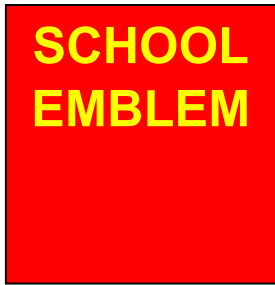




eduresourceza.com

**INFORMATION TECHNOLOGY: PAPER I**

Time: 3 hours

150 marks

Student Name	
---------------------	--

EXAMINERS: Cluster

MODERATORS: Another Cluster

PLEASE READ THE FOLLOWING INSTRUCTIONS CAREFULLY

1. This question paper consists of **15** pages. Please check that your question paper is complete.
2. This question paper is to be answered using object-oriented programming principles. Your program must make sensible use of methods and parameters.
3. This paper is divided into two sections. All candidates must answer both sections.
4. This paper is set in programming terms that are not specific to any particular programming language (Java/Delphi) or database (Access/MySQL/JavaDB).
5. Ensure that you answer the questions in the manner described because marks will be awarded for your solution according to the specifications that are given in the question.
6. Only answer what is asked in each question. For example, if the question does not ask for data validation, then no marks are awarded for it, and therefore no code needs to be written for data validation.
7. If you cannot get a section of code to work, comment it out so that it will not be executed and so that you can continue with the examination. If possible, try to explain the error to aid the marker.
8. When accessing files from within your code, DO NOT use full path names for the files, as this will create problems when the program is marked on a computer other than the one you are writing on. Merely refer to the files using their names and extensions, where necessary.
9. Your programs must be coded in such a way that they will work with any data and not just the sample data supplied or any data extracts that appear in the question paper. You are advised to look at the supplied data files carefully.
10. Make sure that routines such as searches, sorts and selections for arrays are developed from first principles, and that you do not use the built-in features of a programming language for any of these routines.
11. All data structures must be defined and declared by you, the programmer. You may not use components provided within the interface to store and later retrieve data.
12. Read the whole question paper before you choose a data structure. You may find that there could be an alternative method of representing the data that will be more efficient considering the questions that are asked in the paper.

13. You must save all your work regularly on the disk you have been given, or the disk space allocated to you for this examination. You should also create a backup of the original files before you start in case the original version is accidentally modified by your solution.
14. If your examination is interrupted by a technical problem such as a power failure, you will, when you resume writing, be given only the time that was remaining when the interruption began, to complete your examination. No extra time will be given to catch up on work that was not saved.
15. Make sure that your examination number appears as a comment in every program that you code as well as on every page of the hard copy that you hand in.
16. Print a code listing of all the programs/classes/output files that you code. Printing must be done after the examination. You will be given half an hour to print after the examination is finished. Your teacher will tell you what arrangements have been made for the printing of your work.
17. You should be provided with the following two folders (in bold) and files. These files are to be used as data for this examination. Note that the database file is provided in MS Access, format.

Section A:

TourDeFrance2022.mdb

TourDeFrance2022.sql

SQL AnswerSheet.rtf

Section B:

overallstage10.txt

stage11results.txt

SCENARIO:

The Tour de France is an annual men's multiple-stage bicycle race. It consists of 21 stages, each a day long, over the course of 23 days covering a total of 3 343 km.

Riders from many different countries enter the race. Each country records the number of riders in the race, the number of riders who have achieved a ranking and if any of the riders have won a stage.

There are four jersey colours awarded after each day's race for positions earned.

- The current overall Race Leader receives the Yellow Jersey.
- The best Sprinter receives the Green Jersey.
- The King of the Mountain (best rider over the immense climbs) receives the Red polka dot Jersey.
- The Best Young Rider receives the White Jersey.

The rider's full name, age¹ and current position is also stored.

SECTION A STRUCTURED QUERY LANGUAGE

Data of countries, riders and jerseys are stored in a database, **TourDeFrance2022.mdb**.

tblCountries contains the details of the countries whose riders are participating in the Tour de France.

Field Name	Data Type	Description
<u>CountryID</u>	AutoNumber	A unique auto-numbered identification number for each country.
CountryName	Text	The name of the country.
TotalRiders	Integer	The total number of riders from the country.
TotalRankedRiders	Integer	The number of riders of the country who ranked in the first 135 riders.
StageWins	Boolean	Whether or not the country's riders won any stages.

Sample of the data in the table **tblCountries**:

CountryID	CountryName	TotalRiders	TotalRankedRiders	StageWins
1	Australia	9	5	☒
2	Austria	6	5	☐
3	Belgium	18	15	☒
4	Canada	4	2	☒
5	Colombia	3	3	☐
6	Denmark	10	6	☒
7	France	32	25	☒
8	Germany	9	7	☐
9	Great Britain	8	7	☒
10	Israel	1	1	☐

¹ Strictly speaking it is better to store a date of birth rather than a age, but for simplicity purposes age is used.

tblJerseys contains information about the jerseys and which riders won them.

Field Name	Data Type	Description
<u>JerseyID</u>	AutoNumber	A unique auto-numbered identification number for each jersey.
Colour	Text	The colour of the jersey.
Description	Text	The description of the jersey.
RiderID	Integer	The RiderID who was awarded this jersey at the end of the race. This is a foreign key to tblRiders .

Sample of the data in the table **tblJerseys**:

JerseyID	Colour	Description	RiderID
1	Yellow	Overall best rider	18
2	Green	Best sprinter	16
3	Red polkadot	Best rider over the immense climbs	18
4	White	Best young rider	1

tblRiders contains information regarding the participants in the 2022 Tour de France.

Field Name	Data Type	Description
<u>RiderID</u>	AutoNumber	A unique auto-numbered identification number for each rider.
RiderName	Text	The name of the rider.
RiderSurname	Text	The surname of the rider.
Age	Integer	The age of the rider.
Position	Integer	The position the rider obtained.
CountryID	Integer	The CountryID of the country the rider is riding for. This is a foreign key to tblCountries.

Sample of the data in the table **tblRiders**:

RiderID	RiderName	RiderSurname	Age	Position	CountryID
1	Tadej	Pogačar	23	2	22
2	George	Bennett	32		16
3	Mikkel	Bjerg	23	124	6
4	Vegard Stake	Laengen	33		17
5	Rafał	Majka	32		18
6	Brandon	McNulty	24	20	25
7	Marc	Soler	28		24
8	Marc	Hirschi	23	127	25
9	Primož	Roglič	32		22
10	Tiesj	Benoot	28	36	3

QUESTION 1

Write SQL queries for each of the following. Insert your answers in the Word document named **SQL Answersheet**.

- 1.1 Display all the information of countries whose riders have won a stage. Sort results in alphabetical order of the country name.

The correct output is shown below:

CountryID	CountryName	TotalRiders	RidersInCompetition	StageWins
1	Australia	9	5	☒
3	Belgium	18	15	☒
4	Canada	4	2	☒
6	Denmark	10	6	☒
7	France	32	25	☒
9	Great Britain	8	7	☒
14	Luxembourg	3	2	☒
15	Netherlands	10	8	☒
22	Slovenia	5	4	☒

(4)

- 1.2 Write a query to display all the data of riders who are 25 years old or younger or older than 35 years who have earned a position.

A sample of the correct output is shown below:

RiderID	RiderName	RiderSurname	Age	Position	CountryID
1	Tadej	Pogcar	23	2	22
3	Mikkel	Bjerga	23	124	6
6	Brandon	McNulty	24	20	25
8	Marc	Hirschia	23	127	25
16	Jonas	Vingegaarda	25	1	6
17	Geraint	Thomasa	36	3	9
20	Filippo	Gannaa	25	95	11
21	Thomas	Pidcocka	22	17	9
29	Stan	Dewulfa	24	65	3
41	Fabio	Jakobsen	25	130	15

(5)

- 1.3 Write a query to list the top 10 riders. Display the rider's name, surname, and position.

The correct output is shown below:

RiderName	RiderSurname	Position
Jonas	Vingegaarda	1
Tadej	Pogcar	2
Geraint	Thomasa	3
David	Gaudu	4
Aleksandr	Vlasov	5
Nairo	Quintana	6
Romain	Bardet	7
Louis	Meintjes	8
Alexey	Lutsenko	9
Adam	Yatesa	10

(4)

- 1.4 Write a query to display the name, surname and age of all the youngest riders. Your query should work for any number of riders with the same lowest age.

The correct output is shown below:

RiderName	RiderSurname	Age
Kevin	Vermaerkea	21
Quinn	Simmonsa	21

(6)

- 1.5 Generate a code for countries whose riders are all ranked in the competition, meaning that the total number of riders in the country is the same as the number of riders.

The code must consist of the first three letters on the country name and a random number that lies between 1 and 99. Show the country name, the total number of riders, the total number of ranked riders and the generated code.

The correct output is shown below:

CountryName	TotalRiders	TotalRankedRiders	Code
Colombia	3	3	Col30
Israel	1	1	Isr39
Kazakhstan	3	3	Kaz5
Latvia	2	2	Lat7
Russia	2	2	Rus19
Slovakia	1	1	Slo73
South Africa	2	2	Sou7
Switzerland	4	4	Swi18

NOTE: The random numbers generated by your database will be different than that seen in the sample data above

(7)

- 1.6 Write a query to list the average position of the riders in each country. Name the generated field **AvePosition**. Show only countries whose average is 30 or less.

The correct output is shown below:

CountryName	AvePosition
Colombia	20.6667
Luxembourg	30.0000

(8)

- 1.7 Write a query to list the riders who have qualified for a coloured jersey. Display the rider's name, surname, country name and the colour of the jersey.

The correct output is shown below:

RiderName	RiderSurname	CountryName	Colour
Daniel Felipe	Marta-neza	Colombia	Yellow
Jonas	Vingegaarda	Denmark	Green
Daniel Felipe	Marta-neza	Colombia	Polkadot
Tadej	Pogcar	Slovenia	White

(6)

- 1.8 A new competitor needs to be added to the database. Write one query that will add this rider to the database using existing data. The new rider, Sebastian Schonberger ended in position 34. To complete the data set of the new rider, use data from fellow rider Mikkell Frolich (**RiderID 45**).

(6)

- 1.9 The position for Yves Lampaert (**RiderID 46**) was incorrectly entered as 130. This should be 120. Write a query that will change 120 to 130 for **RiderID 46**.

(4)

50 marks

SECTION B OBJECT-ORIENTATED PROGRAMMING

The Tour de France is an annual men's multiple-stage bicycle race. It consists of 21 stages, each a day long, over the course of 23 days covering a total of 3 343 km. The time for each rider is updated at the end of each day to produce a new time. The time is stored in the format hh:mm:ss. After a few days the accumulated hours will be greater than 24.

You are tasked with creating a test program to use a text file containing a Stage's results of one day and update a second text file containing the overall results for the **Yellow** and **Green** Jerseys.

- The Race Leader with the lowest accumulated time across the stages receives the Yellow Jersey.
- The best Sprinter receives the Green Jersey. A green jersey is awarded based on points achieved and not the time.

You will use a text file called **overallstage10.txt** that has the accumulated rider's times and points of the first 10 stages sorted in ascending order of overall time. The **stage11results.txt** text file must be used to update the times and points of the existing riders who have completed stage 11

Each line text file called **overallstage10.txt** contains a rider's name, the team he rides for, and his current overall (total) race time. Some riders have an extra field that holds his total points for sprinting.

For example, at the end of stage 10, POGACAR has a total time of 37 hours, 11 minutes and 28 seconds, and has accumulated 139 Sprint points. The hours value of the rider's total time is greater than 24.

A sample of 40 riders has been used to test your code. An extract of the file is shown below:

Name#Team#OverAllTime#Points

```
POGACAR,UAE TEAM EMIRATES,37:11:28,139
SANCHEZ,BAHRAIN VICTORIOUS,37:13:18,
KONRAD,BORA - HANSGROHE,37:19:16,
URAN,EF EDUCATION - EASYPOST,37:21:55,
BENOOT,JUMBO - VISMA,37:27:44,
MOLLEMA,TREK - SEGAFREDO,37:41:46,
VAN AERT,JUMBO - VISMA,37:43:07,284
VELASCO,ASTANA - QAZAQSTAN TEAM,37:44:11,
BETTIOL,EF EDUCATION - EASYPOST,37:44:41,
VAN BAARLE,INEOS GRENADIERS,37:49:28,
...
TURGIS,TOTALENERGIES,39:24:24,
```

The text file **stage11results.txt** contains the results for Stage 11. Not all riders finished stage 11.

Each line contains a rider's name, and the time he took to complete stage 11. Some riders have an extra field that shows how many Sprint points he scored for stage 11.

For example, DILLER completed stage 11 in a time of 17 hours 20 minutes and 2 seconds. and scored 9 points for Stage 11's Sprint section.

An extract of the file is shown below:

Name#TimeForStage11#Points

DILLIER	17:20:02	9
KONRAD	17:20:03	8
NIV	17:20:04	
TUSVELD	17:20:05	
BAGIOLI	17:21:06	
URAN	17:21:55	
DEWULF	17:22:04	15
SANCHEZ	17:22:05	
BETTIOL	17:23:33	
BISSEGGER	17:23:43	
...		
BJERG	17:32:33	

QUESTION 2

The **YellowRider** class was designed based on the following class diagram. It indicates the fields and methods required. Note the method names and parameter names in the diagram.

YellowRider
Fields: - name : string - team : string - overAllTime: DateTime + <u>JERSEYCOLOUR : string = "[YELLOW JERSEY]"</u>
Methods: + Constructor(n : string, t : string, o : DateTime) + getName() : string + getTeam() : string + getOverAllTime () : DateTime + toString() : string + updateOverAllTime (newTime : string)

- 2.1 Create a new class called **YellowRider**. (1)
- 2.2 Add the **name**, **team** and **overall** fields as indicated in the class diagram. (2)
- 2.3 Create the constant **JERSEYCOLOUR** field as indicated in the class diagram. This field must belong to the class and be visible from outside the class. (2)
- 2.4 Code a constructor method that will accept parameters to initialise the **name**, **team** and **overall** fields of the class. (3)
- 2.5 Code accessor methods for the **name**, **team** and **overall** fields. (2)
- 2.6 Code the **toString** method to return a string combining all the fields in the following format:
<name><tab><team><tab><overall time>
For example: SANCHEZ BAHRAIN VICTORIOUS 37:11:39 (2)

Note: The **updateOverAllTime** method will coded in **Question 7**.

[12]

QUESTION 3

The **GreenRider** class was designed with the following class diagram. This class is a subclass (child) of the **YellowRider** class and will be used to create objects that will store riders in contention for the Green Jersey. The class diagram indicates the fields and methods required. Note the method names and parameter names in the diagram.

GreenRider
Fields: - points: integer + <u>JERSEYCOLOUR : string = "[GREEN JERSEY]"</u>
Methods: + Constructor(n : string, t : string, o : DateTime, p: integer) + getPoints() : integer + setPoints(int p) + toString() : string

- 3.1 Create a new class called **GreenRider**. It is a subclass (child) of **YellowRider**. (2)
- 3.2 Create the **points** field as indicated in the class diagram. (2)
- 3.3 Create the constant **JERSEYCOLOUR** field as indicated in the class diagram. This field must belong to the class and be visible from outside the class. (1)
- 3.4 Code the constructor to initialise the parent class fields and set the class field. (3)
- 3.5 Code the indicated accessor and mutator for the **points** field. The **setPoints** method must *add points* to the **points** field. (2)
- 3.6 Code the **toString** method to override the superclass method and return a string with the addition of **points** field value using the **toString** method in the super class. Use the following format:

<name><tab><team><tab><overall time>

For example: POGACAR UAE TEAM EMIRATES 37:11:28 139
(2)

[12]

QUESTION 4

- 4.1 Create a new class called **RiderManager**. (1)
- 4.2 Create an array called **arrRiders** that can store 200 **YellowRider** objects and an integer counter called **size** to keep track of how many riders are stored in the array. (4)
- 4.3 Code a constructor method for the **RiderManager** class. The method will receive the name of a text file as a parameter. It should read the contents of the text file and populate the **YellowRider** array you have coded in the previous question. The method should do the following:
- Open the file for reading.
 - Loop through the lines of the text file. For each iteration of the loop:
 - Read each line and extract the name, team name and overall time.
 - Convert the rider's time to a **DateTime** object by doing the following.
 - isolate the hours, minutes and seconds from the time using string handling techniques.
 - deduct 24 hours from the hours field. (You may assume all hours will be greater than 24)
 - create a **DateTime** object using the 2022 for the year, 7 for the month, 1 for the day the hours (less 24), minutes and seconds values.
- For example, POGACAR's time is a string in the format 37:11:28. It must be converted to a **DateTime** object with the value: 2022-07-01T13:11:28
- Determine if the line contains data for a **YellowRider** or **GreenRider** object.
 - Use the appropriate constructor and add the object to the array.
 - Increment **size**. (16)
- 4.4 Code a **toString** method to return the information of all the riders in the array as a single string. Each rider's information should be on a new line. (5)
- 4.5 Code a method called **searchRider** to return the integer position of a rider in the array called **arrRiders** given the rider's name. Return -1 if no rider is found. (6)

Note: The **searchRider** method will be used in **Question 7**.

[32]

QUESTION 5

- 5.1 Create a new class called **LeTourUI** that will provide a simple text-based interface. (1)
- 5.2 Create a new **RiderManager** object for the results of stage 10 of the race. The results are stored in the text file **overallstage10.txt**. Send the text file name as a string parameter to the **RiderManager** constructor. (2)
- 5.3 Call the appropriate method to display all the riders' information. (1)

The first 14 lines of output is shown below:

POGACAR	UAE TEAM EMIRATES	2022-07-01T13:11:28	139
SANCHEZ	BAHRAIN VICTORIOUS	2022-07-01T13:13:18	
KONRAD	BORA - HANSGROHE	2022-07-01T13:19:16	
URAN	EF EDUCATION - EASYPOST	2022-07-01T13:21:55	
BENOOT	JUMBO - VISMA	2022-07-01T13:27:44	
MOLLEMA	TREK - SEGAFREDO	2022-07-01T13:41:46	
VAN AERT	JUMBO - VISMA	2022-07-01T13:43:07	284
VELASCO	ASTANA - QAZAQSTAN TEAM	2022-07-01T13:44:11	
BETTIOL	EF EDUCATION - EASYPOST	2022-07-01T13:44:41	
VAN BAARLE	INEOS GRENADIERS	2022-07-01T13:49:28	
DILLIER	ALPECIN - DECEUNINCK	2022-07-01T13:53:05	
TUSVELD	TEAM DSM	2022-07-01T14:00:42	
TRATNIK	BAHRAIN VICTORIOUS	2022-07-01T14:06:15	
LAPORTE	JUMBO - VISMA	2022-07-01T14:10:21	114

[4]

QUESTION 6

- 6.1 Code a new method called **findFastestGreenRider** to find the rider with the most points and return a **GreenRider** object. (8)
- 6.2 Code a typed method called **listJerseyRiders**. It should return the first rider in the array as the current fastest Yellow Jersey rider (rider with the best overall time), as well as the Green Jersey rider (rider with the most points) as a single string. Use the appropriate class constant from the **YellowJersey** and **GreenJersey** classes as headings. (6)
- 6.3 In the **LeTourUI** class add code to call the **listJerseyRiders** method and display the result on the screen.

The correct output is shown below:

[YELLOW JERSEY]			
POGACAR	UAE TEAM EMIRATES	2022-07-01T13:11:28	139
[GREEN JERSEY]			
VAN AERT	JUMBO - VISMA	2022-07-01T13:43:07	284

(2)

[16]

QUESTION 7

Return to the **YellowRider** class.

- 7.1 Refer to the class diagram in Question 2 and code the void method called **updateOverAllTime**. The method must receive a rider's time for stage 11 as a string. The hours, minutes and seconds values must be extracted from the string parameter and added to the **overAllTime** field to update the **overAllTime** field of each rider with stage 11's times. (6)

Return to the **RiderManager** class.

- 7.2 Code a typed method called **stageResults** that will return a string of all the riders who have finished stage 11 with their updated times and points. Use the text file called **stage11results.txt**. Note that not all riders finished stage 11

Use the **searchRider** and **updateOverAllTime** methods you previously coded in your solution.

The method **stageResults** must:

- Open the file called **stage11results.txt**.
- Loop through the lines of the text file. For each iteration of the loop:
 - Extract the rider's name and search for the rider in the array called **riderArr**.
 - If the rider is found:
 - Update the rider's time with extracted time.
 - If points have been awarded for a sprint:
 - Add the points value to rider's current points if the rider is a **GreenJersey** object; alternatively, convert the rider to a **GreenJersey** object using their existing **YellowJersay** values and the extracted points.
 - Add the rider to a string that will be returned at the end of the method.

(16)

7.3 In the **LeTourUI** class, call the **stageResults** method and display the results of the riders who finished stage 11 with their updated times and/or points.

The correct output is shown below:

DILLIER	ALPECIN - DECEUNINCK	2022-07-02T07:15:05	9
KONRAD	BORA - HANSGROHE	2022-07-02T06:42:16	8
NIV	ISRAEL-PREMIER TECH	2022-07-02T07:45:26	
TUSVELD	TEAM DSM	2022-07-02T07:25:42	
BAGIOLI	QUICK-STEP ALPHA VINYL TEAM	2022-07-02T08:10:58	
URAN	EF EDUCATION - EASYPOST	2022-07-02T07:37:55	
DEWULF	AG2R CITROEN TEAM	2022-07-02T07:40:24	15
SANCHEZ	BAHRAIN VICTORIOUS	2022-07-02T06:40:18	
BETTIOL	EF EDUCATION - EASYPOST	2022-07-02T07:40:41	
BISSEGGER	EF EDUCATION - EASYPOST	2022-07-02T08:37:33	
TRATNIK	BAHRAIN VICTORIOUS	2022-07-02T08:13:15	11
LAPORTE	JUMBO - VISMA	2022-07-02T08:27:21	114
TURGIS	TOTALENERGIES	2022-07-02T09:43:24	
VAN KEIRSBULCK	ALPECIN - DECEUNINCK	2022-07-02T08:57:23	
VAN POPPEL	BORA - HANSGROHE	2022-07-02T09:12:28	
BARTHE	B&B HOTELS - KTM	2022-07-02T08:00:52	
DAINESE	TEAM DSM	2022-07-02T08:07:30	
VAN DER HOORN	INTERMARCHE	2022-07-02T08:05:37	17
VAN HOOYDONCK	JUMBO - VISMA	2022-07-02T08:06:31	
PETIT	INTERMARCHE	2022-07-02T07:56:03	
CAPIOT	TEAM ARKEA - SAMSIC	2022-07-02T07:55:50	72
DEGENKOLB	TEAM DSM	2022-07-02T07:55:31	
NIELSEN	EF EDUCATION - EASYPOST	2022-07-02T07:49:19	129
SAGAN	TOTALENERGIES	2022-07-02T08:15:16	86
PHILIPSEN	ALPECIN - DECEUNINCK	2022-07-02T07:54:35	109
MOZZATO	B&B HOTELS - KTM	2022-07-02T08:12:16	
BAUER	TEAM BIKEEXCHANGE-JAYCO	2022-07-02T08:12:19	
TORRES BARCELO	MOVISTAR TEAM	2022-07-02T09:13:12	
VAN MOER	LOTTO SOUDAL	2022-07-02T08:21:17	13
VERMEERSCH	LOTTO SOUDAL	2022-07-02T08:12:26	
BJERG	UAE TEAM EMIRATES	2022-07-02T09:16:25	

(2)

[24]

100 marks