Preliminary Lab Exam

Below are fourteen practical, menu-driven programming exercises. each problem includes:

- A brief description (how the menu should function)
- Any relevant mathematical formulas (where needed)
- Sample outputs

1. ATM System

Description:
Create a program that simulates basic ATM transactions:
1) Deposit money,
2) Withdraw money,
3) View account balance,
4) Exit the system.

Sample Output:

```
=== ATM Machine ===
1. Deposit
2. Withdraw
3. View Balance
4. Exit
Select an option (1-4): 1

Enter amount to deposit: 100
$100.0 deposited successfully.

=== ATM Machine ===
1. Deposit
2. Withdraw
3. View Balance
4. Exit
Select an option (1-4): 3
Your current balance is: $100.0

=== ATM Machine ===
1. Deposit
2. Withdraw
3. View Balance
4. Exit
Select an option (1-4): 4
Thank you for using the ATM. Goodbye!
```

2. Menu-Driven Calculator

Description:

Create a basic calculator with a menu of operations:
1) Addition,
2) Subtraction,
3) Multiplication,
4) Division,
5) Exit.

The user repeatedly chooses an option, then enters two numbers. The program performs the chosen operation and displays the result until the user chooses to exit.

Sample Output:

```
=== Simple Calculator ===
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Choose an operation (1-5): 1
Enter first number: 10
Enter second number: 5
Result: 15.0

=== Simple Calculator ===
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Choose an operation (1-5): 5
Exiting calculator. Goodbye!
```

3. Factorial & Summation Tool

Description:
Create a menu with two arithmetic features:
1) Compute factorial of a positive integer *n*.
2) Compute the summation from 1 to *n*.
3) Exit.

Formulas:

- Factorial: `n! = 1 × 2 × … × n`
- Summation (1 to n): `Sum = n(n + 1)/2`

Sample Output:

```
=== Factorial & Summation Menu ===
1. Compute Factorial
```

```
2. Compute Summation (1 to n)
3. Exit
Choose an option (1-3): 1
Enter a positive integer: 5
Factorial of 5 is 120.

Choose an option (1-3): 2
Enter a positive integer: 5
Sum of numbers from 1 to 5 is 15.

Choose an option (1-3): 3
Exiting program. Goodbye!
```

4. Temperature Converter

Description:

Create a menu-driven program to convert temperatures:

1) Celsius to Fahrenheit,

2) Fahrenheit to Celsius,

3) Exit.

Formulas:

- F = (C × 9/5) + 32
- C = (F - 32) × 5/9

Sample Output:

```
=== Temperature Converter ===
1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
3. Exit
Choose an option (1-3): 1
Enter temperature in Celsius: 25
25.0°C is 77.0°F

Choose an option (1-3): 3
Exiting temperature converter. Goodbye!
```

5. Prime Number Utility

Description:

Create a menu that:

1) Checks if a single number is prime,

2) Displays all prime numbers up to a limit,

3) Exits.

Sample Output:

```
=== Prime Number Utility ===
1. Check if a number is prime
2. Show all primes up to a limit
3. Exit
Choose an option (1-3): 1
Enter an integer (>1): 7
7 is prime.

Choose an option (1-3): 2
Enter the upper limit (>1): 10
Primes up to 10: [2, 3, 5, 7]

Choose an option (1-3): 3
Exiting Prime Number Utility. Goodbye!
```

6. Multiplication Table Generator

Description:

Create a menu that:

1) Prompts for an integer and prints its multiplication table (1 to 10),

2) Exits.

Sample Output:

```
=== Multiplication Table Menu ===
1. Generate a Multiplication Table
2. Exit
Choose an option (1-2): 1
Enter an integer: 3
Multiplication Table for 3:
3 x 1 = 3
3 x 2 = 6
...
3 x 10 = 30

Choose an option (1-2): 2
Exiting Multiplication Table Program. Goodbye!
```

7. Basic Grade Manager

Description:

Build a menu to manage grades:

1) Add a grade (0–100),

2) Show average of all grades,

3) List all grades,

4) Exit.

Sample Output:

```
=== Grade Manager ===
1. Add Grade
2. Show Average Grade
3. List All Grades
4. Exit
Choose an option (1-4): 1
Enter a grade (0-100): 85
Grade added.

Choose an option (1-4): 2
Average grade: 85.00

Choose an option (1-4): 4
Exiting Grade Manager. Goodbye!
```

8. Number Guessing Game

Description:
A menu with:
1) Start or restart the guessing game (with a fixed secret number),
2) Exit.

While playing, prompt user guesses until correct:
- If guess < secret, print "Too Low"
- If guess > secret, print "Too High"
- If guess == secret, print "Correct!"

Sample Output:

```
=== Number Guessing Game ===
1. Start/Restart Game
2. Exit
Choose an option (1-2): 1
Guess the secret number (or -1 to quit): 5
Too Low
Guess the secret number (or -1 to quit): 9
Too High
Guess the secret number (or -1 to quit): 7
Correct!

Choose an option (1-2): 2
Exiting the Number Guessing Game. Goodbye!
```

9. Grocery Bill Calculator

Description:

A menu to manage a grocery bill:

1) Add an item cost,

2) Remove the last item cost,

3) View total,

4) Exit.

Sample Output:

```
=== Grocery Bill Calculator ===
1. Add Item Cost
2. Remove Last Item
3. View Total
4. Exit
Choose an option (1-4): 1
Enter the item cost: 2.99
Added $2.99 to the bill.

Choose an option (1-4): 1
Enter the item cost: 4.5
Added $4.5 to the bill.

Choose an option (1-4): 3
Current total bill: $7.49

Choose an option (1-4): 4
Exiting Grocery Bill Calculator. Goodbye!
```

10. Collatz Conjecture Menu

Description:

Create a menu with:

1) Generate the Collatz sequence for a positive integer,

2) Exit.

Collatz rule:

- If `n` is even, next `n` = `n / 2`

- If `n` is odd, next `n` = `3n + 1`

Continue until `n` becomes 1.

Sample Output:

```
=== Collatz Conjecture Menu ===
1. Generate Collatz sequence
2. Exit
Choose an option (1-2): 1
Enter a positive integer: 6
Collatz sequence: 6 -> 3 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1
```

```
Choose an option (1-2): 2
Exiting Collatz Menu. Goodbye!
```

11. Trigonometric Calculator

Description:
Build a menu that helps compute basic trigonometric values:
1) Calculate sine of an angle,
2) Calculate cosine of an angle,
3) Calculate tangent of an angle,
4) Exit.

The user can choose whether the angle is provided in degrees or radians (or you may restrict to one system for simplicity). Then, display the corresponding trigonometric result.

Formulas (in radians):

- `sin(θ) = opposite / hypotenuse`
- `cos(θ) = adjacent / hypotenuse`
- `tan(θ) = sin(θ) / cos(θ)`

Note: If you accept degrees, convert to radians before applying math library functions: `radians = degrees × (π / 180)`.

Sample Output:

```
=== Trigonometric Calculator ===
1. Sine
2. Cosine
3. Tangent
4. Exit
Choose an option (1-4): 1
Enter angle value: 30
Is this value in (d)egrees or (r)adians? d
sin(30°) = 0.5

Choose an option (1-4): 2
Enter angle value: 0.5235987
Is this value in (d)egrees or (r)adians? r
cos(0.5235987 rad) = 0.8660254

Choose an option (1-4): 4
Exiting Trigonometric Calculator. Goodbye!
```

12. Loan Payment Calculator

Description:
Build a menu:

1) Compute monthly payment given principal, annual interest, and years,
2) Exit.

Formula (Amortization):
If *P* is the principal, *r* is the monthly interest rate, and *n* is the number of monthly payments (`years × 12`), then:

```
M = P × [ r (1 + r)^n ] / [ (1 + r)^n - 1 ]
```

If *r = 0*, then `M = P / n`.

Sample Output:

```
=== Loan Payment Calculator ===
1. Compute Monthly Payment
2. Exit
Choose an option (1-2): 1
Enter loan principal: 10000
Enter annual interest rate (in %): 5
Enter loan duration in years: 2
Estimated monthly payment: $438.71

Choose an option (1-2): 2
Exiting Loan Payment Calculator. Goodbye!
```

13. Distance-Speed-Time Calculator

Description:
Create a menu:
1) Compute time (given distance & speed),
2) Compute distance (given speed & time),
3) Compute speed (given distance & time),
4) Exit.

Formulas:

- `Time = Distance / Speed`
- `Distance = Speed × Time`
- `Speed = Distance / Time`

Sample Output:

```
=== Distance-Speed-Time Calculator ===
1. Compute Time (given Distance & Speed)
2. Compute Distance (given Speed & Time)
3. Compute Speed (given Distance & Time)
4. Exit
Choose an option (1-4): 1
Enter distance (e.g., km): 100
Enter speed (e.g., km/h): 50
Time required: 2.0 hours
```

```
Choose an option (1-4): 4
Exiting Distance-Speed-Time Calculator. Goodbye!
```

14. BMI Calculator

Description:
Design a menu-driven program to calculate the *Body Mass Index (BMI)* for users in either Metric or Imperial units. The menu options:
1) Calculate BMI (Metric),
2) Calculate BMI (Imperial),
3) Exit.

The program asks for the user's weight and height in the selected system, then displays the computed BMI. You may also provide brief guidance on BMI interpretation (e.g., "underweight," "normal," "overweight," etc.).

Formulas:

- *Metric:* $BMI = weight_{(kg)} / [height_{(m)}]^2$
- *Imperial:* $BMI = (703 \times weight_{(lbs)}) / [height_{(in)}]^2$

Sample Output:

```
=== BMI Calculator ===
1. Calculate BMI (Metric)
2. Calculate BMI (Imperial)
3. Exit
Choose an option (1-3): 1
Enter weight in kilograms: 70
Enter height in meters: 1.75
Your BMI is 22.86

Choose an option (1-3): 2
Enter weight in pounds: 154
Enter height in inches: 69
Your BMI is 22.73

Choose an option (1-3): 3
Exiting BMI Calculator. Goodbye!
```
