VIT - Vellore

Name: VAISHNAVI AWASTHI .
Email: vaishnavi.awasthi2021@vitstudent.ac.in
Roll no: 21BCE3997
Phone: 9999999999
Branch: VINILA JINNY_OOPS
Department: admin
Batch: VL2024250501961
Degree: admin

Scan to verify results

BCSE102P_VL2024250501961_ASSESSMENT_SET 3

BCSE102P_VL2024250501961_Assessment 3_Set 1

Attempt : 2
Total Mark : 20
Marks Obtained : 20

Section 1 : Coding

1. Problem Statement

A city water department wants to implement an automated billing system that calculates monthly charges based on water consumption. The system needs to apply penalties for exceeding usage limits and determine potential yearly savings if users optimize their consumption.

The BillCalculator class should be made a friend class of WaterBilling so that it can access the private members and perform the necessary calculations.

Formulas:

baseCharge = baseRate * monthlyUsage

penalty = (monthlyUsage - 100.0) * excessRate if monthlyUsage is greater than 100.0 else it is 0.0

totalCharge = baseCharge + penalty

optimizedCharge = baseRate * optimizedUsage

yearlySavings = (totalCharge - optimizedCharge) * 12

Answer

```
#include <iostream>
#include <iomanip>
using namespace std;

class WaterBilling {
private:
    double monthlyUsage;
    double baseRate;
    double excessRate;
    double optimizedUsage;

public:
    WaterBilling(double usage, double base, double excess, double optimized) {
        monthlyUsage = usage;
        baseRate = base;
        excessRate = excess;
        optimizedUsage = optimized;
    }

// You are using GCC
    //Type your code here
    friend class BillCalculator;
};

class BillCalculator {
    //Type your code here
    public:
    void calculateBill(WaterBilling bill){
        double bc=(bill.baseRate*bill.monthlyUsage);
        double p;
        if(bill.monthlyUsage>100){
            p=(bill.monthlyUsage-100)*bill.excessRate;
        }else{
            p=0;
        }
        double tc=bc+p;
        double oc=bill.baseRate*bill.optimizedUsage;
        double ys=(tc-oc)*12;
        cout<<"Monthly Charge: "<<fixed<<setprecision(2)<<tc<<endl;
        cout<<"Penalty: "<<fixed<<setprecision(2)<<p<<endl;
        cout<<"Optimized Charge: "<<fixed<<setprecision(2)<<oc<<endl;
        cout<<"Yearly Savings: "<<fixed<<setprecision(2)<<ys<<endl;
    }
};

int main() {
    double monthlyUsage, baseRate, excessRate, optimizedUsage;
    cin >> monthlyUsage;
    cin >> baseRate;
    cin >> excessRate;
    cin >> optimizedUsage;

    WaterBilling bill(monthlyUsage, baseRate, excessRate, optimizedUsage);
    BillCalculator calculator;
    calculator.calculateBill(bill);

    return 0;
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statment

Aarav runs a Digital Piano Learning System where students enroll in online piano lessons and track their practice sessions. Each student has a learning level, and their performance is evaluated based on practice hours and test scores. Implement a system using inheritance where a base class stores student details, while derived classes calculate progress scores and determine if a student is ready for an advanced level.

Formulas Used:

Progress Score Calculation (for each student)

progress_score = (practice_hours * 0.5) + (test_score * 0.5)

This ensures equal weightage is given to practice time and test performance.

Average Progress Score Calculation:

Sum up all individual student progress scores and divide by n.

average_progress_score = (sum of all progress scores) / n

Advanced Level Eligibility Criteria:

A student is considered "Advanced Level Ready" if their progress score is 75 or more.

Count the number of students meeting this condition.

Answer

```
#include <iostream>
#include <iomanip>
using namespace std;
const int MAX_STUDENTS = 100;

// You are using GCC
class Student {
   //Write your code here
   public:
   string name;int lvl;double hr;int score;
   void initStudent(string n,int l,double h,int s){
      name=n;
      lvl=l;
      hr=h;
      score=s;
   }

};

class PianoStudent : public Student {
   //Write your code here
   public:



   double progressScore(){
      return (0.5*hr+0.5*score);
   }
   bool isAdvancedLevelReady(){
      double pg=(0.5*hr)+(0.5*score);
```

```
        if(pg>=(double)75){
            return true;
        }
        return false;
    }

};

int main() {
    int n;
    cin >> n;

    PianoStudent pianoStudents[MAX_STUDENTS];

    string name;
    int level, testScore;
    double practiceHours;

    for (int i = 0; i < n; i++) {
        cin >> name >> level >> practiceHours >> testScore;
        pianoStudents[i].initStudent(name, level, practiceHours, testScore);
    }

    double totalProgress = 0;
    int advancedReadyCount = 0;

    for (int i = 0; i < n; i++) {
        totalProgress += pianoStudents[i].progressScore();
        if (pianoStudents[i].isAdvancedLevelReady()) advancedReadyCount++;
    }

    cout << fixed << setprecision(2);
    cout << "Average Progress Score: " << totalProgress / n << endl;
    cout << "Advanced Level Ready: " << advancedReadyCount << endl;

    return 0;
}
```

Status : Correct ***Marks : 10/10***