

Third Year B. Tech (EL & CE)

Semester: V

Subject: Object-Oriented Programming Lab

Name: Shreerang Mhatre

Class: TY

Roll No: 52

Batch: A3

Experiment No: 05

Name of the Experiment: Virtual Function

Performed on: 22/11/2023

Submitted on: 22/11/2023

Problem Statement:

Write a C++ program with base class Employee and derive classes Class1_Employee, Class2_Employee and Class3_Employee.

Salary of an employee is calculated as per his/her designation.

Declare calculate salary () as a pure virtual function in the base class and define it in respective derive classes to calculate salary of an employee.

Output:

```
D:\Object Oriented Programming\exp5>cd "d:\Object Oriented Programming\exp5\" && g++
virtual_function.cpp -o virtual_function && "d:\Object Oriented
Programming\exp5\"virtual_function
Class1_Employee
Salary: $50000
Class2_Employee
Salary: $60000
Class3_Employee
Salary: $70000
```

Code:

```
#include <iostream>

// Base class
class Employee {
public:
    virtual double calculateSalary() const = 0; // virtual function

    virtual void displayType() const {
        std::cout << "Base Employee" << std::endl;
    }
};

// Derived class 1
class Class1_Employee : public Employee {
public:
    double calculateSalary() const override {
        // Implement salary calculation logic for Class1_Employee
        return 50000.0;
    }

    void displayType() const override {
        std::cout << "Class1_Employee" << std::endl;
    }
};

// Derived class 2
class Class2_Employee : public Employee {
public:
    double calculateSalary() const override {
        // Implement salary calculation logic for Class2_Employee
        return 60000.0;
    }

    void displayType() const override {
        std::cout << "Class2_Employee" << std::endl;
    }
};
```

// Derived class 3

```
class Class3_Employee : public Employee {
public:
    double calculateSalary() const override {
        // Implement salary calculation logic for Class3_Employee
        return 70000.0;
    }

    void displayType() const override {
        std::cout << "Class3_Employee" << std::endl;
    }
};

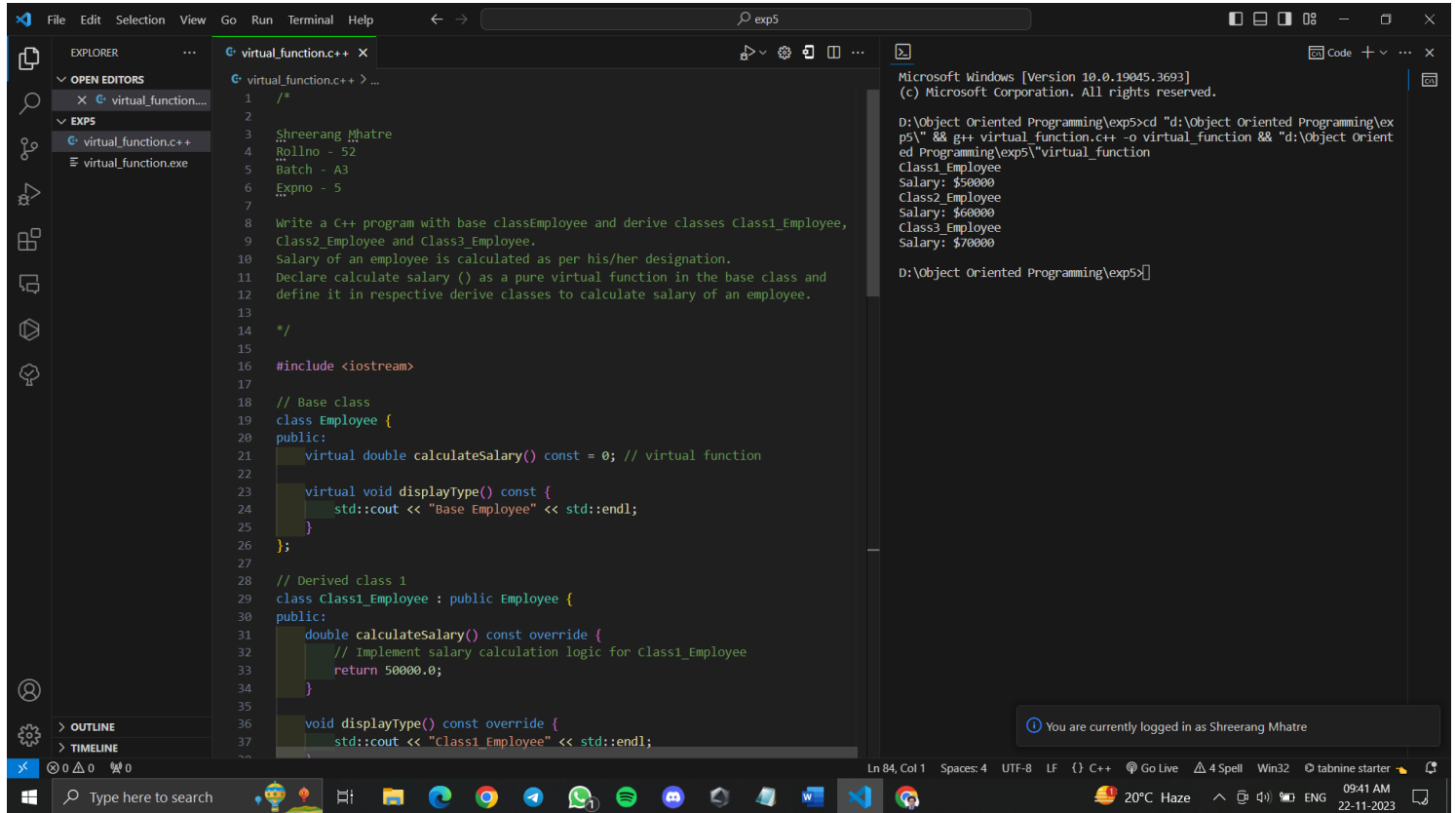
int main() {
    Class1_Employee employee1;
    Class2_Employee employee2;
    Class3_Employee employee3;

    // Displaying employee types and their salaries
    employee1.displayType();
    std::cout << "Salary: $" << employee1.calculateSalary() << std::endl;

    employee2.displayType();
    std::cout << "Salary: $" << employee2.calculateSalary() << std::endl;

    employee3.displayType();
    std::cout << "Salary: $" << employee3.calculateSalary() << std::endl;

    return 0;
}
```



```

1  /*
2
3  Shreerang Mhatre
4  Rollno - 52
5  Batch - A3
6  Expno - 5
7
8  Write a C++ program with base class Employee and derive classes Class1_Employee,
9  Class2_Employee and Class3_Employee.
10 Salary of an employee is calculated as per his/her designation.
11 Declare calculate salary () as a pure virtual function in the base class and
12 define it in respective derive classes to calculate salary of an employee.
13
14 */
15
16 #include <iostream>
17
18 // Base class
19 class Employee {
20 public:
21     virtual double calculateSalary() const = 0; // virtual function
22
23     virtual void displayType() const {
24         std::cout << "Base Employee" << std::endl;
25     }
26 };
27
28 // Derived class 1
29 class Class1_Employee : public Employee {
30 public:
31     double calculateSalary() const override {
32         // Implement salary calculation logic for Class1_Employee
33         return 50000.0;
34     }
35
36     void displayType() const override {
37         std::cout << "Class1_Employee" << std::endl;
38     }
39 };
40
41 // Derived class 2
42 class Class2_Employee : public Employee {
43 public:
44     double calculateSalary() const override {
45         // Implement salary calculation logic for Class2_Employee
46         return 60000.0;
47     }
48
49     void displayType() const override {
50         std::cout << "Class2_Employee" << std::endl;
51     }
52 };
53
54 int main() {
55     Employee* emp1 = new Class1_Employee();
56     Employee* emp2 = new Class2_Employee();
57
58     emp1->displayType();
59     emp1->calculateSalary();
60
61     emp2->displayType();
62     emp2->calculateSalary();
63
64     return 0;
65 }

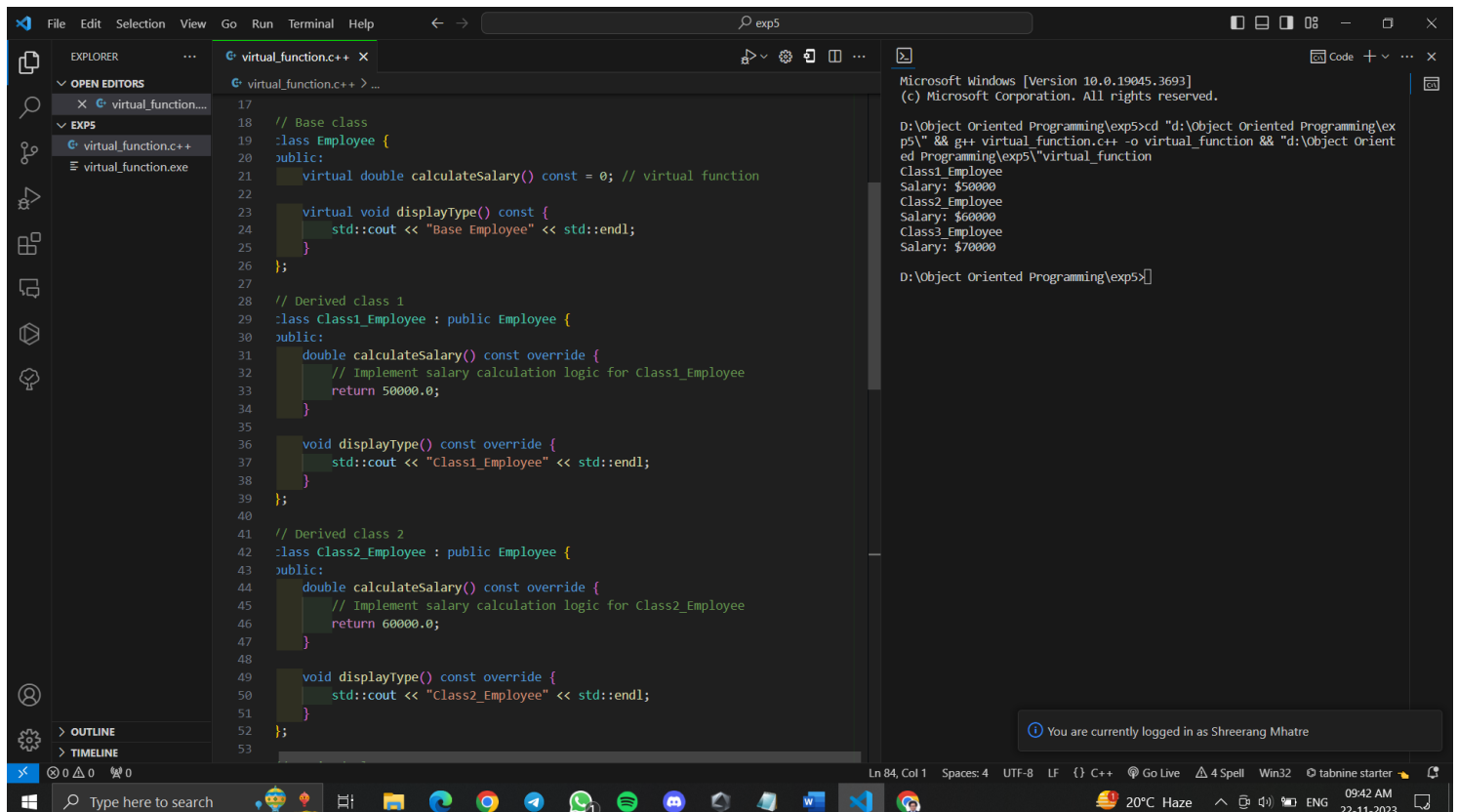
```

Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

D:\Object Oriented Programming\exp5>cd "d:\Object Oriented Programming\exp5" && g++ virtual_function.cpp -o virtual_function && "d:\Object Oriented Programming\exp5\virtual_function"
Class1_Employee
Salary: \$50000
Class2_Employee
Salary: \$60000
Class3_Employee
Salary: \$70000

D:\Object Oriented Programming\exp5>

You are currently logged in as Shreerang Mhatre



```

17 // Base class
18 class Employee {
19 public:
20     virtual double calculateSalary() const = 0; // virtual function
21
22     virtual void displayType() const {
23         std::cout << "Base Employee" << std::endl;
24     }
25 };
26
27 // Derived class 1
28 class Class1_Employee : public Employee {
29 public:
30     double calculateSalary() const override {
31         // Implement salary calculation logic for Class1_Employee
32         return 50000.0;
33     }
34
35     void displayType() const override {
36         std::cout << "Class1_Employee" << std::endl;
37     }
38 };
39
40 // Derived class 2
41 class Class2_Employee : public Employee {
42 public:
43     double calculateSalary() const override {
44         // Implement salary calculation logic for Class2_Employee
45         return 60000.0;
46     }
47
48     void displayType() const override {
49         std::cout << "Class2_Employee" << std::endl;
50     }
51 };
52
53 int main() {
54     Employee* emp1 = new Class1_Employee();
55     Employee* emp2 = new Class2_Employee();
56
57     emp1->displayType();
58     emp1->calculateSalary();
59
60     emp2->displayType();
61     emp2->calculateSalary();
62
63     return 0;
64 }

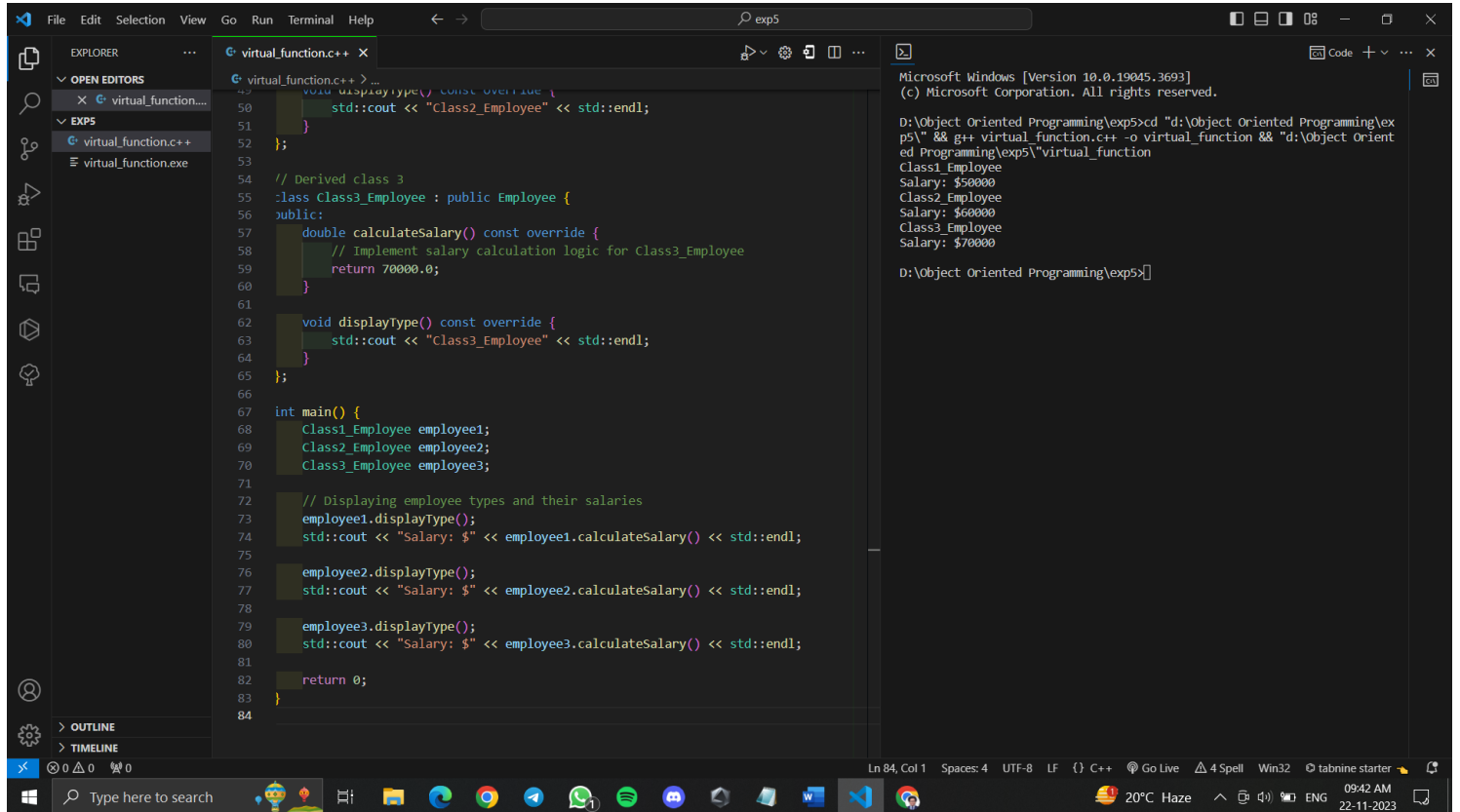
```

Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

D:\Object Oriented Programming\exp5>cd "d:\Object Oriented Programming\exp5" && g++ virtual_function.cpp -o virtual_function && "d:\Object Oriented Programming\exp5\virtual_function"
Class1_Employee
Salary: \$50000
Class2_Employee
Salary: \$60000
Class3_Employee
Salary: \$70000

D:\Object Oriented Programming\exp5>

You are currently logged in as Shreerang Mhatre



The screenshot shows the Visual Studio Code editor with a C++ file named `virtual_function.cpp` open. The code defines a base class `Employee` with a virtual `displayType()` method and a `main()` function that creates three objects: `Class1_Employee`, `Class2_Employee`, and `Class3_Employee`. Each class overrides the `displayType()` method to print its name and salary. The `main()` function calls `displayType()` for each object and also prints the salary using the `calculateSalary()` method.

```

49 void displayType() const override {
50     std::cout << "Class2_Employee" << std::endl;
51 }
52 };
53
54 // Derived class 3
55 class Class3_Employee : public Employee {
56 public:
57     double calculateSalary() const override {
58         // Implement salary calculation logic for Class3_Employee
59         return 70000.0;
60     }
61
62     void displayType() const override {
63         std::cout << "Class3_Employee" << std::endl;
64     }
65 };
66
67 int main() {
68     Class1_Employee employee1;
69     Class2_Employee employee2;
70     Class3_Employee employee3;
71
72     // Displaying employee types and their salaries
73     employee1.displayType();
74     std::cout << "Salary: $" << employee1.calculateSalary() << std::endl;
75
76     employee2.displayType();
77     std::cout << "Salary: $" << employee2.calculateSalary() << std::endl;
78
79     employee3.displayType();
80     std::cout << "Salary: $" << employee3.calculateSalary() << std::endl;
81
82     return 0;
83 }
84

```

The terminal output shows the execution of the program, displaying the class names and salaries for each employee object.

```

D:\Object Oriented Programming\exp5>cd "d:\Object Oriented Programming\exp5" && g++ virtual_function.cpp -o virtual_function && "d:\Object Oriented Programming\exp5\virtual_function.exe"
Class1_Employee
Salary: $50000
Class2_Employee
Salary: $60000
Class3_Employee
Salary: $70000
D:\Object Oriented Programming\exp5>

```