Nishanth P – 24BAD405

Week 8

1. Multiple Inheritance – Professor Salary
   Code

```cpp
#include <iostream>
#include <string>
using namespace std;

class Teacher{
    protected:
        double basic_pay;
        void setSalary(double pay){
            basic_pay=pay;
        }
};

class Researcher{
    protected:
        double allowance;
        void getAllowance(string res){
            if(res=="ai"){
                allowance=6500;
            }
            else if(res=="quantum"){
                allowance=8750;
            }
            else{
                allowance=0;
            }
        }
};

class Professor:public Teacher, Researcher{
    public:
        double total;
        Professor(double sal, string resh){
            setSalary(sal);
            getAllowance(resh);
            totalPay();
        }
        void totalPay(){
            total=basic_pay+allowance;
            cout<<"Total Payment is "<<total<<endl;
        }
};

int main(){
    int salary;
    string res;
    cin>>salary>>res;
    if(salary>0){
        Professor prof(salary, res);
    }

    return 0;
}
```

Output

Enter a salary and research field:
25000 ai
Total Payment is 31500

Process returned 0 (0x0)   execution time : 10.722 s
Press any key to continue.

Week 8

2. Multilevel Inheritance – Student Marks

Code

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

class Person {
protected:
    string name;
    int age;
public:
    void getPersonData() {
        cout << "Enter name: ";
        cin >> name;
        cout << "Enter age: ";
        cin >> age;
    }
    void displayPersonData() {
        cout << "Name: " << name << endl;
        cout << "Age: " << age << endl;
    }
};

class Student : public Person {
protected:
    int marks[3];
public:
    void getStudentData() {
        getPersonData();
        cout << "Enter marks for 3 subjects: " << endl;
        for(int i = 0; i < 3; i++) {
            cout << "Subject " << i+1 << ": ";
            cin >> marks[i];
        }
    }
    void displayStudentData() {
        displayPersonData();
        cout << "Marks: ";
        for(int i = 0; i < 3; i++) {
            cout << marks[i] << " ";
        }
        cout << endl;
    }
};

class GraduateStudent : public Student {
private:
    float average;
    float percentage;
public:
    void calculateResult() {
        int sum = 0;
        for(int i = 0; i < 3; i++) {
            sum += marks[i];
        }
        average = sum / 3.0;
        percentage = (sum / (3.0 * 100)) * 100;
    }
    void displayResult() {
        displayStudentData();
        cout << "Average Marks: " << average << endl;
        cout << "Percentage: " << percentage << "%" << endl;
    }
};

int main() {
    GraduateStudent gs;
    gs.getStudentData();
    gs.calculateResult();
    cout << "\nStudent Details & Result\n";
    gs.displayResult();
    return 0;
}
```

Output

```
Enter name: blk
Enter age: 20
Enter marks for 3 subjects:
Subject 1: 64
Subject 2: 78
Subject 3: 85

Student Details & Result
Name: blk
Age: 20
Marks: 64 78 85
Average Marks: 75.6667
Percentage: 75.6667%

Process returned 0 (0x0)    execution time : 15.654 s
Press any key to continue.
```

Week 8

3. Constructor Execution Order – Bonus Calculation

Code

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

class Base1 {
protected:
    float basic;
public:
    Base1(float b) {
        cout << "Base1 constructor called" << endl;
        basic = b;
    }
};

class Base2 {
protected:
    float bonusPercent;
public:
    Base2(float bp) {
        cout << "Base2 constructor called" << endl;
        bonusPercent = bp;
    }
};

class Employee : public Base1, public Base2 {
    float grossSalary;
public:
    Employee(float b, float bp) : Base1(b), Base2(bp) {
        cout << "Employee constructor called" << endl;
        grossSalary = basic + (basic * bonusPercent / 100);
    }

    void display() {
        cout << "Basic Salary: " << basic << endl;
        cout << "Bonus %: " << bonusPercent << endl;
        cout << "Gross Salary: " << grossSalary << endl;
    }
};

int main() {
    Employee e(basic, bonus);
    cout << "Enter basic salary: ";
    cin >> basic;
    cout << "Enter bonus percentage: ";
    cin >> bonus;

    Employee e(basic, bonus);
    cout << "\nSalary Details\n";
    e.display();

    return 0;
}
```

Output

```
"K:\OOPS\WEEK 8\Q3\Q3.exe

Enter basic salary: 25000
Enter bonus percentage: 7
Base1 constructor called
Base2 constructor called
Employee constructor called

Salary Details
Basic Salary: 25000
Bonus %: 7
Gross Salary: 26750

Process returned 0 (0x0)   execution time : 14.542 s
Press any key to continue.
```

4. Ambiguity in Multiple Inheritance – Area Calculation
Code

```cpp
Q4 > G Q4.cpp
1   #include <iostream>
2   using namespace std;
3
4   class Rectangle {
5   public:
6       void calculateArea(int length, int breadth) {
7           cout << "Area of Rectangle: " << (length * breadth) << endl;
8       }
9   };
10
11  class Triangle {
12  public:
13      void calculateArea(double base, double height) {
14          cout << "Area of Triangle: " << (0.5 * base * height) << endl;
15      }
16  };
17
18  class ShapeCalculator : public Rectangle, public Triangle {
19  public:
20      void rectangleArea(int length, int breadth) {
21          Rectangle::calculateArea(length, breadth);
22      }
23      void triangleArea(double base, double height) {
24          Triangle::calculateArea(base, height);
25      }
26  };
27
28  int main() {
29      ShapeCalculator sc;
30      sc.rectangleArea(10, 5);
31      sc.triangleArea(6.0, 4.0);
32      return 0;
33  }
34
```

Output

```
"K:\OOPS\WEEK 8\Q4\Q4.exe   X   +   v

Area of Rectangle: 50
Area of Triangle: 12

Process returned 0 (0x0)   execution time : 3.627 s
Press any key to continue.
```

Week 8

5. Diamond Problem – Student Grades

Code

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

class Person {
protected:
    string name;
public:
    void setName(string n) {
        name = n;
    }
    void displayName() {
        cout << "Name: " << name << endl;
    }
};

class Test : virtual public Person {
protected:
    int academicMarks;
public:
    void setAcademicMarks(int m) {
        academicMarks = m;
    }
};

class Sports : virtual public Person {
protected:
    int sportsMarks;
public:
    void setSportsMarks(int m) {
        sportsMarks = m;
    }
};

class Result : public Test, public Sports {
public:
    void displayResult() {
        displayName();
```

```cpp
};

class Result : public Test, public Sports {
public:
    void displayResult() {
        displayName();
        cout << "Academic Marks: " << academicMarks << endl;
        cout << "Sports Marks: " << sportsMarks << endl;
        cout << "Total Score: " << (academicMarks + sportsMarks) << endl;
    }
};

int main() {
    Result r;
    r.setName("blk");
    r.setAcademicMarks(75);
    r.setSportsMarks(15);
    r.displayResult();
    return 0;
}
```

Output

```
"K:\OOPS\WEEK 8\Q5\Q5.exe

Name: blk
Academic Marks: 75
Sports Marks: 15
Total Score: 90

Process returned 0 (0x0)    execution time : 3.304 s
Press any key to continue.
```

6. Multilevel – Animal Energy

Code

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

class Animal {
protected:
    double foodEnergy;
public:
    void setFoodEnergy(double cal) {
        foodEnergy = cal;
    }
};

class Mammal : public Animal {
protected:
    double activityEnergy;
public:
    void setActivityEnergy(double cal) {
        activityEnergy = cal;
    }
};

class Human : public Mammal {
private:
    double studyEnergy;
public:
    void setStudyEnergy(double cal) {
        studyEnergy = cal;
    }
    void calculateRemainingEnergy() {
        double remaining = foodEnergy - (activityEnergy + studyEnergy);
        cout << "Remaining Energy: " << remaining << " calories" << endl;
    }
};

int main() {
    Human h;
    h.setFoodEnergy(2000);
    h.setActivityEnergy(500);
    h.setStudyEnergy(300);
    h.calculateRemainingEnergy();
    return 0;
}
```

Output

```
"K:\OOPS\WEEK 8\Q6\Q6.exe

Remaining Energy: 1200 calories

Process returned 0 (0x0)    execution time : 3.029 s
Press any key to continue.
```

Nishanth P – 24BAD405

Week 8

7. Multiple Inheritance – Duck Motion

Code

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

class Flyable {
protected:
    double flyingSpeed, flyingTime;
public:
    void setFlying(double speed, double time) {
        flyingSpeed = speed;
        flyingTime = time;
    }
    double flyDistance() {
        return flyingSpeed * flyingTime;
    }
};

class Swimmable {
protected:
    double swimmingSpeed, swimmingTime;
public:
    void setSwimming(double speed, double time) {
        swimmingSpeed = speed;
        swimmingTime = time;
    }
    double swimDistance() {
        return swimmingSpeed * swimmingTime;
    }
};

class Duck : public Flyable, public Swimmable {
public:
    double totalDistance() {
        return flyDistance() + swimDistance();
    }
};

int main() {
    Duck d;
    double fSpeed, fTime, sSpeed, sTime;

    cout << "Enter flying speed and time: ";
    cin >> fSpeed >> fTime;
    cout << "Enter swimming speed and time: ";
    cin >> sSpeed >> sTime;

    d.setFlying(fSpeed, fTime);
    d.setSwimming(sSpeed, sTime);

    cout << "Total distance covered by Duck: " << d.totalDistance() << endl;

    return 0;
}
```

Output

```
"K:\OOPS\WEEK 8\Q7\Q7.exe"

Enter flying speed and time: 15 10
Enter swimming speed and time: 10 2
Total distance covered by Duck: 170

Process returned 0 (0x0)    execution time : 17.267 s
Press any key to continue.
```

8. Access Specifiers – Marks Calculation

Code

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

class Base {
protected:
    int mark1, mark2, mark3;
public:
    void setMarks(int m1, int m2, int m3) {
        mark1 = m1;
        mark2 = m2;
        mark3 = m3;
    }
};

class Intermediate : public Base {
protected:
    int total;
public:
    void calculateTotal() {
        total = mark1 + mark2 + mark3;
    }
    int getTotal() {
        return total;
    }
};

class Derived : public Intermediate {
private:
    double average;
public:
    void calculateAverage() {
        average = total / 3.0;
    }
    void display() {
        cout << "Total: " << total << endl;
        cout << "Average: " << average << endl;
    }
};

int main() {
    Derived d;
    int m1, m2, m3;

    cout << "Enter 3 marks: ";
    cin >> m1 >> m2 >> m3;

    d.setMarks(m1, m2, m3);
    d.calculateTotal();
    d.calculateAverage();
    d.display();

    return 0;
}
```

Output

```
"K:\OOPS\WEEK 8\Q8\Q8.exe"

Enter 3 marks: 88 79 96
Total: 263
Average: 87.6667

Process returned 0 (0x0)   execution time : 16.486 s
Press any key to continue.
```
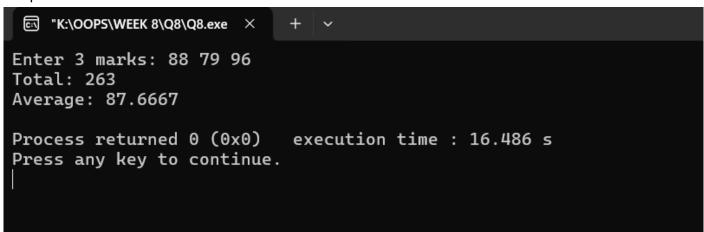
Week 8

9. Multiple Inheritance – Resolving Same Variable

Code

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

class A {
public:
    double x;
    void setCost(double c) {
        x = c;
    }
};

class B {
public:
    double x;
    void setSelling(double s) {
        x = s;
    }
};

class C : public A, public B {
public:
    double calculateProfit() {
        return B::x - A::x;
    }
};

int main() {
    C obj;
    double cost, selling;

    cout << "Enter cost price: ";
    cin >> cost;
    cout << "Enter selling price: ";
    cin >> selling;

    obj.setCost(cost);
    obj.setSelling(selling);

    double profit = obj.calculateProfit();
    if(profit > 0)
        cout << "Profit: " << profit << endl;
    else if(profit < 0)
        cout << "Loss: " << -profit << endl;
    else
        cout << "No Profit No Loss" << endl;

    return 0;
}
```

Output

```
"K:\OOPS\WEEK 8\Q9\Q9.exe

Enter cost price: 875
Enter selling price: 900
Profit: 25

Process returned 0 (0x0)   execution time : 8.620 s
Press any key to continue.
```

Week 8

10. Real-World Manager Salary

Code

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

class Employee {
protected:
    int id;
    string name;
public:
    void setEmployee(int eid, string ename) {
        id = eid;
        name = ename;
    }
    void displayEmployee() {
        cout << "Employee ID: " << id << endl;
        cout << "Name: " << name << endl;
    }
};

class Salary {
protected:
    double basic, DA, HRA;
public:
    void setSalary(double b, double d, double h) {
        basic = b;
        DA = d;
        HRA = h;
    }
    double calculateBaseSalary() {
        return basic + DA + HRA;
    }
};

class Manager : public Employee, public Salary {
private:
    double allowance;
public:
    void setAllowance(double a) {
```

```cpp
class Manager : public Employee, public Salary {
private:
    double allowance;
public:
    void setAllowance(double a) {
        allowance = a;
    }
    void displayGrossSalary() {
        displayEmployee();
        double gross = calculateBaseSalary() + allowance;
        cout << "Gross Salary: " << gross << endl;
    }
};

int main() {
    Manager m;
    int id;
    string name;
    double basic, DA, HRA, allowance;

    cout << "Enter Employee ID and Name: ";
    cin >> id >> name;
    cout << "Enter Basic Pay, DA, HRA: ";
    cin >> basic >> DA >> HRA;
    cout << "Enter Special Allowance: ";
    cin >> allowance;

    m.setEmployee(id, name);
    m.setSalary(basic, DA, HRA);
    m.setAllowance(allowance);

    m.displayGrossSalary();

    return 0;
}
```

Output

```
Enter Employee ID and Name: 2809 blk
Enter Basic Pay, DA, HRA: 20000 2500 7500
Enter Special Allowance: 1500
Employee ID: 2809
Name: blk
Gross Salary: 31500

Process returned 0 (0x0)   execution time : 33.952 s
Press any key to continue.
```