OOP Lab 13

Open Ended Lab

SUBMITTED BY:

UBAID AHMAD

ROLL NO:

2410-0011

SUBMITTED TO:

Ma’am laiba Tanveer

DATE:

30/12/2025

Lab Report

Smart Fitness Tracker System using Object-Oriented Programming in C++

Objective

The objective of this lab is to design and implement ;

- a Smart Fitness Tracker System using Object-Oriented Programming (OOP) concepts in C++.
- The system models fitness tracking devices that monitor distance covered, heart rate, and calories burned.
- It demonstrates core OOP principles including inheritance, polymorphism, constructors and destructors, static data members, operator overloading, copy constructor, and dynamic memory management.

Problem Description

A company requires a C++ program to model smart fitness trackers. Each tracker records distance covered, heart rate, and calories burned.

The system should allow;

- displaying device information,
- combining data from two trackers,
- tracking how many tracker objects exist,
- ensuring proper memory handling.

Class Design

The system consists of a base class FitnessDevice and a derived class SmartTracker. FitnessDevice contains basic device information such as device ID and battery status and declares a pure virtual function for displaying details. SmartTracker extends this class by adding health-related data including distance, heart rate, and calories burned. A static data member is used to keep track of the number of active tracker objects.

Constructors and Destructor

The SmartTracker class includes a parameterized constructor to initialize all data members, a copy constructor to create a deep copy of an existing object, and a destructor to properly release resources and update the object count.

Polymorphism

Runtime polymorphism is achieved by overriding the pure virtual showDetails function of the FitnessDevice class in the SmartTracker class. This function displays complete information about the tracker.

Operator Overloading

The addition operator (+) is overloaded to combine two SmartTracker objects. Distance and calories burned are summed, while battery level and heart rate are averaged. The result is returned as a new SmartTracker object.

Dynamic Memory Management

SmartTracker objects are created dynamically using the new keyword and are properly deleted using delete to avoid memory leaks. The static count variable ensures accurate tracking of active objects.

Conclusion

This lab demonstrates the practical application of Object-Oriented Programming concepts in C++. The Smart Fitness Tracker System effectively uses inheritance, polymorphism, constructors, destructors, static members, operator overloading, and dynamic memory management to model a real-world application.

CODE:

```
#include <iostream>

#include <string>

using namespace std;


class FitnessDevice {

protected:
```

```
    int deviceID;
    int batteryStatus;

public:
    FitnessDevice(int id , int battery) {
        deviceID = id;
        batteryStatus = battery;
    }

    virtual void showDetails() = 0;

    ~FitnessDevice() {};
};

class SmartTracker : public FitnessDevice {
private:
    double distance;
    int heartRate;
    int caloriesBurned;
    static int count;

public:
```

```
SmartTracker(int id, int battery,
        double d, int hr, int cal ) : FitnessDevice(id, battery) {
    distance = d;
    heartRate = hr;
    caloriesBurned = cal;
    count++;
    cout << "SmartTracker [" << deviceID << "] created." << endl;
}

SmartTracker(const SmartTracker &other) :
FitnessDevice(other.deviceID, other.batteryStatus) {
    distance = other.distance;
    heartRate = other.heartRate;
    caloriesBurned = other.caloriesBurned;
    count++;
    cout << "SmartTracker [" << deviceID << "] copied." << endl;
}

~SmartTracker() {
    count--;
    cout << "SmartTracker [" << deviceID << "] destroyed." << endl;
}
```

```
void showDetails() override {
    cout << "Device ID: " << deviceID << endl;
    cout << "Battery: " << batteryStatus << "%" << endl;
    cout << "Distance: " << distance << " km" << endl;
    cout << "Heart Rate: " << heartRate << " bpm" << endl;
    cout << "Calories Burned: " << caloriesBurned << endl;
}

static void showCount() {
    cout << "Total SmartTracker objects: " << count << endl;
}

SmartTracker operator+(const SmartTracker &s) {
      int id=this->deviceID + count;
    SmartTracker temp(id, 0, 0, 0, 0);
        temp.batteryStatus = (this->batteryStatus + s.batteryStatus) / 2;
        temp.distance = this->distance + s.distance;
        temp.heartRate = (this->heartRate + s.heartRate) / 2;
        temp.caloriesBurned = this->caloriesBurned + s.caloriesBurned;

return temp;
}
```

```
};

int SmartTracker::count = 0;

int main() {
    SmartTracker *t1 = new SmartTracker(11, 90, 5.2, 80, 250);
    SmartTracker *t2 = new SmartTracker(12, 85, 7.5, 75, 300);
    SmartTracker *t3 = new SmartTracker(13, 88, 10.0, 78, 400);

    cout << "\n--- Tracker Details ---\n";
    t1->showDetails();
    cout << endl;
    t2->showDetails();
    cout << endl;
    t3->showDetails();

    SmartTracker::showCount();

    cout << "\n--- Combining Trackers ---\n";
    SmartTracker t4 = *t1 + *t2;
    t4.showDetails();
```

```
    SmartTracker::showCount();

    cout << "\n--- Copy Constructor Demo ---\n";
    SmartTracker t5 = *t3;
    t5.showDetails();

    SmartTracker::showCount();

    cout << "\n--- Deleting Objects ---\n";
    delete t1;
    SmartTracker::showCount();

    delete t2;
    SmartTracker::showCount();

    delete t3;
    SmartTracker::showCount();

    return 0;
}
```

OUTPUT:

D:\4th semesteer data\OOP\00 lab 13 Open ended\Smart_fitness_tracker.exe

```
SmartTracker [11] created.
SmartTracker [12] created.
SmartTracker [13] created.

--- Tracker Details ---
Device ID: 11
Battery: 90%
Distance: 5.2 km
Heart Rate: 80 bpm
Calories Burned: 250

Device ID: 12
Battery: 85%
Distance: 7.5 km
Heart Rate: 75 bpm
Calories Burned: 300

Device ID: 13
Battery: 88%
Distance: 10 km
Heart Rate: 78 bpm
Calories Burned: 400
Total SmartTracker objects: 3

--- Combining Trackers ---
SmartTracker [14] created.
Device ID: 14
Battery: 87%
Distance: 12.7 km
Heart Rate: 77 bpm
Calories Burned: 550
Total SmartTracker objects: 4
```

```
--- Copy Constructor Demo ---
SmartTracker [13] copied.
Device ID: 13
Battery: 88%
Distance: 10 km
Heart Rate: 78 bpm
Calories Burned: 400
Total SmartTracker objects: 5

--- Deleting Objects ---
SmartTracker [11] destroyed.
Total SmartTracker objects: 4
SmartTracker [12] destroyed.
Total SmartTracker objects: 3
SmartTracker [13] destroyed.
Total SmartTracker objects: 2
SmartTracker [13] destroyed.
SmartTracker [14] destroyed.
```

END,,,