NATIONAL SENIOR CERTIFICATE EXAMINATION
MAY 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.**

---

IEB Copyright © 2022

## SECTION A

## QUESTION 1

### Question 1.1 – 4 marks

```
SELECT * ✓
FROM tblResearchers ✓
ORDER BY Department, ✓ Surname; ✓
```

### Question 1.2 – 5 marks

```
SELECT *
FROM tblSupervisors ✓
WHERE Department IN ✓ ('MatSci' , 'BusCom') ✓ AND ✓ MaxCapacity >= 3 ✓
```

**alternative solution**

```
SELECT *
FROM tblSupervisors ✓
WHERE Department = ✓ 'MatSci' OR Department =  'BusCom' ✓ AND ✓
MaxCapacity >= 3 ✓
```

### Question 1.3 – 6 marks

```
SELECT ROUND ✓ (AVG(GrantAmount), ✓ 2 ✓) AS AvgGrant ✓
FROM tblResearchProjects
WHERE YEAR(DateStarted) ✓ BETWEEN 2016 AND 2018; ✓
```

**alternative solution using AND**

```
SELECT ROUND ✓ (AVG(GrantAmount), ✓ 2 ✓) AS AvgGrant ✓
FROM tblResearchProjects
WHERE YEAR(DateStarted) ✓ >= 2016 AND YEAR(DateStarted) <= 2018; ✓
```

### Question 1.4 – 5 marks

```
SELECT SupervisorID, COUNT(*) ✓ AS TotalResearchProjects ✓
FROM tblResearchProjects ✓
GROUP BY ✓ SupervisorID; ✓
```

### Question 1.5 – 8 marks

```
SELECT YEAR(DateStarted) ✓ AS Years, SUM(GrantAmount) ✓ AS GrantsGiven✓
FROM tblResearchProjects✓
GROUP BY✓ Year(DateStarted) ✓
HAVING ✓ SUM(GrantAmount) >= 500000✓
```

### Question 1.6 – 8 marks

```
SELECT ProjectName, Firstname &' ' & Surname AS FullName, ✓
SupervisorName, DateStarted
FROM tblResearchProjects, tblResearchers, tblSupervisors,
tblAllocations ✓✓
WHERE tblResearchProjects.ResearchProjectID =
tblAllocations.ResearchProjectID ✓ AND tblAllocations.ResearcherID =
```

IEB Copyright © 2022 

```
tblResearchers.ResearcherID ✓ AND tblResearchProjects.SupervisorID =
tblSupervisors.SuperVisorID✓ AND tblAllocations.isLead = TRUE✓
ORDER BY SupervisorName; ✓ (ACCEPT INNER JOINS)
```

**JAVADB**

```
SELECT ProjectName, Firstname || ' ' || Surname AS FullName, ✓
SupervisorName, DateStarted
FROM tblResearchProjects, tblResearchers, tblSupervisors,
tblAllocations ✓✓
WHERE tblResearchProjects.ResearchProjectID =
tblAllocations.ResearchProjectID ✓ AND tblAllocations.ResearcherID =
tblResearchers.ResearcheID ✓ AND tblResearchProjects.SupervisorID =
tblSupervisors.SuperVisorID AND tblAllocations.isLead = TRUE ✓
ORDER BY ✓ SupervisorName; ✓
```

**MYSQL**

```
SELECT ProjectName, Concat(Firstname,' ',Surname) AS FullName, ✓
SupervisorName, DateStarted
FROM tblResearchProjects, tblResearchers, tblSupervisors,
tblAllocations ✓✓
WHERE tblResearchProjects.ResearchProjectID =
tblAllocations.ResearchProjectID ✓ AND tblAllocations.ResearcherID =
tblResearchers.ResearcherID ✓ AND tblResearchProjects.SupervisorID =
tblSupervisors.SuperVisorID AND tblAllocations.isLead = TRUE ✓
ORDER BY ✓ SupervisorName; ✓
```

## Question 1.7 – 8 marks

```
INSERT INTO ✓ tblSuperVisors ✓ ( SupervisorName, Department,
MaxCapacity ) ✓
SELECT Firstname & ' ' & Surname✓ , Department, 4 ✓
FROM tblResearchers ✓
WHERE ResearcherID ✓ BETWEEN 17 AND 20; ✓
```

**JAVADB**

```
INSERT INTO ✓ tblSuperVisors ✓ ( SupervisorName, Department,
MaxCapacity ) ✓
SELECT Firstname  || ' ' || Surname✓, Department, 4 ✓
FROM tblResearchers ✓
WHERE ResearcherID ✓ BETWEEN 17 AND 20; ✓
```

**MySQL**

```
INSERT INTO ✓ tblSuperVisors ✓ ( SupervisorName, Department,
MaxCapacity ) ✓
SELECT CONCAT(Firstname ,' ' ,Surname)✓, Department, 4 ✓
FROM tblResearchers ✓
WHERE ResearcherID ✓ BETWEEN 17 AND 20; ✓
```

IEB Copyright © 2022 

## Question 1.8 – 6 marks

```
UPDATE✓ tblSupervisors✓ SET Department✓ = LEFT(Department, 3) ✓ & ' & '
&✓ RIGHT(Department, 3) ✓
```

**JAVADB**

```
UPDATE✓ tblSupervisors ✓SET Department ✓= SUBSTR(Department, 1,3) ✓|| '
& ' || ✓SUBSTR(Department, 3, 3) ✓
```

**MySQL**

```
UPDATE✓ tblSupervisors ✓SET Department ✓=  CONCAT( LEFT(Department3) ✓,
' & ' , ✓RIGHT(Department, 3) )✓
```

IEB Copyright © 2022

## JAVA SOLUTION

## QUESTION 2 CANDIDATE CLASS

```
//Question 2
  Q2.1 - 3
✓ header correct
public class Candidate {
    ✓ properties private
    ✓ named and typed correctly
    private String studentID;
    private String name;
    private int year;

    //Q2.2 - 4
    ✓ header correct
    ✓ correct parameter names and types
    public Candidate(String inSID, String inNE, int inYR)
    {
        ✓✓ properties assigned correctly
        studentID = inSID;
        name = inNE;
        year = inYR;
    }
    //Q2.3 - 2
    ✓ correct header and ✓ return types for all three getters
    public String getStudentID()
    {
        return studentID;
    }

    public String getName()
    {
        return name;
    }

    public int getYear()
    {
        return year;
    }
    //Q2.4 - 4
    ✓ header correct
    public String toString()
    {
        ✓ contains all fields
        ✓ fields in correct format
        ✓ returned correctly
        return name + " [" + studentID + "] year " + year;
    }
}
```

IEB Copyright © 2022 

## JAVA SOLUTION

## QUESTION 3 RESEARCHPAPER CLASS

```
//Question 3
//Q3.1 - 3
✓ class header correct
public class ResearchPaper {
    ✓ all properties named and ✓ typed correctly
    private String title;
    private String department;
    private boolean meetsDeptRequiremens;
    private String reqCode;
    private String studentID;

    //Q3.2 - 3
    ✓ constant declared with final/constant
    ✓ named and typed correctly
    ✓ assigned correctly
    public static final int DEANMINPAGES = 200;

    //Q3.3 - 4
    ✓ header correct
    ✓ parameter named correctly
    ✓ typed correctly
    public ResearchPaper(String inTE, String inDT, boolean inMDR, String
inRC , String inSID)
    {
        ✓ properties assigned correctly
        title = inTE;
        department = inDT;
        meetDeptRequirements = inMDR;
        reqCode = inRC;
        studentID = inSID;
    }

    //Q3.4 - 2
    ✓ correct header and ✓ return types for all four getters
    public String getTitle() {
        return title;
    }

    public String getDepartment() {
        return department;
    }

    public String getMeetsDeptRequirements() {
        return meetsDeptRequiremens;
    }

    public String getReqCode() {
        return reqCode;
    }

    public String getStudentID() {
        return studentID;
    }
```

```
//Q6 - 7
    ✓ header correct
    public boolean onDeansList()
    {
        ✓ extract pages, ✓peer reviewed and ✓fieldResearch

        int pages = Integer.parseInt(reqCode.substring(0,
reqCode.length() - 2));

        boolean peerReviewed = reqCode.charAt(reqCode.length()-2) ==
'T';

        boolean fieldResearch = reqCode.charAt(reqCode.length()-1) ==
'T';

       ✓ includes method call to meetsDeptRequirements
       ✓ makes use of the constant
        return meetsDeptRequirements && pages>= DEANMINPAGES &&
peerReviewed && fieldResearch; ✓return correct combined requirements as
boolean
    }
    //Q3.5 - 5
    ✓ header correct
    public String toString()
    {
        String r = "";
       ✓ all fields included
       ✓ format correct
        r = r + "Title: " + title + "\n";
        r = r + "Author: " + studentID + "\n";
       ✓ adding new lines
        r = r + "Meets " + department + "'s requirements: " +
meetsDeptRequirements;

       ✓ correctly returned
        return r;
    }




}
```

## QUESTION 4 & 7.1-2 ,8.1-2 CANDIDATEPAPERMANAGER CLASS

```
//Question 4
//Q4.1 - 1
✓Class header correct
public class CandidatePaperManager {

    //Q4.2 - 2
    ✓ private array of candidate objects
    ✓ size set to 20
    private Candidate cArr[] = new Candidate[20];
```

```
//Question 7
//Q7.1 - 2
✓ //array of ResearchPaper object created with correct size
private ResearchPaper rpArr[] = new ResearchPaper[20];
✓//rpSize variable created and initialize
private int rpSize = 0;


//Q4.3 - 9
✓ constructor header correct
public CandidatePaperManager()
{
    try
    {
       ✓ open file for reading
        Scanner sc = new Scanner(new File("candidates.txt"));

       ✓ loop through all 20 lines
        for (int i = 0; i < 20; i++) {
           ✓ read the line
            String line = sc.nextLine();

           ✓ split the line into the required parts
            String tokens[] = line.split("#");

            String sid = tokens[0];
            String name = tokens[1];
            int year = Integer.parseInt(tokens[2]);
           ✓ create Candidate object
           ✓ add to array
           ✓ correct index
            cArr[i] = new Candidate(sid, name, year);
        }
    }
    catch(FileNotFoundException fne)
    {
        System.out.println("File Missing " + fne)
          ✓ Error message
    }
}
//Q4.4 - 5
✓ header correct
public String displayAllCandidates()
{
    String r = "";
   ✓ loop through array
    for (int i = 0; i < cArr.length; i++) {
       ✓ append to string
       ✓ extra blank line
        r+= cArr[i] + "\n";
    }

    return r; ✓appended string returned
}
```

```
//Q4.5 - 8
✓ Header correct
public void sortByYear()
{
    ✓ loop through the array
     for (int i = 0; i < cArr.length -1; i++) {
        ✓ nested loop
        ✓ correct for sort used
         for (int j = i+1; j < cArr.length ; j++) {
            ✓ compare the object years
            ✓ correct objects referenced for sort
            ✓ correct use of method
             if(cArr[i].getYear() < cArr[j].getYear())
             {
                ✓ swopping elements

                 Candidate temp = cArr[i];

                 cArr[i] = cArr[j];

                 cArr[j] = temp;
             }
         }
    }
}

//Q7.2 - 11
✓ header
public String reviewResearchPapers()
{
    ✓ string to store the accepted and rejected
     String accepted = "Accepted\n";
     String rejected = "Rejected\n";

     try
     {
        ✓ open file
         Scanner sc = new Scanner(new File("researchPapers.txt"));

        ✓ read lines
         while(sc.hasNextLine())
         {
            ✓ split tokens
             String line = sc.nextLine();
             String tokens[] = line.split("#");

            ✓ Create researchPaper object
             ResearchPaper rp = new ResearchPaper(tokens[0] ,
tokens[1] , tokens[2] , tokens[3]);

            ✓ check if it meets requirements
            ✓ must call method
             if(rp.meetsDeptRequirements())
             {
                ✓ add to array
                 rpArr[rpSize] = rp;
                ✓ increase rpsize
                 rpSize++;
```

```
                    accepted += rp + "\n";
                }
                else
                {
                    rejected += rp + "\n";
                }
            }
        }
        catch(FileNotFoundException e)
        {
            System.out.println("File Missing" + e);
        }

        ✓ return string with accepted and reject lists
        return accepted + "\n" + rejected;
    }

    Q8.1 - 17
    ✓header
  public String doctoralCandidates()
    {

        ✓make date formatter
        DateTimeFormatter formatDate =
DateTimeFormatter.ofPattern("YYYY/MM/dd HH:mm:ss");
        ✓format date
        String dateF = LocalDateTime.now().format(formatDate);

        ✓separating departments

        String depts = "MatSci#BusCom#HumArt#LawMed";

        Scanner scDepts = new Scanner(depts).useDelimiter("#");
        String currentDept = "", temp = dateF + "\n";

        ✓getting the list of department doctoral candidates
        while (scDepts.hasNext())
        {
            currentDept = scDepts.next();
            temp = temp + currentDept + "\n";

           ✓loop through all research papers
            for (int i = 0; i < rpSize; i++)
            {
                ✓check if the paper matches the department
                if ( rpArr[i].getDepartment().equals(currentDept))
                {
                      ✓find the research paper's author's name.
                    Candidate candidate =
findCandidate(rpArr[i].getStudentID());

                      ✓add to department list
                    temp = temp + candidate.getName();
                      ✓check if on dean's list
                    if (rpArr[i].onDeansList())
                    {
                        ✓add (D)
                        temp = temp + " (D)";
```

```
                    }
                      ✓add to concatenated list
                      ✓add extra line
                    temp = temp + "\n" + rpArr[i].getTitle() + "\n\n";
                }

            }

        ✓ Create file to write to with correct name
         try
         {
             PrintWriter outFile = new PrintWriter(new
FileWriter("doctors.txt"));

            ✓save to file and close
             outFile.println(temp);

             outFile.close();
         } catch (Exception e)
         {
             System.out.println("Error writing to file " + e);
         }

         return temp;

         }
    public Candidate findCandidate(String sid)
    {
         Candidate c = null;
        ✓loop through candidates
         for (int i = 0; i < cArr.length; i++) {
            ✓check for matching studentid
             if(cArr[i].getStudentID().equalsIgnoreCase(sid))
                 c =  cArr[i];
         }

         return c;
    }


}
```

## QUESTION 5 & 7.3, 8.2 CANDIDATEPAPERUI CLASS

```
//Question 5
//Q5.1 - 1
✓ application class created
public class CandidatePaperUI {
    public static void main (String args[])
    {
        //Q5.2 - 1
        ✓ create CandidatePaperManager object
         CandidatePaperManager cpm = new CandidatePaperManager();

        //Q5.3 - 1
        ✓ calling the displayAllCandidates method
         System.out.println(cpm.displayAllCandidates());
```

```
        //Q5.4 - 2
        ✓ calling the sortByYear method and ✓ redisplaying
         cpm.sortByYear();
         System.out.println(cpm.displayAllCandidates());
        //Q7.3 - 1
        ✓ call the reviewResearchPapers method
         System.out.println(cpm.reviewResearchPapers());

        //Q8.2 - 2
        ✓ call doctoralCandidates method in correct place ✓
         System.out.println(cpm.doctoralCandidates());
    }
}
```

## DELPHI SOLUTION

## QUESTION 2 CANDIDATE CLASS

```
unit uCandidate;

interface
uses SysUtils;

//Q2.1 - 3
✓ Class header correct
type TCandidate = class
  ✓ properties private
  ✓ typed and typed correctly

  private
    studentID : string;
    name : string;
    year : integer;
  public
    constructor Create(inSID : string; inNE:string; inYR:integer);
    function getStudentID() :string;
    function getName() : string;
    function getYear() : integer;
    function toString() : string;
end;

implementation

{ TCandidate }
//Q2.2  - 4
✓ header correct
✓ correct parameter names and types
constructor TCandidate.Create(inSID: string; inNE: string; inYR:
integer);
begin
  ✓✓ properties assigned correctly
  studentID := inSID;
  name := inNE;
  year := inYR;
end;
```

IEB Copyright © 2022 

```
// Q2.3 - 2
✓ Correct header and ✓ return types for all three gettters
function TCandidate.getName: string;
begin
  Result := name;
end;

function TCandidate.getStudentID: string;
begin
  Result := studentID;
end;

function TCandidate.getYear: integer;
begin
  Result := year;
end;

// Q2.4  - 4
✓ header correct
function TCandidate.toString: string;
begin
  ✓ includes all fields
  ✓ fields in correct format
  ✓ returned correctly
  Result :=  name + ' [' + studentID + '] ' + IntToStr(year) ;
end;

end.
```

## QUESTION 3 RESEARCHPAPER CLASS

```
unit uResearchPaper;

interface
  uses SysUtils;
  //Q3.1 - 3
  ✓ class header correct
  type TResearchPaper = class
    private
      ✓ all properties named and ✓typed correctly
       title, department, reqCode, studentID : string;
        meetsDeptRequirements : boolean;
    public
      //Q3.2 - 3
      ✓ constant declared with final/constant
      ✓ named and typed correctly
      ✓ assigned correctly
       const
         DEANMINPAGES = 200;

         constructor Create(inTE : string; inDT:string; inMDR:boolean;
inRC:string ; inSID:string);
         function getTitle() : string;
         function getDepartment() : string;
         function getMeetsDeptRequirements() : boolean;


         function getReqCode() : string;
```

```
        function getStudentID() : string;
        function onDeansList() : boolean;
        function toString() : string;
  end;
implementation

{ TResearchPaper }
```

//Q3.3 - 4
✓ header correct
✓ parameter named correctly
✓ typed correctly

```
constructor TResearchPaper.Create(inTE : string ; inDT : string ; inMDR
: boolean ; inRC : string; inSID: string  );
begin
```

✓ properties assigned correctly

```
  title:=inTE;
  department := inDT;
  meetDeptRequirements := inMDR;
  reqCode:= inRC;
  studentID := inSID;end;
```

//Q3.4 - 2
✓ correct header and ✓ return types for all four getters

```
function TResearchPaper.getDepartment: string;
begin
  Result := department;
end;

function TResearchPaper.getMeetDeptRequirements: boolean;
begin
  Result := meetDeptRequirements;
end;

function TResearchPaper.getReqCode: string;
begin
  Result:= reqCode;
end;

function TResearchPaper.getStudentID: string;
begin
  Result := studentID;
end;

function TResearchPaper.getTitle: string;
begin
  Result:= title;
end;
```

//Q6 - 7
✓ header correct

```
function TResearchPaper.onDeansList: boolean;
var
  peerReviewed, fieldResearch : boolean;
  pages : integer;
begin
```

✓ extract pages, ✓peer reviewed and ✓fieldResearch

IEB Copyright © 2022 

```
  pages := StrToInt(Copy(reqCode,1,reqCode.Length-2));
  peerReviewed := CompareText( Copy(reqCode, reqCode.Length-1,1)  , 'T')
= 0;
  fieldResearch :=  CompareText( Copy(reqCode, reqCode.Length,1)  , 'T')
= 0;
  ✓ includes method call to meetsDeptRequirements
  ✓ makes use of the constant
  Result:= getMeetsDeptRequirements() AND  (pages >= DEANMINPAGES) AND
peerReviewed AND fieldResearch;
end;
✓ return correct combined requirements as boolean

//Q3.5  - 5
✓ header correct
function TResearchPaper.toString: string;
begin

  ✓ all fields included
  ✓ format correct
  Result:=  'Title: ' + title + #13#10;
  Result := Result + 'Author: ' + studentID + #13#10;
  Result := Result + 'Meets ' + department + '''s requirements: ';
  ✓ adding new lines
  ✓ correctly returned
  if getMeetsDeptRequirements() then
    Result:= Result + ' true ' +  #13#10
  else
     Result:= Result + ' false ' +  #13#10;


end;

end.
```

## QUESTION 4 & 7.1-2 ,8.1-2 CANDIDATEPAPERMANAGER CLASS

```
unit uCandidatePaperManager;

interface
  uses SysUtils, uCandidate, uResearchPaper;
//Q4.1 - 1
✓ class header correct
type TCandidatePaperManager = class
  private
    //Q4.2 - 2
    ✓ private array of candidate objects
    ✓ size set to 20
    cArr : array[1..20] of TCandidate;
    //Question 7
    //Q7.1 - 2
    ✓array of ResearchPaper object created with correct size
    rpArr : array[1..20] of TResearchPaper;
    ✓ rpSize created and intialized
    rpSize : integer;
  public
    constructor Create;
```

IEB Copyright © 2022 

```
    function displayAllCandidates : string;
    procedure sortByYear();
    function  reviewResearchPapers() : string;
    function findCandidate(sid : string) : TCandidate;
    function getDeptCandidates(department : string) : string;
    function doctoralCandidates() : string;

end;

implementation

{ TCandidatePaperManager }
//Q4.3 - 9
✓ constructor header correct
constructor TCandidatePaperManager.Create;
var
  infile : textfile;
  line, studentID, name : string;
  year , i: integer;

begin
  if FileExists('candidates.txt') <> true then
    begin
      WriteLn('File Missing'); ✓error message
    end
  else
    begin
      ✓ open file for reading
      AssignFile(infile, 'candidates.txt');
      Reset(infile);

      ✓ loop through all 20 lines
      for i := 1 to length(cArr) do
      begin
        ✓ read the line
        ReadLn(infile,line);
        ✓ split the line into the required parts
        studentID := Copy(line, 1, Pos('#', line) - 1);
        Delete(line, 1, Pos('#', line));
        name := Copy(line, 1, Pos('#', line) - 1);
        Delete(line, 1, Pos('#', line));
        year := StrToInt(line);
        ✓ create Candidate object
        ✓ add to array
        ✓ correct index
        cArr[i] :=  TCandidate.Create(studentID , name, year);
      end;



    end;


end;

//Q4.4 - 5
✓ header correct
function TCandidatePaperManager.displayAllCandidates: string;
```

```
var
  i : integer;
  output : string;
begin
  output := '';
  ✓ loop through array
  for i := 1 to length(cArr) do
    begin
      ✓ append to string
      ✓ extra blank line
       output := output + cArr[i].toString + #13#10;
    end;

  Result := output; ✓appended string returned
end;

//Q4.5 - 8
✓ Header correct

procedure TCandidatePaperManager.sortByYear;
var
  i,j : integer;
  temp : TCandidate;
begin
  ✓ loop through the array
   for i := 1 to length(cArr)-1 do
     begin
      ✓ nested loop
      ✓ correct for sort used
       for j := i+1 to length(cArr) do
         begin
         ✓ compare the object years
         ✓ correct objects referenced for sort
         ✓ correct use of method
           if cArr[i].getYear() <   cArr[j].getYear() then
           begin
           ✓ swopping elements
             temp := cArr[i];
             cArr[i] := cArr[j];
             cArr[j] := temp;
           end;

         end;
     end;
end;


//Q7.2 - 11
✓ header
function TCandidatePaperManager.reviewResearchPapers: string;
var
  accepted , rejected : string;
  inFile : textfile;
  line, title, department, reqCode, studentID : string;
  meetsDeptRequirements : boolean;
  year , i: integer;
  rp : TResearchPaper;
begin
```

✓ string to store the accepted and rejected

```
  accepted := 'Accepted' + #13#10;
  rejected := 'Rejected' + #13#10;

  if FileExists('researchpapers.txt') <> true then
    begin
      WriteLn('File missing');
    end
  else
    begin
      AssignFile(inFile, 'researchpapers.txt');
      Reset(inFile);

      rpSize := 0;
      while NOT EOF(inFile) do
      begin

        ReadLn(infile, line);

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

        department:= Copy(line, 1, Pos('#', line) - 1);
        Delete(line, 1, Pos('#', line));
        meetsDeptRequirements:= StrToBool(Copy(line, 1, Pos('#', line) -
        1));
        Delete(line, 1, Pos('#', line));
        reqCode := Copy(line, 1, Pos('#', line) - 1);
        Delete(line, 1, Pos('#', line));

        studentID:= line;
        rp := TResearchPaper.Create(title,department,
meetsDeptRequirements,reqCode,studentID);
        if rp.getMeetsDeptRequirements() then
          begin
            inc(rpSize);

            rpArr[rpSize] := rp;
            accepted := accepted + rp.toString ;
          end
        else
          begin
            rejected := rejected + rp.toString ;
          end;


      end;

      Result:= accepted + #13#10 + rejected;
```

✓open file

✓ read lines

✓ split tokens

✓ Create researchPaper object

✓ check if it meets requirements

✓ must call method

✓ add to array

✓ increase rpsize

✓ return string with accepted and reject lists

IEB Copyright © 2022

```
    end;


end;
```

Q8.1 - 17
✓ header

```
function TCandidatePaperManager.doctoralCandidates: string;
var
  dept, currentDept, name , title , list: string;
  i : integer;
  f:TextFile;
begin
```

✓make date formatter
✓format date

```
   list := (FormatDateTime('YYYY/MM/dd HH:mm:ss', Now))  + #13#10;

   dept := 'MatSci#BusCom#HumArt#LawMed#';
```

✓separating departments

```
   currentDept := Copy(dept, 1, Pos('#', dept) - 1);
   Delete(dept, 1, Pos('#', dept));
```

✓getting the list of department doctoral candidates

```
   while Length(currentDept) <> 0 do
   begin
     list:= list + currentDept +  #13#10;
```

✓loop through all research papers

```
     for i := 1 to rpSize do
      begin
```

✓checking per department

```
        if CompareText(rpArr[i].getDepartment,currentDept) = 0 then
        begin
```

✓find the research paper's author's name.

```
          name := findCandidate(rpArr[i].getStudentID()).getName();

          title := rpArr[i].getTitle();
```

✓add to department list

```
          list := list + name;
```

✓check if on dean's list

```
          if rpArr[i].onDeansList then
          begin
```

✓add (D)

```
            list := list + ' (D)' + #13#10;
          end
          else
          begin
            list := list + #13#10;
          end;
```

✓add to concatenated list
✓add extra line

```
                list := list + title +  #13#10  +  #13#10;

          end
       end;

       currentDept := Copy(dept, 1, Pos('#', dept) - 1);
       Delete(dept, 1, Pos('#', dept));
    end;


    ✓Create file to write to with correct filename
    AssignFile(f, 'doctors.txt');
    ReWrite(f);

    ✓save to file and close
    Writeln(f,list);
    CloseFile(f);


    Result:= list;
end;

function TCandidatePaperManager.findCandidate(sid:string): TCandidate;
var
  c : TCandidate;
  i : integer;

begin
  c := nil;
  ✓ loop through candidates
  for i := 1 to Length(cArr) do
    begin
    ✓ check for matching studentid
      if CompareText(cArr[i].getStudentID , sid) = 0 then
      begin
        c := cArr[i];
      end;

    end;

  Result := c;
end;




end.
```

## QUESTION 5 & 7.3 , 8.2 CANDIDATEPAPERUI CLASS

```
//Question 5
//Q5.1 - 1
```

```
✓ application class created
program CandidatePaperUI;

{$APPTYPE CONSOLE}

{$R *.res}

uses
  System.SysUtils,
  uCandidate in 'uCandidate.pas',
  uResearchPaper in 'uResearchPaper.pas',
  uCandidatePaperManager in 'uCandidatePaperManager.pas';

var

  cpm : tCandidatePaperManager;
begin
  try
    { TODO -oUser -cConsole Main : Insert code here }
    //Q5.2 - 1
    ✓ create CandidatePaperManager object
    cpm := TCandidatePaperManager.Create();
    //Q5.3  - 1
    ✓ calling the displayAllCandidates method
    WriteLn(cpm.displayAllCandidates());

    //Q5.4 - 2
    ✓ calling the sortByYear method and ✓redisplaying
    cpm.sortByYear();
    WriteLn(cpm.displayAllCandidates());

    //Q7.3 - 1
    ✓ call the reviewResearchPapers method
    WriteLn(cpm.reviewResearchPapers());
    //Q8.2 - 2
    ✓ call the doctoralCandidates method in correct place ✓
    WriteLn(cpm.doctoralCandidates());
    ReadLn;

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

**Total: 150 marks**

IEB Copyright © 2022