NATIONAL SENIOR CERTIFICATE EXAMINATION
NOVEMBER 2022

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 tblCaptains
WHERE CombatTraining = TRUE
ORDER BY DateHired DESC
```

QUESTION 1.2 [4]

```
SELECT *
FROM tblSpaceships
WHERE (Model LIKE 'M*' OR Model LIKE 'T*') AND FuelConsumption <
500
```

Accept % for MySQL/JavaDB

QUESTION 1.3 [5]

```
SELECT *
FROM tblCaptains
WHERE YEAR(NOW()) - YEAR(DateHired) >= 10
```

JAVADB

```
SELECT *
FROM tblCaptains
WHERE YEAR(CURRENT_DATE) - YEAR(DateHired)  >= 10
```

MYSQL

```
SELECT *
FROM tblCaptains
WHERE YEAR(NOW() ) - YEAR(DateHired)  >= 10
```

QUESTION 1.4 [5]

```
SELECT Model , Speed
FROM tblSpaceShips
WHERE Speed > (SELECT AVG(Speed) FROM tblSpaceShips)
ORDER BY Speed DESC
```

QUESTION 1.5 [7]

```
SELECT YEAR(DepartureDate) AS YearOfMission, COUNT(*) AS
NumberOfMissions
FROM tblMissions
WHERE MONTH(DepartureDate) <= 6
GROUP BY YEAR(DepartureDate)
```


MySQL Alias use in GroupBy

```
SELECT YEAR(DepartureDate) AS YearOfMission, COUNT(*) AS
NumberOfMissions
FROM tblMissions
WHERE MONTH(DepartureDate) <= 6
GROUP BY YearOfMission
```

QUESTION 1.6 [9]

```
SELECT tblMissions.MissionID, tblCaptains.Fullname,
tblSpaceships.Model, tblSpaceships.Speed, tblMissions.SolarSystem,
tblMissions.Distance, Round ((Distance/Speed),2) AS TimeInYears
FROM tblMissions, tblCaptains, tblSpaceships
WHERE tblMissions.CaptainID=tblCaptains.CaptainID AND
tblMissions.SpaceshipID=tblSpaceships.SpaceshipID;
```

JAVADB

```
SELECT tblMissions.MissionID, tblCaptains.Fullname,
tblSpaceships.Model, tblSpaceships.Speed, tblMissions.SolarSystem,
tblMissions.Distance, FLOOR ( (Distance/Speed) * 100 + 0.5) / 100
AS TimeInYears
FROM tblMissions, tblCaptains, tblSpaceships
WHERE tblMissions.CaptainID=tblCaptains.CaptainID AND
tblMissions.SpaceshipID=tblSpaceships.SpaceshipID
```

MYSQL

```
SELECT tblMissions.MissionID, tblCaptains.Fullname,
tblSpaceships.Model, tblSpaceships.Speed, tblMissions.SolarSystem,
tblMissions.Distance, Round ((Distance/Speed),2) AS TimeInYears
FROM tblMissions, tblCaptains, tblSpaceships
WHERE tblMissions.CaptainID=tblCaptains.CaptainID AND
tblMissions.SpaceshipID=tblSpaceships.SpaceshipID;
```

Accept Inner Joins

QUESTION 1.7 [7]

```
UPDATE tblMissions
SET MainObjective = 'Scouting'
WHERE DepartureDate > NOW() AND RIGHT(SolarSystem,1) NOT IN
('A','B','C') AND MainObjective = 'Military Exercise'
```

JAVADB

```
UPDATE tblMissions
SET MainObjective = 'Scouting'
WHERE DepartureDate > CURRENT_DATE AND SUBSTR(SolarSystem,
length(SolarSystem)  ,  1) NOT IN ('A' , 'B' , 'C') AND
MainObjective = 'Military Exercise'
```

MYSQL


```
UPDATE tblMissions
SET MainObjective = 'Scouting'
WHERE DepartureDate > NOW() AND RIGHT(SolarSystem,1) NOT IN
('A','B','C')  AND MainObjective = 'Military Exercise'
```

QUESTION 1.8 [9]

```
INSERT INTO tblMissions (CaptainID, SpaceshipID, MainObjective,
DepartureDate, Distance, SolarSystem)
SELECT CaptainID , INT( (RND(CaptainID) * 30) + 1 ) ,
'Exploration' , #01/01/2030# , 5 , 'Alpha-Centauri'
FROM tblCaptains
WHERE CaptainID NOT IN (SELECT CaptainID FROM tblMissions)
```

MYSQL

```
INSERT INTO tblMissions ( CaptainID, SpaceshipID, MainObjective,
DepartureDate, Distance, SolarSystem )
SELECT CaptainID, FLOOR( (RAND() * 30) + 1 ), 'Exploration',
‘01/01/2030’, 5, 'Alpha-Centauri'
FROM tblCaptains
WHERE CaptainID NOT IN (SELECT CaptainID FROM tblMissions)
```

JAVADB

```
INSERT INTO tblMissions  ( CaptainID, SpaceshipID, MainObjective,
DepartureDate, Distance, SolarSystem )
SELECT CaptainID, FLOOR( (RAND(CaptainID) * 30) + 1 ) ,
'Exploration', ‘01/01/2030’ , 5, 'Alpha-Centauri'
FROM tblCaptains
WHERE CaptainID NOT IN (SELECT CaptainID FROM tblMissions)
```


SECTION B OBJECT ORIENTATED PROGRAMMING

JAVA SOLUTION

QUESTION 2 CrewMember.java

```
//Q2.1 - 4
 Class header correct
public class CrewMember {
     fields private
     typed correctly
     named correctly
    private String fullname;
    private int crewID;
    private String department;

    //Q2.2 - 2
    field declared as public
    named and assigned correctly (must be static)
    public static int numPromotedCrew=0;


    //Q2.3 - 3
     header correct
     parameters named and typed correctly
    public CrewMember(String inFN , int inCID, String inDT )
    {
         fields assigned using parameters
        fullname = inFN;
        crewID = inCID;
        department = inDT;
    }

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

    public int getCrewID() {
        return crewID;
    }

    public String getDepartment() {
        return department;
    }
```

```
    //Q2.5 - 5
     Header correct
    public String toString()
    {
         contains all fields
         added tabbed spaces
         formatting correct
         returns string
        return fullname + "\tCrew ID: " + crewID + "\t[" +
department + "]";
    }


}
```

QUESTION 3 Officer.java

```
//Q3.1 - 2
 Class named correctly
 Extends CrewMember
public class Officer extends CrewMember
{
    //Q3.2 - 2
     private, correctly named and typed
     datePromoted uses a suitable date object
    private int rank;
    private LocalDate datePromoted;
    //Q3.3 - 2
    field declared as public
    named and assigned correctly (must be static)
    public static int numPromotedOfficers = 0;

    //Q3.4 - 4
     Constructor header correct

    public Officer(String inFN , int inCID, String inDT ,int inRK,
LocalDate inDP)
    {
         Parent contructor called
         Correct parameters given
        super(inFN, inCID, inDT);

         child class fields assigned using parameters
        rank = inRK;
        datePromoted  = inDP;
    }

    //Q3.5 - 1
     Method header correct and returning correct field

    public int getRank() {
        return rank;
    }
```

```
    //Q3.6 - 5
    Method header correct
    public String getTitle()
    {
        array of strings representing different
          titles in correct order
        String[] titleArr =
        {
            "", "Ensign", "Lieutenant", "Lt Commander",
"Commander", "Captain"
        };

        returning correct title based on rank
        return titleArr[rank];
    }


    //Q3.6 - 5
     **Alternative solution**
     **Switch can be replaced by case statement or correctly
nested if else statement.
    Header correct
    public String getTitle()
    {
        checking for title based on rank
        String title = "";
        switch(rank)
        {
            case 1:
                title = "Ensign";
                break;
            case 2:
                title = "Lieutenant";
                break;
            case 3:
                title = "Lt Commander";
                break;
            case 4:
                title = "Commander";
                break;
            case 5:
                title = "Captain";

        }
        returning correct title
        return title;
    }
```



```
    //Q3.7 - 4
     Method header correct
    public String toString()
    {
         calls toString from parent class
         appends title
         return String
        return super.toString() + " " + getTitle();
    }

    //Q3.8 - 5
    Method header correct
    public void promote()
    {

         Checking for rank less than 5
          (less than equal to 4 also acceptable)
        if (rank < 5)
        {
            increment rank
            rank = rank + 1;
            increment number of promoted officers
            numPromotedOfficers++;

           Updating the promotion date to the current date
            datePromoted = LocalDate.now();


        }


    }
}
```

QUESTION 4 & 6.1,6.2 CrewMemberManager.java

```
//Q4.1 - 1
Class header correct
public class CrewMemberManager
{
    //Q4.2 - 4
    fields private
    fields named correctly
    array of 80 CrewMembers (Type of the parent class)
    size field correct type

    private CrewMember[] cArr = new CrewMember[80];
    private int size = 0;
```

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

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

                extract first 3 fields
                String tokens[] = line.split("#");
                String fullname = tokens[0];
                int crewID = Integer.parseInt(tokens[1]);
                String department = tokens[2];

                check for an officer or a crewmember
                if (tokens.length == 3)
                {
                    creating crewmember object and add to array
                    CrewMember c = new CrewMember(fullname,
crewID, department);
                    cArr[size] = c;
                }
                else
                {
                    get officer rank
                    int rank = Integer.parseInt(tokens[3]);
                    get date and convert date(date must match the
                     object type defined in Officer)
                    LocalDate date = LocalDate.parse(tokens[4],
DateTimeFormatter.ofPattern("dd/MM/yyyy"));
                    create officer object and add to array
                    Officer f = new Officer(fullname, crewID,
department, rank, date);
                    cArr[size] = f;
                }
                 increment size
                 size++;

            }

            sc.close();
        } catch (FileNotFoundException fne)
        {
            System.out.println("File Missing " + fne);
        }
    }
```



```
//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 += cArr[i].toString() + "\n";
    }

    return appended string
    return r;
}

//Q6.1 - 4
find the crew member using crewID
public int findCrewMember(int crewID)
{
    int pos = 0;
    using an appropriate loop
    while (pos < size)
    {

        if crewID matches end search
        if (cArr[pos].getCrewID() == crewID)
        {
            return pos;
        }

        pos++;

    }
    return -1 not found
    return -1;
}
//Q6.2 - 5
Method header correct with return statement
public String processTestResults()
{
    list for output
    String r = "";
    try
    {
        read file
        Scanner sc = new Scanner(new File("testResults.txt"));
        loop using appropriate loop
        for (int i = 0; i < 10; i++)
        {
            read the crewID and testResult
            int crewID = sc.nextInt();
            int testResult = sc.nextInt();
```

```
//Q6.2.1 - 1
find the pos of the crewmember with the crewID
int pos = findCrewMember(crewID);

//Q6.2.2a - 7
check if the crew member is an officer
if (cArr[pos] instanceof Officer)
{
    cast element to an Officer
    Officer f = (Officer) cArr[pos];
    boolean pass = false;
    check if the officer has met the requirements
    correctly nested if statement
    if (f.getRank() == 1 && testResult >= 75)
    {
        pass = true;
    } else if(f.getRank()== 2 && testResult >= 80)
    {
        pass = true;
    } else if(f.getRank()== 3 && testResult >= 85)
    {
        pass = true;
    } else if(f.getRank()== 4 && testResult >= 90)
    {
        pass = true;
    }


    //Q6.2.3 - 6
    6.2.3 - Append the word officer and their
    name
    r +=  "Officer " + f.getFullname();

    //6.2.2a cont.
    check if they passed
    if (pass)
    {
        promote the officer
        f.promote();
        replace object in the array with the new
         object
        cArr[pos] = f;

        6.2.3 - Appending pass text for officer
        r += " has passed and is promoted to " +
             f.getTitle() + " Rank " +
             f.getRank() + "\n";
    }
    else if (f.getRank() == 5)
    {
```

```
                        6.2.3 - appending captain participation
                        r += " participated in the test\n";
                    }
                    else if (pass == false)
                    {
                        6.2.3 -appending failed for an officer
                        r += " has failed\n";
                    }
                }

                //6.2.2b - 6
                else
                {
                    r += "Crew Member " + cArr[pos].getFullname();
                    check if the crew member achieved the minimum
                     to pass and append pass/fail
                    if (testResult >= 75)
                    {
                        increment numPromotedCrew
                        CrewMember.numPromotedCrew++;
                        promote the crewmember by creating new
                          officer with crewmember details using
                        appropriate date
                        Officer newOfficer = new Officer
                              (cArr[pos].getFullname(),
                               cArr[pos].getCrewID(),
                               cArr[pos].getDepartment(),
                               1, LocalDate.now());
                        add new officer object to array
                        cArr[pos] = newOfficer;
                       6.2.3 - Appending passed message to new
                        officer including new title and rank
                        r += " has passed and is promoted to " +
                               newOfficer.getTitle() + " Rank " +
                               newOfficer.getRank() + "\n";
                    } else
                    {
                        6.2.3 - Appending fail for a crewmember
                        r += " has failed\n";
                    }
                }

            }
        } catch (FileNotFoundException e)
        {
            System.out.println("File Missing " + e);
        }

        return r;
    }


}
```

QUESTION 5 , 6.3 , 6.4 CrewMemberUI.java

```
//Q5.1 - 1
 Class header correct
public class CrewMemberUI {
    public static void main(String[] args) {
        //Q5.2 - 2
          CrewMemberManager object created in appropriate place
        CrewMemberManager cm = new CrewMemberManager();

        //Q5.3 - 2
          Calling toString and displaying all crew members
        System.out.println(cm.toString());

        //Q6.3 - 1
         Call processTestResults in appropriate class

        System.out.println(cm.processTestResults());

        //Q6.4 - 2
        Calling the static numPromotedCrew
        Calling the static numPromotedOfficers
        System.out.println("Number of promoted crew members: " +
                           CrewMember.numPromotedCrew);
        System.out.println("Number of promoted officers: " +
                           Officer.numPromotedOfficers);
    }
}
```


DELPHI SOLUTION

QUESTION 2 uCrewMember.pas

```
unit uCrewMember;

interface
uses SysUtils;

//Q2.1 - 4
 Class header correct
type TCrewMember = class
   fields private
   typed correctly
   named correctly
  private
    fullname : string;
    crewID : integer;
    department : string;

  public
    //Q2.2 - 2
    field declared as public
    named and assigned correctly (must be static/class)
    class var numCrewPromoted : integer;
    constructor Create(inFN: string; inCID : integer; inDT :
string );
    function getFullName() : string;
    function getCrewID() : integer;
    function getDepartment() : string;
    function toString() : string ; virtual;

end;

implementation

{ TCrewMember }

//Q2.3 - 3
 Constructor header correct
 Parameters named and typed correctly
constructor TCrewMember.Create(inFN: string; inCID: integer; inDT:
string);
begin
   fields assigned using parameters
  fullname := inFN;
  crewID := inCID;
  department := inDT;
end;
```



```
//Q2.4 - 3
 ALL getter methods named correctly
 All method return types in header correct
 All methods return correct field
function TCrewMember.getCrewID: integer;
begin
  Result := crewID;
end;

function TCrewMember.getDepartment: string;
begin
  Result := department;
end;

function TCrewMember.getFullName: string;
begin
  Result := fullName;
end;

//Q2.5 - 5
 Method header correct
function TCrewMember.toString: string;
begin
   contains all fields
   added tabbed spaces
   formatting correct
   returns string
  Result := fullname + #9 + 'CrewID: ' + IntToStr(crewID) + #9 +
'[' + department + ']';
end;

end.
```

QUESTION 3 uOfficer.pas

```
unit uOfficer;

interface
  uses SysUtils, DateUtils, uCrewMember;
//Q3.1 - 2
Class named correctly
Extends CrewMember

type TOfficer = class(TCrewMember)

    //Q3.2 - 2
     private, correctly named and typed
     datePromoted uses a suitable date object
  private
    rank : integer;
    datePromoted : TDateTime;

  public
     //Q3.3 - 2
```

```
    field declared as public
    named and assigned correctly (static/class)

    class var numPromotedOfficers : integer;
    constructor Create(inFN: string; inCID : integer; inDT :
string; inRK : integer; inDP : TDateTime);
    function getTitle() : string;
    function getRank() : integer;
    function toString : string ; override;
    procedure promote();
end;

implementation
    { TOfficer }
//Q3.4 - 4
Constructor header correct
constructor TOfficer.Create(inFN: string; inCID : integer; inDT :
string; inRK : integer; inDP : TDateTime);
begin
    Parent constructor called
    Correct parameters given
    Inherited Create(inFN, inCID, inDT);
    child class fields assigned using parameters
    rank := inRK;
    datePromoted := inDP;
end;

//Q3.5 - 1
Method header correct and returning correct field
function TOfficer.getRank: integer;
begin
    Result := rank;
end;

//Q3.6 - 4
Method header correct
function TOfficer.getTitle: string;
const
  array of strings representing different titles in correct
    order
   titleArr : array[1..5] of string = ('Ensign','Lieutenant','Lt
Commander','Commander','Captain');
begin
  returning correct title based on rank
  Result:= titleArr[rank];
end;

//3.6 - 4
**Alternative solution**
**Switch can be replaced by case statement or correctly nested if
else statement
Header correct
function TOfficer.getTitle: string;
var
```

```
  title : string;
begin
  checking for title based on rank
  case rank of
    1 : title:= 'Ensign';
    2 : title:= 'Lieutenant';
    3 : title:= 'Lt Commander';
    4 : title:= 'Commander';
    5 : title := 'Captain';

  end;

  returning correct title based on rank
  Result:= title;
end;
}

//Q3.7 - 4
Method header correct
function TOfficer.toString: string;
begin
    calls toString from parent class
    appends title
    return String
    Result := Inherited toString + ' ' + getTitle();
end;

//Q3.8 - 5
Method header correct
procedure TOfficer.promote();
begin
Checking for rank less than 5 (less than equal to 4 also
acceptable)
    if rank < 5 then
    begin
      increment rank
      rank := rank + 1;
      increment number of promoted officers
      Inc(numPromotedOfficers);
      Updating the promotion date to the current date
      datePromoted:=  Now;

    end;


end;

end.
```

QUESTION 4 & 6.1,6.2 uCrewMemberManager.pas

```
unit uCrewMemberManager;

interface
uses SysUtils,DateUtils, uCrewMember, uOfficer;
//Q4.1 - 1
Class header correct

type tCrewMemberManager = class
  //Q4.2 - 4
  Fields private
  Fields named correctly
  Array of 80 CrewMembers (Type of the parent class)
  size field correct type
  private
    cArr : array[1..80] of TCrewMember;
    size : integer;
  public
    constructor Create();
    function toString() : string;
    function processTestResults() : string;
    function findCrewMember(inCID : integer) : integer;

end;
implementation

{ tCrewMemberManager }

//Q4.3 - 11
Constructor method header correct
constructor tCrewMemberManager.Create;
var
  inFile : textfile;
  line, fullname, department, date , d ,m ,y: string;
  crewID, rank : integer;
  datePromoted : TDateTime;
begin
  if FileExists('crewmembers.txt') <> true then
    begin
      WriteLn('File Missing');
    end
  else
    begin
      open file
      AssignFile(inFile, 'crewmembers.txt');
      Reset(inFile);

      size := 0;
      loop through file
      while NOT EOF(inFile) do
        begin
          read line
          ReadLN(inFile, line);
```

```
            increment size
            Inc(size);

            extract the first three fields
            fullname := Copy(line , 1 , Pos('#', line) -1);
            Delete(line, 1, Pos('#', line));
            crewID := StrToInt(Copy(line , 1 , Pos('#', line) -1));
            Delete(line, 1, Pos('#', line));

             check for an officer or a crewmember
            if Pos('#', line) > 0 then
              begin
                    get officer title and rank
                    department := Copy(line , 1 ,
                                  Pos('#', line) -1);
                    Delete(line, 1, Pos('#', line));
                    rank := StrToInt(Copy(line , 1 ,
                                    Pos('#', line) -1));
                    Delete(line, 1, Pos('#', line));
                    date  := line;

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

                    m := Copy(date,1 , Pos('/', date) - 1);
                    Delete(date,1 , Pos('/', date));
                    y := date;
                     get date (date must match the object type
              defined in Officer)

                     datePromoted := EncodeDate(StrToInt(y),
                                                StrToInt(m),
                                                StrToInt(d));
                     create officer object and add to array
                     cArr[size] := TOfficer.Create(fullname,crewID,
                                                   department, rank,
                                                   datePromoted);
              end
            else
              begin
                    department := line;
                     creating crewmember object and add to array
                    cArr[size] := TCrewMember.Create(fullname,
                                                     crewID,
                                                     department);
              end;
          end;
      end;

end;
```



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

 //Q6.1 - 4
 find the crew member using crewID
function tCrewMemberManager.findCrewMember(inCID: integer):
integer;
var
  found : boolean;
  pos : integer;
begin

  found := false;
  pos := 0;
   using an appropriate loop
  while NOT(found) AND (pos <= size) do
    begin
      if crewID matches end search
      Inc(pos);

      if cArr[pos].getCrewID() = inCID  then
      begin
        found := true;
      end;

    end;

   return pos
   Result := pos;
end;
```



```
//Q6.2 - 5
Method header correct
function tCrewMemberManager.processTestResults: string;
var
  infile : textfile;
  list for output
  cLine, tLine, output : string;
  crewID, testResult, pos , i : integer;
  f : TOfficer;
  pass : boolean;
begin
  read file
  if FileExists('testResults.txt') <> true then
    begin
      WriteLn('File Missing');
    end
  else
    begin

      AssignFile(inFile, 'testResults.txt');
      Reset(inFile);
      loop using appropriate loop
      for i := 1 to 10 do
        begin
          read the crewid and testResult
          ReadLn(infile , cLine);
          ReadLn(infile , tLine);

          crewID := StrToInt(cLine);
          testResult := StrToInt(tLine);

          //Q6.2.1 - 1
          find the pos of the crewmember with the crewID
          pos := findCrewMember(crewID);

          //Q6.2.2a - 7
          check if the crew member is an officer
          if cArr[pos] is TOfficer then
          begin
            cast element to an Officer
            f := cArr[pos] as TOfficer;
            pass := false;
            check if the officer has met the requirements
            correctly nested if statement
            if (f.getRank = 1) AND (testResult >= 75) then
            begin
              pass := true;
            end
            else if (f.getRank = 2) AND (testResult >= 80) then
            begin
              pass := true;
            end
            else if (f.getRank = 3) AND (testResult >= 90) then
            begin
```



```
              pass := true;
            end
            else if (f.getRank = 4) AND (testResult >= 95) then
            begin
              pass := true;
            end;

             //Q6.2.3 - 6
             6.2.3 - Appended the word officer and their name
             output :=  output + 'Officer' +  f.getFullName();

             //6.2.2a cont.
             check if they passed
            if pass then
            begin
              promote the officer
              f.promote();
              replace object in the array with the new object
              cArr[pos] := f;

              6.2.3 - Appending pass text for officer
              output := output + ' has passed and is promoted to '
                                  f.getTitle() + ' Rank ' +
                                  IntToStr(f.getRank()) + #13#10;
            end
            else if f.getRank() = 5 then
            begin
                6.2.3 - appending captain participation
                 output := output + ' participated in test ' +
                           #13#10;
            end
            else if pass = false then
            begin
              6.2.3 - appending failed for an officer
              output := output +' has failed' + #13#10;
            end;
          end
          //6.2.2b - 6
          else
          begin
            check if the crew member achieved the minimum to pass
             and append pass/fail
            if testResult >= 75 then
            begin
              increment numPromotedCrew
              Inc(TCrewMember.numCrewPromoted);
              promote the crewmember by creating new officer
                 with crewmember details
              using appropriate date
              f := TOfficer.Create(cArr[pos].getFullName() ,
cArr[pos].getCrewID() , cArr[pos].getDepartment() , 1 , Now);
              add new officer object to array
              cArr[pos] := f;
```

```
                6.2.3 - Appending passed message to new officer
                        including new title and rank
                output := output + f.getFullName() +
                          ' has passed and is promoted to ' +
                            f.getTitle() + ' Rank ' +
                            IntToStr(f.getRank()) + #13#10;
              end
              else
              begin
                 6.2.3 - Appending fail for a crewmember
                output := output + cArr[pos].getFullName + '
                          has failed' + #13#10;
              end;
            end;
          end;
          Result := output;
      end;

end;

end.
```

QUESTION 5, 6.3 , 6.4 uCrewMemberUI.pas

```
program CrewMemberUI;
//Q5.1 - 1
 Class header correct
{$APPTYPE CONSOLE}
{$R *.res}

uses
  System.SysUtils,
  DateUtils,
  uCrewMember in 'uCrewMember.pas',
  uOfficer in 'uOfficer.pas',
  uCrewMemberManager in 'uCrewMemberManager.pas';

var

  cm : TCrewMemberManager;

begin
  try
    { TODO -oUser -cConsole Main : Insert code here }

    //Q5.2 - 2
     CrewMemberManager object created
    in appropriate place
    cm := TCrewMemberManager.create();

    //Q5.3 - 2
     Calling toString and displaying all crew members
    WriteLn(cm.toString());

    //Q6.3 - 1
    Call processTestResults in appropriate class
    WriteLn(cm.processTestResults);
    //Q6.4 - 2
    Calling the static numPromotedCrew
    Calling the static numPromotedOfficers
    WriteLn('Number of promoted crew members ' +
              IntToStr(TCrewMember.numCrewPromoted));
    WriteLn('Number of promoted officers ' +
              IntToStr(TOfficer.numPromotedOfficers));
    ReadLn;

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