

1. We want to calculate the total marks of each student of a class in Physics, Chemistry and Mathematics and the average marks of the class. The number of students in the class are entered by the user. Create a class named Marks with data members for roll number, name and marks. Create three other classes inheriting the Marks class, namely Physics, Chemistry and Mathematics, which are used to define marks in individual subject of each student Roll number of each student will be generated automatically.

```
#include <iostream>
#include <vector>
using namespace std;
class Marks {
public:
    int rollNo;
    string name;
    Marks(string name) : name(name), rollNo(static_cast<int>(nextRollNo++))
    {}
private:
    static int nextRollNo;
};

int Marks::nextRollNo = 1;

class Physics : public Marks {
public:
    int marks;
    Physics(string name, int marks) : Marks(name), marks(marks) {}
};
```

```
class Chemistry : public Marks {
public:
    int marks;
    Chemistry(string name, int marks) : Marks(name), marks(marks) {}
};
```

```
class Mathematics : public Marks {
public:
    int marks;
    Mathematics(string name, int marks) : Marks(name), marks(marks) {}
};
```

```
int main() {
    int numStudents;
    cout << "Enter the number of students: ";
    cin >> numStudents;

    vector<Physics> physicsStudents;
    vector<Chemistry> chemistryStudents;
    vector<Mathematics> mathStudents;

    string name;
    int physicsMarks, chemistryMarks, mathMarks;

    for (int i = 0; i < numStudents; ++i) {
        cout << "Enter name of student " << i + 1 << ": ";
        cin >> name;
```

```

    cout << "Enter Physics marks: ";
    cin >> physicsMarks;
    cout << "Enter Chemistry marks: ";
    cin >> chemistryMarks;
    cout << "Enter Mathematics marks: ";
    cin >> mathMarks;

    physicsStudents.push_back(Physics(name, physicsMarks));
    chemistryStudents.push_back(Chemistry(name, chemistryMarks));
    mathStudents.push_back(Mathematics(name, mathMarks));
}

// Calculate total and average marks
int totalPhysics = 0, totalChemistry = 0, totalMath = 0;
for (int i = 0; i < numStudents; ++i) {
    totalPhysics += physicsStudents[i].marks;
    totalChemistry += chemistryStudents[i].marks;
    totalMath += mathStudents[i].marks;
}

double avgPhysics = static_cast<double>(totalPhysics) / numStudents;
double avgChemistry = static_cast<double>(totalChemistry) / numStudents;
double avgMath = static_cast<double>(totalMath) / numStudents;

// Output results
cout << "Average Physics marks: " << avgPhysics << endl;
cout << "Average Chemistry marks: " << avgChemistry << endl;

```

```

cout << "Average Mathematics marks: " << avgMath << endl;

return 0;
}

```

```

C:\Users\Vishnu Priya\OneDri >
Enter the number of students: 2
Enter name of student 1: priya
Enter Physics marks: 4
Enter Chemistry marks: 8
Enter Mathematics marks: 10
Enter name of student 2: vishnu
Enter Physics marks: 9
Enter Chemistry marks: 8
Enter Mathematics marks: 7
Average Physics marks: 6.5
Average Chemistry marks: 8
Average Mathematics marks: 8.5
-----
Process exited after 25.46 seconds with return value 0
Press any key to continue . . .

```

2. Given an integer array $A[]$ consisting of N non-negative integers representing an elevation map, Where the width of each bar is 1. The task is to compute the total volume of water that can be trapped after rain.

```

#include <iostream>

#include <vector>

using namespace std;

int trap(vector<int>& height) {
    int n = height.size();
    int left = 0, right = n - 1;
    int left_max = 0, right_max = 0;

```

```

int water = 0;
while (left < right) {
    if (height[left] < height[right]) {
        left_max = max(left_max, height[left]);
        water += left_max - height[left];
        left++;
    } else {
        right_max = max(right_max, height[right]);
        water += right_max - height[right];
        right--;
    }
}
return water;
}

int main()
{
    int n;
    cout << "Enter the number of elements: ";
    cin >> n;
    vector<int> height(n);
    cout << "Enter the elements of the array: ";
    for (int i = 0; i < n; i++) {
        cin >> height[i];
    }
    int trapped_water = trap(height);
    cout << "Total water trapped: " << trapped_water << endl;
    return 0;
}

```

}

```
C:\Users\Wishnu Priya\OneDri x + v
Enter the number of elements: 12
Enter the elements of the array: 0
1
0
2
1
0
1
3
2
1
2
1
Total water trapped: 6
-----
Process exited after 59.51 seconds with return value 0
Press any key to continue . . . |
```

3. Define a class Ele_Bill in C++ with the following descriptions: Private Members: Cname of type character array Pnumber of type long No_of_Units of type integer Amount of type float. Calc_amount() this number function should calculate the amount as No_of_Units.

```
#include <iostream>
```

```
#include <cstring>
```

```
using namespace std;
```

```
class Ele_Bill {
```

```
private:
```

```
    char Cname[50];
```

```
    long Pnumber;
```

```
    int No_of_Units;
```

```
    float Amount;
```

```
void Calc_amount() {  
    Amount = No_of_Units;  
}
```

public:

```
Ele_Bill(const char* cname, long pnumber, int units) {  
    strcpy(Cname, cname);  
    Pnumber=pnumber;  
    No_of_Units = units;  
    Calc_amount();  
}
```

```
void displayBill() {  
    cout << "Customer Name: " << Cname << endl;  
    cout << "Phone Number: " << Pnumber << endl;  
    cout << "Number of Units: " << No_of_Units << endl;  
    cout << "Amount: Rs." << Amount << endl;  
}
```

```
};
```

```
int main() {  
    char name[50];  
    long pnun;  
    int units;  
    cout << "Enter customer name: ";  
    cin.getline(name, 50);  
    cout << "Enter phone number: ";  
    cin>>pnun;  
    cout << "Enter number of units consumed: ";
```

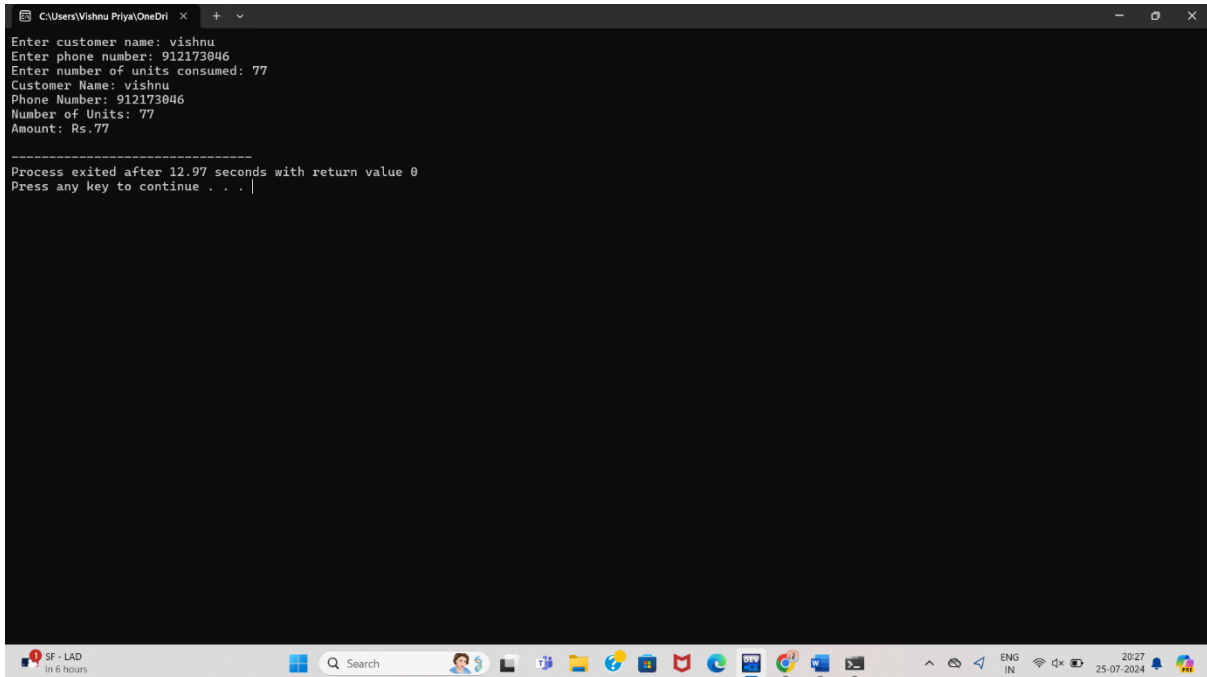
```
    cin >> units;

    Ele_Bill bill(name, pnum, units);

    bill.displayBill();

    return 0;

}
```



```
C:\Users\Vishnu Priya\OneDri >
Enter customer name: vishnu
Enter phone number: 912173046
Enter number of units consumed: 77
Customer Name: vishnu
Phone Number: 912173046
Number of Units: 77
Amount: Rs.77

-----
Process exited after 12.97 seconds with return value 0
Press any key to continue . . .
```