VIT - Vellore

Name: RONIT MEXSON .
Email: ronit.mexson2024@vitstudent.ac.in
Roll no: 24BAI0036
Phone: 9999999999
Branch: ARUMUGA ARUN R_OOPS
Department: admin
Batch: VL2024250502365
Degree: admin

Scan to verify results

BCSE102P_Structured and Object Oriented Programming Lab_VL2024250502365

VIT V_Structured and OOP_Lab 6_COD_Easy_Multi-level inheritance

Attempt : 1
Total Mark : 20
Marks Obtained : 20

Section 1 : Coding

1. Problem Statement

Mohit wants a financial calculator program for Fixed Deposits and Simple Interest. He needs a program that uses multi-level inheritance. The program should have three classes:

class Investment - Holds the principal, interest rate, and time period as attributes.class FixedDeposit - Derived from the Investment class, calculates the maturity amount using a method called calculateMaturityAmount(). class SimpleInterest - Derived from FixedDeposit class, prints the maturity amount using a method called printFD(). It then computes the simple interest and prints it using a method called calculateSimpleInterest().

Formulas used:

Maturity Amount = P * (1 + (R/100)) ^ T where the power value is calculated using pow() function from the math library.Simple Interest = P * N * R/100

where P - principal, R - interest rate, and N - time period in years.

Answer

```
// You are using GCC
#include <iostream>
#include <iomanip>
#include <cmath>

using namespace std;

class Investments {
protected:
    double amt, int_rate, time_period;

public:
    Investments(double principle, double rate, double time)
        : amt(principle), int_rate(rate), time_period(time) {}
};

class FixedDeposit : public Investments {
public:
    FixedDeposit(double principle, double rate, double time)
        : Investments(principle, rate, time) {}

    double calculateMaturityInterest() {
        double maturityAmount = amt * pow((1 + (int_rate / 100)), time_period);
        return maturityAmount - amt;  // Returning only the interest earned
    }

    double getMaturityAmount() {
        return amt * pow((1 + (int_rate / 100)), time_period);
    }
};

class SimpleInterest {
private:
    double amt, int_rate, time_period;
```

```
public:
    SimpleInterest(double principle, double rate, double time)
        : amt(principle), int_rate(rate), time_period(time) {}

    double calculateSimpleInterest() {
        return (amt * int_rate * time_period) / 100.0;
    }
};

int main() {
    double p, r, t;
    cin >> p >> r >> t;

    FixedDeposit fd(p, r, t);
    SimpleInterest si(p, r, t);

    cout << fixed << setprecision(2);
    cout << "Maturity Amount: " << fd.getMaturityAmount() << endl;
    cout << "Simple Interest: " << si.calculateSimpleInterest() << endl;

    return 0;
}
```

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

2. Problem Statement

Amar needs a program to calculate order costs with discount options for James. Help Amar to write a program that uses multi-level inheritance.

class Order - Holds item price, quantity, and discount as attributes.class FinalOrder - Derived from Order class which calculates the total cost for James with a given item price, quantity, and discount percentage.class DiscountedOrder - Derived from FinalOrder class which calculates the final cost for James, considering an additional discount on top of the regular discount.

Formulas used:

Total Cost = (Item Price × Quantity) - (Item Price × Quantity × Discount / 100.0)Final Cost = (Total Cost) - (Total Cost × Additional Discount / 100.0)

Answer

```
// You are using GCC
#include<iostream>
#include<iomanip>
using namespace std;

class Order {
public:
   double price;
   int qty;
   double discount;
   double total;
   double add_dis;
};

class FinalOrder : public Order {
public:
   void display_total() {
      total = (price * qty) - (price * qty * discount / 100.0);
      cout << fixed << setprecision(2) << "Total Cost: " << total << endl;
   }
};

class DiscountedOrder : public FinalOrder {
public:
   void set(double a, int b, double c, double d) {
      price = a;
      qty = b;
      discount = c;
      add_dis = d;
   }

   void display_final() {
      cout << fixed << setprecision(2) << "Final Cost: " << (total) - ((total *
add_dis) / 100.0);
   }
};

int main() {
   double p, d, d1;
   int q;
   cin >> p >> q >> d >> d1;
```

```
    DiscountedOrder obj;
    obj.set(p, q, d, d1);
    obj.display_total();
    obj.display_final();

    return 0;
}
```

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