National University of Singapore
School of Continuing and Lifelong Education

TIC1002: Introduction to Computing and Programming II
Semester II, 2020/2021

Tutorial 8
OOP Concepts

1. Old McDonald's farm has some new animals.

   Just as before, each animal has a name, and makes a sound. Of these, some animals are flyers. These animals can fly—while they are flying, they make the sound "flap" until they stop flying.

   The new class definition is shown below. However, there are some errors below:

```
class Animal {
private:
    string _name; // e.g. Cow
    string _sound; // e.g. Moo

public:
    Animal(string name, string sound) {
        _name = name;
        _sound = sound;
    }

    string getName() { return _name; }

    string getSound() { return _sound; }
};


class Flyer : public Animal {
private:
    string _name;
    string _sound;
    bool _isFlying;

public:
    Flyer(string name, string sound)
    : _name(name), _sound(sound), _isFlying(false) {}

    string getSound() {
        if(_isFlying) return "flap";
        else return Animal::getSound();
    }
    void fly() { _isFlying = true; }

    void stop() { _isFlying = false; }
```

```
};


class Glider : Flyer {
private:
    bool _isGliding;

public:
    Glider(string name, string sound)
    : Flyer(name, sound), _isGliding(false) {}

    void glide() { if(_isFlying) _isGliding = true; }

    void stop() { _isFlying = false; _isGliding = false; }

    string getSound() {
        if(_isGliding) return "woosh";
        else return getSound();
    }
};
```

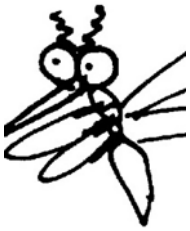
(a) How is overriding demonstrated here, and how is it useful?

(b) Identify and rectify the errors in the code above. There should be 5 errors. (Hint: try compiling the code after you have corrected the errors!)

Similar to Tutorial 1, we want to write a program that sings "Old McDonald". However, flyers and gliders will make different sounds, depending on whether they are currently flying or gliding. The animals on Old McDonald's farm are as follows:

Parrot
"sqwark"
not flying

Cow
"moo"



Mosquito
"buzz"
flying

Sheep
"mehh"

Fish
"blurp"

Picture credits: clipartpanda.com

Use the following template below to solve your problem. Note that you are using the same `sing()` function as Tutorial 1, which does not check whether a particular animal is a flyer or non-flyer.

```
class NewMcDonald {
private:
    vector<Animal*> _farm; // New McDonald had a farm (still has now)
```

```
public:
    NewMcDonald() {
        /* TODO: Create your farm,  an Animal* vector */
    }
    ~NewMcDonald() {
        /* TODO: New McDonald has no (more) farm... */
    }
    void sing() {
        for (Animal *animal : farm) {
            cout << "Old McDonald had a farm, E-I-E-I-O\n";
            cout << "And on his farm he had a " << animal->getName() << ", E-I-E-I-O\n";
            cout << "With a " << animal->getSound() << " " << animal->getSound() << " here a
            cout << "..." << endl;
        }
    }

    void fillThisFarm() {
        _farm.pushback(new Flyer("Parrot", "squark"));
        _farm.pushback(new Animal("Cow", "moo"));
        _farm.pushback(new Flyer("Mosquito", "buzz"));
        ((Flyer*)_farm[2])->fly();
        _farm.pushback(new Animal("Sheep", "mehh"));
        _farm.pushback(new Animal("Fish", "blurp"));
    }
};
```

(c) What is the datatype of `_farm[0]`? Why can a pointer to a Flyer be assigned to `_farm[0]`?

(d) Why can't `((Flyer*)_farm[2])->fly()` be replaced with `_farm[2]->fly()`?

(e) With the given `Animal` and `Flyer` classes above, why will polymorphism not work? Make the necessary change.