**Slip 23: Sample Solutions and Explanations**

**Q1. Account/Bank Management (with Static Member and Dynamic Array)**

**Approach**

* Store accounts with static count and total balance.
* Use a vector to store multiple accounts.
* Accept, display, deposit, and withdraw with validation.

**Code**

#include <iostream>  
#include <vector>  
using namespace std;  
  
class Account {  
 static int accCount;  
 static double totalBalance;  
 int accNo; double balance;  
public:  
 Account() { accCount++; }  
 void accept() { cin >> accNo >> balance; totalBalance += balance; }  
 void display() { cout << accNo << " " << balance << endl; }  
 void deposit(double amt) { balance += amt; totalBalance += amt; }  
 void withdraw(double amt) {  
 if(balance >= amt) { balance -= amt; totalBalance -= amt; }  
 else cout << "Insufficient balance\n";  
 }  
 static void showStats() { cout << accCount << " " << totalBalance << endl; }  
};  
int Account::accCount = 0;  
double Account::totalBalance = 0.0;  
  
int main() {  
 int n; cin >> n;  
 vector<Account> ac(n); for(auto &a:ac) a.accept();  
 for(auto &a:ac) a.display();  
 ac[0].deposit(500); ac[1].withdraw(100);  
 Account::showStats();  
 return 0;  
}

**Explanation**

* Static variables track total number and overall balances.
* Methods allow for deposit, withdrawal, and displaying statistics.
* Demonstrates static data members and dynamic arrays.

**Syntax Definitions**

* **static**: Declares a member function or variable that belongs to the class, not to any object.
* **vector**: A dynamic array from the C++ Standard Library.
* **Dynamic Array**: An array whose size can change at runtime (here, implemented using vector).

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

**Approach**

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

**Code**

#include <iostream>  
#include <vector>  
using namespace std;  
  
// [Student Class Definition]  
class Student {  
 int roll;  
 string name;  
 int accNo;  
public:  
 void accept() {  
 cout << "Roll: "; cin >> roll;  
 cout << "Name: "; cin >> name;  
 cout << "Account No: "; cin >> accNo;  
 }  
 void display() { cout << roll << " " << name << " " << accNo << endl; }  
 int getAccNo() { return accNo; }  
};  
  
int main() {  
 int n;  
 cout << "Number of students: ";  
 cin >> n;  
 vector<Student> students(n);  
 for(auto &s : students) s.accept();  
 int searchAccNo;  
 cout << "Enter account number to search: "; cin >> searchAccNo;  
 bool found = false;  
 for(auto &s : students) {  
 if(s.getAccNo() == searchAccNo) {  
 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 a student by account number and displays their details if found.
* The getAccNo 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. Account/Bank Management (with Static Member and Dynamic Array Case Study)**

**Approach**

* Store accounts with static count and total balance.
* Use vector, input and show all accounts.
* Demonstrate deposit and withdrawal operations.

**Code**

#include <iostream>  
#include <vector>  
using namespace std;  
  
class Account {  
 static int accCount;  
 static double totalBalance;  
 int accNo; double balance;  
public:  
 Account() { accCount++; }  
 void accept() { cin >> accNo >> balance; totalBalance += balance; }  
 void display() { cout << accNo << " " << balance << endl; }  
 void deposit(double amt) { balance += amt; totalBalance += amt; }  
 void withdraw(double amt) {  
 if(balance >= amt) { balance -= amt; totalBalance -= amt; }  
 else cout << "Insufficient balance\n";  
 }  
 static void showStats() { cout << accCount << " " << totalBalance << endl; }  
};  
int Account::accCount = 0;  
double Account::totalBalance = 0.0;  
  
int main() {  
 int n; cin >> n;  
 vector<Account> ac(n); for(auto &a:ac) a.accept();  
 for(auto &a:ac) a.display();  
 ac[0].deposit(500); ac[1].withdraw(100);  
 Account::showStats();  
 return 0;  
}

**Explanation**

* Static variables track total number and overall balances.
* Methods allow for deposit, withdrawal, and displaying statistics.
* Demonstrates static data members and dynamic arrays.

**Syntax Definitions**

* **static**: Declares a member function or variable that belongs to the class, not to any object.
* **vector**: A dynamic array from the C++ Standard Library.
* **Dynamic Array**: An array whose size can change at runtime (here, implemented using vector).