



eduresourceza.com

ST BENEDICT'S COLLEGE



SUBJECT	Information Technology	DATE	July 2016
GRADE	12	MARKS	120
EXAMINER	Mrs Kench	MODERATOR	
NAME	MEMO	DURATION	3 Hours and 30min printing
CLASS			

COGNITIVE LEVELS			
LOW ORDER	29 %	MIDDLE ORDER	41 %
		HIGH ORDER	30 %

INSTRUCTIONS	
COMMENT	
TEACHER'S SIGNATURE	
PARENT'S SIGNATURE	

QUESTION 1

SQL

40 MARKS

- 1.1 The Director at one of the branches of the gym wants an alphabetical list of all banned substances with their classification sorted to class then description. (3)

```
SELECT description, class
FROM tblbannedsubstances✓
ORDER BY Class, ✓ Description✓
```

- 1.2 The Director of the Gym would like to know the surname of the member who has been at the gym the longest. (3)

```
SELECT surname✓
From Tblmembers
order by DateJoined✓ limit 1✓
```

- 1.3 Display all members who have not tested positive and hence have no offenses. (5)

```
SELECT *
FROM tblmembers✓
left join ✓ tboffences✓
on tblmembers.memberid=tboffences.memberid✓
WHERE offenceID IS NULL✓
```

OR

```
SELECT *
FROM tblmembers✓
WHERE memberID ✓ NOT IN✓ (SELECT memberID✓ from tboffences) ✓
```

- 1.4 Ruby Johnstone (ID=11) was frequently caught distributing and using banned substances. From an investigation, it was apparent that she had an accomplice, Maria Baker (ID=16). All the records for Ruby Johnstone, concerning her drug abuse and penalty schedule, must be implemented for Maria Baker as well. (5)

```
INSERT INTO tboffences✓ (Date,Penalty,SubstanceID,memberID) ✓
(SELECT Date,Penalty,SubstanceID,16 ✓
FROM tboffences ✓
WHERE memberID=11) ✓
```

- 1.5 Display the total amount of each type of penalty that was issued. Only count the penalty if it was not a suspension. (6)

```
SELECT penalty, COUNT(*)✓
FROM tboffences✓
WHERE penalty✓ != 'Suspended'✓
GROUP BY✓ penalty ✓
```

- 1.6 Suspensions usually last a year. Display the members who have served their suspension, do not include members who are suspended or whose membership has ended. (6)

```
SELECT firstname, lastname, penalty
FROM tblmembers, tboffences✓
WHERE Tblmembers.memberid = tboffences.memberid ✓
and (year(now()) - year(Date)) >1 ✓✓
and penalty != 'Suspended' ✓and penalty !='Membership Ended'✓
```

- 1.7 Elizabeth West (903) terminated her membership. Delete this gym member. (2)

Delete

FROM Tblmembers✓

WHERE id=903✓

- 1.8 Display the names and surnames of members who have had to pay a fine. Any member with multiple offences must only be listed once. (4)

SELECT Distinct firstname, lastname

FROM tbloffences, Tblmembers

WHERE tbloffences.memberid = Tblmembers.memberid
and penalty like 'fine%'

- 1.9 The National Association of Gyms would like a copy of all members facing criminal charges. A unique code is generated for each offender. The code is generated as follows: First letter of the Gender, first three letters of the first name, last two letters of the surname, a random number of two digits, followed by the date of birth. Generate this code for each member; as an example:

'1', 'Mildred', 'Green', 'mgreen0@archive.org', 'Female', '1978-01-29 00:00:00', '2006-08-24 00:00:00' may give us

FMilen001978-01-29

(6)

Select concat(left(gender,1), ✓left(firstname,3),right(lastname,2), ✓

FLOOR(1 + RAND() * 10) ,FLOOR(1 + RAND() * 10) ✓, date(DATE)) as code✓

FROM Tblmembers, tbloffences✓

WHERE Tblmembers.memberid = tbloffences.memberid and penalty = 'Criminal charges'✓

QUESTION 2**OOP****80 MARKS**

```
public class Dist {
//2.1 [3 marks]
    private int [] dArr = new int [5];

//2.2 [5 marks]
    public Dist(String temp)
    {
        int i = 0;
        Scanner sc = new Scanner(temp).useDelimiter(" ");
        while (sc.hasNext())
        {
            dArr[i] = sc.nextInt();
            i++;
        }
    }

//2.3 [4 marks]
    public int totalKMs()
    {
        int total = 0;
        for (int i = 0; i < dArr.length; i++) {
            total = total + dArr[i];
        }
        return total;
    }
}
```

```

    }
}

//3.1 [3 marks]
public class Runner {
    private String name;
    private Dist dists;
    private String raceTime;
    private String club; ✓type ✓name ✓ private

//3.2 [3 marks]
    public Runner(String name, String club, String dists, String
raceTime) {✓
        this.name = name;
        this.dists = new Dist(dists); ✓
        this.raceTime = raceTime;
        this.club = club; ✓

    }

//3.3 [1 mark]
    public String getName() {
        return name; ✓
    }

//3.4 [5 marks]
    public String toString()✓
    {
        return✓ name + " ran a total of " + dists.totalKMs()✓ +
"\nRace Time: ✓ " + raceTime + " Club: " + club + "\n"; ✓
    }

//3.5 [5 marks]
    public int getTimeInSecs()
    {
        int hour = Integer.parseInt("" + raceTime.charAt(0)); ✓
        int min = Integer.parseInt("" + raceTime.substring (2,4)); ✓
        int sec = Integer.parseInt("" + raceTime.substring (5)); ✓
        return sec + min*60 + hour*3600; ✓✓
    }

    public String getClub() {
        return club;
    }

    public int getTotalDist()
    {
        return dists.totalKMs();
    }

    public String getSurname()
    {
        String [] n = name.split(" ");

```

```

        return n[1];
    }
}

public class RunnerArray {

//4.1 [3 marks]
    private Runner ✓rArr[] = new Runner[100]; ✓
    private int rSize = 0; ✓

//4.2 [8 marks]
    public RunnerArray() {
        try {
            Scanner scFile = new Scanner(new File("Runners.txt")); ✓
            String line, name, club, dist, time;

            while (scFile.hasNext()) { ✓
                line = scFile.nextLine(); ✓
                Scanner sc = new Scanner(line).useDelimiter(";"); ✓
                name = sc.next(); ✓✓
                club = sc.next();
                dist = sc.next();
                time = sc.next();
                sc.close();
                rArr[rSize] = new Runner(name, club, dist, time); ✓
                rSize++; ✓
            }
            scFile.close();
        } catch (FileNotFoundException f) {
            System.out.println("File Not Found - check file name or
path");
        }
    }

//4.3 [4 marks]
    public String toString() { ✓
        String temp = "";
        for (int i = 0; i < rSize; i++) { ✓
            temp = temp + rArr[i] + "\n"; ✓
        }
        return temp; ✓
    }

//4.4 [8 marks]
    public String getWinner() {
        String temp = "";
        int lo = rArr[0].getTimeInSecs(); ✓
        String loName = rArr[0].getName();

        for (int i = 1; i < rSize; i++) { ✓
            if (rArr[i].getTimeInSecs() < lo) { ✓
                lo = rArr[0].getTimeInSecs(); ✓
                loName = rArr[0].getName();
            }
        }
    }
}

```

```

        return loName + " won with a time of " + convertTime(lo) + ✓
    "\n\n";
}

```

```

private String convertTime(long t)
{
    long h = t / 3600;
    long m = (t % 3600) / 60; ✓
    long s = (t % 3600) % 60; ✓
    String mS= "", sS= "";
    if (m < 9)
        mS = "0" + m;
    else mS = "" + m;
    if (s < 9)
        sS = "0" + s;
    else sS = "" + s; ✓
    return h + ":" + mS + ":" + sS;
}

```

//4.5 [9 marks]

```

public void sort() {✓
    for (int i = 0; i < rSize - 1; i++) {✓
        for (int j = i + 1; j < rSize; j++) {✓
            if (rArr[i].getClub().compareTo(rArr[j].getClub()) ✓ >
0) {

                Runner temp = rArr[i]; ✓
                rArr[i] = rArr[j]; ✓
                rArr[j] = temp;
            }
            else
                if (rArr[i].getClub().equals(rArr[j].getClub())) {✓
                    if
(rArr[i].getSurname().compareTo(rArr[j].getSurname()) > 0) {✓
                        Runner temp = rArr[i]; ✓
                        rArr[i] = rArr[j];
                        rArr[j] = temp;
                    }
                }
            }
        }
    }
}

```

//6.1 [13 mark]

```

public String clubStats()
{
    String [] clubArr = {"Jeppe", "BCC", "Rand Road
Warriors", "Pirates"};
    int [] totalArr = new int [4];
    int [] sumTimeArr = new int [4];
    int [] countArr = new int[4]; ✓

    String temp = "";
    int sumTime = 0;
}

```

```

for (int i = 0; i < rSize; i++) {
    for (int j = 0; j < clubArr.length; j++) {
        if (rArr[i].getClub().equals(clubArr[j])) {
            totalArr[j] += rArr[i].getTotalDist();
            sumTimeArr[j] += rArr[i].getTimeInSecs();
            countArr[j]++;
        }
    }
}

for (int i = 0; i < sumTimeArr.length; i++) {
    long ave = Math.round((double) sumTimeArr[i]/countArr[i]);

    temp += clubArr[i] + " has a total dist of " + totalArr[i]
+ "\n"
        + "and an average time of " + convertTime(ave)
+ "\n\n";
}

return temp;
}

}

public class Gr12PracPrelim {

    public static void main(String[] args) {

//5.1 [2 marks]
        RunnerArray ra = new RunnerArray();

//5.2 [1 mark]
        System.out.println(ra);

//5.3 [1 mark]
        System.out.println(ra.getWinner());

//5.4 [1 mark]
        ra.sort();
        System.out.println(ra);

//6.1 [1 mark]
        System.out.println(ra.clubStats());
    }
}

```