higher education & training

Department:
Higher Education and Training
REPUBLIC OF SOUTH AFRICA

NATIONAL CERTIFICATE (VOCATIONAL)

COMPUTER PROGRAMMING

(Second paper)
NQF LEVEL 4

(11011034)

28 November 2024 (X-paper)
09:00–13:00

This question paper consists of 13 pages.

551Q2N2428

Copyright reserved

DEPARTMENT OF HIGHER EDUCATION AND TRAINING
REPUBLIC OF SOUTH AFRICA

NATIONAL CERTIFICATE (VOCATIONAL)
COMPUTER PROGRAMMING
NQF LEVEL 4
(Second paper)
TIME: 4 HOURS
MARKS: 100

INSTRUCTIONS AND INFORMATION

1. Answer all the questions.
2. Read all the questions carefully.
3. Number the answers according to the numbering system used in this question paper.

INSTRUCTIONS TO STUDENTS AND LECTURERS

1. Students must be seated in the computer laboratory 30 minutes before the start of the examination to ensure that all computers have the appropriate programs.
2. These questions must be completed on a computer by making use of MS Office 2010 or a later version and VB.NET 2005 or VB.NET Express or VB.NET 2008 or a later version.
3. Students must make sure that they print their work immediately after completing a question. Ensure that your examination number appears on each printout. No questions without a printed EXAMINATION NUMBER will be marked.
4. In the event of problems such as: a power failure; computer breakdown or a printer breakdown, the invigilator will make the necessary arrangements to continue with the examination process once the problem has been resolved. Students will not be penalised with time lost under these circumstances.
5. All work on the computer should be saved at regular intervals to prevent loss of work. No additional time may be allowed for work lost due to lack of saving.
6. No student may print his/her work for another student, or make a memory stick available to another student or access another student's work on the network. Any attempt to access any information from or transfer information to another student in whatever manner is a contravention of the examination rules and regulations and will be viewed in an extremely serious light.

Copyright reserved

7. All printouts or screen prints done during the session must be handed in.

8. Make sure that your examination number appears on all printouts.

9. Students may not copy the source code from VB.NET to MS Word and then print it from the word processor.

10. Steps to print a VB.NET form or any other screen:

    10.1 Run the solution.

    10.2 Hold the ALT key down and press the print screen button (This will copy the active screen).

    10.3 Open MS Word and paste the form/screen in a new document.

    10.4 Add your EXAMINATION NUMBER to the top right-hand side of your page.

    10.5 Print the page.

11. You are allowed to make use of the help files in VB.NET.

12. The following files will appear on your hard drive in the folders:

    12.1 QUESTION 1
    C:\CP L4 P2\ HikersExplorer
    HikersExplorer (Microsoft Visual Studio Solution)

    12.2 QUESTION 2
    C:\ CP L4 P2\ Assessment (Microsoft Visual Studio Solution)

    12.2 QUESTION 3
    C:\CP L4 P2\Databases\
    C:\ CP L4 P2\ StudentRegister (Microsoft Visual Studio Solution)

    12.3 QUESTION 4
    C:\CP L4 P2\QUESTION 4
    Images/environ.jpg
    Style.css

13. Students may use the following or any other program of their choice to create web pages:

    Notepad ++, Netbeans, Blue voda, Dreamweaver

Copyright reserved

QUESTION 1

LifeStyle-Hiking asked you to design a pre-hike quotation application for their company as a means to create added value for their customers.

A customer must be able to enter the `size` of the hiking group as well as select a hiking time (`day` or `night`). The application must use this information to generate a quotation for the trip based on the selections and inputs made.

Day-time hiking is charged at R250 per person and night-time hiking at R500 per person. However, discounts are applicable based on the size of the group as shown in the table below. From the table you will note that groups that are smaller than 4 (four) in `size` will NOT qualify for a discount.

Group	Size	Discount
Small	4 - 10	5%
Medium	11 - 20	15%
Big	More than 20	20%

The following IPO chart will assist you in better understanding the dynamics of the user requirements as well as how to complete the required calculations

INPUT	PROCESSING	OUTPUT
Time Size	Set the `price` based on the Time: Day hiking: `price` = 250 Night hiking: `price` = 500 Set the `discount` based on the Size: Size less than 4 → `discount` = 0 Size 4-10 → `discount` = (price*size)*0.05 Size 11-20 → `discount` = (price*size)*0.15 Size greater than 20 → `discount` =(price*size)*0.2 Calculate the total price of the hiking trip: TotalPrice = (`price` x size) – `discount`	Time Size Price Discount TotalPrice

FIGURE 1: Below is a printout of what the user interface looks like:

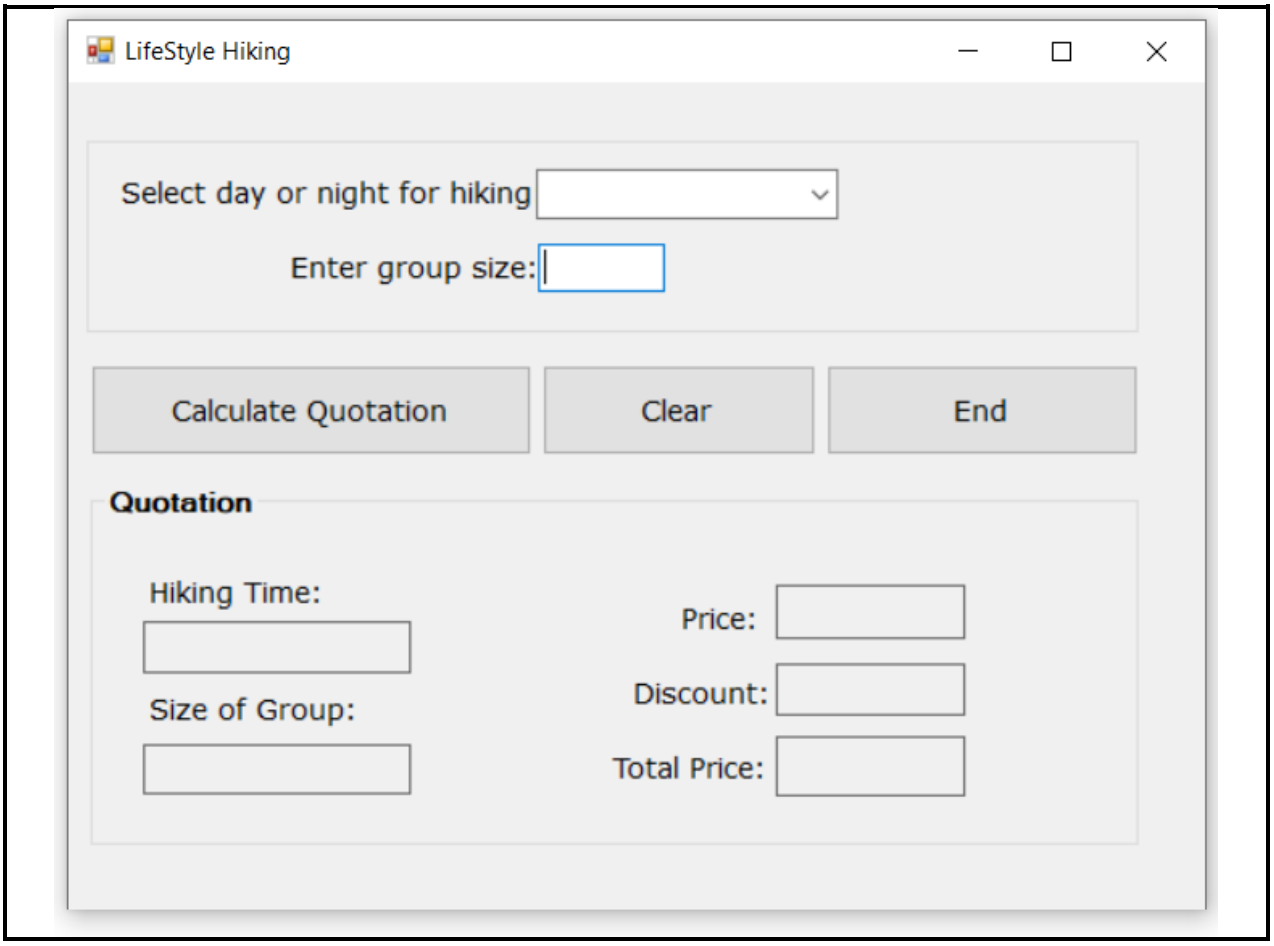


FIGURE 1

STEP 1 Clicking the CALCULATE QUOTATION button after supplying the applicable inputs will display the calculated values in the “Quotation” panel.

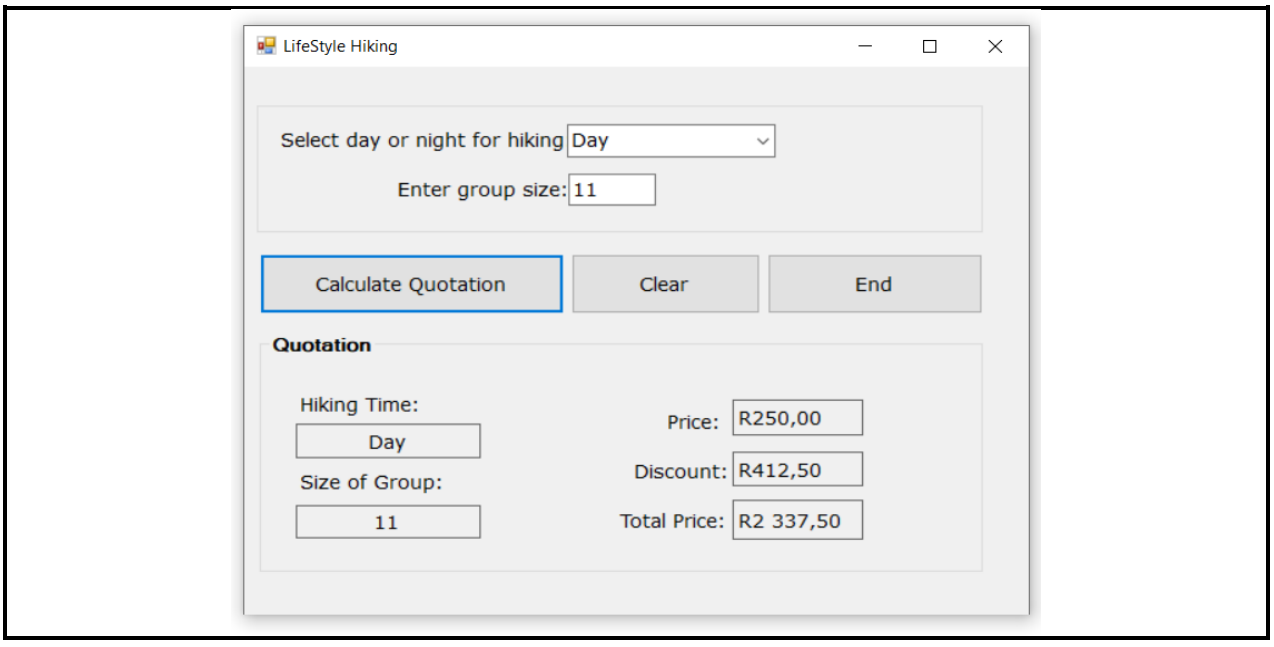
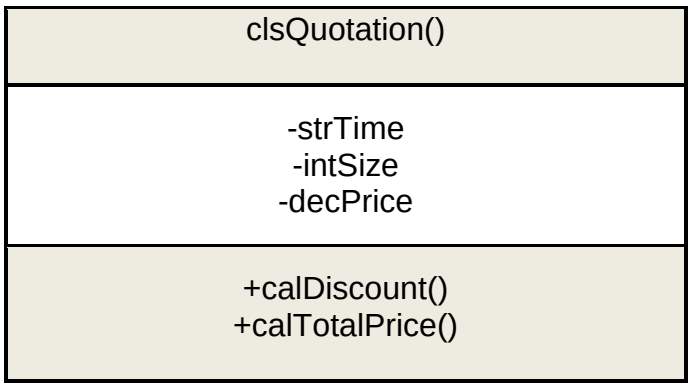


FIGURE 1.1

Copyright reserved

Use the following class diagram, *clsQuotation* to help you understand and complete the application as instructed below.



The program interface with the partially completed code have been saved on the hard drive in a folder called …/QUESTION1. Open the solution named *HikersExplorer* and follow the instructions below to complete the program

1.1 Create a class called *clsQuotation* and add the following program code.

1.1.1 Create all the private fields as listed in the class diagram. (2)

1.1.2 Create the associated public properties for the private fields created in QUESTION 1.1.1 (4)

1.1.3 Create a public *constructor* to instantiate the *clsQuotation* class object and set the private fields created in QUESTION 1.1.1 to their default values. (2)

1.1.4 Create a public function called *calDiscount*() to calculate the discount a client qualifies for according to the size of the group entered. (4)

1.1.5 Create a public function called *calTotalPrice()* to calculate the total price for the group minus the discount returned by *calDiscount().* (3)

1.2 Add the following program code to the click event of the CALCULATE QUOTATION button one the main *HikersExplorer* form.

1.2.1 Declare all the variable to be used in the calculations and related data displays and assign it the values from the applicable controls. (3)

1.2.2 Declare and instantiate an object of the *clsQuotation* class. (1)

1.2.3 Assign all the applicable values and properties to the variables declared in QUESTION 1.2.1. (3)

1.2.4 Assign the calculated values to the applicable label controls as shown in the “Quotation Panel” on the screenshots above.
NOTE: All R/c amounts must be formatted to display as currency. (4)

Copyright reserved

Add your EXAMINATION NUMBER as a comment line to the program code of the main *HikersExplorer* form and make a printout of the code.

1.3 Change the text in the title bar of the form to your EXAMINATION NUMBER. (1)

Run the program with the test data below and make a screenshot printout of the result.

TIME	GROUP SIZE
DAY	9
NIGHT	24

[27]

QUESTION 2

Design an application that will allow a user to input TWO sets of assessment results for the theoretical and practical components of a subject. Both subject components have the same set of assessments namely, Test1, Test2, Assignment1, Assignment2 and an Internal Exam. When the user has captured the marks, the information must be displayed as shown in the screenshot in STEP 1.

FIGURE 2 below is a printout of what the user interface looks like

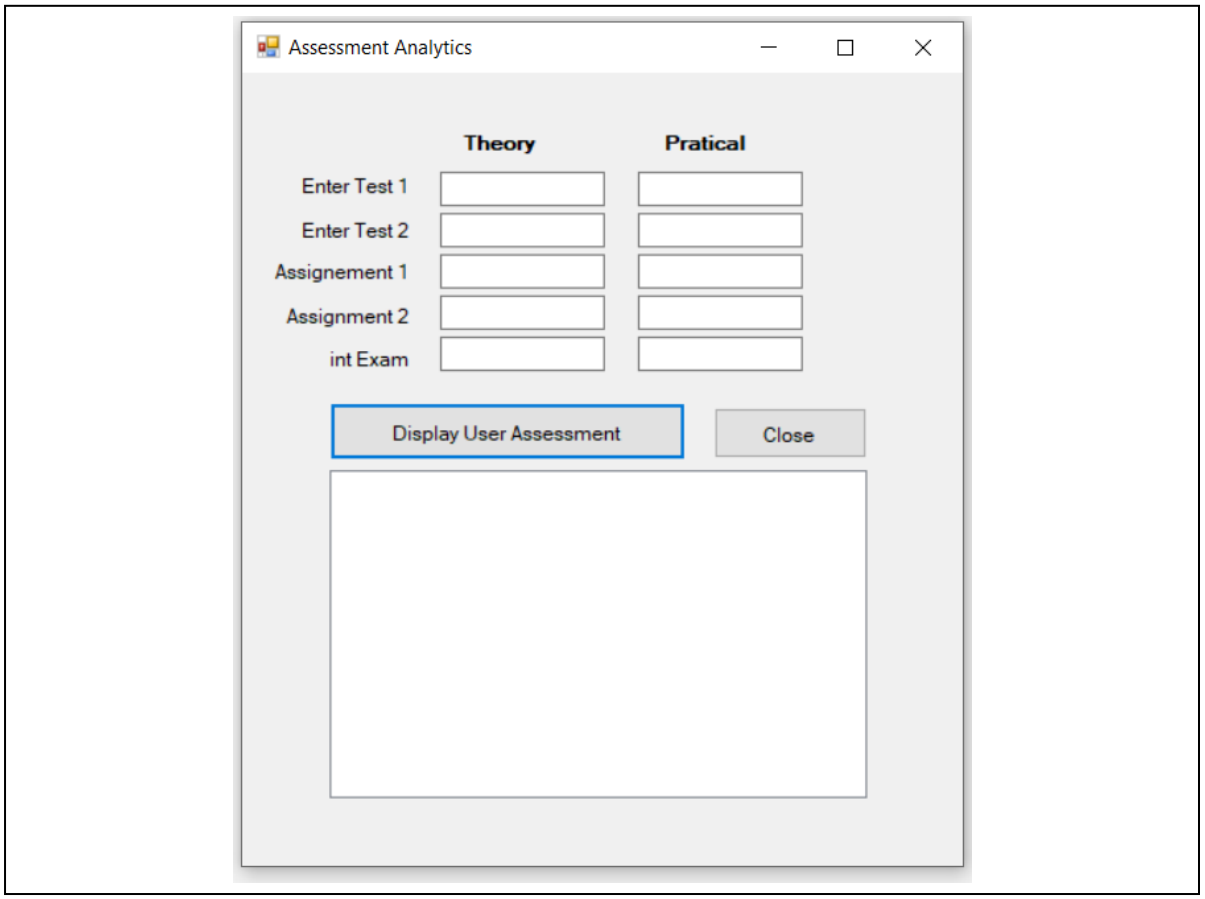
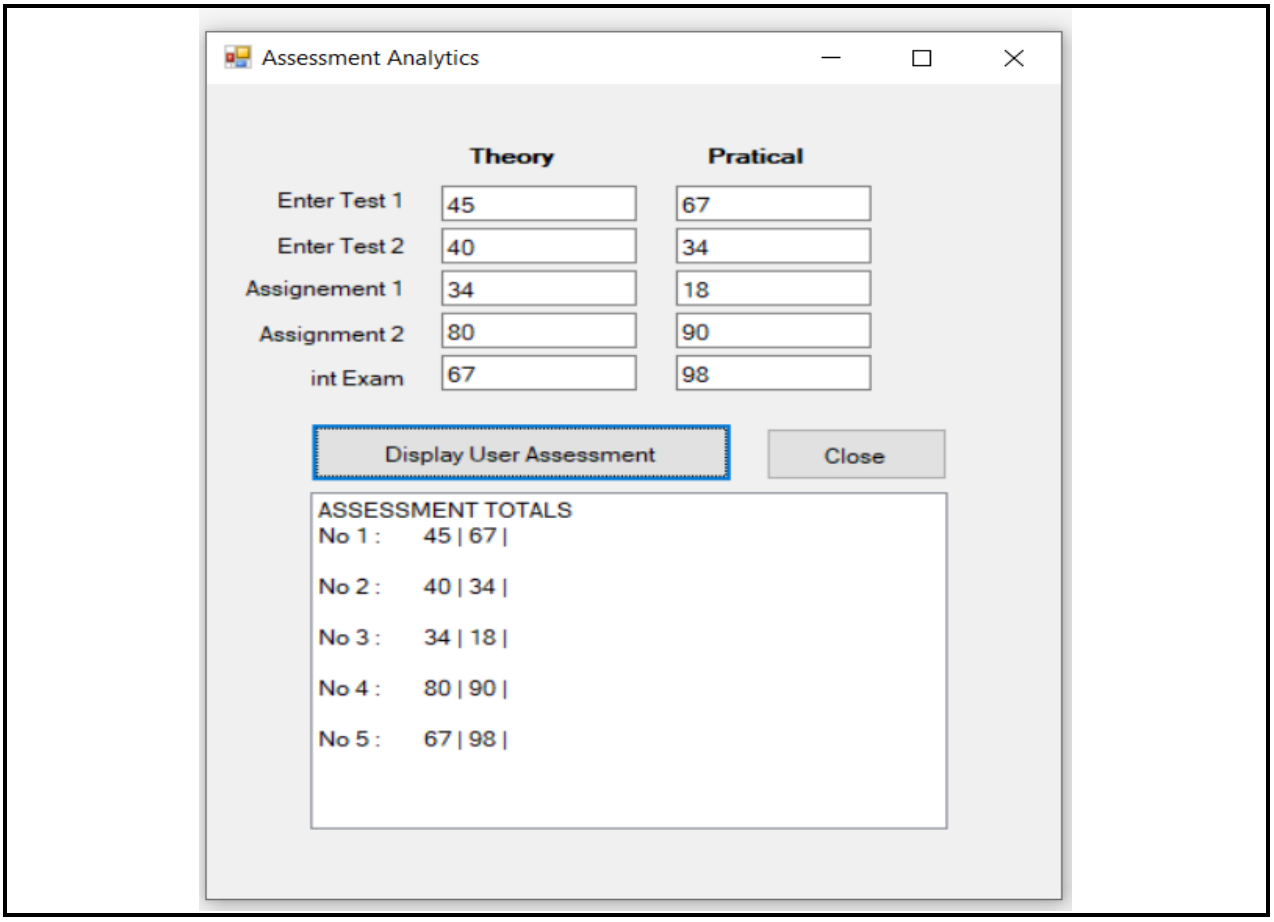


FIGURE 2

The following steps explain how to use the program to display the elements as indicated

STEP 1: Run the program. When the form loads, the user will be prompted to supply the input as indicated. Once completed, clicking the Display User Assessment will format and display the output as shown below.



The partly completed interface has been saved on the hard drive. Open the solution *Assessments* (… \QUESTION 2\) and follow the instructions below to complete the program.

Add the following code to the click event of the DISPLAY USER ASSESSMENT button.

2.1 Declare a 2D array with 5 rows and 2 columns to store the assessment marks that the user will input for the theory and practical components. Make sure that the array is dimensioned correctly. (1)

2.2 Declare two integer variables to use as counters when iterating through the rows and columns of the array declared in QUESTION 2.1 (1)

2.3 Use a nested FOR .. LOOP to iterate through the user inputs and add it to the 2D array. (3)

2.4 Make use of the *GetUpperBound()* method together with a nested FOR … LOOP to retrieve the elements from the 2D array and display in the format as shown in the screenshots above. (5)

Copyright reserved

Add your EXAMINATION NUMBER as a comment line to the program code of the main *Assessments* form and make a printout of the code.

[10]

QUESTION 3

The e-Learning Manager asked you to design an application that will enable them to register students for any available courses they may have on offer. They would prefer that the students' details be stored in a database as it would make managing the related administrative tasks so much easier.

The database called *Admissions* has already been created and is saved on the hard drive in the application folder … \QUESTION 3\Database\. It consists of TWO tables *blStudent* and *tblCourse*, which are related, in a one-to-many relationship.

The image below represents a graphical layout of the *Admissions.mdb* table structures.

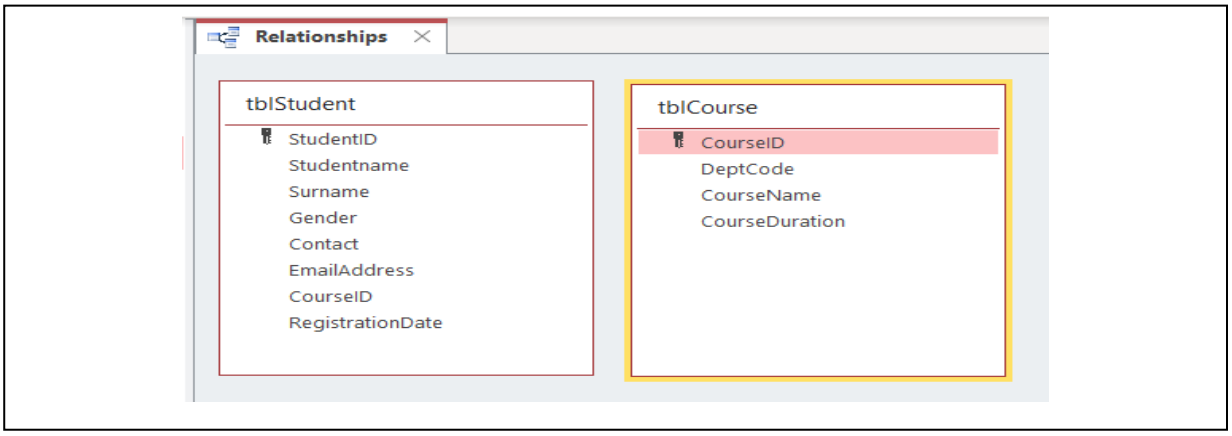


FIGURE 3: Below is a printout of what the user interface looks like

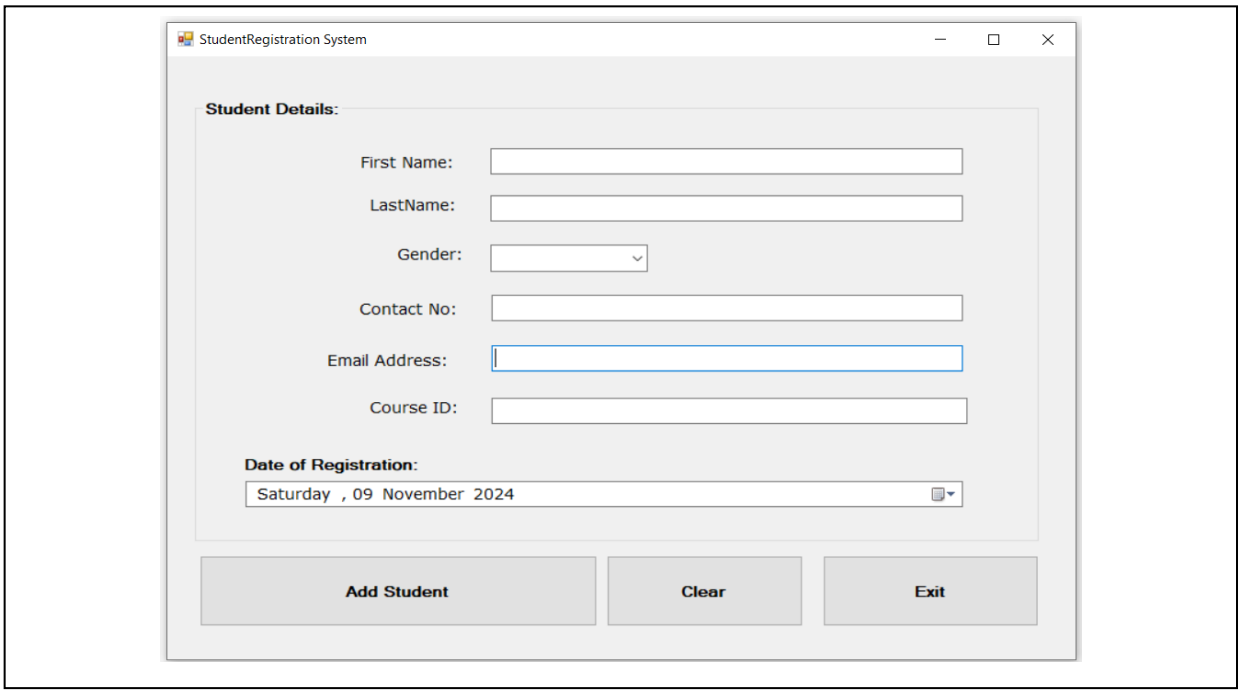
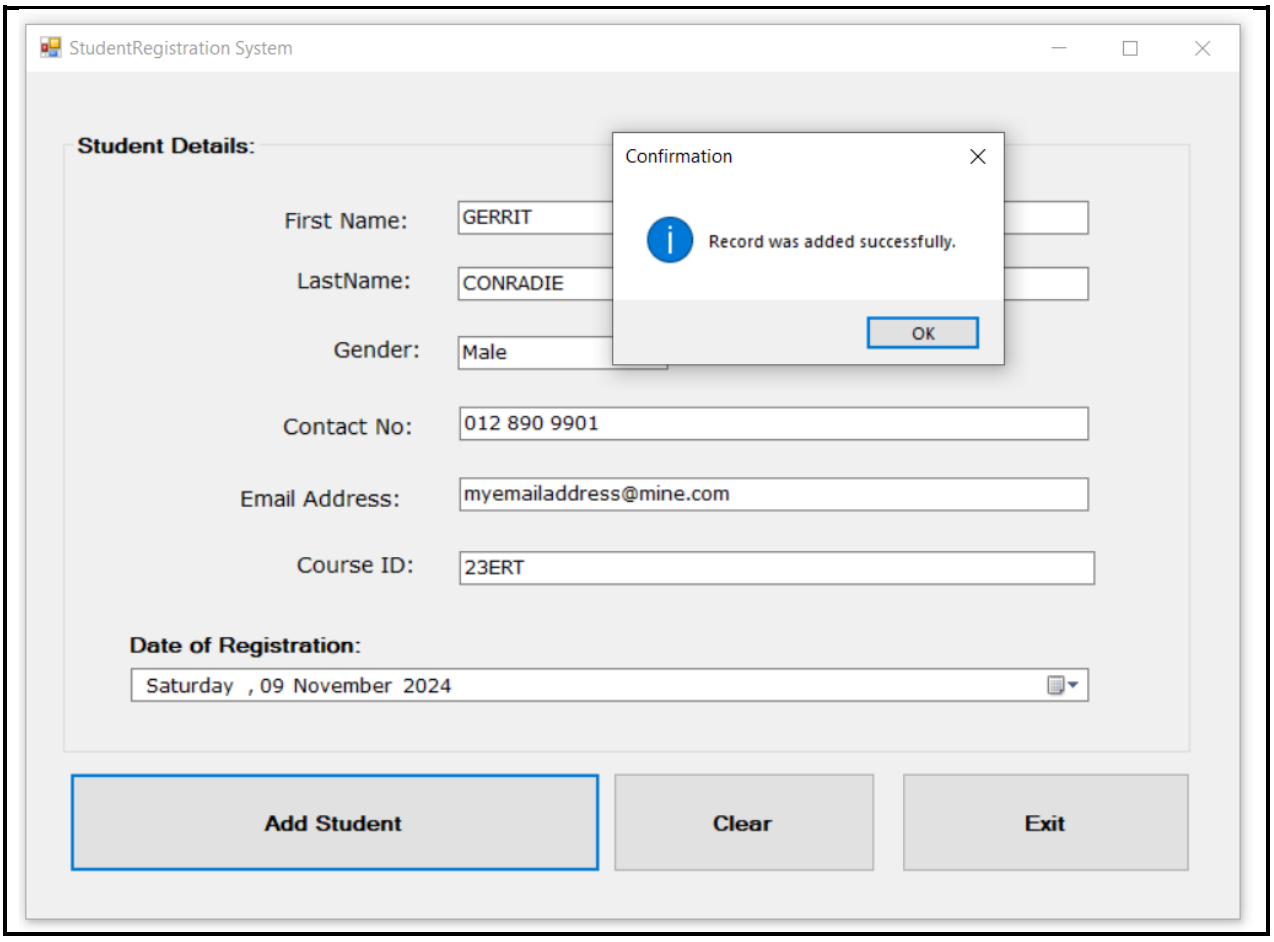


FIGURE 3

The following step explains how to use the program to insert records into the *tblStudent* table.

STEP 1 Run the program and complete the field inputs as shown on the form. To save or add the record to the *tblStudent* table in the database, the user has to click the 'ADD STUDENT' button. If the record was added successfully, the user will be informed accordingly as indicated in the screenshot.



Both the database, *Admissions.mdb* and the partially completed Visual Basic Solution *RegistrationSystem,* are stored in the folder … \QUESTION3\. Open the solution and follow the instructions below to complete the application.

NOTE: You MAY NOT make use of the database wizard to complete the questions

3.1 The click event of the 'ADD STUDENT' button invokes a sub procedure called *getRegister().* Add the required program code in the *getRegister()* sub procedure that will do the following:

Make use of *Try … Catch* to trap any exceptions that may occur during the execution of the program code.

3.1.1 Declare a variable called *redate* and assign it the value from the selected date (Date of Registration on the screenshots). Set the format as a *short date.* (2)

3.1.2 Establish a connection to the *Admissions* database. (6)

3.1.3 Compose the SQL statement that will insert the student details into the database table *tblStudent*. (4)

3.1.4 Define and set the values of all the required insert query parameters. (8)

3.1.5 Write the line of code that will execute the query to insert the record. (2)

3.1.6 If no exceptions occurred then inform the user that the "*Record was added successfully*". (2)

3.1.7 Close the connection to the database. (1)

Add your EXAMINATION NUMBER as a comment line to the program code of the main *RegistrationSystem* form and make a printout of the code.

[25]

QUESTION 4

You are to design a registration web page for your college on behalf of the SRC Educational officer. The page must offer students the opportunity to register for Environmental Awareness Day, which your college will be hosting. On this day the College will be handing out branded t-shirts, but only to those who pre-registered for the event.

FIGURE 4 below is a screen printout of what the web page will look like.



FIGURE 4

All required images and references to external data sources have been saved on your hard drive in the …\QUESTION4 folder.

Follow the instructions below to complete the web form as displayed above. Save the page as *index.html.*

4.1 The web page is divided into FIVE divisions (<DIV></DIV>). The *pagewrapper* division includes all the other divisions i.e., *header*, *navigation*, *content* and *footer* divisions

Understanding the above, Add the following HTML code to create the web page:

4.1.1 Add a link to the external cascading style sheet *style.css* which contains the class styling for the various divisions. (1)

4.1.2 Create the *pagewrapper* division. (1)

4.1.3 Create the *header* division and insert the logo saved as *environ.jpg* in the folder *Images*. Set the width to 185px and the height to 121px. (2)

4.1.4 Create the navigation division and code the navigation menu as shown in the left side-bar underneath the logo. (4)

4.1.5 Create the *content* division and insert the page title, “Environmental Awareness Day!” as an h1 HTML element. (2)

4.1.6 Create a form and name it *event.* Set the form’s method to POST and its action to “mailto:environ@college.edu.za”. (2)

4.1.7 Create a borderless table with five rows and two columns. Use this layout to complete the form inputs as displayed in FIGURE 4 i.e., name, surname, contact, etc. The t-shirt size input should be a dropdown list with three options: Small, Medium, Large and X-Large as on the web page (6)

4.1.8 Create a *fieldset* with the *legend* ‘Select your t-shirt color” and complete it with the radio button elements as shown in FIGURE 4. (4)

4.1.9 Create a borderless table with one row and two columns to contain the *submit* and *clear* buttons, respectively. Set the *submit* button’s onclick event to invoke a function called *Validate().* (2)

4.1.10 Create the footer division and add your EXAMINATION NUMBER as copyright. (1)

[25]

QUESTION 5

Testing the integrity of the information entered by the students requires a "validation process" that must be invoked when the student clicks the *submit* button. You are therefore required to create a JavaScript function called *Validate()* that will be executed as soon as the student submits the form.

The function must check for VALID inputs, specifically making sure that the quantity field contains a numeric value and that the user selected a colour for the t-shirt. If any of the inputs are NOT VALID, then the function must inform the user with an appropriate message.

Complete the instructions below to add the *Validate()* JavaScript function to the index.html page created in QUESTION 4.

5.1 Write the code required to create the *Validate()* JavaScript function in the *index.html* document. (2)

5.2 Declare all the required variables for the form input fields and assign it the values as entered by the user. (5)

5.3 Determine whether or not the *quantity* input field holds a numeric value. If not, alert the user with an appropriate message. (2)

5.4 Determine whether or not the user has selected a colour for the t-shirt. If not alert the user with an appropriate message. (3)

5.5 Add your EXAMINATION NUMBER as a comment line at the TOP of the *Index.html* page and make a printout of the source code, including a screenshot of the browsed web page. (1)

[13]

TOTAL: 100

Copyright reserved