**B Tech Computer Science and Engineering (Data Science)**

**-311 Program Semester – II**

**Academic Year 2023-24**

**Object Oriented Programming and Design**

**Mini Project**

**Aptitude Test Analyzer**

[**presenation link**](https://docs.google.com/presentation/d/1sbhUjbNAUMwqc2J0dPYjJ8o2hOy6rEGZYmU74j7h1RA/edit?usp=sharing)

Developed by

L004 – Ishaa Bhalekar

L005 - Arpita Das

L007- Sini Goyal

L014 - Saanvi Kanodia

**Faculty In charge**

**Dr. Dhirendra Mishra**

**SVKM’s NMIMS**

**Mukesh Patel School of Technology Management and Engineering**

**Vile Parle West Mumbai-56**

**Content**

| **Chapter No.** | **Topics** | **Page Numbers** |
| --- | --- | --- |
| 1 | Abstract |  |
| 2 | Introduction |  |
| 3 | Scope of the Project -Societal need of this project |  |
| 4 | Organization of project |  |
| 5 | Research Question |  |
| 6 | Overall Block diagram of functionalities |  |
| 7 | Problem Statement |  |
| 8 | List of classes and its members planned |  |
| 9. | USE CASE Diagram |  |
| 10. | Class Diagram |  |
| 11. | Activity Diagram |  |
| 12. | Sequence Diagram |  |
| 13. | Object Oriented concepts [Encapsulation, Inheritance, Polymorphism, Friend class/function etc.] to be implemented in the code with justification. |  |
| 14 | Algorithms |  |
| 15 | Flowcharts |  |
| 16 | Code written in C++ |  |
| 17 | Output screenshots of all functions |  |
| 18 | List of Errors obtained and its resolution. |  |
| 19 | Scope of improvements [plans to improve it] |  |
| 20 | References/List of research papers, books, white papers referred. (in IEEE format) |  |

**1.Abstract:** This program aims to create an Aptitude test analyser that allows users to answer a series of questions and find out the details of their performance. We aim to imitate online tests while adding additional features and creating a user-friendly environment for the user. For this purpose, the program makes use of Object-oriented programming to create individual blocks of relevant data and retrieve it during run time. This is aided by the use of classes and source files.

Where different types of data are allied to exist in different sections so as to not cause any unwanted modification of manipulation. This will also restrict data handling to those functions

who are given the access to do so.

**2.Introduction:** While there are many projects that create test analysis apps using various framework/stylistic languages, we say a lack in the existence of a coding language that manned the entire project. Additionally, we noticed that while more basic analysis sites might provide free services, those which provided detailed analysis were only available through paid access. To create a program that can be accessed without payment we decided to use C++ language to man the process of testing and analyzing users based on their aptitude.

**Scope of the Project – Societal need of this project:**  Students can evaluate potential career choices based on their strengths and weaknesses by taking an aptitude test. The student is able to identify their weaker domains and work on them. The program guarantees an in-depth and open examination of the test. Additionally, the test gives students a specific amount of time to complete the questions so they can gauge how well they can respond to time constraints. A free testing program is also very necessary to have.

**3.Organisation of project:**

**4.Research Objective**: The aim of the project is to create an aptitude test for students that provides detailed performance analysis based on their critical thinking, logical reasoning, and problem-solving skills respectively. Additionally, we aim to integrate these features with a constantly running timer block that will help to provide required time analysis.

**5.Research Question:** How to create an aptitude test that benefits the students in analysing their strengths and weaknesses using Object-oriented programming?

**6.** **Overall Block diagram of functionalities**

**7.** **Problem Statement:** The aptitude test helps the students to identify their potential by analysing their presence of mind, expertise in a particular domain. Also, the test helps students evaluate their progress and control over time by comparing their test scores and speeds in different domains like critical thinking, logical reasoning, and problem-solving skills respectively. User will enter answer to eachquestion and based on the time taken for every question and section, program will output detailed analysis and review. User also has option to enter notes and review them later.

**8.List of classes and its members planned:**

1.  **DATABASE Class**:

a. Attributes: question (string), method (string), ans1 (char).

b. Methods: print\_quest() (returns a string), print\_method() (returns a string).

2. **Calculate Class**:

a. Attributes: score (int), sect\_mrks[4] (int array), accuracy (double), indicator[] (string array).

b. Methods: review() ( return string), check() (returns an int), strength\_weak(), analysis().

3. **USER Class**:

a. Attributes: input1 (char), notes (string).

b. Methods: store\_ans() (returns a char), create\_notes() (returns a string).

4. **TIMER Class**:

a. Attributes: time (float), total\_time (float).

b. Method: print\_time() (returns a float).

5. **TIME ANALYSIS Class**:

a. Attributes: speed (double), user\_time (float).

b. Methods: avg\_speed(), section\_time() (both return double values)

**9.USE CASE DIAGRAM:**

**Description for Use Case Diagram:**

1. **Actors**:

· The primary actor is the User (presumably a test-taker or an instructor).

· Other actors include:

§ Calculate Actor Service: Responsible for storing answers and creating notes.

§ Time Actor Service: Manages timing-related tasks, such as starting the timer and calculating average speed.

§ Database Actor Service: Handles displaying questions and printing methods used to solve them.

§ Timer Actor Service: Calculates the time taken by the user.

2. **Use Cases** (Actions Performed by the User):

· Checks Answers: The user can verify their answers against the correct ones.

· Evaluates Strength and Weaknesses: Assess their performance in different areas.

· Reviews Entire Test: Allows a comprehensive review of the entire test.

· Analysis of Performance According to the Domain: Provides insights based on the test domain.

3. **Interactions**:

· The user interacts with the system through these use cases.

· The services (actors) assist in various tasks, such as giving answers, managing time, and displaying relevant information.

This diagram captures the high-level interactions between the user and the system, emphasizing functionality from the user’s perspective.

**10.CLASS DIAGRAM:**

**Description of Class Diagram:**

6. **DATABASE Class**:

· Attributes: question (string), method (string), ans1 (char).

· Methods: print\_quest() (returns a string), print\_method() (returns a string).

· Purpose: The DATABASE class seems to store information related to questions, their solution methods, and answers.

7. **Calculate Class**:

· Attributes: score (int), sect\_mrks[4] (int array), accuracy (double), indicator[] (string array).

· Methods: review() (returns a string), check() (returns an int), strength\_weak(), analysis().

· Purpose: The Calculate class likely performs calculations based on data from the DATABASE. It checks user input and provides analysis.

8. **USER Class**:

· Attributes: input1 (char), notes (string).

· Methods: store\_ans() (returns a char), create\_notes() (returns a string).

· Purpose: The USER class interacts with the Calculate class, providing input and receiving feedback.

9. **TIMER Class**:

· Attributes: time (float), total\_time (float).

· Method: print\_time() (returns a float).

· Purpose: The TIMER class displays time information to the user and provides timing data for the TIME ANALYSIS.

10. **TIME ANALYSIS Class**:

· Attributes: speed (double), user\_time (float).

· Methods: avg\_speed(), section\_time() (both return double values).

· Purpose: The TIME ANALYSIS class analyzes timing data received from the TIMER related to the user’s performance or actions.

**11.ACTIVITY DIAGRAM:**

**Description for Activity Diagram:**

11. **Displaying Questions and Timer**:

· When user will start the Test, the system displays a question for you to answer.

· Simultaneously, a timer begins counting down. It will be displayed at the top of the screen.

12. **Recording Your Answering Time**:

· As users will think through the question, the system records how long it takes you to answer.

· This process repeats a total of 20 questions.

13. **Storing Your Answer**:

· After answering each question, the user’s response is stored by the system.

14. **Entering Notes**:

· Once the user has completed all 20 questions, user will have the option to enter any notes or comments related to your answers.

15. **Viewing Total Marks and Accuracy**:

· Next, the system displays the user's total marks and accuracy based on the user's answers.

16. **Choosing Between Review and Analysis**:

· Users can decide whether to review their performance or analyze it further.

17. **Review Option**:

· If the user chooses to review, the system shows you the questions along with their correct answers.

· Users can also view any notes they have entered and even print them if needed.

18. **Analysis Option**:

· Alternatively, if user selects analysis, the system provides additional insights:

§ Average speed (how quickly you answered questions)

§ Section-wise time distribution

§ Strengths and weaknesses based on the user’s performance

19. **Exiting the Activity**:

· Finally, when the user is done reviewing or analyzing, you can exit the activity.

**12. SEQUENCE DIAGRAM:**

**Description for Sequence Diagram:**

When the program starts, the screen displays the homepage and instructions for the test. The user selects the "Start Test" option to begin. The first question is displayed, along with a timer indicating the total time remaining in the test. The user has the option to enter notes for reference.

After answering each question, the user can choose the "Next" option to proceed to the next question. This process repeats for 20 questions. The last question presents a "Finish Test" option, which terminates the test.

Upon completion, the screen displays the total marks and accuracy of the test. Accuracy is determined by the number of correct answers out of the total questions.

After completing the test, the user is presented with three options:

- View Review: Displays each question, the correct answer, the user's answer, the methodology, and any previously created notes.

- View Analysis: Shows the total time taken to complete the test, the average speed per question, the time taken in each section (logical, problem-solving, critical thinking, arithmetic) marks scored in each section and based on that which section is a strength of the user, and which is a weakness.

- Exit: Allows the user to exit the program. After selecting this option, an exit page is displayed.

**13. CONCEPTS**

1. Class:

**14. ALGORITHM STEPS**

1) Start

2) Display Start page & instructions stored in database

3) User will start the test

4) Initialize question number to i

5) If i<20 go to step 6 , else if i=20 go to step 14

6) Display test question, question[i]

7) Print timer from variable time

8) Record the time when question was printed in time\_start

9) Store user answers in store[]

10) If the user knows the answer goto step 11 else go to step 6

11) Store will be initialized with (a/b/c/d)

12) Store user notes in notes[] (optional)

13) When user clicks next option, go to step 6

14) User finishes test

15) Record the time when the test was finished in time\_stop

16) Calculate and Display total marks scored in the test

17) Calculate and display accuracy

18) Display review, analysis, and exit option

19) If user selects review goto step 20, Else if analysis goto step 25, else if exit go to step 30

20) Display Review interface. Print Question[]

21) Display answer[]

22) Display store[]

23) Display method[]

24) Display notes[]

25) Display Analysis interface. Print total\_time

26) Display avg\_speed

27) Display section\_time

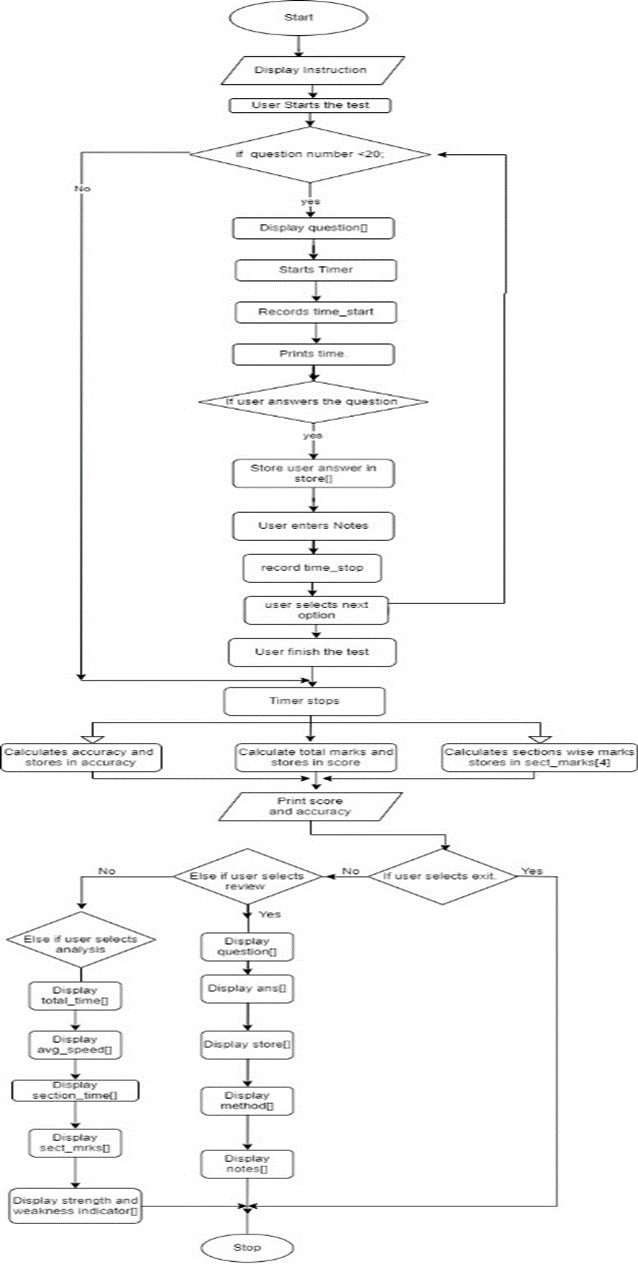
28) Display Sect\_mrks[]

29) Display strengths & weaknesses of the user by printing indicator[] goto step 25

30) Display Exit page

31) End

**15. FLOWCHART DIAGRAM**



**20. References/List of research papers,books,white papers referred. (in IEEE Format).​​￼**

**​​​**

**​​References**

**​​**

| **​[1]** | **​R. C. Seacord, Secure Coding in C and C++, 2 ed., Addison-Wesley, 2013, pp. 33,37.** |
| --- | --- |

**​​References**

**​​​​​​￼**

| **​[1]** | **​E.Balaguruswamy, Object Oriented Programming with c++, 4th Edition ed., Dehi, Telangana: Tata McGraw-Hill Publishing Company Limited, 1990.** |
| --- | --- |
| **​[2]** | **​R. C. Seacord, Secure Coding in C and C++, 2 ed., Addison-Wesley, 2013, pp. 33,37.** |

**​​​​​￼**

**​​​￼**

**​​​**

**​​[1]​**

**​​Bibliography**

**​​**

| **​[1]** | **​E.Balaguruswamy, Object Oriented Programming with c++, 4th Edition ed., Dehi, Telangana: Tata McGraw-Hill Publishing Company Limited, 1990.** |
| --- | --- |
| **​[2]** | **​R. C. Seacord, Secure Coding in C and C++, 2 ed., Addison-Wesley, 2013, pp. 33,37.** |

**​​​￼**

**​​​**

**​​Bibliography**

**​​**

| **​[1]** | **​R. C. Seacord, Secure Coding in C and C++, 2 ed., Addison-Wesley, 2013, pp. 33,37.** |
| --- | --- |