National University of Computer and Emerging Sciences, Lahore Campus

Course:	**OOP Lab**	**Course Code:**	**CL1004**
Program:	**BS(CS, DS)**	**Semester:**	**Spring 2023**
Duration:	**2 Hour 45 Minutes**	**Total Marks:**	**100**
Paper Date:	**20th May, 2023**	**Weight:**	**40%**
		Page(s):	**5**
Exam:	**Lab Final**	**Reg. No.**	

Read below Instructions Carefully:

- Understanding the question statement is also part of the exam, so do not ask for any clarification. In case of any ambiguity, make suitable assumptions.
- You have to complete exam in specified hrs. No extra time will be given for submission.
- Submit a single **.cpp file** for each question named as **21L-1122 (Q#)**
- Place all .cpp file into the **folder** named as **21L-1122**
- Submit folder on **cactus** by following path: \\cactus1\ Xeon\ Spring 2023\Final OOP Slot-2\Your Section
- Your code should be **intended** and **commented** properly. Use **meaningful variable names**.
- It is your responsibility to save your code from being copied. All matching codes will be considered cheating cases. **PLAGIARISM** will result in forwarding of **case to Disciplinary Committee**.

QUESTION 1 (60 Points)

Consider a Virtual Zoo Management System. In this system, there are several entities, such as Animals, Zookeepers, and Enclosures. The Zoo has different types of Animals, like Birds, Mammals, and Reptiles, and Zookeepers specialized in taking care of each type. Each Enclosure is dedicated to a specific type of animal and is maintained by a designated Zookeeper. Each Animal can perform actions like eating, sleeping, and making sounds, and each Zookeeper can feed animals, clean enclosures, and monitor animal health. Each Enclosure hosts a group of Animals and needs regular maintenance.

Create a base class **`Animal`** with the following data members and functions:
Data Members:

1. `name`: a string representing the name of the animal
2. `age`: an integer representing the age of the animal
3. `gender`: a string representing the gender of the animal

Functions:

- `Animal(animalType, name, age, gender)`: a constructor method that initializes the data members of the Animal class
- `string getName()`: a method that returns the name of the Animal object
- `int getAge()`: a method that returns the age of the Animal object
- `string getGender()`: a method that returns the gender of the Animal object
- `void makeSound()`: A pure virtual function

- `void display()`: A virtual function that display the name, age, gender and animal type

Create three classes `**Bird**` , '**Mammel'** and '**Reptile'** which are inherited from the Animal class and have an additional function called makeSound( ) because each animal makes different sounds. For Example birds chirp, Mammals roar and reptiles hiss.
Following is the structure of these classes which includes additional data members and functions:

```
class Bird : public Animal {
private:
    string color;
public:
    Bird(string name, int age, string gender, string color);
    void display(;
    void makeSound();
};


class Mammal : public Animal {
private:
    float weight;

public:
    Mammal(string name, int age, string gender, float weight);
    void display();
    void makeSound();
};



class Reptile : public Animal {
private:
    float length;

public:
    Reptile(string name, int age, string gender, float weight);
    void display();
    void makeSound();
};
```

Now create a **Zookeeper** class which has following data members and functions:

```
class Zookeeper {
private:
    string zookeeperType;
    string name;
    int age;
    string gender;

public:
    Zookeeper(string zookeeperType, string name, int age, string gender);
    string getZookeeperType() const;
    string getName() const;
    int getAge() const;
    string getGender() const;
    void feedAnimal(const Animal& animal) const;
};
```

The Zookeeper has a name, age, gender and a zooKeeperType which is a string representing the type of the zookeeper (e.g. "BirdKeeper", "MammalKeeper", "ReptileKeeper"). It has getters of all the data members and a function feedAnimal which receives an animal objects and prints the name of the animal it is feeding. For example:
"Zookeeper feeds African Grey parrot"

Create a class **'Enclosure'** which has following data members and functions:

- `enclosureType`: a string representing the type of the enclosure (e.g. "BirdEnclosure", "MammalEnclosure", "ReptileEnclosure")
- `capacity`: an integer representing the maximum number of animals the enclosure can hold
- `animals`: a list of Animal objects currently in the enclosure
- `zookeeper`: the Zookeeper object responsible for maintaining the enclosure

Functions:

- `Enclosure(enclosureType, capacity, zookeeper)`: a constructor method that initializes the data members of the Enclosure class
- `getEnclosureType()`: a method that returns the enclosure type of the Enclosure object
- `getCapacity()`: a method that returns the maximum capacity of the Enclosure object
- `getCurrentAnimalCount()`: a method that returns the number of animals currently in the Enclosure object

```
class Enclosure {
private:
    string enclosureType;
    int capacity;
    vector<Animal*> animals;
    Zookeeper* zookeeper;

public:
    Enclosure(string enclosureType, int capacity, Zookeeper* zookeeper);
    ~Enclosure();
    string getEnclosureType() const;
    int getCapacity() const;
    int getCurrentAnimalCount() const;
    void addAnimal(Animal* animal);
    void removeAnimal(Animal* animal);
};
```

The Enclosure class has a composition relationship with the Animal class and Zookeeper class has a aggregation relationship with the Animal class.
In the main function:

1. Create pointers of 'Animal' class to store the objects of different types of animals (birds, mammals, reptiles)
2. Create three objects of enclosures named birdEnclosure, mammalEnclosure, reptileEnclosure and store some data through constructors in each object
3. Add animals created in Step-1 to each of the Enclosure objects. For example, add bird object to birdEnclosure using addAnimal function

4. Create 3 objects of ZooKeeper class for birds, mammal and reptile
5. Call the feedAnimal through each of the object of ZooKeeper class passing the different type of animals.

QUESTION 2 (40 Points)

You are tasked with implementing a program to swap the lower triangular half of the matrix with upper triangular (Without classes). Your program should have following functions:

1. **int** AllocateMemory(int& rows, int& cols)** that takes size of a char matrix (rows and columns) from user, allocates memory for the matrix and return its pointer.
2. **void InputMatrix(int** matrix, const int rows, const int cols) which inputs the data from console and store the values in the matrix.**
3. **void DisplayMatrix(int** matrix, const int& rows, const int& cols)** that displays the matrix in proper format.
4. **void swapTriangular(int** matrix, const int& rows, const int& cols)**: This function swaps the upper triangular halves of matrix with lower triangular if the matrix is not Upper Triangular

After swapping lower triangular halve with upper triangular half of Matrix C. It will become as following:

$$C=\begin{bmatrix}4 & 1 & 9\\ 8 & 6 & 2\\ 5 & 0 & 4\end{bmatrix} \quad \text{After swap} \Rightarrow \quad C=\begin{bmatrix}4 & 0 & 5\\ 2 & 6 & 8\\ 9 & 1 & 4\end{bmatrix}$$

Write a program which includes following tasks:

1. Create a 3x3 matrix and initialize its elements with values of your choice.
2. Display the matrix.
3. Swap the upper triangular halve with lower triangular halve.

Provide the implementation of the `Matrix` class and the program code that demonstrates its usage.

```
int main() {
    int rows, cols;
    cout << "Enter the number of rows: ";
    cin >> rows;

    cout << "Enter the number of columns: ";
    cin >> cols;

    //Allocate memory for matrix
    int** matrix = AllocateMemory(rows, cols)

    // Input matrix elements from the user
    InputMatrix( matrix, rows, cols)

    // Display the input matrix
    DisplayMatrix(matrix, rows, cols)
```

```
    swapTriangular(matrix, rows, cols);
    DisplayMatrix(matrix, rows, cols);

    return 0;
}
```