CS112 Section B-Object-Oriented Programming, Spring 2023, Midterm Examination
Faculty of Computer Science and Engineering, GIK Institute
Instructor: Prof. Dr. Zahid Halim and Mr. Said Nabi
Friday, 24th March 2023 Time: 60 Minutes Total Marks: 40

Vetted by (Name and Signatures):

You are advised to READ these notes:

- Maximum points for each question are written in the brackets [ ]
- This exam. is closed books, closed notes. Please see that the area in your threshold is clean. You will be charged for any material which can be classified as 'helping in the paper' found near you.
- The invigilator present is not supposed to answer any questions. If there is any missing parameter, write your assumption clearly and move forward.

Q1 *[Points: 12, difficulty level: Medium, CLO-2]*: Write a C++ program that takes a string input from the user and converts all lowercase characters to uppercase characters using pointers.

The program should prompt the user to enter a string of up to 100 characters. The program should then use a pointer to iterate through the string and convert any lowercase letters to uppercase letters. The program should output the resulting string to the console.

Here's an example of the program's input and output:

Input:

Enter a string: Hello, world!

Output:

The uppercase version of the string is: HELLO, WORLD!

Q2 *[Points: 8, difficulty level: Low, CLO-3]*: Implement a program that simulates a bank account management system using object-oriented programming principles in C++.

The program should have the following functionality:

- Allow the user to create a new bank account with an initial balance.
- Allow the user to deposit money into an existing account.
- Allow the user to withdraw money from an existing account.
- Allow the user to transfer money from one account to another.
- Allow the user to close an existing account.
- Display the current balance of an existing account.

The program should maintain the following data structures:

A BankAccount class that encapsulates the account information, including the account number, the account holder's name, and the account balance.

A collection of BankAccount objects, implemented as a dynamic array or vector.

The program should use object-oriented programming principles to implement the functionality, including the use of member functions, constructors, and private member variables. The program should also handle errors, such as attempting to withdraw more money than is available in an account or attempting to transfer money to a non-existent account.

Q3 *[Points: 15, difficulty level: High, CLO-2]*: Create a class called "Matrix" that contains a private two-dimensional array of integers called "data", as well as private integer variables called "rows" and "columns" that represent the dimensions of the matrix.

The class should have the following public member functions:

- A constructor that takes the number of rows and columns as arguments, and initializes the "data" array to all zeros.
- A destructor that deallocates the "data" array.
- A copy constructor that creates a new Matrix object with the same dimensions and data as an existing Matrix object.
- An overloaded assignment operator that assigns the data and dimensions of an existing Matrix object to another Matrix object.
- A member function called "set" that takes a row index, column index, and integer value as arguments, and sets the corresponding element in the "data" array to the given value.
- A member function called "get" that takes a row index and column index as arguments, and returns the value of the corresponding element in the "data" array.
- A static member function called "identity" that takes an integer argument "n", and returns a new Matrix object that represents the identity matrix with dimensions "n x n". The identity matrix is a square matrix with ones on the diagonal and zeros elsewhere.

The "identity" member function should be implemented using the following algorithm:

- Create a new Matrix object with dimensions "n x n".
- Set all elements in the "data" array to zero.
- Loop over the diagonal elements of the "data" array (i.e., elements with row index equal to column index), and set them to 1.

Q4 *[Points: 5, difficulty level: low, CLO-1]*: Write a recursive function to find factorial of a number. DO NOT write main.

Happy coding!