**Slip 20: Sample Solutions and Explanations**

**Q1. School Hierarchy System (Multi-level Inheritance)**

**Approach**

* Define a base class Person with common attributes (name, age).
* Derive a Teacher class from Person that adds subject information.
* Derive an Admin class from Teacher that adds duty information.
* Each class provides its own accept method to input relevant data.

**Code**

#include <iostream>  
using namespace std;  
  
// [Base Class: Person]  
class Person {  
protected: string name; int age;  
public: void accept(){ cin>>name>>age; }  
};  
// [Derived Class: Teacher]  
class Teacher: public Person {  
 string subj;  
public: void accept(){ Person::accept(); cin>>subj; }  
};  
// [Further Derived: Admin]  
class Admin: public Teacher {  
 string duty;  
public: void accept(){ Teacher::accept(); cin>>duty; }  
};  
  
int main() {  
 Admin a; a.accept();  
 return 0;  
}

**Explanation**

* Person is the base class for name and age.
* Teacher inherits from Person and adds subject.
* Admin inherits from Teacher and adds duty.
* Each class's accept method calls its parent to handle inherited data.

**Syntax Definitions**

* **protected**: Members are accessible in the class and its derived classes.
* **public inheritance**: Derived class inherits all public and protected members of the base class.

**Q2. Student Class: Accept, Display, and Search by Subject**

**Approach**

* Create a Student class with attributes: roll number, name, and subject.
* Accept details for n students and store them in a vector.
* Search for students by subject and display their details.

**Code**

#include <iostream>  
#include <vector>  
using namespace std;  
  
// [Student Class Definition]  
class Student {  
 int roll;  
 string name, subject;  
public:  
 void accept() {  
 cout << "Roll: "; cin >> roll;  
 cout << "Name: "; cin >> name;  
 cout << "Subject: "; cin >> subject;  
 }  
 void display() { cout << roll << " " << name << " " << subject << endl; }  
 string getSubject() { return subject; }  
};  
  
int main() {  
 int n;  
 cout << "Number of students: ";  
 cin >> n;  
 vector<Student> students(n);  
 for(auto &s : students) s.accept();  
 string searchSubject;  
 cout << "Enter subject to search: "; cin >> searchSubject;  
 bool found = false;  
 for(auto &s : students) {  
 if(s.getSubject() == searchSubject) {  
 s.display();  
 found = true;  
 }  
 }  
 if(!found) cout << "Student not found." << endl;  
 return 0;  
}

**Explanation**

* The Student class encapsulates student data and provides methods to accept and display it.
* The program reads n students, then searches for students by subject and displays their details if found.
* The getSubject method is used for searching.

**Syntax Definitions**

* **class**: A user-defined type that groups data and functions.
* **vector**: A dynamic array from the C++ Standard Library.

**Q3. School Hierarchy System (Multi-level Inheritance Case Study)**

**Approach**

* Use base class Person, derived class Teacher, and further derived class Admin to model a school hierarchy.
* Each class adds its own specific data and calls parent methods for inherited data.
* Demonstrate object creation and data input in main.

**Code**

#include <iostream>  
using namespace std;  
  
class Person {  
protected: string name; int age;  
public: void accept(){ cin>>name>>age; }  
};  
class Teacher: public Person {  
 string subj;  
public: void accept(){ Person::accept(); cin>>subj; }  
};  
class Admin: public Teacher {  
 string duty;  
public: void accept(){ Teacher::accept(); cin>>duty; }  
};  
  
int main() {  
 Admin a; a.accept();  
 return 0;  
}

**Explanation**

* Shows inheritance chain, each class accepts its own plus parent fields.
* Demonstrates multi-level inheritance for a school system.

**Syntax Definitions**

* **Inheritance**: Mechanism by which one class acquires the properties and behaviors of another class.
* **Method Overriding**: Redefining a base class method in a derived class.