Programming in C and C++ - Practice Sheet 2

First Name: _____

Last Name: _____

Matriculation Number: _____

Instructions:
1. Read all questions before starting.
2. Write your name above immediately.
3. Write clearly. C/C++ are case-sensitive.
4. Indent your code properly.
5. No books, notes, or electronic devices allowed.
6. You may use the back of pages for extra space.
7. You have 2 hours to complete this test.
8. Cheating results in immediate failure.
9. By signing below, you confirm understanding of all rules.

Signature: _____

%	**Grade**	%	**Grade**	%	**Grade**	%	**Grade**
0.00 - 39.49	5.0	39.50 - 44.49	4.7	44.50 - 49.49	4.3	49.50 - 54.49	4.0
54.50 - 59.49	3.7	59.50 - 64.49	3.3	64.50 - 69.49	3.0	69.50 - 74.49	2.7
74.50 - 79.49	2.3	79.50 - 84.49	2.0	84.50 - 89.49	1.7	89.50 - 94.49	1.3
94.50 - 100.00				1.0			

Problem 1: String Operations (3 points)

Language: C
Write a function `int count_digits(const char *str)` that returns the number of digits (0-9) in a given string. Do not use any string.h functions.

Problem 2: 2D Array Manipulation (4 points)

Language: C
Write a complete C program that:

1. Reads integers n and m from standard input (dimensions)
2. Dynamically allocates an n×m integer matrix
3. Initializes all elements to 0
4. Sets the main diagonal elements to 1 (where i = j)
5. Prints the matrix in a readable format
6. Frees all allocated memory

Problem 3: Pointer Analysis (3 points)

Language: C
What is the exact output of this program?

```c
#include <stdio.h>
int main() {
    int x = 5;
    int *p = &x;
    int **q = &p;
    **q = 10;
    *p = *p + 5;
    printf("x=%d, *p=%d, **q=%d\n", x, *p, **q);
    return 0;
}
```

Problem 4: Recursive Function (5 points)

Language: C
Write a recursive function that calculates the sum of digits of a positive integer:

```c
int sum_digits(int n) {
    // Your code here
}
```

Example: `sum_digits(1234)` should return 10.

Problem 5: String Reversal (4 points)

Language: C
Implement `void reverse_string(char *str)` that reverses a string in-place without using string.h functions or additional arrays.

Problem 6: Linked List Implementation (6 points)

Language: C
Define a structure for student records:

```
struct Student {
    char name[50];
    float grade;
    struct Student *next;
};
```

Write functions to:

1. Add a student at the end of the list
2. Calculate the average grade
3. Display all students with grades above average

Problem 7: File Operations (3 points)

Language: C
Write a program that reads integers from "input.txt", filters out odd numbers, and writes only even numbers to "output.txt". Include proper error handling.

Problem 8: Bitwise Operations (2 points)

Language: C
Using bitwise operators, write:

```c
int is_power_of_two(unsigned int n) {
    // Return 1 if n is power of two, 0 otherwise
}
```

Problem 9: C++ Class Design (5 points)

Language: C++
Design a `BankAccount` class with:

- Private: accountNumber (int), balance (double), owner (string)
- Constructor with parameters
- Getter/setter for balance (no negative values)
- Methods: deposit(amount), withdraw(amount)
- Overloaded `<<` operator for output

Problem 10: Copy Constructor (4 points)

Language: C++
Implement a deep copy constructor for:

```cpp
class String {
    char *data;
    int length;
public:
    String(const char *str = "");
    String(const String &other); // Implement this
    ~String();
};
```

Explain why deep copy is needed.

Problem 11: Inheritance Hierarchy (6 points)

Language: C++
Create an abstract `Shape` class with pure virtual `area()` and `perimeter()`. Derive `Circle` and `Rectangle` classes implementing these methods. Use `M_PI` for .

Problem 12: Operator Overloading (3 points)

Language: C++
Overload the `+` operator for `Vector2D` class:

```
class Vector2D {
    float x, y;
public:
    Vector2D operator+(const Vector2D &other) const;
};
```

Problem 13: Exception Handling (2 points)

Language: C++
Write code demonstrating `try-catch` for array index out of bounds using `std::out_of_range`.