checking the hoursLogged
     using minHourFactor
     correct colour for multiplier
```

```
      if hoursLogged >= (minHourFactor * 3) then
      begin
        c := 'Gold';
      end
      else if hoursLogged >= minHourFactor * 2 then
       begin
         c := 'Silver';
       end
      else
      begin
         c := 'Bronze' ;
      end;

       returning award colour
      Result := c;
end;

//Q2.7 - 5
header correct
function TAward.toString: string;
begin
  contains all fields
  formatting correct
  includes method call for awardColour
  returns string
  Result := fullname + ' ' + description + ' ' +
IntToStr(hoursLogged) + ' hrs (' + getAwardColour() + ')';
end;

end.
```


QUESTION 3 uSpecialAward.pas

```
unit uSpecialAward;

interface
  uses SysUtils, DateUtils, uAward;
//Q3.1 - 2
Class named correctly
Extends ServiceAward
type TSpecialAward = class(TAward)
  private
    //Q3.2 - 3
    fields private
    correctly named and typed
    dateStarted uses a suitable date object
    project : string;
    dateStarted : TDateTime;

  public
    //Q3.3 - 1
    field declared as public name and typed correctly
    class var numSpecialAwards : integer;
    constructor Create (inFN: string; inDN: string; inHD :
integer; inPT : string; inDS : TDate );
    function getProject() : string;
    function getDateStarted() : TDate;
    function getNumYears() : integer;
    function toString() : string ; override;


end;

implementation

{ TSpecialAward }
//Q3.4 - 4
Header correct
constructor TSpecialAward.Create(inFN: string; inDN: string; inHD
: integer; inPT : string; inDS : TDate);
begin
  Parent constructor called
  Correct parameters given
  Inherited Create(inFN, inDN , inHD);
   child class fields assigned using parameters
  project := inPT;
  dateStarted := inDS;
  Inc(numSpecialAwards);
end;

//Q3.5 - 2
getter method correct returning correct type
returning correct type
function TSpecialAward.getDateStarted: TDate;
```

```
begin
  Result:= dateStarted;
end;

//Q3.6 - 5
header correct
function TSpecialAward.getNumYears: integer;
var
 years : integer;
  months , days : integer;
begin

    Using appropriate class
    Using the custom date 18/5/2023
    determining the difference in the dates
    years := YearOf(EncodeDate(2023,5, 18)) - YearOf(dateStarted);
    months :=  MonthOf(EncodeDate(2023,5, 18) )  -
MonthOf(dateStarted);
    days := DayOf(EncodeDate(2023,5, 18)) - DayOf(dateStarted);


    if  months < 0 then
    begin
      years := years - 1;
    end
    else if (months = 0) AND (days  < 0) then
    begin
      years := years - 1;
    end;

     return the number of years
    Result := years;

end;

function TSpecialAward.getProject: string;
begin
  Result:= project;
end;

//Q3.7 - 4
Method header correct
function TSpecialAward.toString: string;
begin
 contains relevant fields
 formating correct
 returning string
  Result:= Inherited toString + ' ' + project + ' project '  +
IntToStr(getNumYears());

end;
end.
```

QUESTION 4 & 6.1 uAwardManager.pas

```
unit uAwardManager;



interface
uses SysUtils, DateUtils, uAward, uSpecialAward;
//Q4.1 - 1
Class header correct
type tAwardManager = class
  private
    //Q4.2 - 4
    fields private
    fields named correctly
    array of 50 Awards (Type of parent class)
    size field correct type
    awArr : array[1..50] of tAward;
    size : integer;

  public
    constructor Create();
    function toString : string;
    procedure sort();
    procedure deleteAward( pos : integer);
    function changeRequirements() : string;




end;

implementation

{ tAwardManager }




//Q4.3 - 11
Constructor method header correct
constructor tAwardManager.Create;
var
  inFile : textfile;
  line, fullname, description , project ,date , d , m , y  :
string;
  hoursLogged : integer;
  dateStarted : TDate;
begin
  if FileExists('awards.txt') <> true then
    begin
      WriteLn('File Missing');
```

```
    end
  else
    begin
    open file
      AssignFile(inFile , 'awards.txt');
      Reset(inFile);

      size:= 0;
      loop through file
      while NOT EOF(inFile) do
        begin
        read line
          ReadLn(inFile , line);
          increment size
          Inc(size);


          extract first 3 fields
          fullName := Copy(line , 1 , Pos('#', line) - 1);
          Delete(line , 1 , Pos('#',line));

          description := Copy(line , 1 , Pos('#', line) - 1);
          Delete(line , 1 , Pos('#',line));

          check for special service award tokens

          if Pos('#' , line) > 0 then
            begin
               get fields for special service award

              hoursLogged := StrToInt(Copy(line , 1 , Pos('#',
line) - 1));
              Delete(line , 1 , Pos('#',line));


              project := Copy(line , 1 , Pos('#', line) - 1);
              Delete(line , 1 , Pos('#',line));
               parsing date (must be same as class type)
              date := line;


              y := Copy(date,1 , Pos('-', date) - 1);
              Delete(date,1 , Pos('-', date));

              m := Copy(date,1 , Pos('-', date) - 1);
              Delete(date,1 , Pos('-', date));
              d := date;



              dateStarted := EncodeDate(StrToInt(y),StrToInt(m),
StrToInt(d));
```

```
              create special service award object
              awArr[size] := TSpecialAward.Create(fullname,
description, hoursLogged, project, dateStarted );

            end
          else
            begin
              create service award object
              hoursLogged := StrToInt(line);
              awArr[size] := TAward.Create(fullname , description,
hoursLogged);
            end;

        end;
    end;


end;



//Q4.4 - 5
 method header
function tAwardManager.toString: string;
  var
  i : integer;
  output : string;
begin
  output := '';
   loop through array
  for i := 1 to size do
    begin
      appending to string
      add new line
      output := output + awArr[i].toString() + #13#10;
    end;
  return appended string
  Result := output;

end;

//Q4.5 - 7
method header correct
procedure tAwardManager.sort;
var
  i , j : integer;
  temp : TAward;
begin
  outside for loop correct
  for i := 1 to size do
  begin
    inside for loop correct
```

```
    for j := 1 to size-1 do
      begin
        checking order
        if CompareStr(awArr[j].getFullname() ,
awArr[j+1].getFullname) > 0 then
        begin
            swop
            correct temp type
            using correct value from loops
             temp := awArr[j];
             awArr[j] := awArr[j+1];
             awArr[j+1] := temp;
        end;

      end;
  end;


end;

 //Q4.6 - 5
    Method header correct
procedure tAwardManager.deleteAward(pos: integer);
var
  i: Integer;
begin
  looping through array
  starting at deletion position
  for i := pos to size do
  begin
      shift correct
       awArr[i] := awArr[i+1];
  end;
   decreament size
  size := size - 1;
end;

//Q6.1 - 16
Method header correct

function tAwardManager.changeRequirements: string;
var
  specialStr : string;
  delString : string;
  i : integer;
  temp : TSpecialAward;

begin
list for student who do not get a special award
     specialStr := 'Students who do not achieve a Special Award' +
#13#10;
      list for deleted students
     delString := 'Deleted students' + #13#10;
```

```
     setting the static property to 12 using the accessor method
     TAward.setMinHourFactor(12);
     resetting the number of special awards to 0 using the static
property
     TSpecialAward.numSpecialAwards := 0;

     loop through array
     i := 1;

     while  i < size do
     begin
        check the minimum hourslogged
       if awArr[i].getHoursLogged() < TAward.getMinHourFactor()
then
       begin
         appending deleted student name
         delString := delString + awArr[i].getFullname + #13#10;
         delete element using method created in Q4.6
         deleteAward(i);
         reduce the loop to account for the shift
         i := i - 1;
       end
        check for instance of special awards
       else if awArr[i] is TSpecialAward then
        begin
        casting to instance of speciaAward
             temp := awArr[i] as TSpecialAward;
               check if award is not gold
             if NOT (CompareStr(temp.getAwardColour , 'Gold') = 0
) then
             begin
                append to list
               specialStr := specialStr + temp.getFullname() +
#13#10 ;
               replace special award with normal award using the
data from the special award
               awArr[i] := TAward.Create( temp.getFullname ,
temp.getDescription , temp.getHoursLogged);
             end;


        end;

        i := i + 1;
     end;
     return lists
     Result:= specialStr + #13#10 + #13#10 + delString;
end;



end.
```


QUESTION 5 & 6.2,6.3 AwardUI.java

```
program AwardUI;
 //Q5.1 - 1
Class header correct
{$APPTYPE CONSOLE}

{$R *.res}

uses
  System.SysUtils,
  DateUtils,
  uAward in 'uAward.pas',
  uSpecialAward in 'uSpecialAward.pas',
  uAwardManager in 'uAwardManager.pas';
var
  testAward : TAward;
  am : TAwardManager;
begin
  try
    { TODO -oUser -cConsole Main : Insert code here }

     //Q5.2 - 1
     creating AwardManager object in appropriate place

    am := TAwardManager.Create();

     //Q5.3 - 1
    Calling the sort method
    am.sort();
    redisplay sorted list
    WriteLn(am.toString());

    //Q5.4 - 1
    Calling static class property with label
    WriteLn('Number of special award Recipients: ' +
IntToStr(TSpecialAward.numSpecialAwards));

    //Q6.2 - 2
    Calling static
    class property with label
    WriteLn(am.changeRequirements) ;

    ReadLn;


  except
    on E: Exception do
      Writeln(E.ClassName, ': ', E.Message);
  end;
end.
```