Name: Patel Manan Maheshkumar
Roll No: CE-111
PPS-2(Lab-9)

1. Play with coins

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

int main() {
   int itemCount;
   cin >> itemCount;
   vector<int> cost(itemCount);
   vector<int> quantity(itemCount);
   vector<int> resultCost;
   vector<int> resultQuantity;
   int amount;
   int remainingAmount;

   for(int i = 0; i < itemCount; i++)
      cin >> cost[i] >> quantity[i];

   cin >> amount;
   remainingAmount = amount;

   for(int i = 0; i < itemCount - 1; i++)
      for(int j = i + 1; j < itemCount; j++)
         if(cost[i] < cost[j]) {
            int temp1 = quantity[i];
            quantity[i] = quantity[j];
            quantity[j] = temp1;

            int temp2 = cost[i];
            cost[i] = cost[j];
            cost[j] = temp2;
         }
```

```
    for(int i = 0; i < itemCount; i++) {
        int currentCost = cost.at(i);
        int currentQuantity = quantity.at(i);
        int maxPossible = remainingAmount / currentCost;

        if(currentQuantity >= maxPossible && maxPossible != 0) {
            resultCost.push_back(currentCost);
            resultQuantity.push_back(maxPossible);
            remainingAmount -= (maxPossible * currentCost);
        }
        else if(currentQuantity <= maxPossible && maxPossible != 0) {
            resultCost.push_back(currentCost);
            resultQuantity.push_back(currentQuantity);
            remainingAmount -= (currentQuantity * currentCost);
        }

        if(remainingAmount == 0) break;
    }

    if(remainingAmount == 0) {
        cout << resultCost.size() << endl;
        for(int i = 0; i < (int)resultCost.size(); i++)
            cout << resultCost[i] << " " << resultQuantity[i] << endl;
    }
    else
        cout << "-1";

    return 0;
}
```

2. Find the closest to the given number

```
#include <vector>
#include <iostream>
using namespace std;
int binarysearch(vector<int> &v1,int key,int low,int high,int n)
{

    while(low <= high) {
        int mid = low + (high - low) / 2;
        if(v1[mid] == key)
            return v1[mid];
        if(v1[mid] < key)
            low = mid + 1;
```

```
        else if(v1[mid] > key)
            high = mid - 1;
    }
        if(low==0)
            return v1[low];
        else if(low==(n-1))
            return v1[n-1];

    int a=v1[low-1],b=v1[low];
    return (abs(a-key)>abs(b-key))?b:a;
}

int main() {
    int n;
    cin >> n;
    vector<int> v1(n);
    vector<int> v2;
    for(int i=0;i<n;i++)
        cin >> v1[i];
    int temp;
    while(cin >> temp)
        v2.push_back(temp);
    for(int i=0;i<(int)v2.size();i++)
       cout << binarysearch(v1,v2.at(i),0,n-1,n) << " ";
    cout << endl;
    return 0;
}
```