**Slip 19: Sample Solutions and Explanations**

**Q1. Login and Registration System**

**Approach**

* Define a User class with attributes for username and password.
* Implement methods for registration (accepting credentials) and login (verifying credentials).
* Demonstrate registration and login in main.

**Code**

#include <iostream>  
using namespace std;  
  
// [User Class Definition]  
class User {  
 string uname, pwd;  
public:  
 void reg() { cin >> uname >> pwd; }  
 bool login(string u, string p) { return uname==u && pwd==p; }  
};  
  
int main() {  
 User u; u.reg();  
 string u1, p1; cin >> u1 >> p1;  
 if(u.login(u1, p1)) cout << "Success\n"; else cout << "Fail\n";  
 return 0;  
}

**Explanation**

* The User class manages user credentials and provides methods for registration and login.
* The login method checks if the entered username and password match the stored credentials.
* Demonstrates basic authentication logic.

**Syntax Definitions**

* **class**: A user-defined type that groups data and functions.
* **Method**: A function defined inside a class.

**Q2. Student Class: Accept, Display, and Search by Marks Above 75**

**Approach**

* Create a Student class with attributes: roll number, name, and marks.
* Accept details for n students and store them in a vector.
* Display details of students whose marks are above 75.

**Code**

#include <iostream>  
#include <vector>  
using namespace std;  
  
// [Student Class Definition]  
class Student {  
 int roll;  
 string name;  
 double marks;  
public:  
 void accept() {  
 cout << "Roll: "; cin >> roll;  
 cout << "Name: "; cin >> name;  
 cout << "Marks: "; cin >> marks;  
 }  
 void display() { cout << roll << " " << name << " " << marks << endl; }  
 double getMarks() { return marks; }  
};  
  
int main() {  
 int n;  
 cout << "Number of students: ";  
 cin >> n;  
 vector<Student> students(n);  
 for(auto &s : students) s.accept();  
 cout << "Students with marks above 75:\n";  
 for(auto &s : students)  
 if(s.getMarks() > 75) s.display();  
 return 0;  
}

**Explanation**

* The Student class encapsulates student data and provides methods to accept and display it.
* The program reads n students, then displays those with marks above 75.
* The getMarks method is used for filtering.

**Syntax Definitions**

* **vector**: A dynamic array from the C++ Standard Library.
* **auto**: Automatically deduces the type of the variable from its initializer.

**Q3. Login and Registration System (Case Study)**

**Approach**

* Class for user with username/password; methods for registration & login.
* Demonstrate registration and login in main.

**Code**

#include <iostream>  
using namespace std;  
  
class User {  
 string uname, pwd;  
public:  
 void reg() { cin >> uname >> pwd; }  
 bool login(string u, string p) { return uname==u && pwd==p; }  
};  
  
int main() {  
 User u; u.reg();  
 string u1, p1; cin >> u1 >> p1;  
 if(u.login(u1, p1)) cout << "Success\n"; else cout << "Fail\n";  
 return 0;  
}

**Explanation**

* Register and login; mimics minimal credential system.
* The login method checks if the entered username and password match the stored credentials.

**Syntax Definitions**

* **class**: A user-defined type that groups data and functions.
* **Method**: A function defined inside a class.