

SINGLE INHERITANCE :-

```
#include <iostream>

using namespace std;

class Account {

    public:

    float salary = 60000;

};

class Programmer: public Account {

    public:

    float bonus = 5000;

};

int main()

{

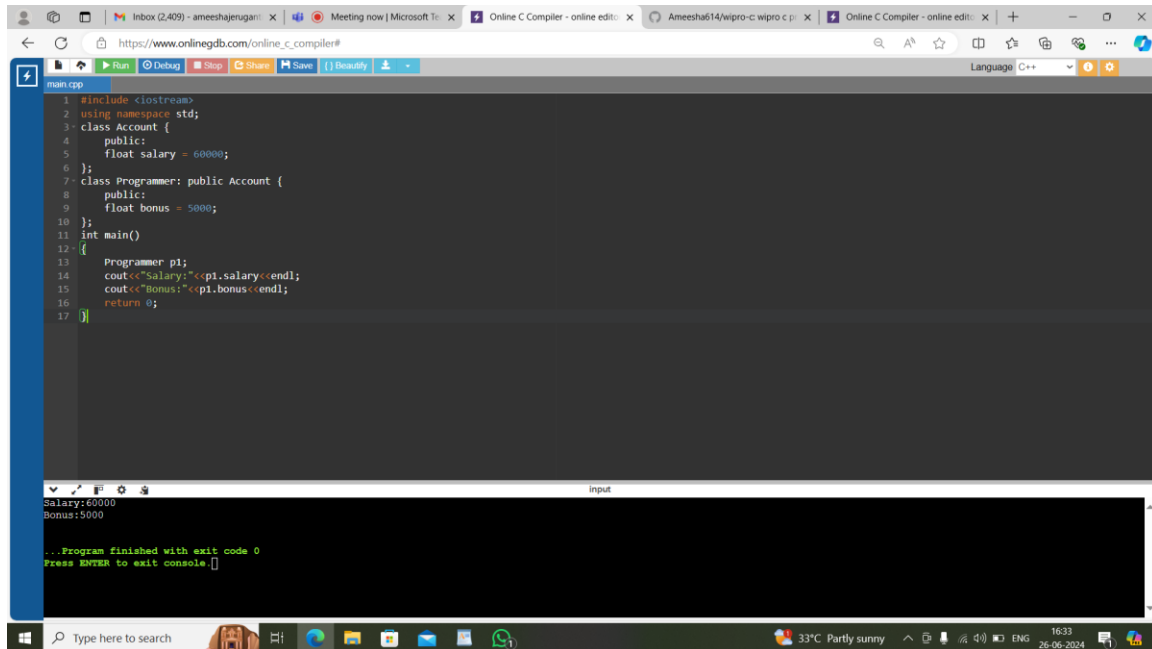
    Programmer p1;

    cout<<"Salary:"<<p1.salary<<endl;

    cout<<"Bonus:"<<p1.bonus<<endl;

    return 0;

}
```



The screenshot shows a web browser window with the URL https://www.onlinegdb.com/online_c_compiler#. The code editor contains the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3 class Account {
4     public:
5     float salary = 60000;
6 };
7 class Programmer: public Account {
8     public:
9     float bonus = 5000;
10 };
11 int main()
12 {
13     Programmer p1;
14     cout<<"Salary:"<<p1.salary<<endl;
15     cout<<"Bonus:"<<p1.bonus<<endl;
16     return 0;
17 }
```

The output window shows the following results:

```
Salary:60000
Bonus:5000

... Program finished with exit code 0
Press ENTER to exit console
```

AMBIGUITY IN INHERITANCE :-

```
#include <iostream>
```

```
using namespace std;
```

```
class A {
```

```
    public:
```

```
    void display() {
```

```
        cout<<"Class A"<<endl;
```

```
    }
```

```
};
```

```
class B {
```

```
    public:
```

```
    void display() {
```

```
        cout<<"Class B"<<endl;
```

```
    }
```

```
};
```

```
class C: public A, public B {
```

```

public:

void view() {

    A :: display();

    B :: display();

}

};

int main() {

    C c;

    c.view();

}

```

The screenshot shows a web browser window with the URL https://www.onlinegdb.com/online_c_compiler#. The code editor contains the following C++ code:

```

1 #include <iostream>
2 using namespace std;
3 class A {
4 public:
5     void display() {
6         cout<<"Class A"<<endl;
7     }
8 };
9 class B {
10 public:
11     void display() {
12         cout<<"Class B"<<endl;
13     }
14 };
15 class C: public A, public B {
16 public:
17     void view() {
18         A :: display();
19         B :: display();
20     }
21 };
22 int main() {
23     C c;
24     c.view();
25 }

```

The output window shows the following text:

```

Class A
Class B
...Program finished with exit code 0
Press ENTER to exit console.

```

POLYMORPHISM :-

```
#include <iostream>
```

```
#include <string>
```

```
#include <vector>
```

```
#include <numeric>          // For accumulate
```

[illegible]

```

        return gpa;
    }

    double calculateSemesterGPA(const std::vector<double>& grades) const {           // Function
to calculate semester GPA

        if (grades.empty()) return 0.0;

        double total = std::accumulate(grades.begin(), grades.end(), 0.0);

        return total / grades.size();
    }

    void getDetails() const override {                                           // Overriding getDetails
function

        Person::getDetails();

        std::cout << "Major: " << major << ", GPA: " << gpa << std::endl;

    }
};

class Faculty : public Person {                                                // Derived class Faculty

private:

    std::string department;

    std::string title;

public:

    Faculty(const std::string& n, int i, const std::string& d, const std::string& t)

        : Person(n, i), department(d), title(t) {}

    void setDepartment(const std::string& d) {                                // Function to set the
department

        department = d;

    }

    std::string getDepartment() const {                                        // Function to get the
department

        return department;

    }
}

```

```

void setTitle(const std::string& t) {                                // Function to set the title
    title = t;
}

std::string getTitle() const {                                    // Function to get the title
    return title;
}

void teachCourse(const std::string& courseName) const {          // Function to assign a
course to teach
    std::cout << title << " " << name << " is assigned to teach " << courseName << std::endl;
}

void getDetails() const override {                                // Overriding getDetails
function
    Person::getDetails();
    std::cout << "Department: " << department << ", Title: " << title << std::endl;
}
};

class Staff : public Person {                                    // Derived class
Staff

private:
    std::string position;

public:
    Staff(const std::string& n, int i, const std::string& p)      // Constructor
        : Person(n, i), position(p) {}

    void setPosition(const std::string& p) {                      // Function to set
the position
        position = p;
    }

    std::string getPosition() const {                             // Function to get the
position
        return position;
    }
}

```

```

    }

    void getDetails() const override {                // Overriding
getDetails function

        Person::getDetails();

        std::cout << "Position: " << position << std::endl;

    }

};

int main() {

    Student s1("Alice", 1001, "Computer Science", 3.8);    // Create a Student

    s1.getDetails();

    std::vector<double> grades = {3.9, 4.0, 3.7, 3.8};

    std::cout << "Semester GPA: " << s1.calculateSemesterGPA(grades) << std::endl;

    Faculty f1("Dr. Bob", 2001, "Physics", "Professor");    // Create a Faculty

    f1.getDetails();

    f1.teachCourse("Quantum Mechanics");

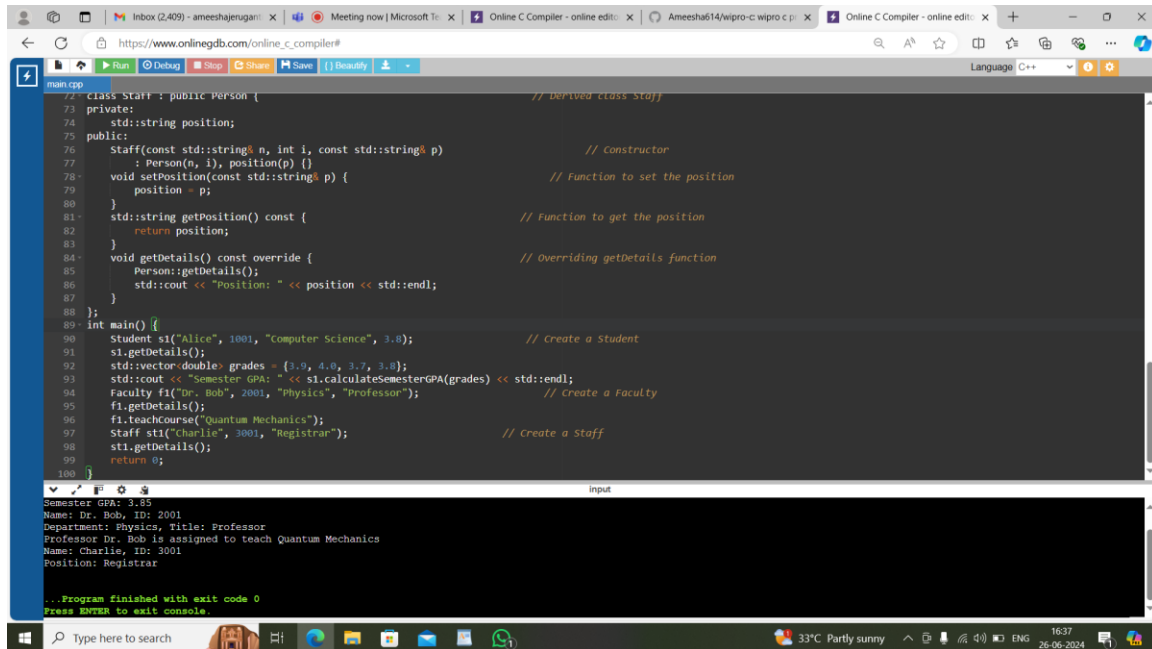
    Staff st1("Charlie", 3001, "Registrar");    // Create a Staff

    st1.getDetails();

    return 0;

}

```



```
1 // Derived class Staff
2 class Staff : public Person {
3 private:
4     std::string position;
5 public:
6     Staff(const std::string& n, int i, const std::string& p) // Constructor
7         : Person(n, i), position(p) {}
8     void setPosition(const std::string& p) { // Function to set the position
9         position = p;
10    }
11    std::string getPosition() const { // Function to get the position
12        return position;
13    }
14    void getDetails() const override { // Overriding getDetails function
15        Person::getDetails();
16        std::cout << "Position: " << position << std::endl;
17    }
18 };
19
20 int main() {
21     Student s1("Alice", 1001, "Computer Science", 3.8); // Create a Student
22     s1.getDetails();
23     std::vector<double> grades = {3.9, 4.0, 3.7, 3.8};
24     std::cout << "Semester GPA: " << s1.calculateSemesterGPA(grades) << std::endl;
25     Faculty f1("Dr. Bob", 2001, "Physics", "Professor"); // Create a faculty
26     f1.getDetails();
27     f1.teachCourse("Quantum Mechanics");
28     Staff st1("Charlie", 3001, "Registrar"); // Create a staff
29     st1.getDetails();
30     return 0;
31 }
```

Semester GPA: 3.85
Name: Dr. Bob, ID: 2001
Department: Physics, Title: Professor
Professor Dr. Bob is assigned to teach Quantum Mechanics
Name: Charlie, ID: 3001
Position: Registrar
... Program finished with exit code 0
Press ENTER to exit console.

FUNCTION OVERLOADING :-

```
#include <iostream>
```

```
using namespace std;
```

```
class cal {
```

```
public:
```

```
static int add(int a, int b) {
```

```
    return a + b;
```

```
}
```

```
static int add(int a, int b, int c)
```

```
{
```

```
    return a + b + c;
```

```
}
```

```
};
```

```
int main(void) {
```

```
    cal C;
```

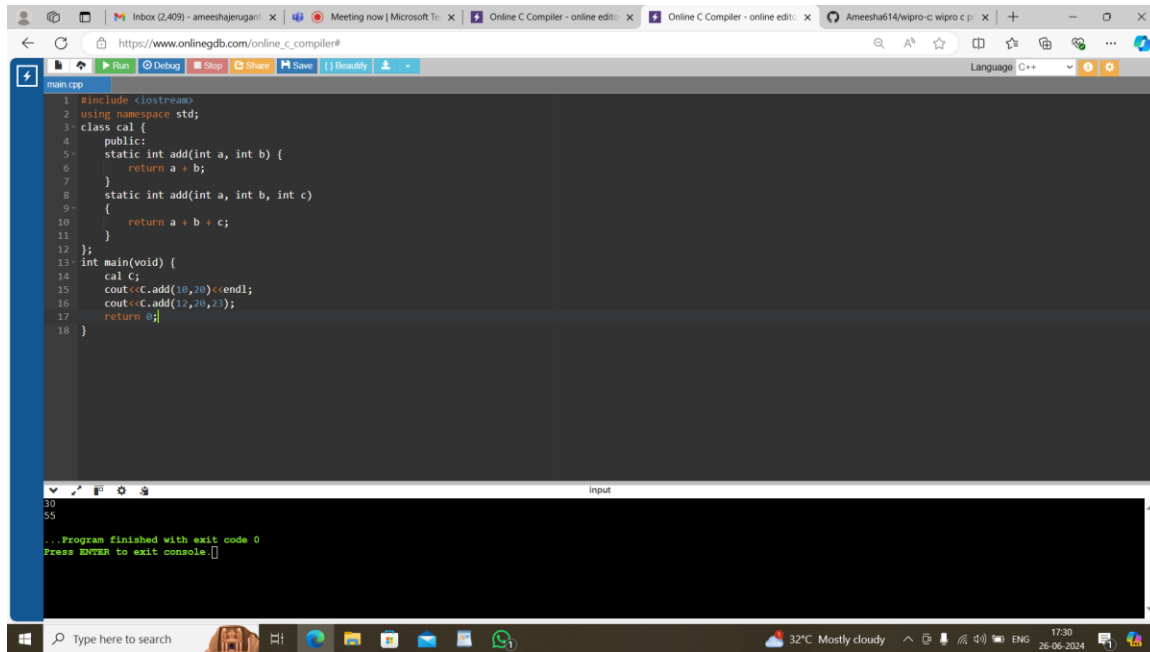
```
    cout<<C.add(10,20)<<endl;
```



```
cout<<C.add(12,20,23);

return 0;

}
```



The screenshot shows a web browser window with the URL https://www.onlinegdb.com/online_c_compiler#. The browser has several tabs open, including 'Inbox (2,409) - ameshajugan...', 'Meeting now | Microsoft Te...', 'Online C Compiler - online edit...', and 'Online C Compiler - online edit...'. The main content area displays a C++ program in a dark-themed editor. The program defines a class 'cal' with two static methods: 'add(int a, int b)' and 'add(int a, int b, int c)'. The 'main' function creates an object 'C' of the 'cal' class and calls 'C.add(10,20)' followed by 'C.add(12,20,23)', printing the results. The output window at the bottom shows the program finished with exit code 0 and the output '55'. The Windows taskbar at the bottom indicates the system time is 17:30 on 26-06-2024, with a temperature of 32°C and 'Mostly cloudy' weather.

```
1 #include <iostream>
2 using namespace std;
3 class cal {
4 public:
5     static int add(int a, int b) {
6         return a + b;
7     }
8     static int add(int a, int b, int c)
9     {
10         return a + b + c;
11     }
12 };
13 int main(void) {
14     cal C;
15     cout<<C.add(10,20)<<endl;
16     cout<<C.add(12,20,23);
17     return 0;
18 }
```

55

...Program finished with exit code 0
Press ENTER to exit console.