**Slip 18: Sample Solutions and Explanations**

**Q1. Shopping Cart System: Add Product, Check Stock, Compute Bill**

**Approach**

* Define a Product class with attributes: name, price, and stock.
* Accept details for multiple products and store them in a vector.
* Allow the user to add products to a cart, check stock, and compute the total bill.

**Code**

#include <iostream>  
#include <vector>  
using namespace std;  
  
// [Product Class Definition]  
class Product {  
 string name;  
 double price;  
 int stock;  
public:  
 void accept() { cin >> name >> price >> stock; }  
 bool available(int qty) { return stock >= qty; }  
 double bill(int qty) { return price \* qty; }  
 void updateStock(int qty) { stock -= qty; }  
 void display() { cout << name << " " << price << " " << stock << endl; }  
};  
  
int main() {  
 int n; cin >> n;  
 vector<Product> prods(n);  
 for(auto &p : prods) p.accept();  
 string query; int qty;  
 cin >> query >> qty;  
 for(auto &p : prods)  
 if(p.available(qty)) {  
 cout << "Bill: " << p.bill(qty) << endl;  
 p.updateStock(qty); break;  
 }  
}

**Explanation**

* The Product class manages product details and provides methods for checking stock, computing bill, and updating inventory.
* The program accepts product details, then processes a purchase by checking stock and updating it if the sale is successful.

**Syntax Definitions**

* **vector**: A dynamic array from the C++ Standard Library.
* **Method**: A function defined inside a class.

**Q2. Student Class: Accept, Display, and Search by Name Starting with 'A'**

**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 names start with 'A'.

**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; }  
 string getName() { return name; }  
};  
  
int main() {  
 int n;  
 cout << "Number of students: ";  
 cin >> n;  
 vector<Student> students(n);  
 for(auto &s : students) s.accept();  
 cout << "Students with names starting with 'A':\n";  
 for(auto &s : students)  
 if(!s.getName().empty() && s.getName()[0] == 'A')  
 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 iterates through the list, displaying only those whose names start with 'A'.
* The check !s.getName().empty() ensures the name is not empty before accessing the first character.

**Syntax Definitions**

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

**Q3. Shopping Cart System (Case Study)**

**Approach**

* Use a class Product for each product (name, price, stock).
* Let user add products to a cart, check stock & compute total bill.

**Code**

#include <iostream>  
#include <vector>  
using namespace std;  
  
class Product {  
 string name;  
 double price;  
 int stock;  
public:  
 void accept() { cin >> name >> price >> stock; }  
 bool available(int qty) { return stock >= qty; }  
 double bill(int qty) { return price \* qty; }  
 void updateStock(int qty) { stock -= qty; }  
 void display() { cout << name << " " << price << " " << stock << endl; }  
};  
  
int main() {  
 int n; cin >> n;  
 vector<Product> prods(n);  
 for(auto &p : prods) p.accept();  
 string query; int qty;  
 cin >> query >> qty;  
 for(auto &p : prods)  
 if(p.available(qty)) {  
 cout << "Bill: " << p.bill(qty) << endl;  
 p.updateStock(qty); break;  
 }  
}

**Explanation**

* Checks for stock, reports bill, decrements inventory after sale.
* Demonstrates basic shopping cart logic for a retail system.

**Syntax Definitions**

* **vector**: A dynamic array from the C++ Standard Library.
* **Method**: A function defined inside a class.