perplexity

read each pdf (name corresponds to the module number), read through these pdfs and compile a list of all programming questions module wise and make a table of it module wise. And after list of programs, include code for complex programs and question and answers for theory questions, Avoid repetition of questions, and if one follow up question seems redundant because we have discussed same topic before in an older question, ignore that.

Here's a structured breakdown of **programming questions from Module 1** (based on the provided PDF content). I've excluded redundant questions and focused on unique programming tasks:

Module 1: Programming Questions Table

Q.No	Program Description	Key Concepts Tested	Complexity Level
6	Calculate area and perimeter of a rectangle	Basic I/O, arithmetic operations	Beginner
8b	Accept student details (name, age, GPA) and display formatted output	Data types, formatted I/O (`printf`, `scanf`)	Beginner
19	Compute factorial of a number	Loops, data types (`unsigned long long`)	Intermediate
25b	Check if a number is even/odd using `if-else`	Conditional statements	Beginner
26a	Check voting eligibility based on age	Simple `if-else` logic	Beginner
27a	Check if a number is less than 10 and print "OK"	Basic `if` statement	Beginner
28	Check if a number is positive/negative	`if-else` with relational operators	Beginner
29	Find the smallest number among three numbers	Nested `if` statements	Beginner
32	Check if a number is an Armstrong number	Loops, arithmetic, math functions (`pow`)	Advanced

Q.No	Program Description	Key Concepts Tested	Complexity Level
33	Menu-driven program for arithmetic operations (+, -, *, /)	Switch-case, loops, error handling	Intermediate
37	Generate a numeric pattern (pyramid)	Nested loops	Intermediate
38	Find the sum of digits of a number	Loops, modulus operator	Intermediate
39	Demonstrate nested loops (e.g., print coordinate pairs)	Nested `for` loops	Beginner
41	Print even numbers between 1 and 20 using `for` loop	Loop control (`i += 2`)	Beginner
44	Check if a number is a palindrome	Loops, arithmetic operations	Intermediate
45	Calculate total marks, percentage, and assign grades	`if-else` ladder, floating-point arithmetic	Intermediate
46	Find the sum of the first and last digits of a number	Loops, modulus/division operators	Intermediate
47	Display multiplication table of a number	`for` loop	Beginner

Code for Complex Programs

1. Armstrong Number Check (Q32)

```
#include <stdio.h>
#include <math.h>

int main() {
    int num, originalNum, remainder, result = 0, n = 0;
    printf("Enter a number: ");
    scanf("%d", &num);
    originalNum = num;

    // Count digits
    while (originalNum != 0) {
        originalNum /= 10;
        n++;
    }

    originalNum = num;

    // Calculate sum of digits^n
    while (originalNum != 0) {
        remainder = originalNum % 10;
        result += pow(remainder, n);
        originalNum /= 10;
    }

    if (result == num)
```

```
        printf("%d is an Armstrong number.\n", num);
    else
        printf("%d is not an Armstrong number.\n", num);

    return 0;
}
```

2. Menu-Driven Calculator (Q33)

```
#include <stdio.h>

int main() {
    int choice;
    float num1, num2, result;

    while (1) {
        printf("\nMenu:\n1. Addition\n2. Subtraction\n3. Multiplication\n4. Division\n5.
        printf("Enter your choice: ");
        scanf("%d", &choice);

        if (choice == 5) break;

        printf("Enter two numbers: ");
        scanf("%f %f", &num1, &num2);

        switch (choice) {
            case 1: result = num1 + num2; break;
            case 2: result = num1 - num2; break;
            case 3: result = num1 * num2; break;
            case 4: result = (num2 != 0) ? num1 / num2 : 0; break;
            default: printf("Invalid choice!\n");
        }

        if (choice >= 1 && choice <= 4)
            printf("Result: %.2f\n", result);
    }

    return 0;
}
```

Theory Questions (Summarized Answers)

1. **Comments in C**: Use `//` for single-line and `/* ... */` for multi-line comments.
2. **Constants vs Variables**: Constants are fixed (`const float PI = 3.14`), variables can change.
3. **Data Types**: `int`, `float`, `double`, `char` with specific sizes and use cases.
4. **Operators**: Arithmetic, relational, logical, bitwise, and their precedence.
5. **Control Statements**: `if-else`, `switch`, loops (`for`, `while`, `do-while`).

Note: For Modules 2-4, provide PDFs or specific questions, and I'll generate similar structured breakdowns!

⁂