UNIVERSITY OF THE WEST INDIES, CAVE HILL CAMPUS

Department of Computer Science, Mathematics & Physics

COMP1210 Computing 2

Class Assignment Due Date: March 25th 2021 - Midnight

This assignment is worth 20% of your coursework

Background:

One of Barbados' most famous residents is the green monkey. They can be legitimately called the Barbados green monkey because you won't find them naturally existing anywhere else in the world. The green monkey was initially brought over to a Barbados as a pet from regions of West Africa during the slave trade over 350 years ago.

The green monkey poses a severe treat to the agriculture of Barbados; they are infamous for picking at crops, especially fruits. Due to a spike in complaints from farmers about crop damage caused by the green monkey, the ministry of Environment have convened a sub-committee of various stakeholders such as biodiversity conservation agencies, animal rights groups, agricultural development groups and tourism sector groups, to come up with a workable solution to the problem.

The sub-committee have decided that to chart a way forward they would need to have accurate data of the green monkey activities amount they troops and within they habitat. This is where you the computer programmers comes in, because the sub-committee has commissioned you to simulate the activities of the green monkey, so that they could get an accurate measure of how the monkeys interact among each other and propagate in the environment.

PART ONE

Specifications:

In this part of the assignment, you will be given three (3) files, these files are **NOT TO BE** modified:

• **M_names.txt**	- List of male names
• **F_names.txt**	- List of female names
• **green_monkey.h**	- Prototype for the green monkey class

The first two files are self-explanatory so we would go into more details about the third one **green_monkey.h** shown below, this header file has class declaration for the class **GreenMonkey**.

```cpp
#pragma once
#include <iostream>
#include <string>
#include <random>
#include <chrono>
#include <fstream>
#include<iomanip>
#define minAge 1
#define maxAge 4
#define minWeight 0.5f
#define maxWeight 3.5f
#define minIncrease 0.1f
#define maxIncrease 1.0f
using namespace std;


class GreenMonkey
{
public:
	GreenMonkey();
	int getRandomNumber(int, int);
	float getRandomNumber(float, float);
	void print();
	void eat();
	void age_me();
	void fight(GreenMonkey&);
	void set_weight(float);
	void set_age(short int);
	void set_name(string);
	void set_gender(char);
	void set_alive(bool);
	void set_mutant(bool);
	void set_injured(short int);
	void set_ID(int);
	float get_weight();
	short int get_age();
	string get_name();
	char get_gender();
	bool get_alive();
	bool get_mutant();
	short int get_injured();
	int get_ID();
	void set_next(GreenMonkey*);
	GreenMonkey* get_next();

private:
	int ID;
	float weight;
	short int age;
	string name;
	char gender;
	bool alive;
	bool mutant;
	int injured;
	GreenMonkey* next;
};
```

Let's give a description of the private data members of the **GreenMonkey** class:

ID – stores a unique number for each green monkey object in a linked list, as a new green monkey object is inserted into the linked list, its ID value is set to one more than the last green monkey object that was added to the list

weight – stores the weight of the green monkey object

age – stores the age of the green monkey object

name – stores the name of the green monkey object

gender – stores the gender of the green monkey object

alive – stores the boolean value indicating whether the green monkey object is dead or alive

mutant – stores the boolean value indicating whether the green monkey object is a mutant or not

injured – stores the number of times a green monkey object is injured

next – stores a pointer to the next green monkey in the linked list

Let's give the specifications of the public member functions of the **GreenMonkey** class:

1. **getRandomNumber()** – This is an overloaded function used to generate a random number in the range of the parameters passed in. The first parameter is the minimum value and the second parameter is the maximum value to be returned by this function. [6 Marks]

2. **GreenMonkey() -** This is the constructor for the green monkey class, here a number of things are to be done when this constructor is called, which are:
    - Assign 'M' or 'F' randomly to the **gender** data member of the object, each value has a 50% change of being assigned [4 Marks]
    - Assign a name to the green monkey, by using the **gender** of the monkey the appropriate text file should be open and a name picked at random, this picked name will be assigned to the **name** data member of the object. They are two text files provided with this assignment, F_names.txt and M_names.txt, the capital letter indicates the gender of the names in the file. [8 Marks]
    - Assign a random **weight** using the constants *minWeight* and *maxWeight* as the parameters to the getRandomNumber(). [2 Marks]
    - Assign a random **age** using the constants *minAge* and *maxAge* as the parameters to the getRandomNumber(). [2 Marks]
    - Assign *true* to data member **alive**. [0.5 Marks]
    - Assign *false* to data member **mutant**. [0.5 Marks]
    - Assign *zero* to data member **injured**. [0.5 Marks]
    - Assign *NULL* to data member **next**. [0.5 Marks]

3. **print()** - This function simply displays the data members, **ID, name**, **gender**, **age**, **weight**, **alive**, **mutant** and **injured**. Note **alive** should be printed as "*Alive*" or "*Dead*" and **mutant** should be printed as "*Normal*" or "*Mutant*". [4 Marks]

4. **eat()** - This function will increase the weight of the GreenMonkey by a random amount from 0.1% to 1.0% of the GreenMonkey s' current weight. If the monkey is a mutant its weight will not change regardless of the number of times it eats. This should use the constants *minIncrease* and *maxIncrease*. [6 Marks]

5. **age_me()** – This function increments the object's age by one. It will also include a 2% chance that the monkey will become a mutant each time it ages. Further, the GreenMonkey has a chance of dying each time it ages. This chance is equivalent to the integer part of the GreenMonkey's weight.
   i.e. A 5.36 pound GreenMonkey has a 5% chance of death. A 25.15 pound GreenMonkey has a 25% chance of death.
   In this function the green monkey's injuries will also be checked, the monkey will die if it has a value more than three (3) within the **injured** data member. [8 Marks]

6. **fight()** - This function accepts another GreenMonkey object as a parameter. It will have the calling GreenMonkey attack the other (passed in) GreenMonkey . The winner is based on a "fight ratio": it is calculated as

   (calling_GreenMonkey _weight/other_GreenMonkey _weight) * 50

   A random value from 1 to 100 is obtained. If it is greater than the fight ratio, the calling GreenMonkey **injured** data member is increased by one; otherwise the other GreenMonkey **injured** data member is increased by one. [8 Marks]

7. **set_** and **get_** functions should be defined as the mutators and accessors of the associated data members. [8 Marks]

In your driver code (the file with the main function), you should implement the following five functions:

- a **create_troop** function which accepts one integer parameter, representing the size of the troop. It will populate a vector with objects of the GreenMonkey class, creating a troop of monkeys.This function returns the vector containing the troop. [5 Marks]

- an **age_troop** function that accepts a vector of GreenMonkeys as an argument, and it will traverses the troop vector and invokes the age_me function on each GreenMonkey object. [5 Marks]

- a **feed_troop** this function that accepts a vector of GreenMonkeys as an argument, and it will traverses the troop vector and invokes the eat function on each GreenMonkey object. [5 Marks]

- an **enage_troop** function which randomly picks two GreenMonkey objects from the troop vector. If the two objects are the same gender the younger of the two will start a fight (calling the fight function) with the other one, if they are the same age then whom starts the fight is chosen at random. If they are different genders then they will mate and a new GreenMonkey will be created and added to the troop. If one or both of the picked monkeys are a mutant then they will not mate nor will they fight. [5 Marks]

- a **statistics_on_troop** function which accepts a troop vector as a parameter and should produce the following information:

    - number of male alive monkeys
    - number of female alive monkeys
    - number of mutant monkeys
    - number of monkeys less than or equal to 3 year old
    - number of male dead monkeys
    - number of female dead monkeys
    - average age of the monkeys
    - average weight of the monkeys [4 Marks]

In your main function you should run the following simulation:
- Create a troop of 50 monkeys
- Display statistics
- Age the troop 10 times
- Feed the troop 6 times
- Engage the troop 8 times
- Display statistics [12 Marks]

Notes:
- Think very carefully about writing the above functions and how they should be used. There are indeed circumstances when some functions should not execute. For example, a dead monkey shouldn't eat anything or a mutant monkey cannot reproduce

Penalties:

1. Runtime or compile errors [5 Marks]
2. Use of hard-coded elements [2 Marks]
3. Modifying the given files [5 Marks]
4. No comments in your code [2 Marks]

THE END