#include <iostream>

#include <fstream>

#include <vector>

#include <ctime>

using namespace std;

struct Item {

    int id; string name; int price; int prepMin;

};

class Menu {

public:

    vector<Item> items;

    Menu() {

        items = {

            {1,"Espresso",100,7},{2,"Cappuccino",150,10},

            {3,"Latte",120,8},{4,"Cold Coffee",80,5},

            {5,"Sandwich",50,6},{6,"Burger",80,8}

        };

    }

    void showMenu() {

        cout << "\n==== MENU ====\n";

        for (auto &it: items)

            cout << it.id << ". " << it.name

                 << " - Rs." << it.price

                 << " (" << it.prepMin << " min)\n";

    }

    Item\* getItem(int id) {

        for (auto &it: items) if (it.id==id) return &it;

        return nullptr;

    }

};

class Billing: public Menu {

    static int token;

public:

    void takeOrder() {

        token++;

        vector<pair<Item\*,int>> order;

        char more='y'; int total=0, prep=0;

        cout << "\nStarting Order (Token " << token << ")\n";

        do {

            showMenu();

            int ch,qty; cout<<"Enter item id: ";cin>>ch;

            Item\* it=getItem(ch);

            if(!it){cout<<"Invalid!\n"; continue;}

            cout<<"Enter qty: ";cin>>qty;

            order.push\_back({it,qty});

            total+=it->price\*qty;

            prep+=it->prepMin\*qty;

            cout<<"More items? (y/n): ";cin>>more;

        }while(more=='y'||more=='Y');

        // Show original prep time

        int original\_prep = prep;

        cout << "\nOriginal estimated prep time: " << original\_prep << " min.\n";

        // Ask for adjustment

        cout << "Do you want to adjust prep time? (y/n): ";

        char adj; cin >> adj;

        int adjusted\_prep = original\_prep;

        if(adj=='y' || adj=='Y'){

            int extra;

            cout<<"Enter minutes to add (use negative to reduce): ";

            cin>>extra;

            adjusted\_prep = original\_prep + extra;

            if(adjusted\_prep < 0) adjusted\_prep = 0;  // no negative prep time

        }

        // Calculate ready time AFTER adjustment

        time\_t now = time(nullptr);

        time\_t ready = now + adjusted\_prep \* 60;

        // Save bill (include original & adjusted for clarity)

        ofstream fout("bill.txt", ios::app);

        fout << "\n=== BILL ===\nToken " << token << "\n";

        for (auto &p : order)

            fout << p.first->name << " x" << p.second << " = Rs."

                 << p.first->price \* p.second << "\n";

        fout << "Total: Rs." << total << "\n";

        fout << "Original Prep time: " << original\_prep << " min\n";

        if (adjusted\_prep != original\_prep)

            fout << "Adjusted Prep time: " << adjusted\_prep << " min\n";

        fout << "Ready by: " << ctime(&ready) << "\n";

        fout.close();

        // Console output

        cout << "\nBill saved!\nTotal Rs." << total << "\n";

        if (adjusted\_prep == original\_prep) {

            cout << "Order ready in " << adjusted\_prep << " min at " << ctime(&ready) << "\n";

        } else {

            cout << "Original prep: " << original\_prep << " min\n";

            cout << "Adjusted prep: " << adjusted\_prep << " min\n";

            cout << "Order ready at " << ctime(&ready) << "\n";

        }

    }

};

int Billing::token = 0;

int main() {

    Billing b; int ch;

    do {

        cout<<"\n1.New Order\n2.Exit\nChoice: ";

        cin>>ch;

        if(ch==1) b.takeOrder();

    }while(ch!=2);

}