**Problem Name:** Apply Discount Every n order

**Topics:** Array, Hash Table, Design

**Companies:** Facebook

**Level:** Medium

**Language:** C++

**Problem Statement:** There is a supermarket that is frequented by many customers. The products sold at the supermarket are represented as two parallel integer arrays products and prices, where the ith product has an ID of products[i] and a price of prices[i].

When a customer is paying, their bill is represented as two parallel integer arrays product and amount, where the jth product they purchased has an ID of product[j], and amount[j] is how much of the product they bought. Their subtotal is calculated as the sum of each amount[j] \* (price of the jth product).

The supermarket decided to have a sale. Every nth customer paying for their groceries will be given a **percentage discount**. The discount amount is given by discount, where they will be given discount percent off their subtotal. More formally, if their subtotal is bill, then they would actually pay bill \* ((100 - discount) / 100).

Implement the Cashier class:

* Cashier(int n, int discount, int[] products, int[] prices) Initializes the object with n, the discount, and the products and their prices.
* double getBill(int[] product, int[] amount) Returns the final total of the bill with the discount applied (if any). Answers within 10-5 of the actual value will be accepted.

**Input Format:**

First line of the in contain two space separated integers nth value and discount.

Second line contains integer value representing size of the product z.

Next line contains z space separated integer representing product id.

Next line contains z space separated integer representing product price.

Next line contains no. of queries q.

Next line contains no of groceries to buy y.

Next y line contains two space separated integers representing Product index and

quantity.

**Output Format:** Print q lines representing total bill after discount.

**Constraints:**

* 1 <= n <= 104
* 0 <= discount <= 100
* 1 <= products.length <= 200
* prices.length == products.length
* 1 <= products[i] <= 200
* 1 <= prices[i] <= 1000
* The elements in products are **unique**.
* 1 <= product.length <= products.length
* amount.length == product.length
* product[j] exists in products.
* 1 <= amount[j] <= 1000
* The elements of product are **unique**.
* At most 1000 calls will be made to getBill.
* Answers within 10-5 of the actual value will be accepted.

**Examples:**

**Input:**

3 50

7

1 2 3 4 5 6 7

100 200 300 400 300 200 100

1

2

1 2

1 2

**Solution:**

**Explanation:** Make an unordered map tags to map all index value with price and do simple math problem to calculate cost and make a variable cnt and increment at each calculation and if cnt equals to nth value apply discount formula.

**Code:**

#include<bits/stdc++.h>

using namespace std;

class Cashier {

public:

    // 'count' will store the serial no. of a customer

    int count,n,discount;

    unordered\_map<int,int>m;

    Cashier(int n, int discount, vector<int>& products, vector<int>& prices) {

        // initialise the values

        count=0;

        this->n=n;

        this->discount=discount;

        // populate the unordered map

        for(int i=0;i<products.size();i++) {

            m[products[i]]=prices[i];

        }

    }

    double getBill(vector<int> product, vector<int> amount) {

        // as a new customer comes

        count++;

        // 'total' will be the actual bill

        double total=0;

        // calculate the total bill

        for(int i=0;i<product.size();i++) {

            total+=(double)(amount[i]\*m[product[i]]);

        }

        // if he/she is the nth customer then give the discount

        if(count%n==0) {

            total=total\*((100-(double)discount)/100);

        }

        // finally return the bill

        return total;

    }

};

int main(){

    int n, discount;

    cin>>n>>discount;

    vector<int> products, prices;

    int x;

    cin>>x;

    for(int i=0; i<x; i++){

        int temp;

        cin>>temp;

        products.push\_back(temp);

    }

    for(int i=0; i<x; i++){

        int temp;

        cin>>temp;

        prices.push\_back(temp);

    }

    Cashier \*obj = new Cashier(n, discount, products, prices);

    int q;

    cin>>q;

    while(q--){

        vector<int> prod, amt;

        cin>>x;

        for(int i=0; i<x; i++){

            int temp;

            cin>>temp;

            prod.push\_back(temp);

        }

        for(int i=0; i<x; i++){

            int temp;

            cin>>temp;

            amt.push\_back(temp);

        }

        double ans = obj->getBill(prod, amt);

        cout<<ans<<endl;

    }

}

**Time Complexity**: O(N)

**Space Complexity:** O(N)

**Optimized Solution:**

**Explanation:** Make a map tags to map all index value with price and do simple math problem to calculate cost and make a variable cnt and increment at each calculation and if cnt equals to nth value apply discount formula.

**Code:**

#include<bits/stdc++.h>

using namespace std;

class Cashier {

public:

    map<int, int> tags;

    int cnt, mod;

    double pct;

    Cashier(int n, int discount, vector<int>& products, vector<int>& prices) {

        cnt = 0, mod = n, pct = (100.0 - discount) / 100.0;

        for(int i=0; i<products.size(); ++i){

            tags[products[i]] = prices[i];

        }

    }

    double getBill(vector<int> product, vector<int> amount) {

        cnt++;

        double ans(0.0);

        for(int i=0; i<product.size(); ++i){

            ans += tags[product[i]] \* amount[i];

        }

        return cnt % mod ? ans : ans \* pct;

    }

};

int main(){

    int n, discount;

    cin>>n>>discount;

    vector<int> products, prices;

    int x;

    cin>>x;

    for(int i=0; i<x; i++){

        int temp;

        cin>>temp;

        products.push\_back(temp);

    }

    for(int i=0; i<x; i++){

        int temp;

        cin>>temp;

        prices.push\_back(temp);

    }

    Cashier \*obj = new Cashier(n, discount, products, prices);

    int q;

    cin>>q;

    while(q--){

        vector<int> prod, amt;

        cin>>x;

        for(int i=0; i<x; i++){

            int temp;

            cin>>temp;

            prod.push\_back(temp);

        }

        for(int i=0; i<x; i++){

            int temp;

            cin>>temp;

            amt.push\_back(temp);

        }

        double ans = obj->getBill(prod, amt);

        cout<<ans<<endl;

    }

}

**Time Complexity**: O(n)

**Space Complexity:** O(n)