Python Pseudocode - Greenwich University Quiz Application

Module 1: main.py

```
ALGORITHM MainApplication
BEGIN
    PRINT welcome messages
    PRINT application information
    CALL run_login_app() FROM login_app module
    PRINT session ended messages
END
```

Module 2: login_app.py

```
CLASS App INHERITS FROM Tk
BEGIN
    METHOD __init__()
    BEGIN
        CALL super().__init__()
        SET window title = "Greenwich University Project"
        SET window icon = logo.ico file
        SET window geometry = "800x600"

        TRY
            LOAD original_logo FROM png file
            SET logo_image = original_logo.subsample(3, 3)
            CREATE logo_label WITH logo_image
            PACK logo_label WITH padding
        EXCEPT
            CREATE logo_label WITH text "Greenwich University"
            PACK logo_label WITH padding
        END TRY

        CREATE title_label WITH text "Exam Quiz"
        PACK title_label

        CREATE info_label WITH text "Enter your student data below."
        PACK info_label

        CREATE prompt_name WITH text "Enter your full name: "
        CREATE name_entry AS Entry widget
        PACK prompt_name
        PACK name_entry

        CREATE prompt_email WITH text "Enter your email: "
        CREATE email_entry AS Entry widget
        PACK prompt_email
        PACK email_entry

        CREATE prompt_id WITH text "Enter your ID: "
        CREATE id_entry AS Entry widget
        PACK prompt_id
        PACK id_entry

        CREATE login_button WITH text "Login to Quiz"
```

```
        SET login_button.command = handle_login
        PACK login_button

        INITIALISE student_data = empty dictionary
    END

    METHOD handle_login()
    BEGIN
        GET name FROM name_entry.get().strip()
        GET email FROM email_entry.get().strip()
        GET student_id FROM id_entry.get().strip()

        IF name IS empty OR email IS empty OR student_id IS empty THEN
            SHOW error message "Please fill in all fields."
            RETURN
        END IF

        SET student_data = {
            "name": name,
            "email": email,
            "id": student_id
        }

        SHOW success message "Welcome [name]! Starting quiz application..."
        CALL self.destroy()
        CALL start_quiz_app()
    END

    METHOD start_quiz_app()
    BEGIN
        TRY
            IMPORT create_app FROM app module
            SET quiz_app = create_app()
            SET quiz_app.student_data = self.student_data
            CALL quiz_app.mainloop()
        EXCEPT ImportError
            SHOW error message "Could not load quiz application"
        END TRY
    END
END

FUNCTION create_login_app()
BEGIN
    RETURN new App instance
END

FUNCTION run_login_app()
BEGIN
    SET app = create_login_app()
    CALL app.mainloop()
END
```

Module 3: app.py

```
CLASS App INHERITS FROM Tk
BEGIN
    METHOD __init__()
    BEGIN
        CALL super().__init__()
        SET window geometry = "600x600"
        SET window title = "Greenwich University Project - Quiz"
        INITIALISE student_data = empty dictionary
        CALL init_quiz_data()
        CALL setup_ui()
    END

    METHOD init_quiz_data()
    BEGIN
        SET quiz_data = [
            [software_engineering_questions],
            [logic_design_questions],
            [algorithm_questions]
        ]
        // Each category contains array of question objects with:
        // - question: string
        // - choices: array of strings
        // - answer: string
    END

    METHOD setup_ui()
    BEGIN
        CREATE container AS Frame
        PACK container WITH fill="both", expand=True
        SET container grid configuration

        INITIALISE frames = empty dictionary

        CREATE main_menu = MainMenu(container, self)
        SET frames["MainMenu"] = main_menu
        GRID main_menu

        FOR each quiz_page IN [SoftwareQuiz, LogicDesignQuiz, AlgorithmQuiz]
        BEGIN
            CREATE frame = QuizPage(container, self, category_index, title)
            SET frames[quiz_page_key] = frame
            GRID frame
        END FOR

        CALL show_frame("MainMenu")
    END

    METHOD show_frame(page_name)
    BEGIN
        GET frame FROM frames[page_name]
        CALL frame.tkraise()
    END
END

FUNCTION create_app()
BEGIN
    RETURN new App instance
END
```

Module 4: quiz_components.py

```
CLASS MainMenu INHERITS FROM Frame
BEGIN
    METHOD __init__(parent, controller)
    BEGIN
        CALL super().__init__(parent)
        SET self.controller = controller

        CREATE main_container AS Frame
        PACK main_container WITH expand=True, fill="both"

        CREATE content_frame AS Frame
        PACK content_frame WITH expand=True

        TRY
            LOAD original_logo FROM png file
            SET logo_image = original_logo.subsample(3, 3)
            CREATE logo_label WITH logo_image
            PACK logo_label
        EXCEPT
            CREATE logo_label WITH text "Greenwich University"
            PACK logo_label
        END TRY

        CREATE title_label WITH text "Exam Quiz"
        PACK title_label

        CALL create_student_info_section(content_frame)

        CREATE buttons_frame AS Frame
        PACK buttons_frame

        FOR each button_config IN quiz_buttons
        BEGIN
            CREATE button WITH text and command
            PACK button
        END FOR
    END

    METHOD create_student_info_section(parent)
    BEGIN
        IF controller HAS student_data AND student_data IS NOT empty THEN
            GET student_data FROM controller.student_data

            CREATE info_frame WITH background colour
            PACK info_frame

            CREATE welcome_label WITH text "Welcome, [student_name]!"
            PACK welcome_label

            CREATE details_label WITH text "ID: [id] | Email: [email]"
            PACK details_label

            CREATE logout_button WITH text "Logout"
            SET logout_button.command = logout
            PACK logout_button
        END IF
    END

    METHOD logout()
    BEGIN
        CLEAR controller.student_data
        CALL controller.show_frame("LoginPage")
```

```
    END
END

CLASS QuizPage INHERITS FROM Frame
BEGIN
    METHOD __init__(parent, controller, category_index, title)
    BEGIN
        CALL super().__init__(parent)
        SET self.controller = controller
        SET self.category_index = category_index
        SET self.title_text = title

        GET self.questions FROM controller.quiz_data[category_index]
        SET self.total_questions = length of questions
        SET self.current_question = 0

        INITIALISE self.user_answers = array of -1 values

        CALL create_header_section()
        CALL create_question_section()
        CALL create_choices_section()
        CALL create_feedback_section()
        CALL create_navigation_buttons()
        CALL create_results_section()
        CALL show_current_question()
    END

    METHOD show_current_question()
    BEGIN
        GET current_q FROM questions[current_question]

        SET progress_text = "Q " + (current_question + 1) + " / " + total_questions
        UPDATE progress_label WITH progress_text
        UPDATE question_text WITH current_q["question"]

        DESTROY all widgets IN choices_container
        CLEAR radio_buttons array

        SET selected_answer = user_answers[current_question]

        FOR each choice IN current_q["choices"]
        BEGIN
            CREATE radio_button WITH choice text
            SET radio_button.variable = selected_answer
            SET radio_button.value = choice_index
            PACK radio_button
            ADD radio_button TO radio_buttons array
            INCREMENT choice_index
        END FOR

        CLEAR feedback_message
        CLEAR result_text

        IF current_question == total_questions - 1 THEN
            SET next_button.text = "Finish"
        ELSE
            SET next_button.text = "Next"
        END IF

        ENABLE next_button
        CALL enable_answer_choices()
    END

    METHOD go_to_next_question()
    BEGIN
```

```
    GET selected_choice FROM selected_answer.get()

    IF selected_choice == -1 THEN
      SET feedback_message = "Please select an option before continuing."
      RETURN
    END IF

    SET user_answers[current_question] = selected_choice

    IF current_question == total_questions - 1 THEN
      SET final_score = calculate_final_score()
      SET score_text = "Your score: " + final_score + " / " + total_questions
      UPDATE result_text WITH score_text
      DISABLE next_button
      CALL disable_answer_choices()
    ELSE
      INCREMENT current_question
      CALL show_current_question()
    END IF
  END

  METHOD calculate_final_score()
  BEGIN
    SET correct_answers = 0

    FOR question_index FROM 0 TO length of questions
    BEGIN
      GET question FROM questions[question_index]
      GET user_choice_index FROM user_answers[question_index]

      IF user_choice_index != -1 THEN
        GET user_choice_text FROM question["choices"][user_choice_index]
        GET correct_answer FROM question["answer"]

        IF user_choice_text == correct_answer THEN
          INCREMENT correct_answers
        END IF
      END IF
    END FOR

    RETURN correct_answers
  END

  METHOD restart_quiz()
  BEGIN
    SET current_question = 0

    FOR i FROM 0 TO total_questions
    BEGIN
      SET user_answers[i] = -1
    END FOR

    CALL show_current_question()
  END

  METHOD go_back_to_menu()
  BEGIN
    CALL restart_quiz()
    CALL controller.show_frame("MainMenu")
  END
END
```

Main Program Flow

```
ALGORITHM QuizApplicationFlow
BEGIN
    START main.py
    CALL run_login_app()

    WHILE user has not logged in successfully
    BEGIN
        DISPLAY login form
        WAIT for user input
        VALIDATE input
        IF validation fails THEN
            SHOW error message
            CONTINUE
        END IF
    END WHILE

    STORE student data
    CLOSE login window
    START quiz application
    PASS student data to quiz app

    WHILE quiz application is running
    BEGIN
        DISPLAY main menu with student info
        WAIT for quiz selection

        IF quiz selected THEN
            LOAD quiz questions
            FOR each question IN selected quiz
            BEGIN
                DISPLAY question and choices
                WAIT for answer selection
                VALIDATE answer selection
                STORE user answer
            END FOR
            CALCULATE and DISPLAY final score
        END IF

        IF logout selected THEN
            RETURN to login screen
        END IF
    END WHILE

    END application
END
```

Data Structures

```
STRUCTURE QuestionObject
BEGIN
   question: STRING
   choices: ARRAY OF STRING
   answer: STRING
END

STRUCTURE StudentData
BEGIN
   name: STRING
   email: STRING
   id: STRING
END

STRUCTURE QuizData
BEGIN
   software_questions: ARRAY OF QuestionObject
   logic_questions: ARRAY OF QuestionObject
   algorithm_questions: ARRAY OF QuestionObject
END
```