Bangladesh University of Business and Technology

BUBT

Assignment

On

Course Title : Object Oriented Programming

Course Code : CSE 122

Submitted to :

Khan Md. Hasib

Lecturer

Department of CSE

BUBT

Submitted by :

Name : Faria Akther Meghla

ID : 21225103120

Intake : Intake-49

Date :

Assignment-1

```
#ifndef ANIMAL_H
#define ANIMAL_H

#include <iostream>
#include <utility>

class Animal {
public:
   Animal();
   Animal(int age, double x, double y);
   virtual ~Animal();

   void setAge(int age);
   int getAge() const;

   void setLocation(double x, double y);
   std::pair<double, double> getLocation() const;

   void setAlive(bool alive);
   bool isAlive() const;

   virtual void move(double x, double y);
```

```
    virtual void sleep();
    virtual void eat();

    virtual std::ostream& print(std::ostream& os) const;

private:
    static long idCounter;
    long id;
    int age;
    std::pair<double, double> location;
    bool alive;
};

std::ostream& operator<<(std::ostream& os, const Animal& animal);

#endif /* ANIMAL_H */
```

```
}

#include "Animal.h"

long Animal::idCounter = 0;

Animal::Animal()
    : id(++idCounter), age(0), location(std::make_pair(0.0, 0.0)), alive(true) {}

Animal::Animal(int age, double x, double y)
    : id(++idCounter), age(age), location(std::make_pair(x, y)), alive(true) {}

Animal::~Animal() {}

void Animal::setAge(int age) {
    this->age = age;
}

int Animal::getAge() const {
    return age;
}

void Animal::setLocation(double x, double y) {
    location.first = x;
    location.second = y;
}
```

```
std::pair<double, double> Animal::getLocation() const {
    return location;
}

void Animal::setAlive(bool alive) {
    this->alive = alive;
}

bool Animal::isAlive() const {
    return alive;
}

void Animal::move(double x, double y) {
    location.first = x;
    location.second = y;
}

void Animal::sleep() {
    std::cout << "Animal is sleeping" << std::endl;
}

void Animal::eat() {
    std::cout << "Animal is eating" << std::endl;
}

std::ostream& Animal::print(std::ostream& os) const {
    os << "ID: " << id << ", Age: " << age << ", Location: (" << location.first << ", " << location.second << "), Alive: " << alive;
    return os;
}

std::ostream& operator<<(std::ostream& os, const Animal& animal) {
    return animal.print(os);
}
```