**U18CO018**

**Shubham Shekhaliya**

1->

#include <bits/stdc++.h>

using namespace std;

int main() {

    cout << "Enter the number of model in tables: \n";

    map<string, pair<int, int> > mp;

    int n;

    cin >> n;

    while (n--) {

        string name;

        cout << "Enter the model name: ";

        cin >> name;

        int total\_unit, cost;

        cout << "Enter the total number of units sold: ";

        cin >> total\_unit;

        cout << "Enter the cost of model: ";

        cin >> cost;

        mp[name] = {total\_unit, cost};

    }

    while (true) {

        cout << "1 Enter a model name\n2 Exit\n";

        int choice;

        cin >> choice;

        if (choice == 1) {

            cout << "Enter model name: ";

            string name;

            cin >> name;

            if (mp.find(name) == mp.end()) {

                cout << "Model not found !!\n";

            } else {

                pair<int, int> p = mp[name];

                cout << "total units sold: " << p.first << endl;

                cout << "cost of model: " << p.second << endl;

                cout << "total cost: " << p.first \* p.second << endl;

            }

        } else {

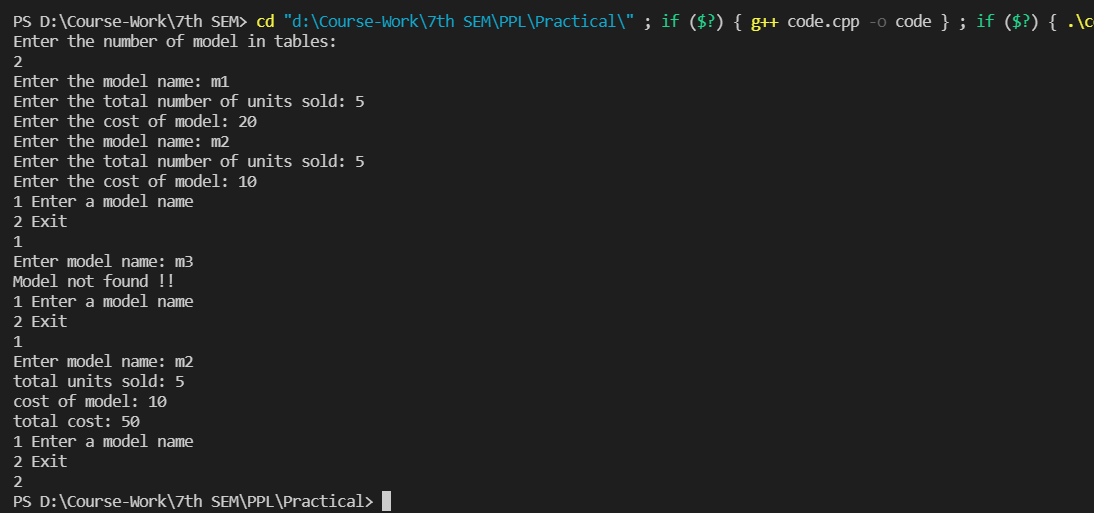
            break;

        }

    }

    return 0;

}

****

**2->**

patient(p21, shubham, address(shlimar\_park, delhi, 395001), [sub(d1, flu), sub(d2,commoncold)] ).

patient(p22, darshan, address(jbnagar, delhi, 395004), [sub(d2, commoncold), sub(d3,chickenpox)] ).

patient(p23, sagar, address(svnit, surat, 395007), [sub(d1, chickenpox), sub(d4,measles)] ).

% flattens list of lists into lists

flatten([],[]).

flatten([H|T],W):-flatten(T,W1), append(W1,H,W).

% remove duplicates from List1 and form List2

remove\_duplicates([],[]).

remove\_duplicates([H|T], [H|T2]):- not(member(H,T)), remove\_duplicates(T,T2).

remove\_duplicates([H|T], L2):- member(H,T), remove\_duplicates(T,L2).

%q1

total\_diseases:-patient(\_,X,\_,Y), length(Y,L), write(X), write(": "), write(L).

%q2

name\_and\_zip:-patient(\_,X,address(\_,\_,Zip),\_),write(X), write(" has zip code: "), write(Zip).

%q3

citydelhi:- patient(PID,Name,address(\_,delhi, \_),\_), write(PID), write(" has a name "), write(Name), nl.

%q4

contains\_doc1([]):-fail.

contains\_doc1([treatment(d1,\_)|\_]):-!.

contains\_doc1([\_|T]):-contains\_doc1(T).

patients\_doc1:-patient(\_,Name,\_,X), contains\_doc1(X), write(Name), nl.

%q5

contains\_cold([]):-fail.

contains\_cold([sub(\_,commoncold)|\_]):-!.

contains\_cold([\_|T]):-contains\_cold(T).

common\_cold\_patient:- patient(id,\_,\_,X), contains\_cold(X), write(id), nl.

%q6

cityaddress:- patient(\_,\_,address(Building, \_, Code),\_), write("("), write(Building),

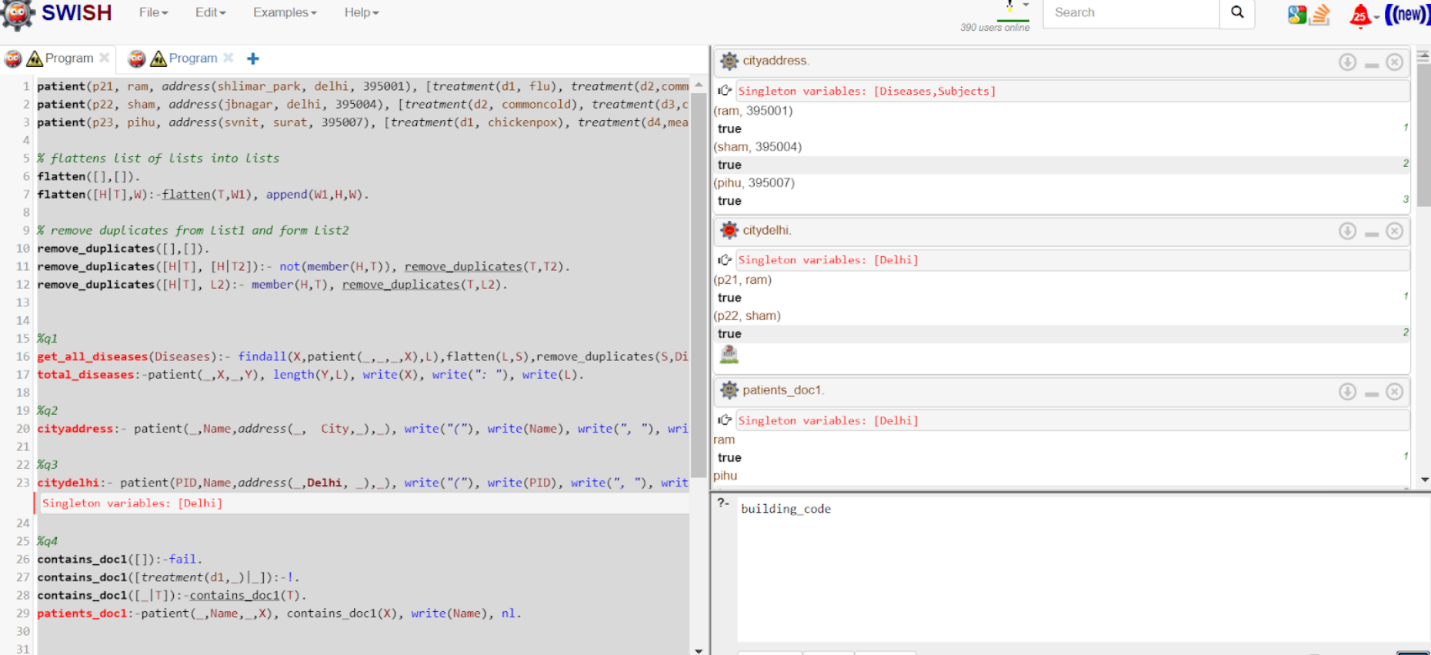
write(", "), write(Code), write(")"), nl.

%q7

extract\_doctor([],[]).

extract\_doctor([treatment(T,\_)|Rest], [T|Tail]):-extract\_doctor(Rest,Tail).

doctor\_for\_patients:- patient(\_,Name,\_,Y),extract\_doctor(Y,Z), write(Name), write(": "), write(Z), nl.

****