EXAMINATION	NATIONAL SENIOR CERTIFICATE
GRADE	12
DATE	NOVEMBER 2024
SUBJECT	INFORMATION TECHNOLOGY
PAPER	1
MARK TOTAL	150
DURATION (HOURS)	3
NUMBER OF PAGES	24

SOUTH AFRICAN COMPREHENSIVE ASSESSMENT INSTITUTE

SUID-AFRIKAANSE KOMPREHENSIEWE ASSESSERINGSINSTITUUT

INSTRUCTIONS TO CANDIDATES

1. In your examination folder, DATA2024 you'll find the folders and incomplete programs for four questions.
2. Rename the folder DATA2024 by adding on (appending) your final examination number e.g. Data2024_62493.
3. You may make backup copies of the folder in a backup folder of your own, **but you should do all your examination work in your renamed** Data2024_XXX folder.
4. **YOU ARE REQUIRED TO ANSWER ALL FOUR QUESTIONS.**
5. Two blank pages are included at the back of the paper for planning purposes.
6. During the control session you should check all your units, namely **Question1_U, Question2_U, clsQuestion3_U, Question3_U** and **Question4_U.**
   - Can the project open and run?
   - Is your exam number entered as a comment in the unit?
7. You may comment out a specific part of your code if that part doesn't compile. All commented code will be marked if there is no alternative code.
8. Notes on **QUESTION 2**:

   Field names for the tables are included as comments, you may copy and paste these field names in your answers for **QUESTION 2**.
9. The files that you need to complete this question paper have been provided to you on the disk space allocated to you.

   The following list of files will be available in the folder named **DATA2024:**

QUESTION 1:
15 image files
Question1_P.dpr
Question1_P.dproj
Question1_P.res
Question1_U.dfm
Question1_U.pas

QUESTION 2:
Esports - Copy.mdb
Esports.mdb
Question2_P.dpr
Question2_P.dproj
Question2_P.res
Question2_U.dfm
Question2_U.pas
TDBConnection_U.pas

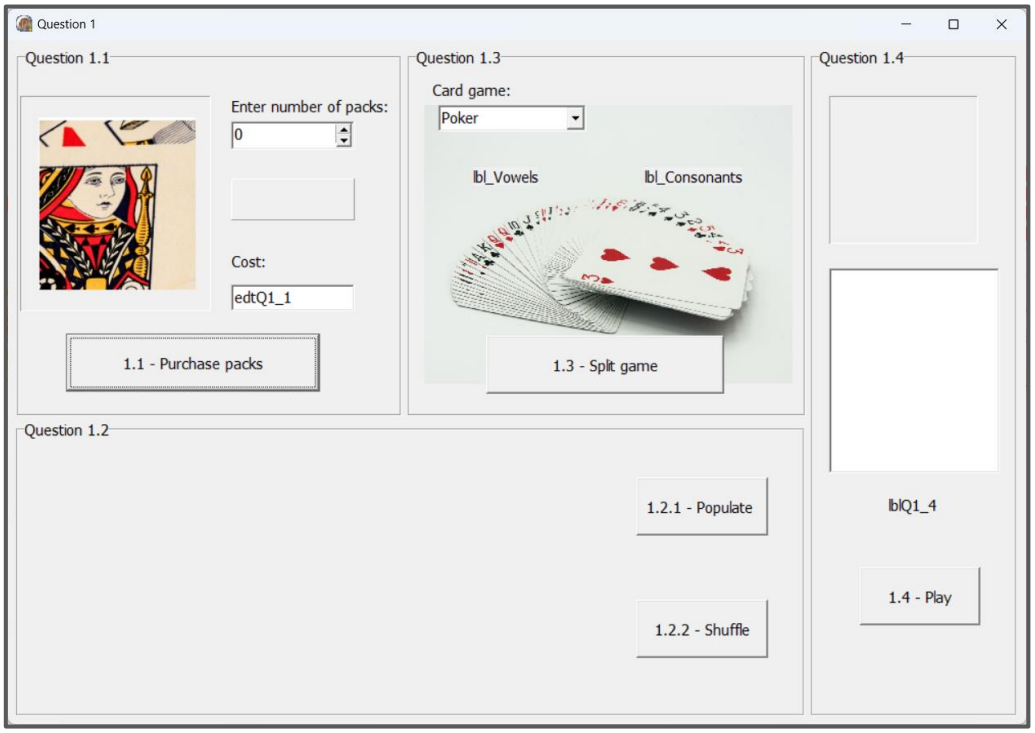
QUESTION 3:
clsQuestion3_U.pas
Question3_P.dpr
Question3_P.dproj
Question3_P.res
Question3_U.dfm
Question3_U.pas
Folder with images

QUESTION 4:
Question4_P.dpr
Question4_P.dproj
Question4_P.res
Question4_U.dfm
Question4_U.pas
Folder with images

SECTION A

QUESTION 1: GENERAL PROGRAMMING

Open the incomplete **Question1_p.dpr** program in your **DATA2024/Question 1** folder. The following screen shot shows the interface:



1.1 **btnQ1_1 [1.1 – Purchase packs]**

A person gets a free deck/pack of cards for every 5 packs of cards bought. Therefore, you pay only for four card packs.

Thus, if a person buys 10 packs of cards, he/she will only pay for 8.

The constant, **rPrice,** per pack of cards is set to R20.

Write code to do the following:

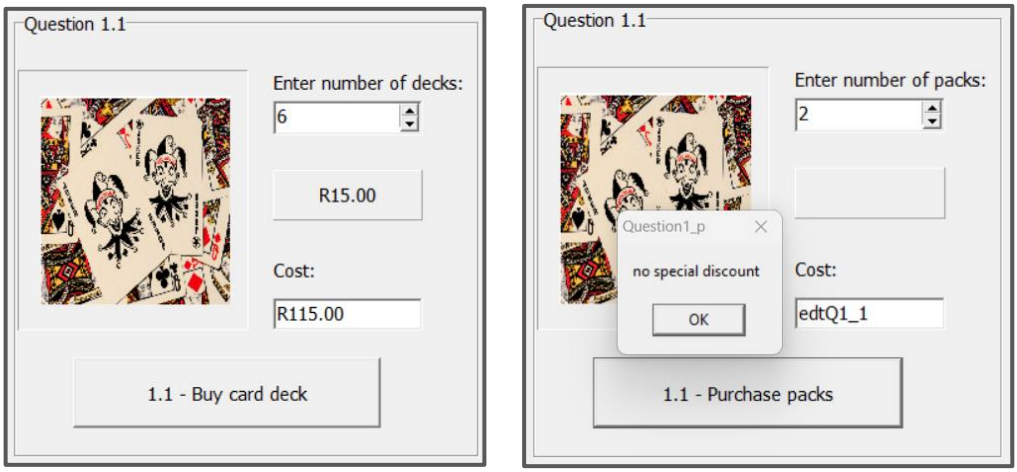
1.1.1
- Set the picture that is displayed in the **imgQ1_1** image to be displayed in its entirety. (See the sample output screen on the next page.)
- Extract the number of card packs to buy from the **spnQ1_1** spin edit.
- Use the constant **rPrice** to calculate the cost for the number of card packs. (3)

1.1.2 Test if the number of card packs is less than 5.
- Display the message "No special discount", if the number of card packs is less than 5.
- Calculate the amount due by subtracting the cost of one card pack for every 5 packs of cards. (7)

1.1.3 Calculate 15% VAT on the amount due and display the VAT on the **pnlQ1_1** panel. (5)

1.1.4 Calculate the final amount (amount due + VAT) on the **edtQ1_1** edit box, as currency. (3)

Example of output:



[18]

1.2 1.2.1 **btnQ1_2_1 [1.2.1 – Populate]**

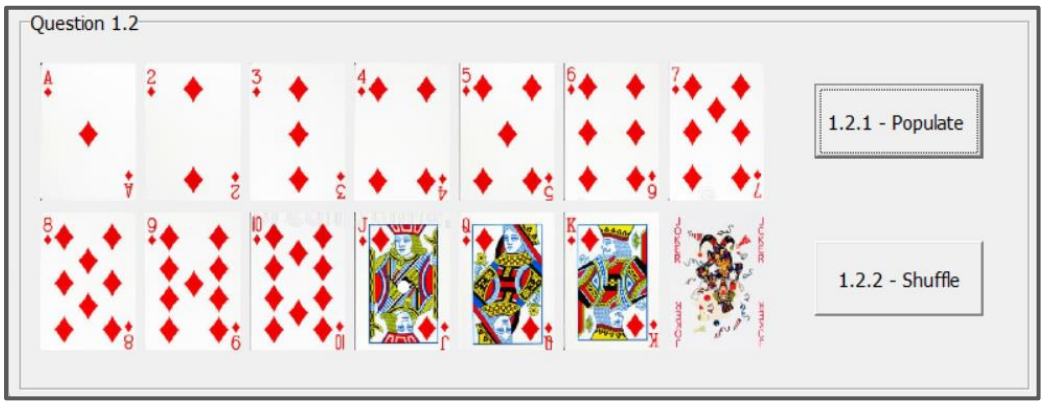
A list of the numbers on a card deck is necessary to display the cards. An array, **arrNumber,** is declared to store the list.

```
arrNumber: array[1..14] of string;
```

Write code to assign the numerical values 1 to 14 as elements in the **arrNumber** array.

NOTE: Code is provided to display the cards.

Example of output:



(4)

1.2.2 **btnQ1_2_2 [1.2.2 – Shuffle]**

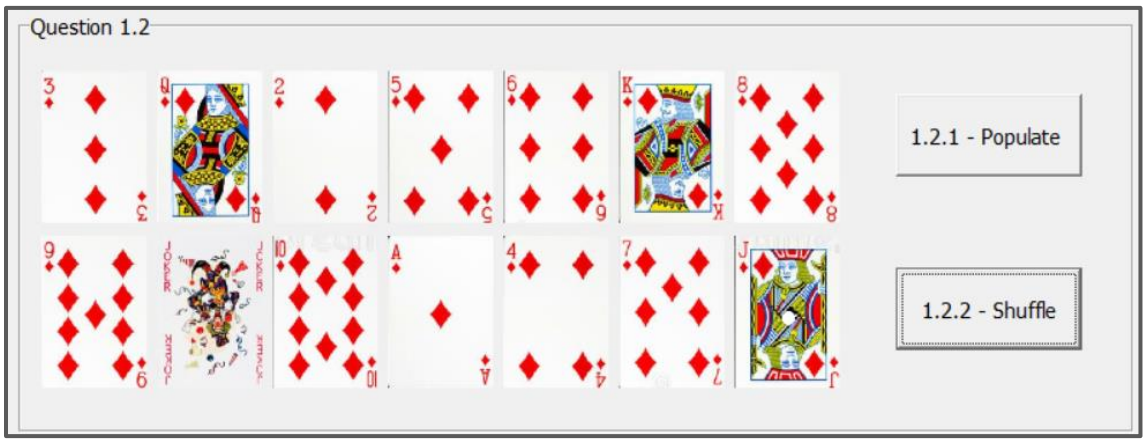
A card deck must be shuffled before a game can be played. The order of the shuffled cards is saved in a text file called **Images.txt**.

NOTE: The code to display a single card is provided.
Use this code in your answer to display the shuffled cards.

Write code to do the following:

- Read each line (card number) in the text file and save the line in a string variable.
- Call the **DisplaySingleCard** procedure with the string variable as parameter.
- You do not have to test that the file exists.

Example of output:



(6)

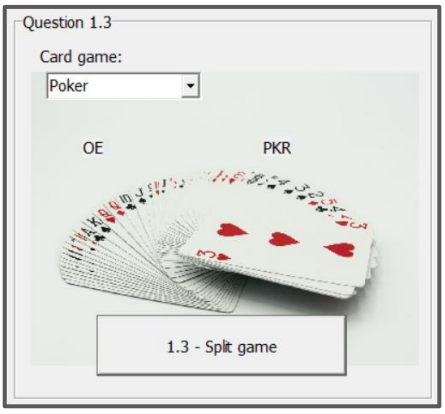
[10]

1.3 **btnQ1_3 [1.3 – Split game]**

Write the code to do the following:

- Extract the card game selected from the combo box.
- Convert the card game to uppercase.
- Split the card game into a string containing all the vowels (a, e, i, o, u) and another string containing all the consonants including special characters like a space.
- Display the string with vowels and string with consonants on the appropriate labels, **lbl_Vowels** and **lbl_Consonants**.

Examples of output:



(10)

1.4 **btnQ1_4 [1.4 – Play]**

In the card game "Burst", a player enters card numbers, which are added together, to try and get as close as possible to a specific number (in this case 50). The player may choose to get another card. The numbers of the cards selected/chosen are displayed in the rich edit.

NOTE: An attempt to code the solution to the problem is provided. Unfortunately, some logical and syntax errors were made (8 in total).

Remove the curly brackets "{" and "}" from the code and correct the errors in the code. (8)

[18]

- Enter your examination number as a comment in the first line of the program file.
- Save your program.

TOTAL SECTION A: [46]

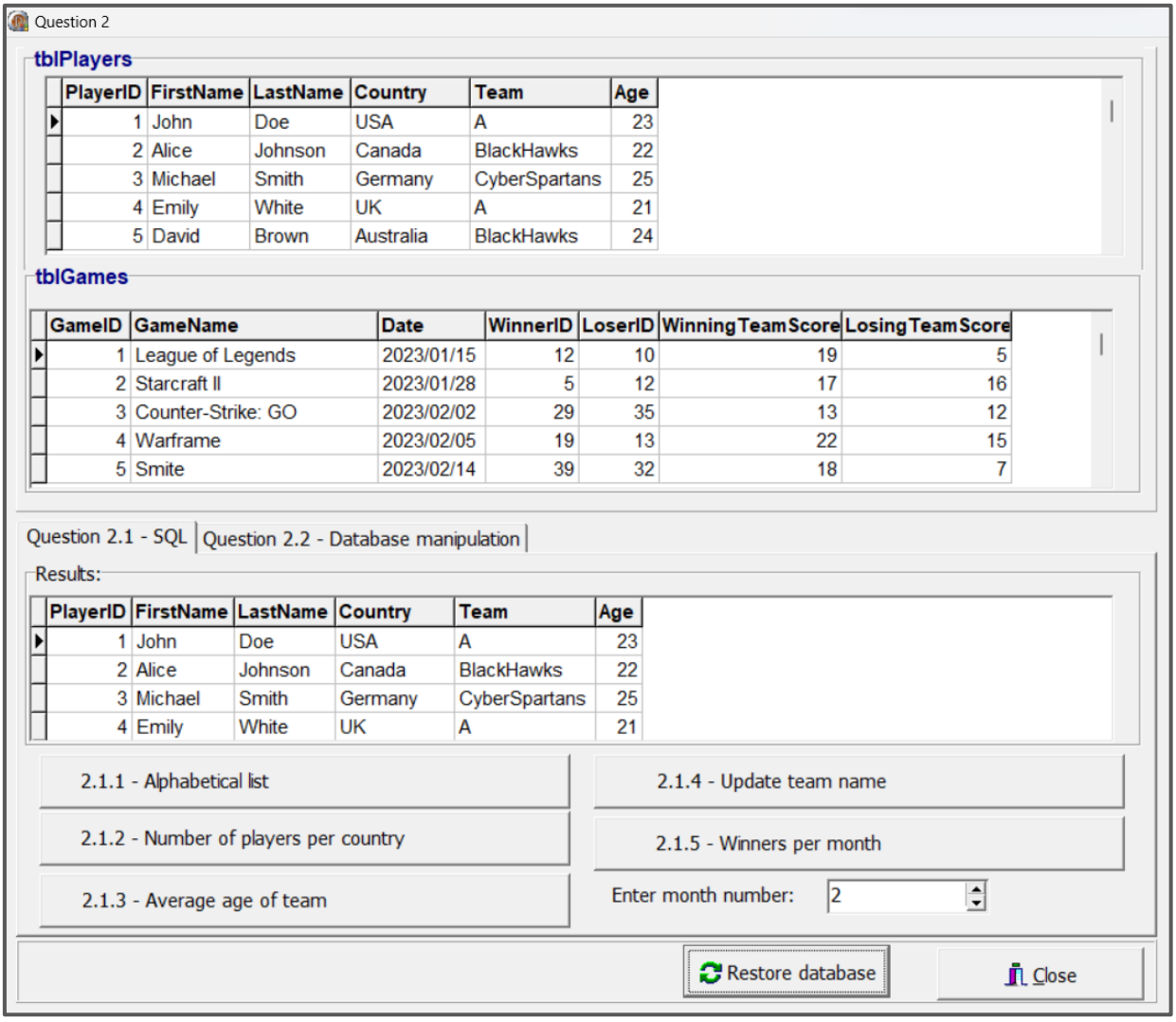
SECTION B

QUESTION 2: DELPHI PROGRAMMING AND DATABASE MANIPULATION

Information on e-sport players and their results are saved in the Access database **ESports.** Appendix A shows screenshots of the database's layout and content.

Open the incomplete **Question2_p.dpr** program in your **DATA2024/Question 2** folder.

The following screen shot shows the interface:



WORK ON THE Question 2.1 – SQL TABSHEET

Complete the **SQL** text statements for each button of **QUESTION 2.1**, as described in **QUESTION 2.1.1** to **QUESTION 2.1.5.**

NOTE: Table and fieldnames are provided as comments in the code to use in your answers for **QUESTION 2**.

2.1.1 btnQ2_1_1 [2.1.1 – Alphabetical list]

Write an **SQL** statement to **display** all the records and fields from the **tblPlayers** table sorted by team, last name and first name.

An example of output for the first four records:

Results:

PlayerID	FirstName	LastName	IDNumber	Country	Team	Age
16	Mia	Cooper		UK	A	21
24	Emily	Davis		Australia	A	23
1	John	Doe		USA	A	23
22	Sarah	Johnson		Canada	A	22

(4)

2.1.2 btnQ2_1_2 [2.1.2 – Number of players per country]

Write an **SQL** statement to **display** the number of players from each country.

An example of output for the first four records:

Results:

Country	Number of players
Australia	6
Brazil	1
Canada	7
China	2

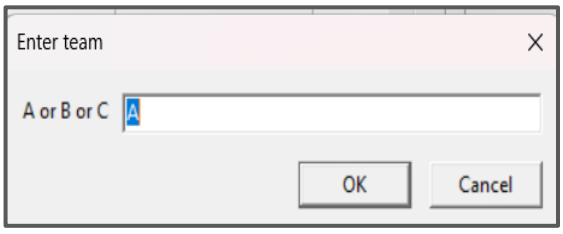
(4)

2.1.3 btnQ2_1_3 [2.1.3 – Average age per team]

Write an **SQL** statement to **display** the average age for the players in the team/teams which start with the same character as a whole number. The first character of the name of the team/teams is entered by the user.

NOTE: Code to enter the first character of the team's name is provided.

Example output:



Results:

average age
23

(5)

2.1.4 **btnQ2_1_4 [2.1.4 – Update team name]**

Team A's name is not stored in full in the database.

Write an **SQL** statement to **update** all the records for team A to display *"Avengers"* as team name.

A message dialogue box is provided to confirm the change in the database:

An example of output for the first four records after the update has been applied:

Results:

PlayerID	FirstName	LastName	Country	Team	Age
1	John	Doe	USA	Avengers	23
2	Alice	Johnson	Canada	BlackHawks	22
3	Michael	Smith	Germany	CyberSpartans	25
4	Emily	White	UK	Avengers	21

(3)

2.1.5 **btnQ2_1_5 [2.1.5 – Winners per month]**

Write an **SQL** statement to display all the winners**,** (first name and last name) from **tblPlayers** for the games played (game name and date) from **tblTeamGames** in the month selected in the **spnQ2** spin edit.

Example of output for the first four records when 2 were selected:

Results:

GameName	Date	FirstName	LastName
Counter-Strike: GO	2023/02/02	Daniel	Garcia
Warframe	2023/02/05	Benjamin	Taylor
Smite	2023/02/14	Noah	Kim
Dota 2	2024/02/01	Michael	Smith

(6)

[22]

WORK ON THE QUESTION 2.2 – DATA MANIPULATION TABSHEET.

Use database table statements (Delphi code) for each button of **QUESTION 2.2**, as described in **QUESTION 2.2.1** and **QUESTION 2.2.2**. **NO** marks will be allocated for SQL-statements in **QUESTION 2.2.**

2.2.1 **btnQ2_2_1 [2.2.1 - Correction]**

The winner ID (**WinnerID**) and the loser ID (**LoserID**) was wrongly recorded.

Write Delphi code to swap the winner ID and loser ID for the selected record.

Example of output:

Before the button is clicked:

tblGames

GameID	GameName	Date	WinnerID	LoserID
1	League of Legends	2023/01/15	12	10

After button is clicked:

tblGames

GameID	GameName	Date	WinnerID	LoserID
1	League of Legends	2023/01/15	10	12

(5)

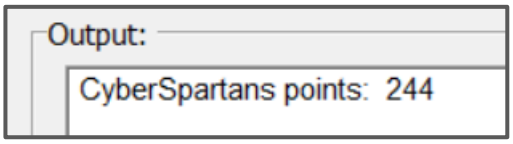
2.2.2 **btnQ2_2_2 [2.2.2 – 2023 Team points]**

Write code to calculate the total WINNING team points for the CyberSpartans team.

NOTE: Code has been provided to display the total.

HINT: Use the **WinnerID** to find the team name in the **tblPlayers** table.

Example of output:



(8)

- Enter your examination number as a comment in the first line of the program file.
- Save your program.

TOTAL SECTION B: [35]

SECTION C

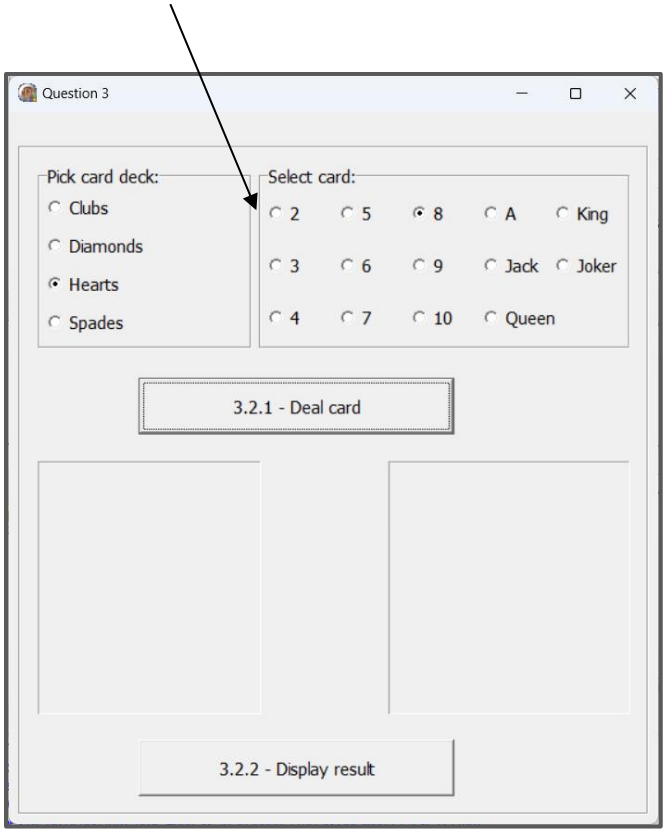
QUESTION 3: OBJECT ORIENTED PROGRAMMING

Open the incomplete **Question3_p.dpr** program in your **DATA2024/Question 3** folder.

The user is guessing which random card will be dealt by the computer.

The following screen shot shows the user interface (GUI):

TAKE NOTE that a card deck starts at 2.



3.1 An **incomplete** object class is saved in the **clsQuestion3_u** unit.

The following UML class diagram represents the incomplete **TCard** class:

TAnimalConsultation
- fSuit : string; - fRank : integer; - fCard : string; - fWin : real;
+ create(); + getValue:real; + getFileName:string; + DealCard; + getPcCard:string; + SetWinnings;

Some notes on the attributes:

The **fSuit** field stores the name of the suit, i.e. clubs ♣, diamonds ♦, hearts ♥ or spades ♠ selected by the user.

The **fRank** field stores the rank of the card (i.e. card number).

A standard deck of cards has 13 ranks in each of the four suits (clubs, diamonds, hearts, and spades). These ranks, from highest to lowest, are:

- Ace (often abbreviated as A)
- King (K)
- Queen (Q)
- Jack (J)
- 10
- 9
- 8
- 7
- 6
- 5
- 4
- 3
- 2

REMEMBER that a card deck starts at 2.

The **fCard** field stores the file name and path for the image of the card, that the computer dealt, e.g. **Cards\Clubs3.png**.

The **fWin** field stores the prize money that can be won if the user guesses the correct card.

Complete the code in the given class (**TCard)** as part of the **clsQuestion3_u** unit as described in **QUESTION 3.1.1** to **QUESTION 3.1.6.**

3.1.1 Write code for a constructor method named **Create** that will receive the suit and index from the radio group option selected by the user:

- Assign the suit parameter to the **fSuit** attribute.
- Assign the index parameter value + 2 to the **fRank** attribute.
- Set the **fCard** attribute to an empty string.
- Set the **fWin** attribute field to 0. (6)

3.1.2 Write the code to create a mutator method named **DealCard,** that will compose and set the value of the **fCard** attribute as follows:

- Let the computer select a random number from 0 to 13 (both included) as card number.
- Set the **bJoker** variable to *true* if the random number is 13. **bJoker** is a provided global variable.
- Set the **bJoker** variable to *false* if the random number is not 13.
- Let the computer select a random number, 0 to 3, as card suit, where 0 is clubs, 1 is diamonds, 2 is hearts and 3 is spades.
- Set the **fCard** attribute to the image filename in the following format:

  Cards\ + SuitName+CardNumber+.png

  e.g.
  'Cards\Clubs4.png'
  'Cards\Hearts10.png'

REMEMBER that a card deck starts at 2. (8)

3.1.3 Write the code for an accessor method named **getFilename** that will return the file name of the card guessed by the user in the following format:

Cards\ + SuitName+CardNumber+.png

e.g.
'Cards\Clubs4.png' (3)

3.1.4 Write the code for an accessor method named **getPcCard** that will return the value of the **fCard** attribute AFTER dealing a card.

NOTE: The **DealCard** method must be called. (3)

3.1.5 Write the code for an accessor method named **getValue** that will return the value of the **fWin** attribute. (2)

3.1.6 Write the code for a mutator method named **SetWinnings** that will calculate and set the **fWin** prize money (if any) based on the following criteria:

- If the correct card (card filename constructed in **3.1.3**) that is guessed by the user is the same as the card dealt (card file name constructed in **3.1.2**) the prize money is the card number multiplied by R100.
- If the Joker is dealt, the prize money is R1 000 regardless of what the user guessed. (6)

[28]

3.2 Complete and code the following for the **Question3_u** main unit:

3.2.1 **btnQ3_2_1 [3.2.1 – Deal Card]**

The user must guess the card dealt by die computer by selecting the card suit and the card number in the provided radio groups.

Write code to create an instance of the object **objCard** with the:

- suit selected in the **rgpQ3** radio group and the
- index of the selection in the **rgpQ3b** radio group as parameters. (4)

3.2.2 **btnQ3_2_2 [3.2.2 – Display result]**

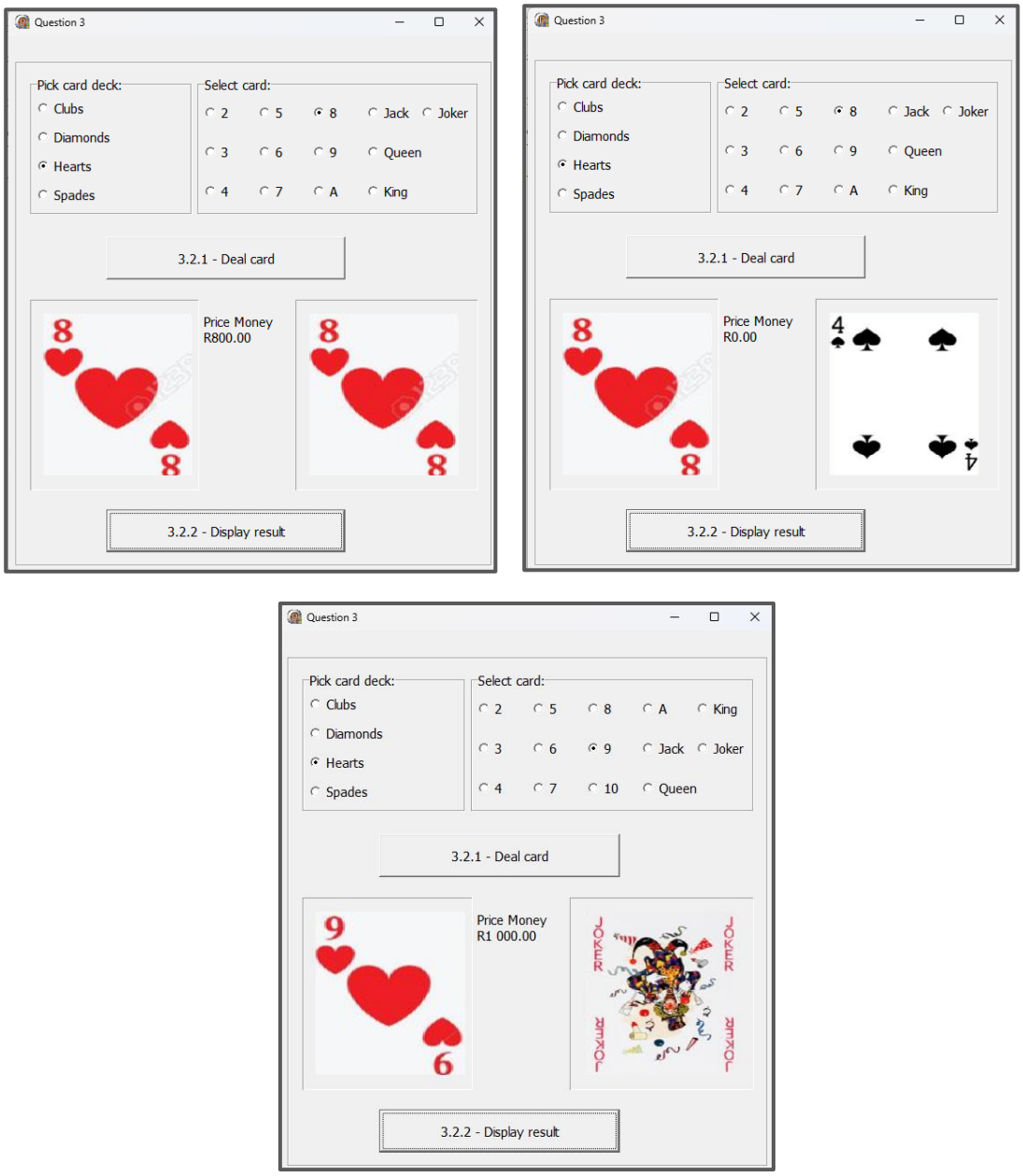
Write the object method calls to:

- Load the card selected/guessed by the user to **imgPlayer**.
- Load the card dealt by the computer to **imgPC.**
- Determine the prize money.
- Display the prize money on the provided label, **lblQ3**.

Example of output (on the next page):

NOTE: Your output may differ based on the user selection and dealt card.

The card the user selects is displayed on the left, and the card randomly chosen / dealt by die computer is displayed on the right.



(6)

[10]

- Enter your examination number as a comment in the first line of the object class and the form class.
- Save your program.

TOTAL SECTION C: [38]

SECTION D

QUESTION 4: PROBLEM SOLVING AND ARRAYS

Open the incomplete **Question4_p.dpr** program in your **DATA2024/Question 4** folder.

Blackjack is a popular card game. The program is created to play a simplified version of Blackjack. A card deck consists of 52 (13 x 4) cards in four suits: clubs, ♣ diamonds, ♦ hearts ♥ and spades. ♠

The following screenshot shows the user interface:

NOTE: Code is provided to handle basic card operations.

The following arrays are declared:

arrDeck: array[1..52] of string;	Stores all the card names in the deck, e.g. 2 of hearts.
arrPlayers: array[1..2,1..10] of string;	Stores the dealer (computer) and a player's cards that were dealt.
arrPlayerTotal: array[1..2] of Integer;	Stores the total value of cards for the dealer and a player.
arrPlayerNames: array[1..2] of string;	Stores the dealer's and a player's names.

The following class scope variable is declared:

iNumDeckCards : Integer = 52;	Stores the number of cards in the deck.

NOTE: The dealer is player 1 and the player is player 2.

4.1 **btnQ4_1 [4.1 – Display deck]**

All the card names in the card deck must be displayed neatly in four columns.

NOTE: Code to set the tabs in the rich edit is provided.
The order of the cards will differ because the array **Deck** is shuffled randomly.
Each suite consists of 13 cards, meaning each column must have 13 card names.

Write code to display the card names in four neat columns.

Example of output:

Cards in deck:

King of Clubs	6 of Hearts	7 of Spades	Ace of Hearts
6 of Diamonds	8 of Clubs	4 of Clubs	Jack of Diamonds
3 of Clubs	8 of Diamonds	Queen of Clubs	7 of Clubs
2 of Clubs	Jack of Clubs	Queen of Hearts	2 of Diamonds
10 of Hearts	Jack of Hearts	3 of Hearts	3 of Spades
King of Hearts	7 of Hearts	9 of Diamonds	2 of Spades
5 of Diamonds	8 of Spades	4 of Diamonds	King of Diamonds
2 of Hearts	10 of Spades	4 of Spades	5 of Clubs
King of Spades	9 of Clubs	Jack of Spades	5 of Hearts
Ace of Spades	7 of Diamonds	Ace of Diamonds	6 of Spades
10 of Diamonds	5 of Spades	3 of Diamonds	Queen of Spades
9 of Spades	6 of Clubs	Ace of Clubs	9 of Hearts
Queen of Diamonds	10 of Clubs	8 of Hearts	4 of Hearts

(6)

4.2 You need to complete the code for provided functions and procedures before a user can click the **[4.2 – Play]** button.

NOTE: Some procedures/functions have been written for you. You will use one of these namely the **CardValue** function in question 4.2.2.

4.2.1 **ShuffleDeck procedure**

The card deck must be shuffled before a card game can commence.

Complete the code in the **ShuffleDeck** procedure to do the following:

For each of the 52 cards:

- Get a random index/position in the card deck.
- Swap the current card in the deck with the card in the random index/position.

(5)

CardValue function

A complete function, **CardValue,** that receives the name of the card and returns the value of the card is provided.

NOTE: You do not need to code anything in this function.

4.2.2 **CalculateTotal function**

Write the code to calculate the total value for ALL the cards in each of the dealer's and one player's hand and save the dealer and player totals in the **arrPlayertotal** array.

e.g. if the dealer's hand (**arrPlayers[1]**) consists of the queen of hearts and the queen of spades the total points for the dealer will be 20 in **arrPlayertotal[1]**.

NOTE: The **CardValue** method call statement is provided and must be used in your answer.
The maximum number of cards that can be in a player's hand is 10. You need to determine the number of cards dealt.
The dealer is player 1 and the player is player 2. (6)

4.2.3 **RemoveFromDeck procedure**

Every time the player or dealer receives a card, the card that is currently at the top of the deck (index/position 1 in the Deck array) is dealt.

Write code to remove the first card in the Deck array and decrease the **iNumDeckCards** that keeps track of the number of cards in the deck. (4)

4.2.4 **HandSize function**

The number of cards currently in the player's or dealer's hand is necessary to determine whether another card can be dealt or not.

Write code to complete the code in the **HandSize** function to return the number of card (elements) saved in the **arrPlayers** array for either the dealer's hand (playerindex 1) or the player's hand (playerindex 2).

NOTE: The "hand" refers to the cards dealt to a player. (4)

4.2.5 DetermineWinner procedure

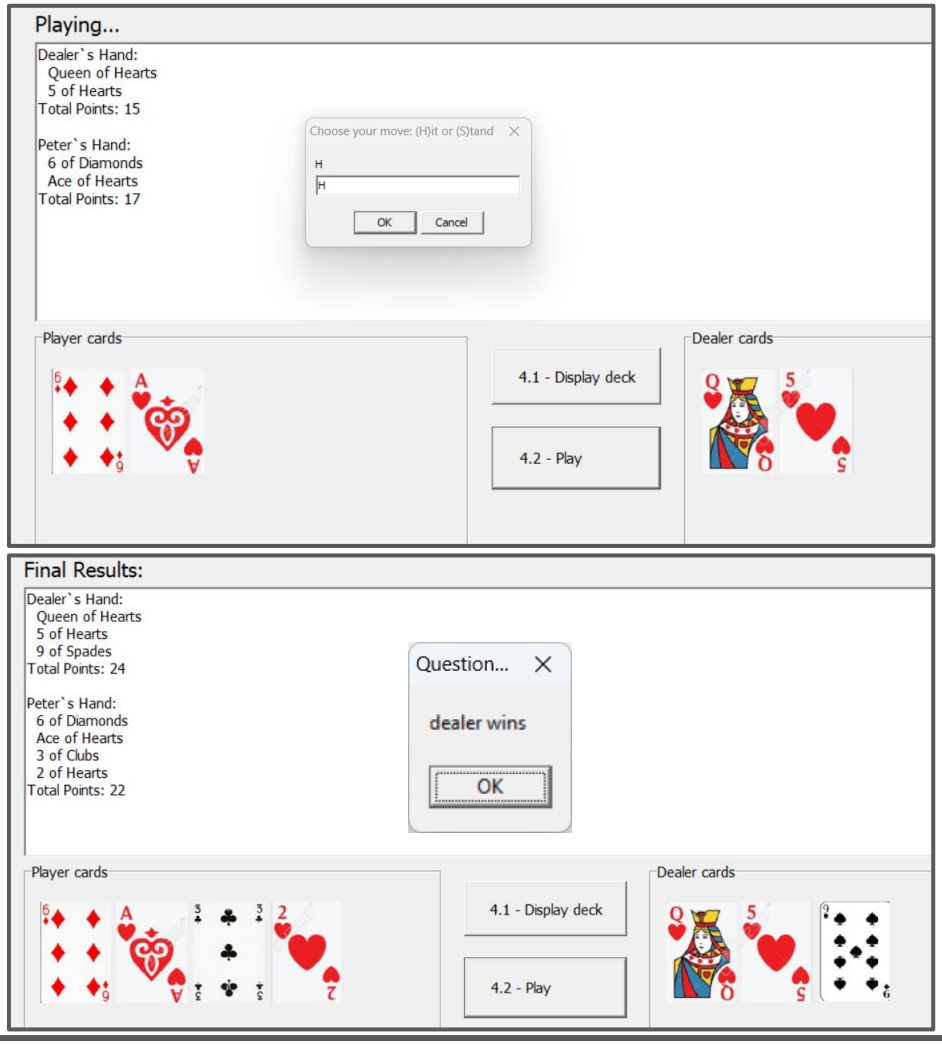
Write code to complete the **DetermineWinner** procedure to display the winner, in the **redQ4** rich edit.

Criteria to determine the winner of a game:

- If the player's total card value (second value in the **arrPlayerTotal** array) is less than or equal to 21 and the player's total card value is less than the dealer's total card value (first value in the **arrPlayerTotal** array), the player wins.
- If the player's and dealer's total card value is the same, it is a draw.
- If the dealer's total card value is less than the player's total card value, the dealer wins.

Examples of possible output:

NOTE: Code to ask the player's name is provided in the **[4.2 – Play]** event handler.



(6)

- Enter your examination number as a comment in the first line of the program file.
- Save your program.

TOTAL SECTION D: [31]

GRAND TOTAL: [150]

INFORMATION TECHNOLOGY P1

DATABASE INFORMATION OF ESPORTS.MDB FOR QUESTION 2:

The design of the database tables is as follows:

The **ESports** database consists of two tables i.e. **tblPlayers** and **tblTeamGames**

Design view of the tblPlayers table:

tblPlayers

Field Name	Data Type
PlayerID	AutoNumber
FirstName	Short Text
LastName	Short Text
Country	Short Text
Team	Short Text
Age	Number

The list of the first ten records in the tblPlayers table:

tblPlayers

PlayerID	FirstName	LastName	Country	Team	Age
1	John	Doe	USA	A	23
2	Alice	Johnson	Canada	BlackHawks	22
3	Michael	Smith	Germany	CyberSpartans	25
4	Emily	White	UK	A	21
5	David	Brown	Australia	BlackHawks	24
6	Jessica	Lee	South Korea	CyberSpartans	26
7	Brian	Miller	USA	A	23
8	Olivia	Davis	Canada	BlackHawks	22
9	Ethan	Harris	Germany	CyberSpartans	25
10	Emma	Robinson	UK	A	21

Design view of the tblTeamGames table:

tblTeamGames

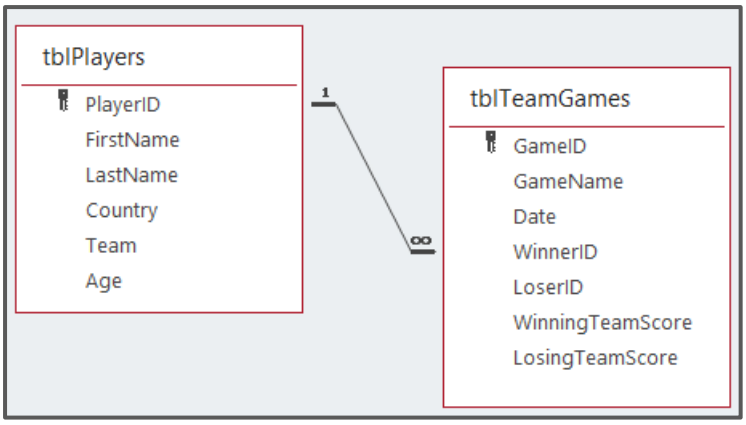
Field Name	Data Type
GameID	Number
GameName	Short Text
Date	Date/Time
WinnerID	Number
LoserID	Number
WinningTeamScore	Number
LosingTeamScore	Number

A screenshot of the first 10 records in the tblTeamGames table:

tblTeamGames

GameID	GameName	Date	WinnerID	LoserID	WinningTeamScore	LosingTeamScore
1	League of Legends	2023/01/15	12	10	19	5
2	Starcraft II	2023/01/28	5	12	17	16
3	Counter-Strike: GO	2023/02/02	29	35	13	12
4	Warframe	2023/02/05	19	13	22	15
5	Smite	2023/02/14	39	32	18	7
6	Dota 2	2023/03/10	11	32	22	17
7	Street Fighter V	2023/03/20	21	28	28	23
8	Overwatch	2023/04/05	3	10	23	13
9	Paladins	2023/04/10	23	24	30	23
10	Clash Royale	2023/05/02	21	34	28	12

Relationship between tables:



INFORMATION TECHNOLOGY PAPER 1
INFORMATION SHEET *(to be completed by candidate)*

EXAMINATION STICKER

FOLDER NAME: _____

Candidate to tick the file name(s) of each of the questions that were attempted/modified and saved.

Question	File name	Saved (✔)	Maximum mark	Mark achieved
1	Question1_U		47	
2	Question2_U		34	
3	Question3_U		10	
	clsQuestion3_U		28	
4	Question4_U		31	
TOTAL			**150**	

BLANK PAGE FOR PLANNING PURPOSES

BLANK PAGE FOR PLANNING PURPOSES