higher education & training

Department:
Higher Education and Training
REPUBLIC OF SOUTH AFRICA

NATIONAL CERTIFICATE (VOCATIONAL)

PRINCIPLES OF COMPUTER PROGRAMMING

NQF LEVEL 3

(Second paper)

(10041023)

28 November 2023 (X-paper)
09:00–13:00

CANDIDATES ARE ALLOWED 15 MINUTES TO READ THE INSTRUCTIONS BEFORE THE START OF THE EXAMINATION.

This question paper consists of 18 pages.

487Q2N2328

Copyright reserved

DEPARTMENT OF HIGHER EDUCATION AND TRAINING
REPUBLIC OF SOUTH AFRICA

NATIONAL CERTIFICATE (VOCATIONAL)
PRINCIPLES OF COMPUTER PROGRAMMING L3 P2
NQF LEVEL 3
TIME: 4 HOURS
MARKS: 150

INSTRUCTIONS AND INFORMATION

1. Candidates MUST be seated in the computer laboratory 45 minutes before the start of the examination to ensure that the allocated computer workstation has the appropriate software and can make printouts.

2. Answer all the questions.

3. Read all the questions carefully.

4. Number the answers according to the numbering system used in this question paper.

5. This question paper is set with programming terms that are specific to the BASIC programming language and must be completed using Visual Studio.NET or VB.NET Community/Express together with MS Office or applicable word-processing software.

6. Answer the questions according to the instructions and specifications given in each question. Marks will be awarded according to the set requirements.

7. Answer ONLY what is required in each question, for example if the question does not ask for data validation, no marks will be awarded for data validation.

8. Routines such as *search*, *sort*, *selection* and *file handling* must be hard-coded by you. You may NOT make use of the built-in features of the programming language.

9. All data structures must be defined by you, unless the question has a partially completed source code.

10. If a question has a partially completed source code, do NOT delete the source code.

11. Candidates are ALLOWED to make use of the Help Function of the Integrated Development Environment.

12. You must SAVE your work REGULARLY on the hard disk/flash disk/network drive that you have been given, to prevent loss of work. NO additional time will be allowed for work lost when work was not saved.

Copyright reserved

13. In the event of problems such as a Power Failure or computer or printing device breakdown, the invigilator will make the necessary arrangements to continue with the examination session. Candidates will Not Be Penalised for Time Lost under such circumstances.

14. No candidate may print his/her work or make a flash drive available for another candidate or access another candidate's work via the network. Any attempt to access any information from or transfer information to another candidate in whatever form is a contravention of the examination rules and regulations and will be viewed in an extremely serious light.

15. Candidates must make sure that they print their work immediately after they completed a question. The candidate's EXAMINATION NUMBER must appear on ALL printouts. No questions without a printed EXAMINATION NUMBER will be marked. Source code printouts and screenshot printouts with handwritten examination numbers will NOT be accepted or marked.

16. The source code must be printed directly from the Integrated Development Environment (IDE). Candidates are NOT ALLOWED to copy the source code from the IDE and paste in MS Word for printing purposes.

17. All PRINTOUTS of the source code and screenshots printed during the examination session MUST be handed in to the invigilator.

18. Steps to print a User Interface or any other screenshot:

    18.1 Run/Debug the solution (for Visual Studio.NET ONLY).

    18.2 Hold down the ALT key and press the print screen (Prt Sc) key.

    18.3 Open a blank MS Word document and paste the screen shot.

    18.4 Insert a blank THREE-column header.

        18.4.1 Add your EXAMINATION NUMBER against the left margin.

        18.4.2 Add your QUESTION NUMBER against the right margin.

    18.5 Make a PRINTOUT of the page.

19. The following data files have been saved on the computer in the **PCP L3 P2 NOV23 *Examination* folder**:

- QUESTION 1 → …\QUESTION1\Question_1.sln
- QUESTION 2 → …\QUESTION2\Question_2.sln
- QUESTION 3 → …\QUESTION3\Question_3.sln
- QUESTION 4 → …\QUESTION1\Question_4.sln
- QUESTION 5 → …\QUESTION1\MZANSI TVET.mdb

Copyright reserved

QUESTION 1

Mzansi TVET College recently launched their in-house mobile clinic for students and staff including the surrounding community. You have subsequently been approached to develop a software application that will be used to assist visitors to the clinic to determine whether or not they can afford to join the college's HEALTH 4-ALL Medical Aid Scheme.

The Scheme offers three levels of benefits, namely GOLD, SILVER and COPPER. Each level is priced individually for the Main Member (MM) and will increase depending on whether or not the member wants to include a number of dependents e.g., children, spouse, parents, etc. In order to encourage younger people to also sign-up, the scheme offers a substantial discount at each level to all prospective members who are younger than 35.

For example, if a person younger than 35 wants to join the HEALTH 4-ALL medical aid scheme on the GOLD LEVEL, he/she will pay an instalment of R1500.00 pm (see table below). If the same person wants to include one or more dependents, the instalment will increase by 45% for each of the dependents. Assuming the main member wants to include 3 dependents, the instalment will be calculated using the **Final Payment (FP)** formula below:

FP = instalment + (instalment x dependent%) x no-of-dependents

Using this formula with the given options, the person's final payment will be:

FP = 1500.00 + ((1500.00 x 0.45) x 3) = R3300.00 pm

The cost, benefits and discounts for each level are summarised in the Costing Table below.

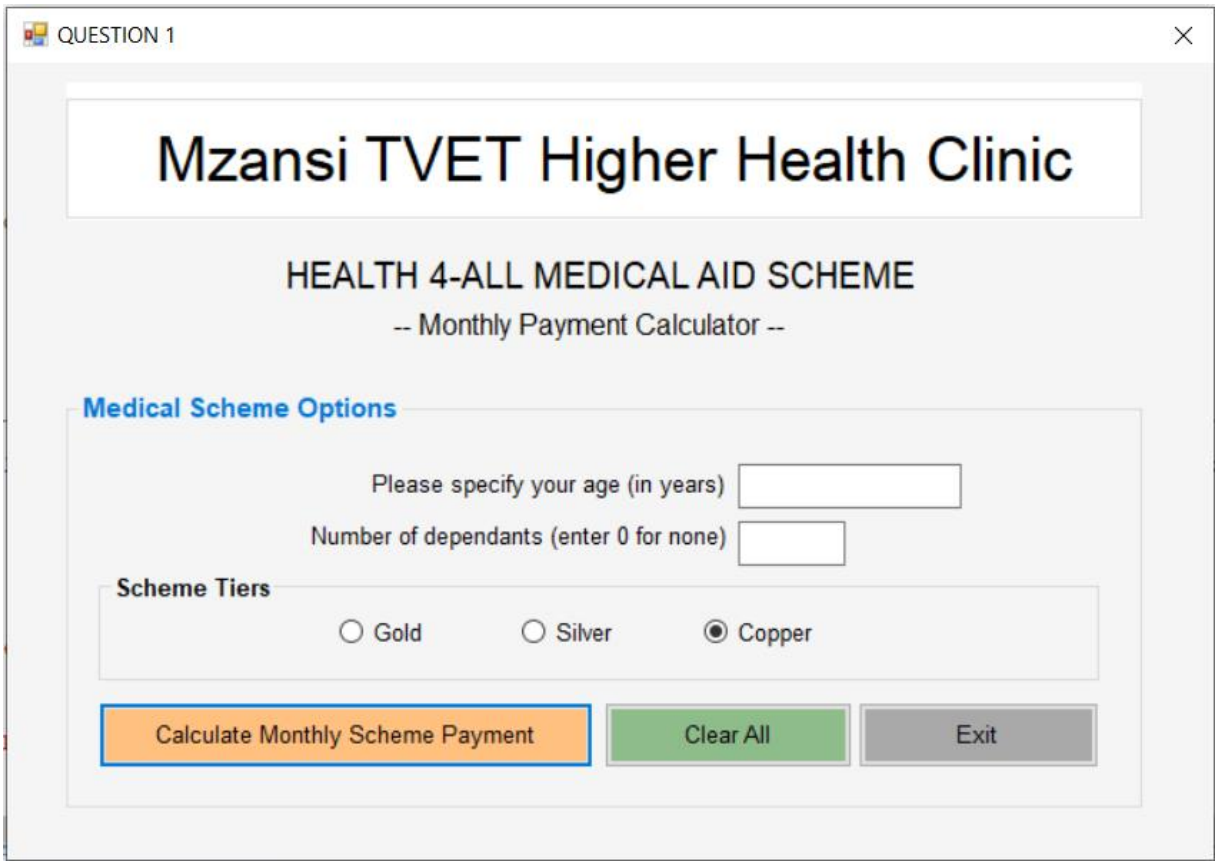
HEALTH 4-ALL Community Health Medical Aid Scheme			
BENEFIT TIER / LEVEL	**MONTHLY INSTALMENT**		**PER DEPENDENT %**
	YOUNGER THAN 35	**35 AND OLDER**	
GOLD	R1500.00	R1800.00	45%
SILVER	R1100.00	R1400.00	35%
COPPER	R750.00	R900.00	25%

TABLE 1.1

Copyright reserved

FIGURE 1.1 below is a screen shot of what the graphical user interface (GUI) should look like.

FIGURE 1.1 – Application GUI



The following steps show how to use the application to calculate the final medical aid installment based on the user's selections and inputs.

STEP 1: Run the program and supply the information as prompted i.e., provide an age, a number of dependents or enter 0 (zero) if there are none. Lastly, select the "*Scheme Tier*" or Level BEFORE clicking the "*Calculate Monthly Scheme Payment"* button. Please note that the default selected level is COPPER.

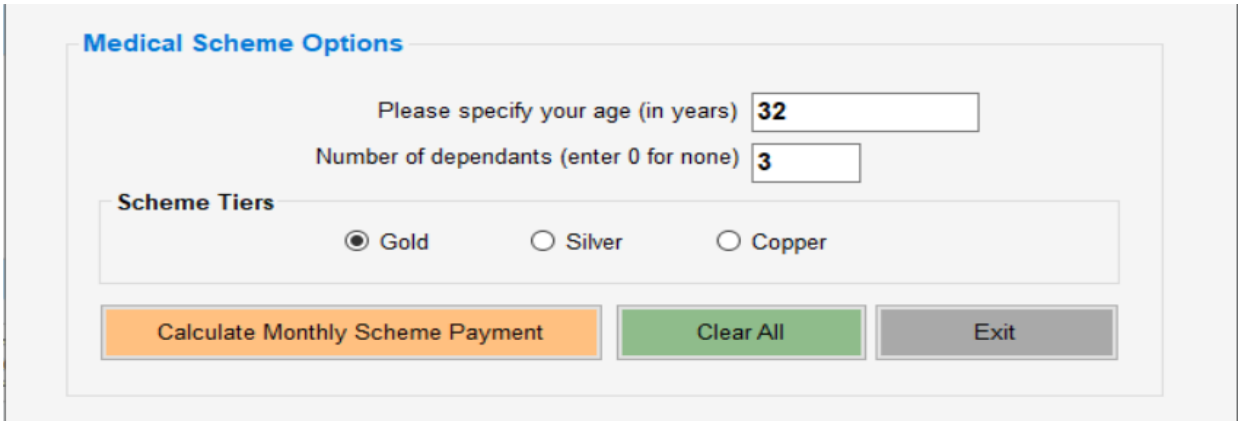


FIGURE 1.2 – User inputs & selections

Copyright reserved

STEP 2: Clicking the "*Calculate Monthly Scheme Payment*" button will calculate and display the prospective member's final instalment as in FIGURE 1.3 below.

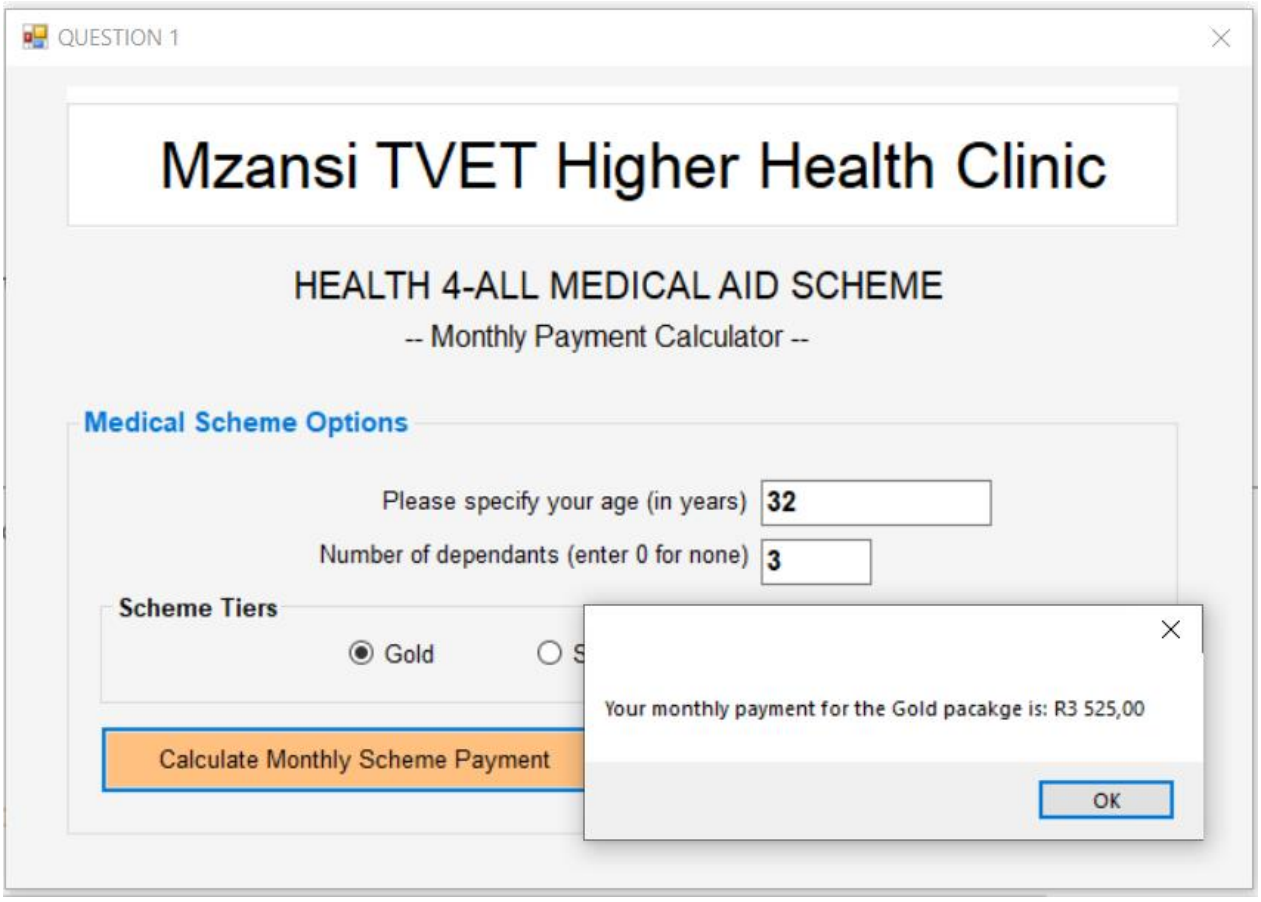


FIGURE 1.3 – Medical Aid Instalment Calculation

STEP 3: If the user clicks the "*Calculate Monthly Scheme Payment*" button without supplying either an age or a number of dependents, the system must display a request to comply with the required inputs as in FIGURE 1.4 below

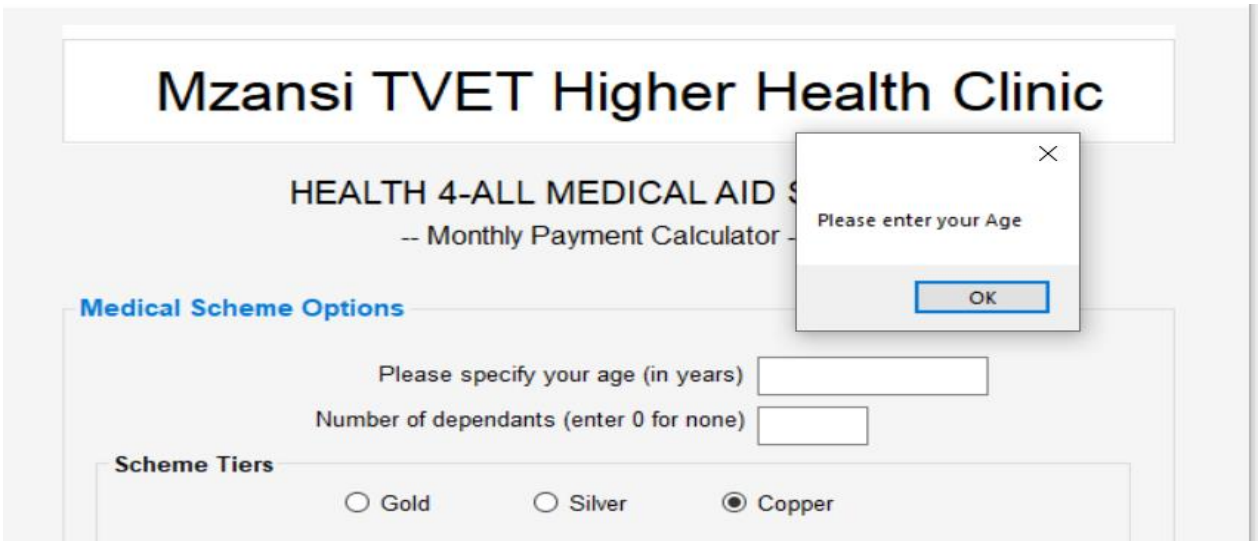


FIGURE 1.4 – Input Validation

Copyright reserved

The partially completed VB.Net solution, *Question_1.sln,* has been saved on the hard drive in the folder …\PCP L3 P2 NOV23\QUESTION1. Open the file in your Visual Basic IDE and follow the instructions below.

1.1 Add the VB.Net programming code to the 'Click Event' of the *Calculate Medical Scheme Payment* button that will do the following.

1.1.1 Declare all the required variables. (2)

1.1.2 Validate that the values for "*age*" AND "*number of dependents*" are not "empty". If either one of the inputs are empty, you have to inform the user with an applicable message as illustrated in FIGURE 1.4 above. (4)

1.1.3 Convert the values the user entered for "*age*" and "*number of dependents*", to Integer values. (2)

1.1.4 Make use of VB's "NESTED IF" selection structures to calculate and display the final medical aid instalment as demonstrated in FIGURE 1.3 above.

Use the **Final Payment (FP)** formula provided in the preceding program specifications as well as the data in TABLE 1.1 to code your calculations (12)

1.1.5 Change the Text in the title bar of the main form to your EXAMINATION NUMBER and run your program with the test data below. Make a SCREENSHOT printout of each result.

TIER / LEVEL	AGE	NO of DEPENDENTS
Silver	25	4
Gold	44	0

(2)

1.1.6 Enter your examination number, computer workstation number and date as comments in the first three (3) lines of the source code. Save and make a printout of the source code.

[22]

QUESTION 2

Not all visitors to Mzansi TVET College's HIGHER HEALTH CLINIC can afford to join a medical aid scheme. For this reason, the College decided to introduce an "INSTALMENT PLAN" solution to help some of their Clinic clients pay off their outstanding balances.

Mzansi TVET once again approached you to develop a computer program that will work out a detailed instalment plan for clients who want to make use of this option. The program must allow the user to enter an **account balance** and select a **payment period** before listing all the instalments, including the **total amount paid** over the given period.

FIGURE 2.1 below shows an example of an instalment plan that was calculated on an account balance of R1800.00 over a period of 6 months.

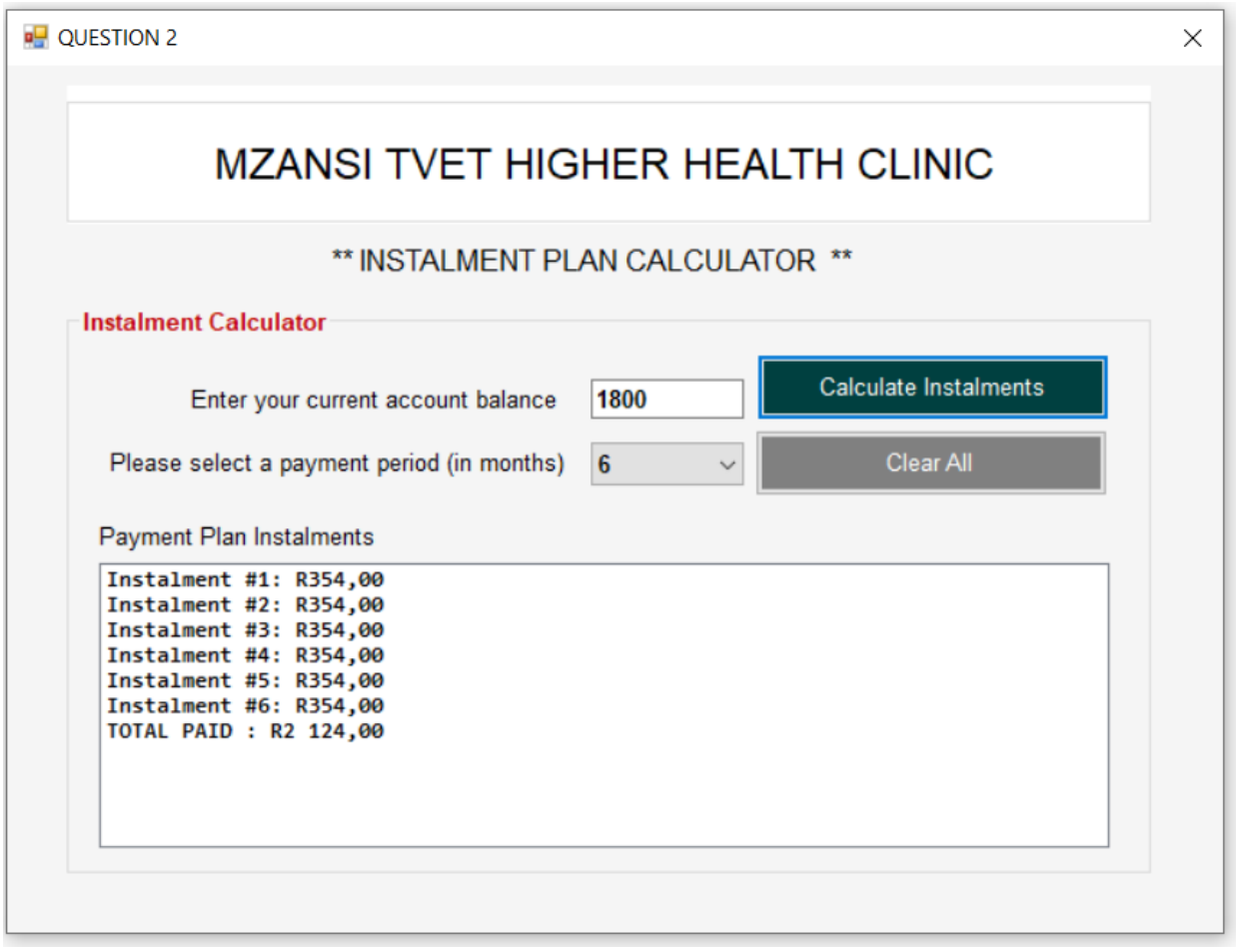


FIGURE 2.1

Clients can choose one of three payment periods namely, **3 months**, **6 months** or **9 months**. Each payment period attracts a different interest percentage as indicated in the list below

- **3 months:** *account balance* + 12%
- **6 months:** *account balance* + 18%
- **9 months:** *account balance* + 26%

Please note that for any given *payment period,* the applicable interest must be ADDED to the current balance BEFORE the instalments are calculated.

When the user clicks the *Calculate Instalments* button, the program must list a detailed breakdown of the instalments, including the TOTAL PAID, in the list box window "PAYMENT PLAN INSTALMENTS", illustrated in FIGURE 2.1 above.

The partially completed VB.Net solution, *Question_2.sln,* has been saved on the hard drive in the folder …\PCP L3 P2 NOV23\QUESTION2. Open the file in your Visual Basic IDE and follow the instructions below.

2.1 Add the programming code to the 'Click Event' of the *Clear All* button that will clear the "account balance" and the "list box control".

2.2 Write the program code in the 'Click Event' of the *Calculate Instalments* button that will do the following.

2.2.1 Declare all the required variables. (2)

2.2.2 Convert the string values for *account balance* and *payment period* to their applicable numeric values. (2)

2.2.3 Use VB.Net's "SELECT CASE" selection structure to determine which *payment* period was selected. For each selected payment period, calculate the *instalment* using the information provided in the preceding program specifications. LIST *the breakdown in payments,* including the TOTAL AMOUNT PAID over the given period, as illustrated in FIGURE 2.1 above. (14)

2.2.4 Change the Text in the title bar of the main form to your EXAMINATION NUMBER and run your program with the test data below. Make a SCREENSHOT printout of each result.

ACCOUNT BALANCE	PAYMENT PERIOD
R2200.00	6
R1380.00	9

(2)

2.2.5 Enter your examination number, computer workstation number and date as comments in the first three (3) lines of the source code. Save and make a printout of the source code.

[22]

Copyright reserved

QUESTION 3

Mzansi TVET College asked you to create an application to help track their water usage over the five (5) days that the college is operational. The program must allow the user to input the water usage in *kilolitres* for each of the five days i.e., from Monday (#1) to Friday (day #5). Currently the college pays R23.75 per *kilolitre* of water used. The program must use the values supplied for each day, together with the given cost per *kilolitre* to analyse the water usage in terms of the following:

- the **total kilolitres** used over the 5-day period
- the **total cost** incurred for the total amount of water used
- the **average kilolitres** used over the 5-day period
- the **average cost** incurred over the 5-day period
- the **highest consumption** in kilolitres over the 5-day period
- the **highest cost** incurred over the 5-day period
- the **most expensive day** in terms of cost over the 5-day period

FIGURE 3.1 below shows a screenshot of what the user interface should look like

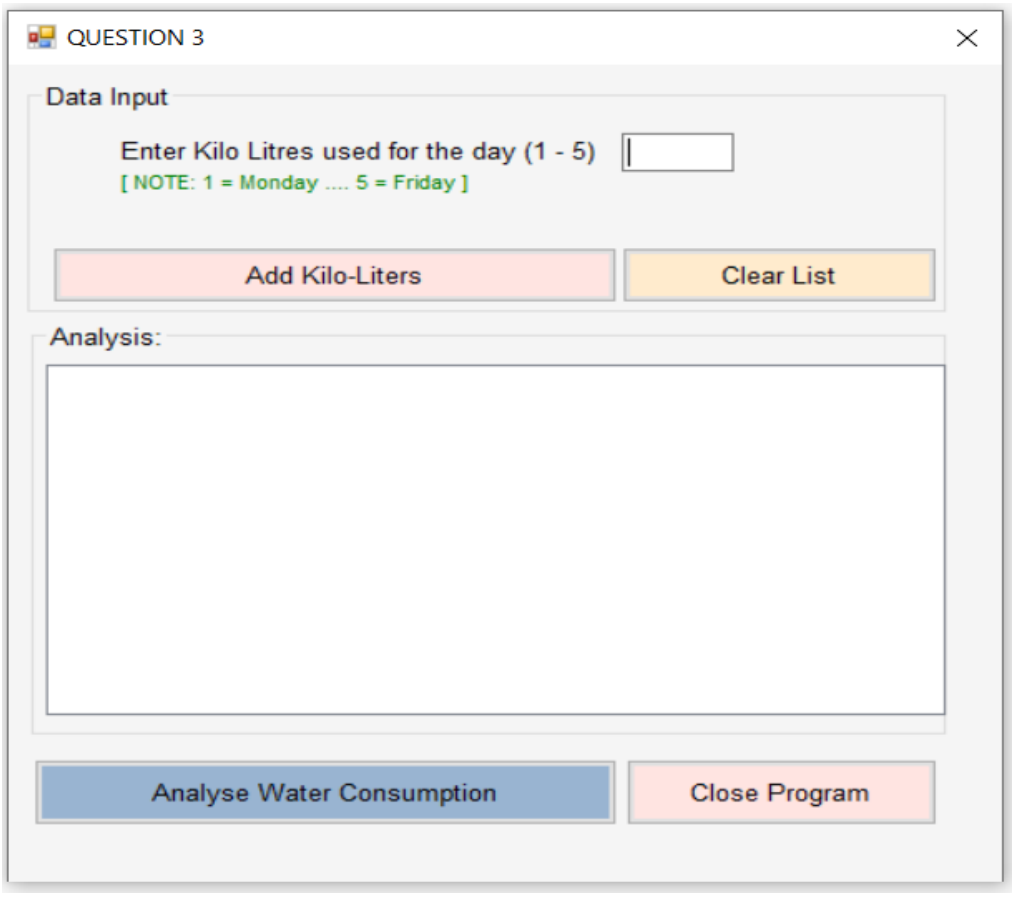


FIGURE 3.1

The following steps will explain how to use the program as well as what the required programmatic functionalities are.

STEP 1: When the program runs, the user must first be asked to enter the values for the water used per day, starting from Monday to Friday. All the daily values MUST first be entered before any analysis can be computed i.e.; the user MUST supply a value FOR EACH of the 5 days.

After entering an amount in *kilolitres,* the user has to click the "*Add Kilo-Litres"* button in order to "store" the value for that particular day. Each value entered, must be listed in the "*Analysis"* window BEFORE the user is prompted for the next value UNTIL ALL the days have been completed.

FIGURE 3.2 below shows a "completed 5-day list" as entered by the user.

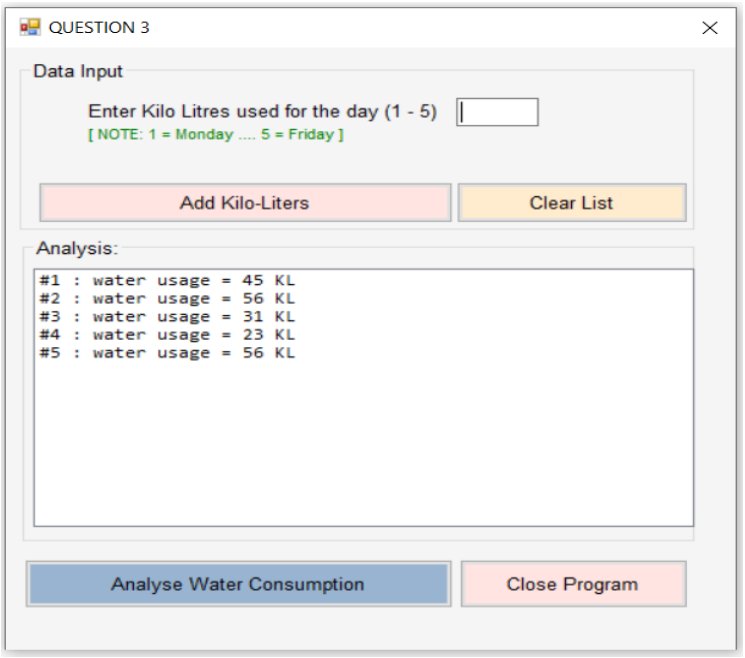


FIGURE 3.2

STEP 2: Before any analysis can be completed, the program must VALIDATE that a **kilolitre** value has been entered for each day. Also, if users attempt to enter MORE THAN 5 values the program must inform them that the "*List is full – only 5 values are allowed*", as shown in FIGURE 3.3 below

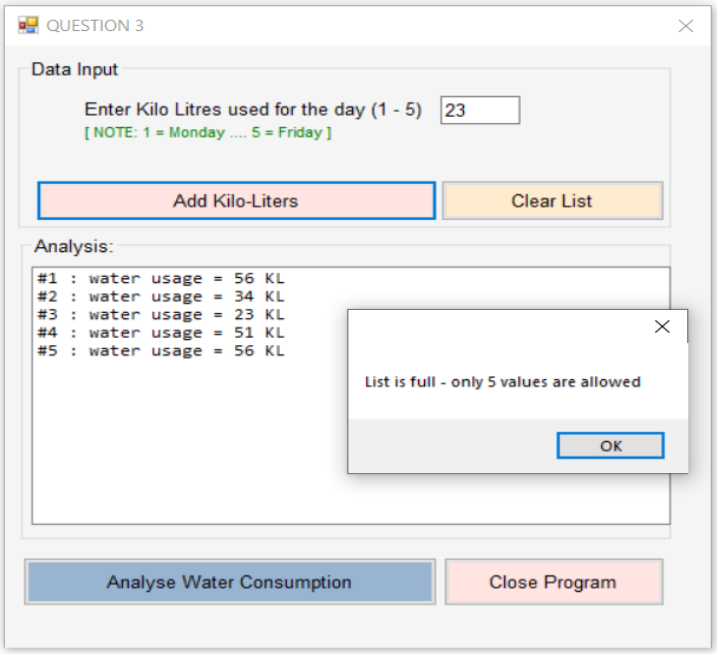


FIGURE 3.3

STEP 3: If users attempt to "Analyse Water Consumption" WITHOUT providing a proper list of **kilolitre** values for the five days, the program must inform them that "*Valid values are required to compute analysis".* In FIGURE 3.4 below, the user attempted to do an "Analysis" after only providing values for TWO days.

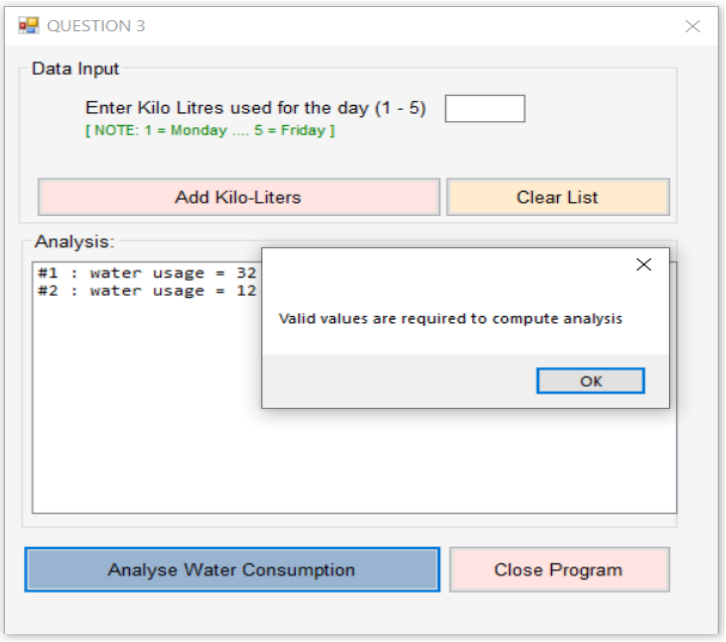


FIGURE 3.4

Copyright reserved

STEP 4: If the user clicks the "*Analyse Water Consumption"* button after providing a VALID SET of data, the program must calculate and display the required statistical values in the "Analysis" list as shown in FIGURE 3.5 below.

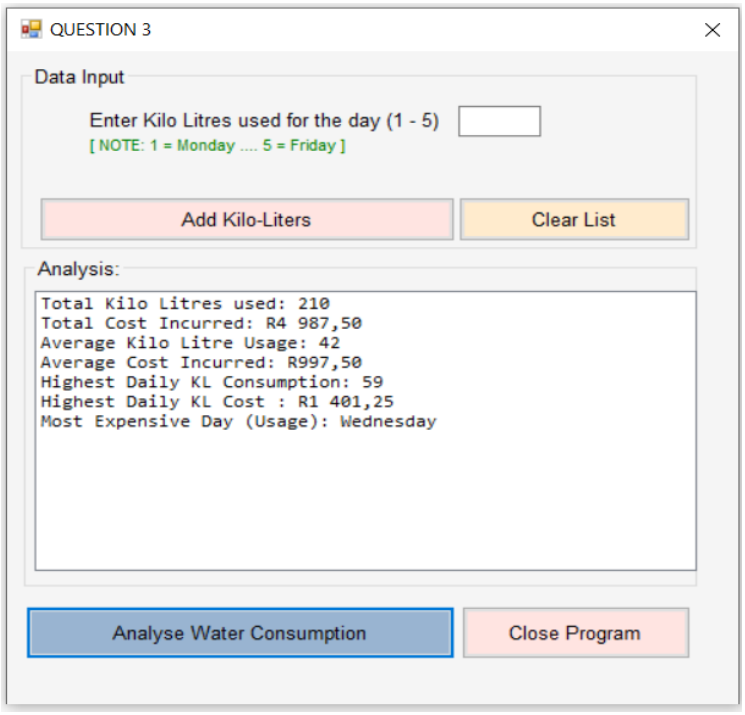


FIGURE 3.5

The partially completed VB.Net solution, *Question_3.sln,* has been saved on the hard drive in the folder …\PCP L3 P2 NOV23\QUESTION3. Open the file in your Visual Basic IDE and follow the instructions below.

3.1 Declare a single-dimension array in which to store the **kilolitre** amounts entered for each of the FIVE days. (1)

3.2 Declare all required string and numeric variables. (2)

3.3 Add the programming code to the 'Click Event' of the *Add Kilo-Litres* button that will do the following:

3.3.1 Validate the **kilolitre** input. If the value is "empty", inform the user to "*Please provide a value in Kilo Litres*". (2)

3.3.2 Determine if the user has ALREADY supplied a **kilolitre** value for ALL of the 5 days. If so, then inform the user that the "*LIST is full – only 5 values are allowed".* (2)

3.3.3 Display the entered **kilolitre** amounts in the 'Analysis" list as shown in FIGURE 3.2 above. Clear and set the "focus" on the **kilolitre** text field, ready for the next input value. (3)

Copyright reserved

3.4 Add the programming code to the 'Click Event' of the *Analyse Water Consumption"* button that will do the following:

3.4.1 Determine if a VALID list of **kilolitre** amounts is available to analyse. If NOT, then inform the user that "*Valid values are required to compute analysis*". (2)

3.4.2 Use a LOOP programming structure to do the following:

- Find the HIGHEST number of **kilolitres** consumed
- Determine which DAY had the HIGHEST **kilolitres**.
- Calculated the TOTAL number of **kilolitres** consumed (6)

3.4.3 Write the code to calculate the following values:

- TOTAL cost for the number of **kilolitres** consumed
- AVERAGE number of **kilolitres** consumed
- AVERAGE COST for the **kilolitres** consumed
- Cost for HIGHEST number of **kilolitres** *consumed* in a day. (4)

3.4.4 Display ALL the calculated values in the "*Analysis*" view as illustrated in FIGURE 3.5 above. (4)

3.4.5 Enter your examination number, computer workstation number and date as comments in the first three (3) lines of the source code. Save and make a printout of the source code

[26]

QUESTION 4

The mathematics lecturer at Mzansi TVET College requested your help in designing an application to capture students' contact information. FIGURE 4.1 below shows a screenshot of what the GUI looks like. The GUI should also give you an idea of which information about the students, the lecturer wants to capture.

When the lecturer clicks the '*Submit Student Record*' button, all the filled in details, per student, must be saved or written to a comma separated values (csv) file called *students.csv*,

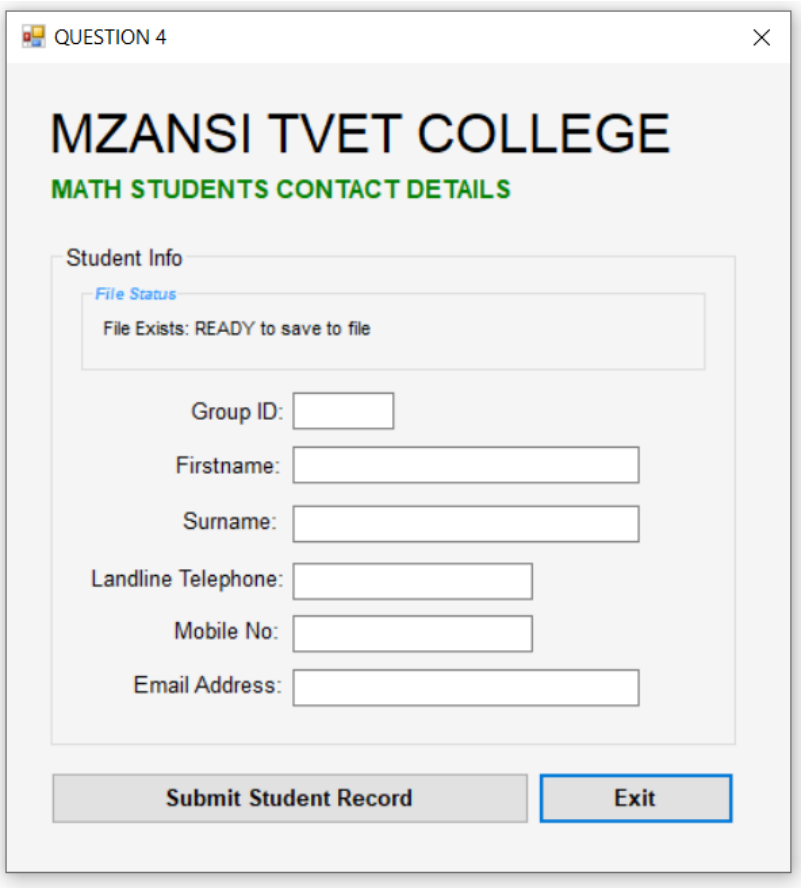


FIGURE 4.1

The partially completed VB.Net solution, *Question_4.sln,* has been saved on the hard drive in the folder …\PCP L3 P2 NOV23\QUESTION4. Open the file in your Visual Basic IDE and follow the instructions below.

4.1 Write the VB.Net code in the "*Form_Load"* sub procedure that will determine if the *students.csv* file exists or not.

If the file exists then display the message "*File Exists: Ready to save to file*" in the ***File Status*** label as indicated in FIGURE 4.1 above.

If the file DOES NOT exist, change the ***File Status*** message to "*File will be created when the SUBMIT button is clicked".* (3)

4.2 Write VB.Net code in the "Click Event" of the '*Submit Student Record'* button that will save the student information to the *students.csv* file. Make use of an appropriate 'exception handling' technique that will 'catch' any errors that may occur during the record submission process.

☆ If the record was written successfully, proceed to clear the input fields and set the *focus* to the *groupID* text box in readiness for the next record submission. Inform the user with the message "*SUCCESS: Student record saved to file*" that the record was saved.

If the record COULD NOT be saved, inform the user of the issue with the message "*ERROR: Student record was NOT saved*". (9)

4.3 Enter your examination number, computer workstation number and date as comments in the first three (3) lines of the source code. Save and make a printout of the source code

[12]

QUESTION 5

☆

The Deputy Principal at Mzansi TVET college asked you to create a database management system to assist with managing staff working in various departments. To simulate the work scenario, the database must consist of TWO tables called DEPARTMENT and EMPLOYEE, respectively.

The diagrams below show the data specification and structural definitions of each of the mentioned tables.

- **TABLE 1**: DEPARTMENT Table Structure

FIELD NAME	DATA TYPE	FIELD SIZE
Dep_ID	Number	Long Integer
Dep_Name	Short Text	22
Dep_Location	Short Text	30

- **TABLE 2**: EMPLOYEE Table Structure

FIELD NAME	DATA TYPE	FIELD SIZE	DESCRIPTION
Emp_ID	Autonumber	Long Integer	Automatic unique code (PK)
Emp_Fname	Short Text	25	First name
Emp_Lname	Short Text	25	Surname
Emp_Experience	Number	Integer	Number of years of experience
Dep_ID	Number	Long Integer	Foreign key

☆

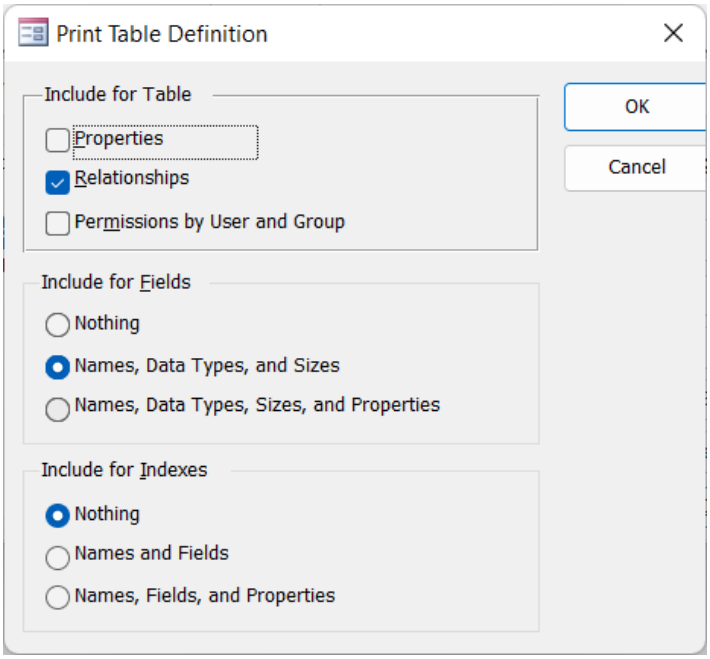
The database *MZANSI TVET.mdb,* consisting of the two above mentioned data tables, has already been created, and can be located in the folder …\PCP L3 P2 NOV23\ QUESTION5, on your workstation's hard drive.

Both tables have also been populated with sample data that reflects the current staff composition. Use MS Access to open the database and answer the questions below.

5.1 Create a "relationship" between the two tables, with enforced referential integrity, using the business rule below.

An *employee* can work in ONE and ONLY ONE *department,* but ONE *department* may have MANY *employees.*

Use the "Database Documenter" to set the *Print Table* definition options as in FIGURE 5.1 below, and print the database table structures for BOTH tables.



(2)

FIGURE 5.1

5.2 Create a *database query* in Design View using BOTH the *Employee* and *Department* tables. The query should display the following fields from each data table:

EMPLOYEE: ID, name, surname, experience
DEPARTMENT: name

Set the "Criteria" to only retrieve records from the "SALES" and "FINANCE" *departments* for employees with MORE THAN 15 years' experience. Save the query as ***qryExperience***.

Make a screenshot printout of the query form in "Design View" – make sure that all the columns are clearly visible (4)

5.3 Run the query called *qryExperience* created in QUESTION 5.2 and make a printout of the results (2)

5.4 Create a query using ONLY the *Employee* data table. All the query has to do, is COUNT the number of employees in the SALES department. (3)

Save the query as *qrySales* and make a screenshot printout of the query form in Design View.

5.5 Run the query called *qrySales* created in QUESTION 5.4 and make a printout of the results. (2)

5.6 Create a Database Report using the query called *qryExperience*, created in QUESTION 5.2. The report should "Group" the report detail per *Department* and also calculate and display the *average* years of experience for ALL employees.

Make sure your EXAMINATION NO appears in the report header. Save the report as *Service_Report* and print it. (5)

[18]

TOTAL: 100

Copyright reserved