Innovative Pedagogical Approach:

AI-Enabled Formative Assessment on C Programming Fundamentals

Course: Problem Solving and Programming

Faculty

Syed Muhammed Fahd

Innovative Pedagogical Approach:

AI-Enabled Formative Assessment on C Programming Fundamentals

Concept and Design

The *Problem Solving and Programming (PSP) Quiz* was designed as an **interactive, formative assessment** to strengthen students' foundational understanding of C-programming syntax, logic, and execution flow. Rather than testing rote recall, this quiz engages students in **code interpretation and logic assessment**, requiring them to mentally simulate program behaviour before responding.

Implemented through **Google Forms**, the quiz provides **immediate feedback** upon submission and allows **multiple attempts**, with the highest score retained. This structure encourages iterative improvement and reflective learning—key features of an effective formative assessment.

Content Focus of the Quiz

The content of the quiz was carefully structured to assess the core building blocks of C programming that students often find conceptually challenging. The quiz content spans the following sections:

1. **Console Input/Output Operations**
   Questions assess how students use printf( ) and scanf( ) with correct format specifiers, handle different data types, and interpret output behaviour. They also test understanding of syntax errors and runtime issues associated with I/O operations.
2. **Identifier Rules and Variable Declarations**
   Items evaluate students' grasp of valid naming conventions, data-type requirements, and declaration syntax—concepts essential for formulating correct algorithms and avoiding compilation errors.
3. **Operators, Precedence, and Associativity**
   A multi-step question guides students to determine the exact order in which operators are executed in expressions such as
   int result = a + sqrt(b * c) - d / e;.
   This helps reinforce the logical structure underlying arithmetic and relational expressions, directly addressing *conditional and iterative reasoning* outcomes.
4. **Control Structures (if–else)**
   Code-based True/False statements focus on branching logic, indentation, and the correct use of braces. Students analyse program flow and determine the actual output, building confidence in conditional evaluation.
5. **Logical and Conditional Operators**
   The questions also assess the understanding of operators such as &&, ||, ==, and =, clarifying distinctions between comparison and assignment—common sources of logical errors in early programming.

Collectively, these question types reinforce the *Understand*-level competencies of **CO1, CO2, and CO6**.

Integration of Generative AI in Quiz Creation and Evaluation

A key feature of the quiz is its use of **AI tools for design and automation**:

1. **Quiz Creation via ChatGPT and Google Apps Script**
   The entire Google Form was automatically generated using a **Google Apps Script** produced through **Generative AI (ChatGPT)**. The script inserted all questions, options, and marking schemes programmatically, ensuring consistency and eliminating repetitive manual entry.
   This process demonstrates how teachers can leverage AI-based code generation to streamline quiz preparation.
2. **Automated Score Analysis**
   Student responses were collected in a linked Google Sheet. A **Python script**, also generated through Generative AI (Github Copilot), was used to process responses and compute scores.
   This integration demonstrates an AI-based formative assessment that uses technology to give quick feedback, helping instructors track student progress and identify areas for improvement.

Pedagogical Significance

This activity serves as an innovative pedagogical tool aligned with the NEP 2020 framework, emphasizing active, technology-enabled, and outcome-based learning. Its key features include::

- **Active Learning:** Students engage in real-time code reasoning rather than passive memorization.
- **Immediate Feedback:** Automated results encourage reflection and correction after every attempt.
- **Formative Purpose:** The quiz serves as a non-graded assessment for learning.
- **Outcome Mapping:**
  - **CO1:** Formulating algorithms and understanding syntax.
  - **CO2:** Applying conditional branching and logical operators.
  - **CO6:** Testing and executing programs while correcting errors.
- **Digital and AI Integration:** The use of Apps Script and Python demonstrates practical AI applications in educational assessment, enhancing both faculty efficiency and student digital literacy.

Summary

This activity shows how traditional quizzes can be transformed into AI-based formative assessments that improve understanding, encourage self-learning, and support data-driven teaching.

By integrating **Google Forms, Apps Script automation, and AI-generated Python script based evaluation**, the activity aligns with **CO1, CO2, and CO6**, strengthening students' understanding of program logic, control structures, and debugging. It represents a practical example of a technology-supported pedagogy in engineering education.

PSP Quiz1

This quiz consists of five questions. You can take this quiz any number of times. Your highest score will be considered.

* Indicates required question

1. Email *

2. Roll Number *

   Enter the full roll no. eg: B24MEB23

3. Enter your full name *

   Full Name in BLOCK LETTERS as given in ETLAB

4. Question 1: Select all True Statements (1 mark each) * 7 pc

Tick all that apply.

- [ ] The 'printf("Hello, World!”);' function is used to display the output Hello, World! on the console.
- [ ] A variable in C cannot be declared without specifying its data type, so 'x = 5 is invalid.
- [ ] An identifier in C can start with a digit, such as 1variable.
- [ ] Identifiers in C can only consist of letters (A-Z, a-z), digits (0-9), and underscores (_).
- [ ] The code 'float a = 10.525; printf("Value of a: %.2f", a);' will display the valu of a as 10.525.
- [ ] The statement 'int x = 5, y = 10; printf("%d + %d = %d", x, y);' will output 5 + 1 = 15.
- [ ] In 'printf("%d", 'A');', the %d format specifier will display its ASCII value 65.

5. Question 2: Select all TRUE statements (1 mark each) * 6 pc

Tick all that apply.

- [ ] The code 'float a; a=5/2; printf("Result: %f", a);' will output Result: 2.5.
- [ ] The code 'int a = 5; printf("%d", a++);' will print 5.
- [ ] The result of the expression 'int a = 10, b = 3; printf("%d", a % b);' will be 1.
- [ ] The statement 'int a = 10, b = 20; int max = (a > b) ? a : b;' will assign 10 to max.
- [ ] The statement 'int a = 1, b = 0; printf("%d", a || b);' will print 1.
- [ ] The equality operator (==) has the same precedence as the assignment operator (=).

6. Question 3: Select the order of operations considering operator precedence and associativity. * 5 pc

Expression:
int result = a + sqrt(b * c) - d / e;

For each row (step number), choose the operator executed at that step.

Mark only one oval per row.

	+	sqrt( )	*	-	/
1	◯	◯	◯	◯	◯
2	◯	◯	◯	◯	◯
3	◯	◯	◯	◯	◯
4	◯	◯	◯	◯	◯
5	◯	◯	◯	◯	◯

7. Question 4: Select the TRUE statements about the following code: * 6 pc

```
int x;
printf("Enter a number: ");
scanf("%d", &x);
if (x > 5)
   printf("Greater than 5");
else
   printf("Smaller than or equal to 5”);
   printf("Always printed");
```

Tick all that apply.

- [ ] If x = 5, the output will be: Smaller than or equal to 5Always printed.
- [ ] If x = 10, the output will be: Greater than 5.
- [ ] The code will give a compilation error because the indentation is incorrect
- [ ] The code will produce a compilation error because the else block does not have braces.
- [ ] If x = 7, the output will be: Greater than 5Always printed.
- [ ] The statement 'printf("Always printed”);' is always executed regardless of t value of x

8. Question 5: The following code is an INCORRECT implementation of the leap year programme. Select the TRUE statements about the following code: * 6 pc

```
if (year % 400 == 0)
    printf("%d is a leap year.\n", year);
else if (year % 4 == 0 ll year % 100 != 0)
    printf("%d is a leap year.\n", year);
else
    printf("%d is not a leap year.\n", year);
```

Tick all that apply.

- [ ] If year = 2000, the output will be 2000 is a leap year.
- [ ] If year = 2023, the output will be 2023 is not a leap year.
- [ ] If year = 2024, the output will be 2024 is a leap year.
- [ ] The program produces incorrect results because of the use of the || operat instead of the correct && operator in the else if condition.
- [ ] If year = 1900, the output will be 1900 is not a leap year.
- [ ] The program correctly implements the leap year rules for SOME years.

Generating Google Forms Quiz

Use the following prompt:

Generate a Google Apps Script that creates a Google Form quiz from the questions in the attached pdf. Set the form title and a short description. Add two text fields at the beginning: one for Roll Number and one for Student Name, both required. Add each question from the PDF as a quiz item, preserving question type (checkbox, multiple choice, or grid). For each question, include all options and mark the correct answers as true. Assign point values per question (based on the number of correct answers). Include code at the end to log both the Form URL and the Edit URL .

```
function createCQuizForm() {
  // Create a new form and set it to quiz mode.
  var form = FormApp.create('C Programming Quiz');
  form.setIsQuiz(true);
  form.setDescription('This quiz consists of five questions. Note: For checkbox questions, grading is by default "all-or-nothing."');

  // ------------------------------------------------
  // Add user information fields at the beginning.
  // ------------------------------------------------
  // Field 1: Roll Number with help text.
  var rollNumberItem = form.addTextItem();
  rollNumberItem.setTitle("Roll Number")
                .setHelpText("Enter the full roll no. eg: B24MEB23")
                .setRequired(true);

  // Field 2: Full Name.
  var fullNameItem = form.addTextItem();
  fullNameItem.setTitle("Enter your full name")
              .setRequired(true);

  // ------------------------------------------------
  // Question 1: Select all True Statements (1 mark each)
  // ------------------------------------------------
  var q1 = form.addCheckboxItem();
  q1.setTitle("Question 1: Select all True Statements (1 mark each)")
    .setChoices([
      q1.createChoice('The printf("Hello, World!"); function is used to display output on the console.', true),
      q1.createChoice('In printf("%d", \'A\');, the %d format specifier will display its ASCII value 65.', true),
      q1.createChoice('A variable in C cannot be declared without specifying its data type, so x = 5; is invalid.', true),
      q1.createChoice('Identifiers in C can only consist of letters (A-Z, a-z), digits (0-9), and underscores (_).', true),
      q1.createChoice('An identifier in C can start with a digit, such as 1variable.', false),
      q1.createChoice('The code int a = 10; printf("Value of a: %f", a); will correctly display the value of a as 10.000000.', false),
      q1.createChoice('The statement int a; scanf("%d", &a); printf("You entered: %d", a); will output You entered: 25 if the user inputs 25.', true),
      q1.createChoice('The statement int x = 5, y = 10; printf("%d + %d = %d", x, y); will output 5 + 10 = 15.', false)
    ])
    .setPoints(5)  // Total of 5 marks (1 mark per correct statement)
    .setRequired(true);

  // ------------------------------------------------
  // Question 2: Select all TRUE statements (1 mark each)
  // ------------------------------------------------
  var q2 = form.addCheckboxItem();
  q2.setTitle("Question 2: Select all TRUE statements (1 mark each)")
    .setChoices([
      q2.createChoice('The code float a; a=5/2; printf("Result: %f", a); will
```

```
output Result: 2.5.', false),
      q2.createChoice('The code int a = 5; printf("%d", a++); will print 6.', false),
      q2.createChoice('The result of the expression int a = 10, b = 3; printf("%d", a % b); will be 1.', true),
      q2.createChoice('The statement int x = 7; float y = x / 2; printf("%.1f", y); will print 3.5.', false),
      q2.createChoice('The statement int a = 1, b = 0; printf("%d", a && b); will print 1.', false),
      q2.createChoice('The statement int a = 10, b = 20; int max = (a > b) ? a : b; will assign 10 to max.', false),
      q2.createChoice('In int x = 8, y = 10; if (x > y) printf("y is greater "); else printf("x is greater");, the output will be y is greater because x < y.', false),
      q2.createChoice('The equality operator (==) has the same precedence as the assignment operator (=).', false)
    ])
    .setPoints(1)  // Only 1 mark for the single correct statement
    .setRequired(true);

  // ----------------------------------------------------
  // Question 3: Grid Question — Order of Operations (Matrix of Radio Buttons)
  // ----------------------------------------------------
  // Expression: int result = a + sqrt(b * c) - d / e;
  // Expected order (per step):
  //   Row 1: "*"    (first operation)
  //   Row 2: "sqrt()" (second operation)
  //   Row 3: "/"    (third operation)
  //   Row 4: "+"    (fourth operation)
  //   Row 5: "-"    (fifth operation)
  var q3 = form.addGridItem();
  q3.setTitle("Question 3: Select the order of operations considering operator precedence and associativity.\n\n" +
              "Expression:\nint result = a + sqrt(b * c) - d / e;\n\n" +
              "For each row (step number), choose the operator executed at that step.");
  // Define rows (steps 1 to 5)
  q3.setRows(["1", "2", "3", "4", "5"]);
  // Define columns (operator choices)
  q3.setColumns(["+", "sqrt()", "*", "-", "/"]);
  q3.setRequired(true);
  // NOTE: Due to API limitations, points for grid questions must be set manually in the Form Editor.

  // ----------------------------------------------------
  // Question 4: Select the TRUE statements about the following code (1 mark each)
  // ----------------------------------------------------
  var q4 = form.addCheckboxItem();
  q4.setTitle(`Question 4: Select the TRUE statements about the following code:

int x;
```

```
printf("Enter a number: ");
scanf("%d", &x);
if (x > 5)
    printf("Greater than 5");
    printf("Always printed");
else
    printf("Smaller than or equal to 5");`)
    .setChoices([
      q4.createChoice('If x = 5, the output will be Smaller than or equal to 5.', false),
      q4.createChoice('If x = 10, the output will be Greater than 5Always printed', true),
      q4.createChoice('If x = 3, the output will be Always printedSmaller than or equal to 5', true),
      q4.createChoice('The code will produce a compilation error because the else block does not have braces.', false),
      q4.createChoice('If x = 7, the output will be Always printed.', false),
      q4.createChoice('The statement printf("Always printed"); is always executed regardless of the value of x', true)
    ])
    .setPoints(3)  // Total of 3 marks (1 mark per correct statement)
    .setRequired(true);

  // ------------------------------------------------
  // Question 5: Select the TRUE statements about the following code (1 mark each)
  // ------------------------------------------------
  // Code:
  // if (year % 400 == 0)
  //     printf("%d is a leap year.\n", year);
  // else if (year % 4 == 0 || year % 100 != 0)
  //     printf("%d is a leap year.\n", year);
  // else
  //     printf("%d is not a leap year.\n", year);
  //
  // Statements:
  // 1. If year = 2000, the output will be 2000 is a leap year. Answer: True
  // 2. If year = 2023, the output will be 2023 is not a leap year. Answer: False
  // 3. If year = 2024, the output will be 2024 is a leap year. Answer: True
  // 4. If year = 1900, the output will be 1900 is not a leap year. Answer: False
  // 5. The program produces incorrect results because of the use of the || operator instead of the correct && operator in the else if condition. Answer: True
  // 6. If year = 1900, the output will be 1900 is a leap year. Answer: True
  // 7. The program correctly implements the leap year rules for all years. Answer: False
  // 8. If year = 2023, the output will be 2023 is a leap year. Answer: True
  var q5 = form.addCheckboxItem();
```

```
  q5.setTitle(`Question 5: Select the TRUE statements about the following code:

if (year % 400 == 0)
    printf("%d is a leap year.\\n", year);
else if (year % 4 == 0 || year % 100 != 0)
    printf("%d is a leap year.\\n", year);
else
    printf("%d is not a leap year.\\n", year);`)
    .setChoices([
      q5.createChoice('If year = 2000, the output will be 2000 is a leap year.', true),
      q5.createChoice('If year = 2023, the output will be 2023 is not a leap year.', false),
      q5.createChoice('If year = 2024, the output will be 2024 is a leap year.', true),
      q5.createChoice('If year = 1900, the output will be 1900 is not a leap year.', false),
      q5.createChoice('The program produces incorrect results because of the use of the || operator instead of the correct && operator in the else if condition.', true),
      q5.createChoice('If year = 1900, the output will be 1900 is a leap year.', true),
      q5.createChoice('The program correctly implements the leap year rules for all years.', false),
      q5.createChoice('If year = 2023, the output will be 2023 is a leap year.', true)
    ])
    .setPoints(5)  // There are 5 correct statements (#1, #3, #5, #6, and #8)
    .setRequired(true);

  // Log the published and edit URLs so you can access your quiz form.
  Logger.log("Form URL: " + form.getPublishedUrl());
  Logger.log("Edit URL: " + form.getEditUrl());
}

```

In [54]:
```
import pandas as pd

# Import the CSV file
file_path = 'PSP Quiz1 6.csv'
df = pd.read_csv(file_path)
```

In [55]:
```
import pandas as pd

# Create an empty DataFrame with specified columns
columns = ['Username','Roll Number','Name', 'Total score', 'Responses']
empty_df = pd.DataFrame(columns=columns)

# Display the empty DataFrame
print(empty_df)
```

```
Empty DataFrame
Columns: [Username, Roll Number, Name, Total score, Responses]
Index: []
```

In [56]:
```
# Convert the column 'Total score' to string for the data frame df
df['Total score'] = df['Total score'].astype(str)
# In the dataframe df, in the column 'Total score' remove / 30 and conver
df['Total score'] = df['Total score'].str.replace('/ 30', '').astype(floa
```

In [57]:
```
grades = empty_df.copy()

# The column 'Username' is filled with the unique values from the column
grades['Username'] = df['Username'].unique()

# The column 'Roll Number' is filled with corresponding values in the col
grades['Roll Number'] = df.drop_duplicates('Username').set_index('Usernam

# The column 'Name' is filled with corresponding values in the column 'Us
grades['Name'] = df.drop_duplicates('Username').set_index('Username')['Na

# The column 'Total score' is filled with the maximum of 'Total score' in
grades['Total score'] = df.groupby('Username')['Total score'].max().reind

# The column 'Responses' is filled with the number of responses for a uni
grades['Responses'] = df['Username'].value_counts().reindex(grades['Usern
```

In [67]:
```
import numpy as np

# Ensure 'Total score' column is of numeric type
grades['Total score'] = pd.to_numeric(grades['Total score'], errors='coer

# Add a column Marks to the dataframe grades
# The column 'Marks' is filled with the values based on the 'Total score'
# The column 'Marks' should be out of 10, whereas column 'Total score' is
# Roundup the values in the column 'Marks' to the nearest integer
grades['Total score'] = pd.to_numeric(grades['Total score'], errors='coer
grades['Marks'] = np.ceil(grades['Total score'] / 3)

#grades['Marks'] = np.rint(grades['Total score'] / 3)
```

In [58]:
```
#In the data frame grades, check in the column 'Roll Number' if there is
#For example if the entry is 67 then it should be 'B24MEB67'
#The change should be there only if the entry doesn't have 'B24MEB' in it
grades['Roll Number'] = grades['Roll Number'].apply(lambda x: 'B24MEB' +
```

In [59]:
```
#In the data frame grades, check in the column 'Roll Number' starting fro
grades['Roll Number'] = grades['Roll Number'].str.strip()  # Remove leadi

for i in range(1, 73):
    roll_number = 'B24MEB' + str(i).zfill(2)
    if roll_number not in grades['Roll Number'].values:
        print(roll_number)
```

```
B24MEB02
B24MEB18
B24MEB45
B24MEB53
B24MEB59
```

In [60]:
```
#In the data frame grades, check in the column 'Roll Number' starting fro
# Also create a new column 'Attendance' and fill it with 'Absent' for all
# Ensure 'Roll Number' column has no leading/trailing spaces
grades['Roll Number'] = grades['Roll Number'].str.strip()

# Create a list to hold missing entries
missing_entries = []

for i in range(1, 73):
    roll_number = 'B24MEB' + str(i).zfill(2)
    if roll_number not in grades['Roll Number'].values:
        missing_entries.append({'Username': None, 'Roll Number': roll_num

# Concatenate the missing entries to the original dataframe
if missing_entries:
    missing_df = pd.DataFrame(missing_entries)
    grades = pd.concat([grades, missing_df], ignore_index=True)

#print(grades)
```

In [ ]:
```
```

In [68]:
```
#Sort the data frame grades based on the column 'Roll Number' in ascendin
grades = grades.sort_values('Roll Number')

#Save the data frame to a new CSV file
grades.to_csv('PSP_grades.csv', index=False)
```

In [72]:
```
# Sample DataFrame
data = {
    'Question1': ["response1;response2;'code;snippet';response3"],
    'Question2': ["response1;response2;response3"],
    'Question3.1': ["response1"],
    'Question3.2': ["response2"],
    'Question3.3': ["response3"],
    'Question3.4': ["response4"],
    'Question3.5': ["response5"],
    'Question4': ["response1;response2;response3"],
    'Question5': ["response1;response2;'code;snippet';response3"]
}

# Extract the first row
first_row = df.iloc[0]

# Custom function to split responses by ; but ignore ; within single quot
def split_responses(response):
    result = []
    current = []
    in_quotes = False
    for char in response:
        if char == "'":
            in_quotes = not in_quotes
        elif char == ";" and not in_quotes:
            result.append(''.join(current).strip())
            current = []
        else:
            current.append(char)
    result.append(''.join(current).strip())
    return result

# Initialize the dictionary
response_dict = {}

# Iterate through the columns from Question1 to Question5
for column in first_row.index:
    if column.startswith('Question1') or column.startswith('Question2') o
        # Split the responses for these questions
        response_dict[column] = split_responses(first_row[column])
    elif column.startswith('Question3'):
        # Store the values as a list of a single element for Question3.1
        response_dict[column] = [first_row[column]]

# Display the dictionary
print(response_dict)
```

```
{'Question1': ['The printf("Hello, World!”); function is used to display
the output Hello, World! on the console.', 'A variable in C cannot be dec
lared without specifying its data type, so x = 5; is invalid.', 'Identifi
ers in C can only consist of letters (A-Z, a-z), digits (0-9), and unders
cores (_).', 'In printf("%d", A’)', ', the %d format specifier will displ
ay its ASCII value 65.'], 'Question2': ['The code int a = 5; printf("%d",
a++); will print 5.', 'The result of the expression int a = 10, b = 3; pr
intf("%d", a % b); will be 1.', 'The statement int a = 1, b = 0; print
f("%d", a || b); will print 1.'], 'Question3.1': ['*'], 'Question3.2': ['
sqrt( )'], 'Question3.3': ['/'], 'Question3.4': ['+'], 'Question3.5': ['-
'], 'Question4': ['If x = 5, the output will be: Smaller than or equal to
5Always printed.', 'If x = 7, the output will be: Greater than 5Always pr
inted.', 'The statement printf("Always printed”); is always executed rega
rdless of the value of x'], 'Question5': ['If year = 2000, the output wil
l be 2000 is a leap year.', 'If year = 2024, the output will be 2024 is a
leap year.', 'The program produces incorrect results because of the use o
f the || operator instead of the correct && operator in the else if condi
tion.', 'The program correctly implements the leap year rules for SOME ye
ars.']}
```

In [94]:
```
#Print every element in the value of the key 'Question1' in the dictionar
for response in response_dict['Question1']:
    print(response)
```

```
The printf("Hello, World!”); function is used to display the output Hell
o, World! on the console.
A variable in C cannot be declared without specifying its data type, so x
= 5; is invalid.
Identifiers in C can only consist of letters (A-Z, a-z), digits (0-9), an
d underscores (_).
In printf("%d", A’)
, the %d format specifier will display its ASCII value 65.
```

In [93]:
```
import pandas as pd

# Sample DataFrame
data = {
    'Question1': ["response1;response2;'code;snippet;with;quotes';respons
    'Question2': ["response1;response2;response3"],
    'Question3.1': ["response1"],
    'Question3.2': ["response2"],
    'Question3.3': ["response3"],
    'Question3.4': ["response4"],
    'Question3.5': ["response5"],
    'Question4': ["response1;response2;response3"],
    'Question5': ["response1;response2;'code;snippet;with;quotes';respons
}



# Extract the first row
first_row = df.iloc[0]

# Custom function to split responses by ; but ignore ; within single quot
def split_responses(response):
    result = []
    current = []
```

```
    in_quotes = False
    quote_level = 0
    i = 0
    while i < len(response):
        char = response[i]
        if char == "'":
            if in_quotes:
                # Check if the next character is also a single quote (esc
                if i + 1 < len(response) and response[i + 1] == "'":
                    current.append(char)
                    i += 1  # Skip the next single quote
                else:
                    quote_level -= 1
                    if quote_level == 0:
                        in_quotes = False
            else:
                in_quotes = True
                quote_level += 1
        elif char == ";" and not in_quotes:
            result.append(''.join(current).strip())
            current = []
        else:
            current.append(char)
        i += 1
    result.append(''.join(current).strip())
    return result

# Initialize the dictionary
response_dict = {}

# Iterate through the columns from Question1 to Question5
for column in first_row.index:
    if column.startswith('Question1') or column.startswith('Question2') o
        # Split the responses for these questions
        response_dict[column] = split_responses(first_row[column])
    elif column.startswith('Question3'):
        # Store the values as a list of a single element for Question3.1
        response_dict[column] = [first_row[column]]

# Display the dictionary
print(response_dict)
```

Response Sample

PSP Quiz1

This quiz consists of five questions. You can take this quiz any number of times. Your highest score will be considered.

The respondent's email (**shamilnaseeh@gmail.com**) was recorded on submission of this form.

Roll Number *

Enter the full roll no. eg: B24MEB23

B24MEB23

Enter your full name *

Full Name in BLOCK LETTERS as given in ETLAB

NASEEH SHAMIL T K

Question 1: Select all True Statements (1 mark each) * 7 points

- [x] The 'printf("Hello, World!”);' function is used to display the output Hello, World! on the console.
- [x] A variable in C cannot be declared without specifying its data type, so 'x = 5;' is invalid.
- [ ] An identifier in C can start with a digit, such as 1variable.
- [x] Identifiers in C can only consist of letters (A-Z, a-z), digits (0-9), and underscores (_).
- [ ] The code 'float a = 10.525; printf("Value of a: %.2f", a);' will display the value of a as 10.525.
- [ ] The statement 'int x = 5, y = 10; printf("%d + %d = %d", x, y);' will output 5 + 10 = 15.
- [x] In 'printf("%d", 'A');', the %d format specifier will display its ASCII value 65.

Question 2: Select all TRUE statements (1 mark each) * 6 points

- [ ] The code 'float a; a=5/2; printf("Result: %f", a);' will output Result: 2.5.
- [x] The code 'int a = 5; printf("%d", a++);' will print 5.
- [x] The result of the expression 'int a = 10, b = 3; printf("%d", a % b);' will be 1.
- [ ] The statement 'int a = 10, b = 20; int max = (a > b) ? a : b;' will assign 10 to max.
- [x] The statement 'int a = 1, b = 0; printf("%d", a || b);' will print 1.
- [ ] The equality operator (==) has the same precedence as the assignment operator (=).

Question 3: Select the order of operations considering operator precedence and associativity. * 5 points

Expression:
int result = a + sqrt(b * c) - d / e;

For each row (step number), choose the operator executed at that step.

	+	sqrt( )	*	-	/
1	○	○	◉	○	○
2	○	◉	○	○	○
3	○	○	○	○	◉
4	◉	○	○	○	○
5	○	○	○	◉	○

Question 4: Select the TRUE statements about the following code: * 6 points

```
int x;
printf("Enter a number: ");
scanf("%d", &x);
if (x > 5)
    printf("Greater than 5");
else
    printf("Smaller than or equal to 5”);
    printf("Always printed");
```

- [x] If x = 5, the output will be: Smaller than or equal to 5Always printed.
- [ ] If x = 10, the output will be: Greater than 5.
- [ ] The code will give a compilation error because the indentation is incorrect
- [ ] The code will produce a compilation error because the else block does not have braces.
- [x] If x = 7, the output will be: Greater than 5Always printed.
- [x] The statement 'printf("Always printed”);' is always executed regardless of the value of x

Question 5: The following code is an INCORRECT implementation of the leap year programme. Select the TRUE statements about the following code: * 6 points

```
if (year % 400 == 0)
    printf("%d is a leap year.\n", year);
else if (year % 4 == 0 || year % 100 != 0)
    printf("%d is a leap year.\n", year);
else
    printf("%d is not a leap year.\n", year);
```

- [x] If year = 2000, the output will be 2000 is a leap year.
- [ ] If year = 2023, the output will be 2023 is not a leap year.
- [x] If year = 2024, the output will be 2024 is a leap year.
- [x] The program produces incorrect results because of the use of the || operator instead of the correct && operator in the else if condition.
- [ ] If year = 1900, the output will be 1900 is not a leap year.
- [x] The program correctly implements the leap year rules for SOME years.

Evaluation Sheet

Username	Roll Number	Name	Total score	Responses
adharshak720@gmail.com	B24MEB01	A K AKASH	30	2
abdulrasheedph11@gmail.com	B24MEB03	ABDUL RASHEED P H	18	1
abdullahshesin21@gmail.com	B24MEB04	ABDULLAH SHESIN K K	16	1
abhijithspabhijithsp@gmail.com	B24MEB05	ABHIJITH. S. P	30	2
abhinandvk005@gmail.com	B24MEB06	ABHINAND V K	30	7
abhinavsatheesh321@gmail.com	B24MEB07	Abhinav Satheesh	24	2
abhirammuchu@gmail.com	B24MEB08	ABHIRAM P T	1	1
abinbinny2005@gmail.com	B24MEB09	Abin Binny	30	2
abindas1016@gmail.com	B24MEB10	ABIN DAS	30	2
adhilmohammedj@gmail.com	B24MEB11	ADHIL MOHAMMED J	24	2
adithdev97@gmail.com	B24MEB12	ADITH DEV K P	30	2
adithyan10pb@gmail.com	B24MEB13	ADITHYAN PB	10	1
adithyanj284@gmail.com	B24MEB14	ADITHYAN J	15	1
adwaithof06@gmail.com	B24MEB15	ADWAITH A	30	2
afnannizam2@gmail.com	B24MEB16	AFNAN N	24	1
afsalasif07@gmail.com	B24MEB17	AFSAL ASIF	30	2
akashvpradeep@gmail.com	B24MEB19	AKASH V PRADEEP	30	1
akshaydasm0@gmail.com	B24MEB20	Akshay Das M	30	3
alankuriakose6526@gmail.com	B24MEB21	ALAN KURIAKOSE	30	3
alexmvash@gmail.com	B24MEB22	ALEX M VARGHESE	30	5
shamilnaseeh@gmail.com	B24MEB23	NASEEH SHAMIL T K	17	2
amalunissar@gmail.com	B24MEB23	AMAL NISSAR	30	2
ajaydevp10@gmail.com	B24MEB23	AJAY DEV P	12	1
anandhu2k05@gmail.com	B24MEB24	ANANDHU KRISHNAN. A	30	2
arjunkannan3245@gmail.com	B24MEB25	ARJUN S GIREESH	18	1
arunkandamkulam303@gmail.com	B24MEB26	ARUN K	30	2
athulrajmv79@gmail.com	B24MEB27	ATHULRAJ M V	6	1
zhimaster666@gmail.com	B24MEB28	AVINASH B	30	2
ayashpsaleem2020@gmail.com	B24MEB29	AYYASH PBSALEEM	29	2
bimalsureshpvonakkunnu@gmail.com	B24MEB30	BIMAL SURESH P V	10	1
devanand82006@gmail.com	B24MEB31	DEV ANAND A L	30	1
devaprabhahari@gmail.com	B24MEB32	DEVAPRABHA D	30	5
dheerajtr09082005@gmail.com	B24MEB33	DHEERAJ TR	30	2
diyaravi5476@gmail.com	B24MEB34	DIYA RAVI	0	1
faihathayattil@gmail.com	B24MEB35	Faiha	1	1
rejafathima100@gmail.com	B24MEB36	FATHIMA REJA	30	1
fidhasadikfaiz@gmail.com	B24MEB37	FIDHA SADIK	30	2
kgoutham911@gmail.com	B24MEB38	GOUTHAM KRISHNA AR	24	1
jasonmanu2006@gmail.com	B24MEB39	JASON JOSEPH MANU	3	1
jithusjohn280@gmail.com	B24MEB40	JITHU S JOHN	30	3
karthiktroy349@gmail.com	B24MEB41	KARTHIK TROY	28	2
kiransurendranmk@gmail.com	B24MEB42	KIRAN SURENDRAN	24	3
meenakshirs6002@gmail.com	B24MEB43	MEENAKSHI S	1	1
mumthazmizhab123@gmail.com	B24MEB44	MISHAB MOHAMMED	1	1
mhmdnabeel1966@gmail.com	B24MEB46	MOHAMMED NABEEL A.V	30	2
mhdrayan416@gmail.com	B24MEB47	MUHAMAD RAYAN	30	2
muhammedridhwandobre43@gmail.com	B24MEB48	MUHAMMED RIDHWAN	1	1
rinasrazab@gmail.com	B24MEB49	MUHAMMAD RINAS	30	2
rushilkoodathil1551@gmail.com	B24MEB50	MUHAMMED RUSHIL ZUBAIR	17	1
8shd987@gmail.com	B24MEB51	MUHAMMED SHAHID	17	1
nandanakannan22@gmail.com	B24MEB52	NANDANA KANNAN	1	2

naveensbabu76@gmail.com	B24MEB54	NAVEEN S BABU	30	2
navneetall41@gmail.com	B24MEB55	NAVNEET B	1	1
nivedsnair06@gmail.com	B24MEB56	NIVED S NAIR	30	3
nivedithasudheersudheer@gmail.com	B24MEB57	NIVEDITHA OS	18	1
gayathripsjpsp10d@gmail.com	B24MEB58	P GAYATHRI	16	1
regulkrishna@gmail.com	B24MEB58	REGUL KRISHNA	30	2
sarathjose2005@gmail.com	B24MEB60	SARATH JOSE B	30	2
shahin.st007@gmail.com	B24MEB61	SHAHIN S	30	1
talksidks11@gmail.com	B24MEB62	SIDHARTH K S	30	2
sobinbiju131@gmail.com	B24MEB63	SOBIN BIJU	5	1
sreesanth3737@gmail.com	B24MEB64	SREESANTH S	17	2
sreyabinu09@gmail.com	B24MEB65	SREYA MARIAM BINU	30	2
suryajith034@gmail.com	B24MEB66	SURYAJITH SUNIL	30	2
vighneshbaiju@gmail.com	B24MEB67	VIGHNESH BAIJU RAJASREE	30	3
vighneshsb12@gmail.com	B24MEB68	VIGHNESH S B	5	1
viswajithvinodkumar@gmail.com	B24MEB69	viswajith km	23	1
vyshnavraj453393@gmail.com	B24MEB70	Vyshnav Raj	30	3
sonashinu16@gmail.com	B24MEB71	SONA SHINU	30	1
anzals690@gmail.com	B24MEB72	ANZAL NIZAM	28	1
muhammed.fahd@gmail.com	B24MEB90	TEST	30	1