Contents

[Bài 4.1: Đảo ngược một danh sách liên kết đơn 2](#_Toc72795247)

[Bài 4.2: Tính diện tích tam giác 5](#_Toc72795248)

[Bài 4.3: Tính tích có hướng của 2 vector 7](#_Toc72795249)

[Bài 4.4: Thao tác với vector 9](#_Toc72795250)

[Bài 4.5: DFS 13](#_Toc72795251)

[Bài 4.6: BFS 16](#_Toc72795252)

[Bài 4.7: Set 18](#_Toc72795253)

[4.8 Map 21](#_Toc72795254)

[4.9 Dijitra 23](#_Toc72795255)

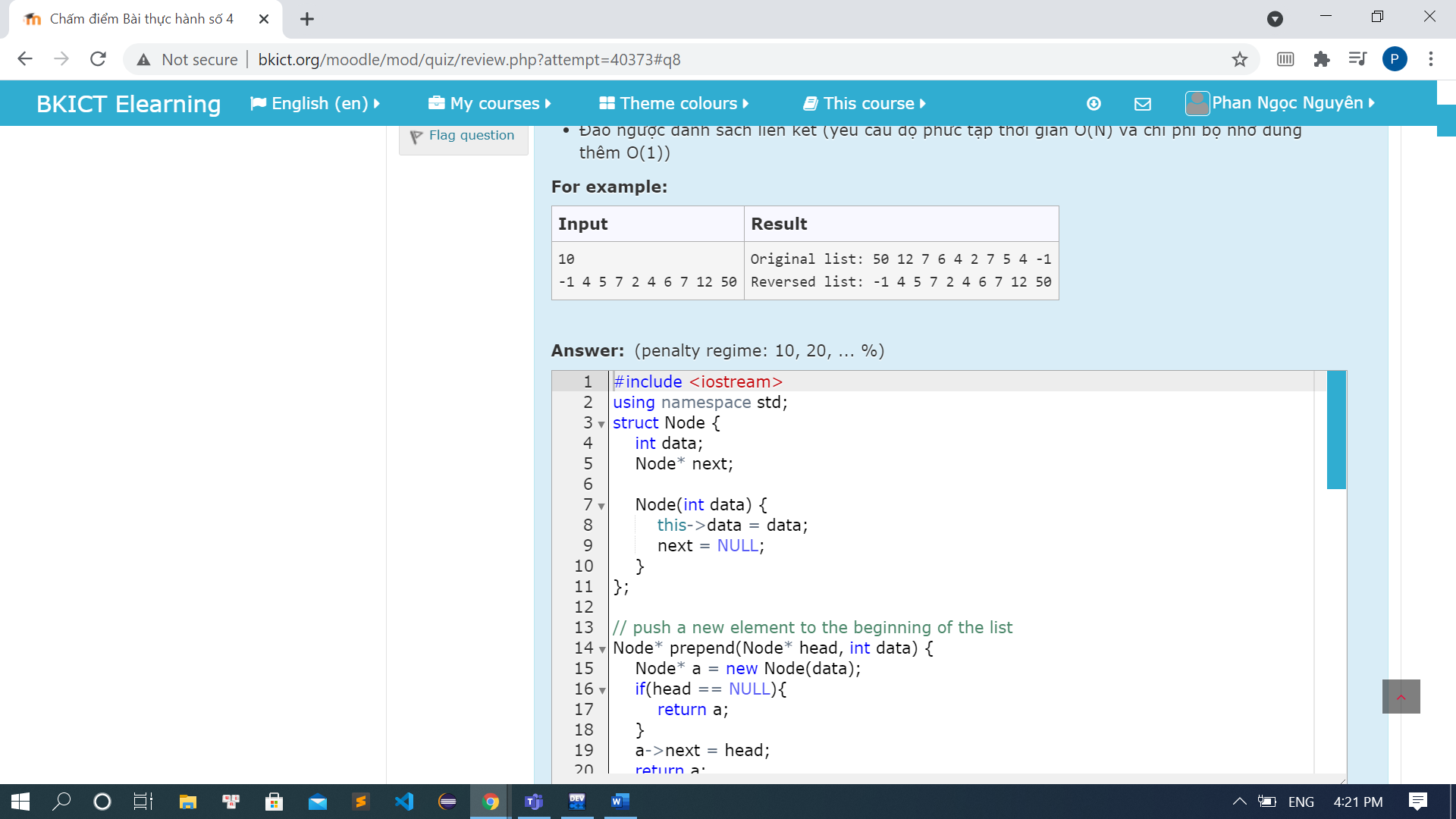
[4.10 Search Engine 25](#_Toc72795256)

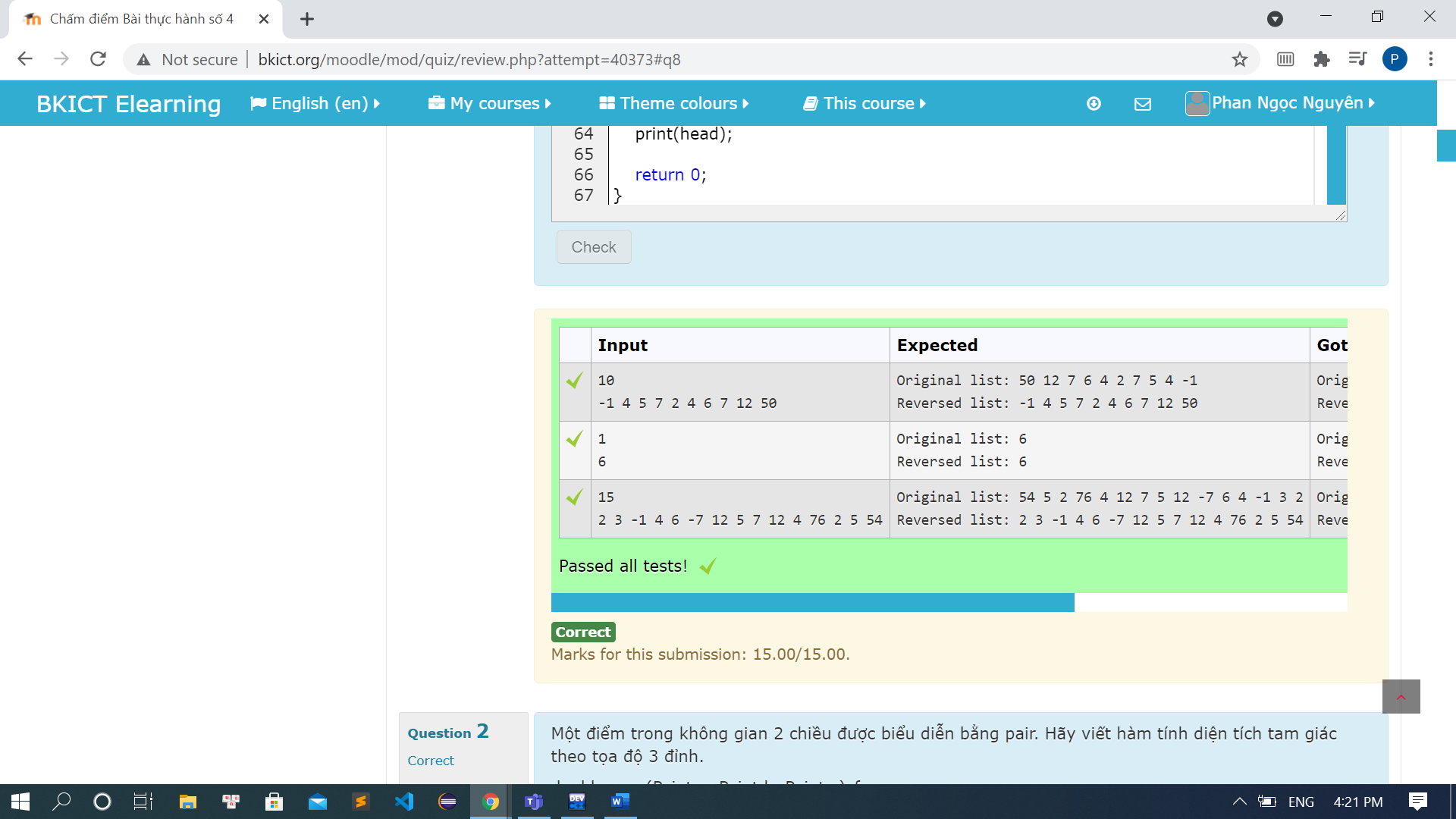
[4.11 Bảo vệ lâu đài: 29](#_Toc72795257)

[4.12 Lược đồ 31](#_Toc72795258)

[4.13 Đếm xâu con 33](#_Toc72795259)

### Bài 4.1: Đảo ngược một danh sách liên kết đơn





Code:

#include <iostream>

using namespace std;

struct Node {

int data;

Node\* next;

Node(int data) {

this->data = data;

next = NULL;

}

};

// push a new element to the beginning of the list

Node\* prepend(Node\* head, int data) {

Node\* a = new Node(data);

if(head == NULL){

return a;

}

a->next = head;

return a;

}

// print the list content on a line

void print(Node\* head) {

Node\* h = head;

while(h!= NULL){

cout << h->data << " ";

h = h->next;

}

cout << endl;

}

// return the new head of the reversed list

Node\* reverse(Node\* head) {

Node\* prev = NULL;

Node\* current = head;

Node\* next = NULL;

while(current != NULL){

next = current->next;

current->next = prev;

prev = current;

current = next;

}

head = prev;

return head;

}

int main() {

cout << "Phan Ngoc Nguyen - 20194134" << endl;

int n, u;

cin >> n;

Node\* head = NULL;

for (int i = 0; i < n; ++i){

cin >> u;

head = prepend(head, u);

}

cout << "Original list: ";

print(head);

head = reverse(head);

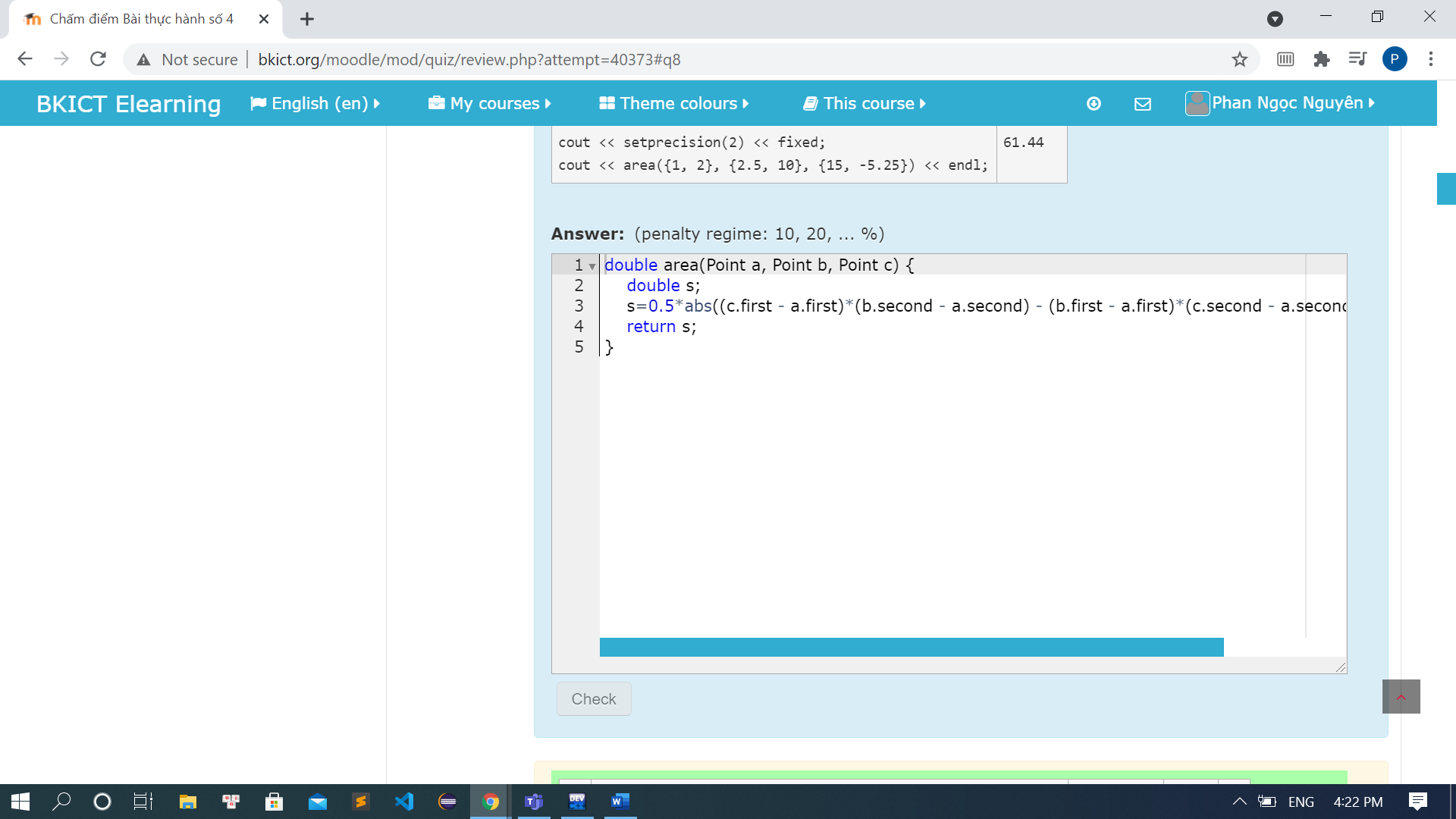
cout << "Reversed list: ";

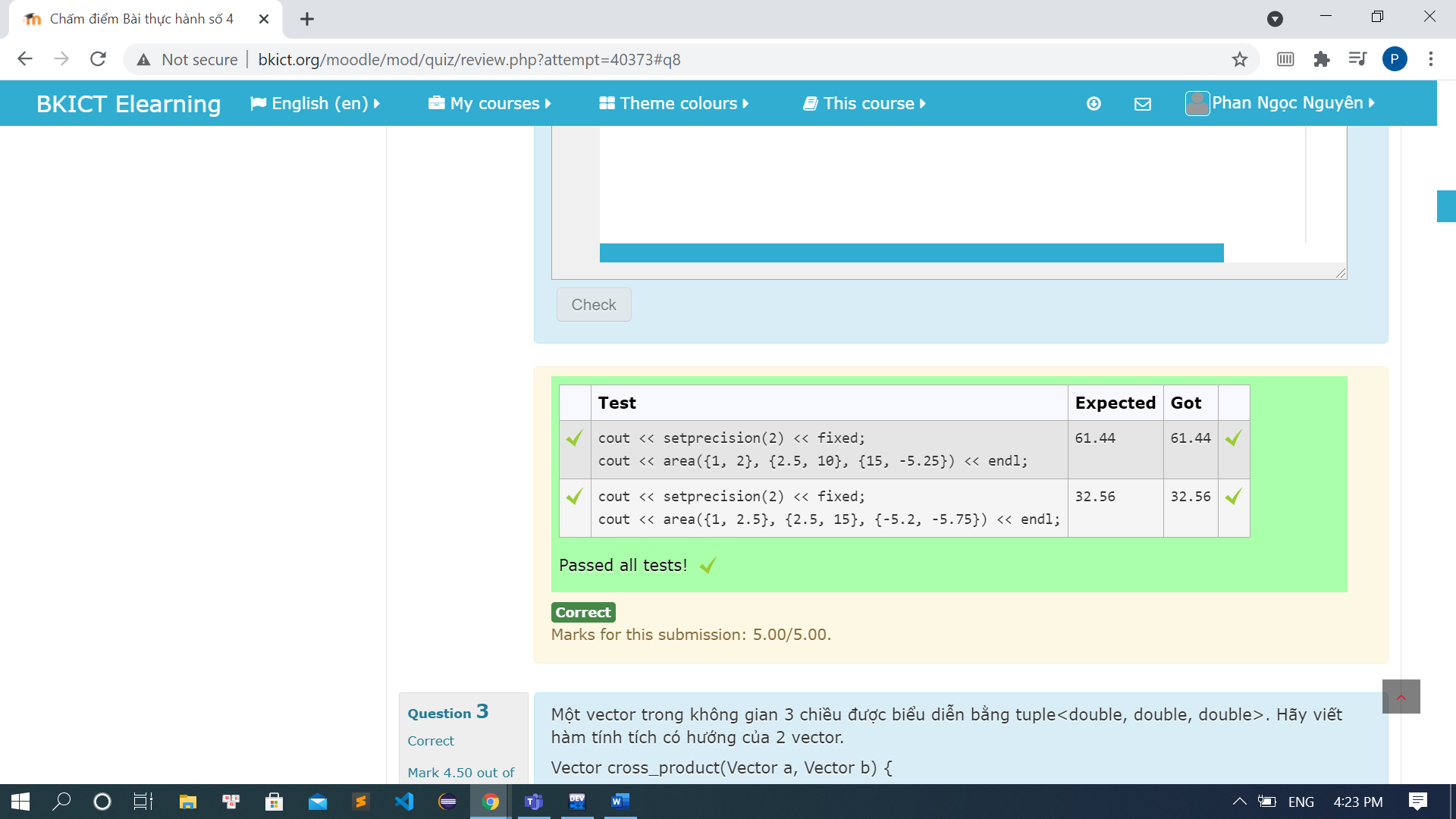
print(head);

return 0;

}

### Bài 4.2: Tính diện tích tam giác





Code:

#include <iostream>

#include <cmath>

#include <iomanip>

#include <utility>

using namespace std;

using Point = pair<double, double>;

double area(Point a, Point b, Point c) {

double s;

s=0.5\*abs((c.first - a.first)\*(b.second - a.second) - (b.first - a.first)\*(c.second - a.second));

return s;

}

int main() {

cout << "Phan Ngoc Nguyen - 20194134" << endl;

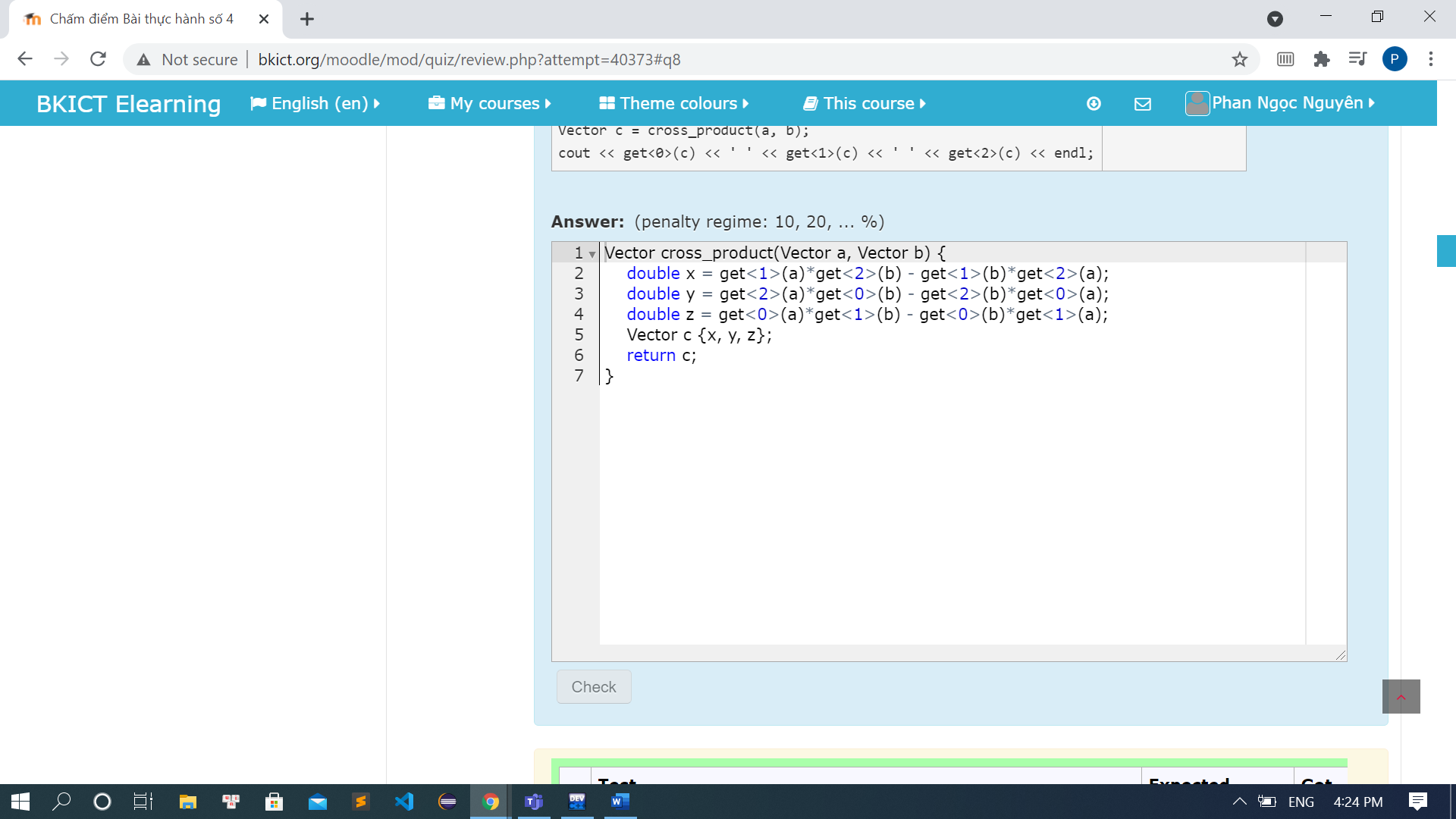
cout << setprecision(2) << fixed;

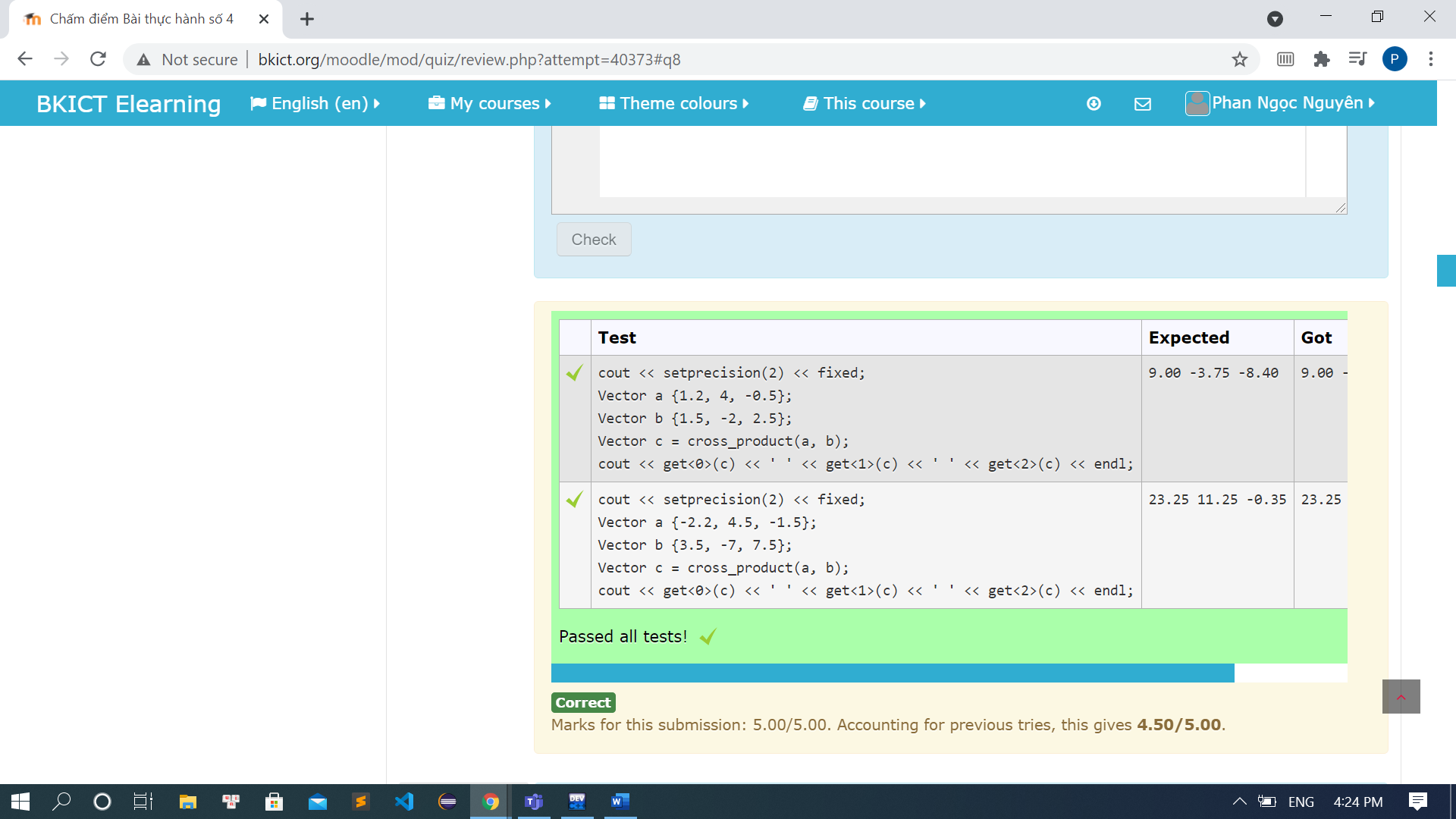
cout << area({1, 2}, {2.5, 10}, {15, -5.25}) << endl;

return 0;

}

### Bài 4.3: Tính tích có hướng của 2 vector





Code:

#include <iostream>

#include <cmath>

#include <iomanip>

using namespace std;

using Vector = tuple<double, double, double>;

Vector cross\_product(Vector a, Vector b) {

double x = get<1>(a)\*get<2>(b) - get<1>(b)\*get<2>(a);

double y = get<2>(a)\*get<0>(b) - get<2>(b)\*get<0>(a);

double z = get<0>(a)\*get<1>(b) - get<0>(b)\*get<1>(a);

Vector c {x, y, z};

return c;

}

int main() {

cout << "Phan Ngoc Nguyen - 20194134" << endl;

cout << setprecision(2) << fixed;

Vector a {1.2, 4, -0.5};

Vector b {1.5, -2, 2.5};

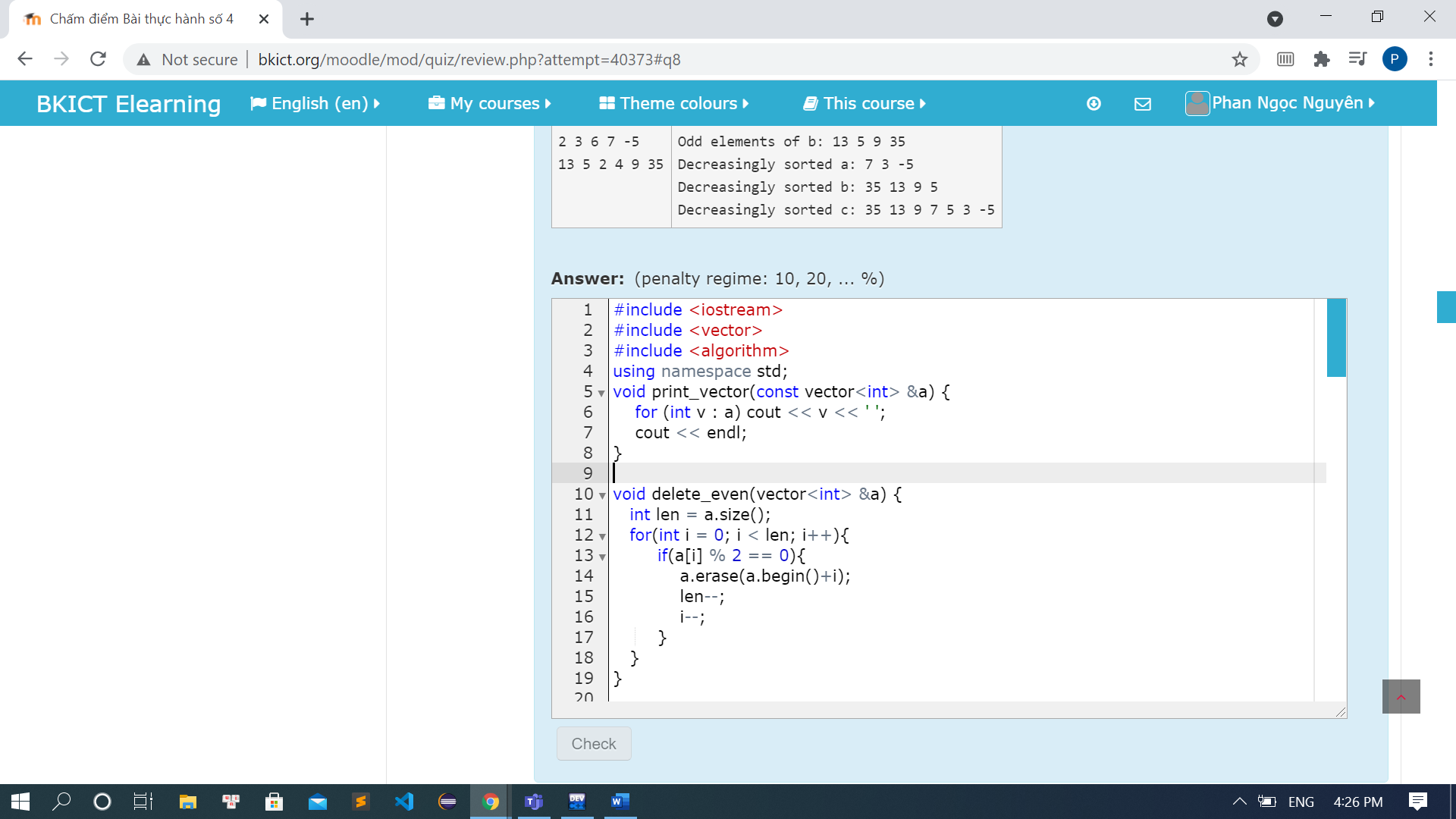
Vector c = cross\_product(a, b);

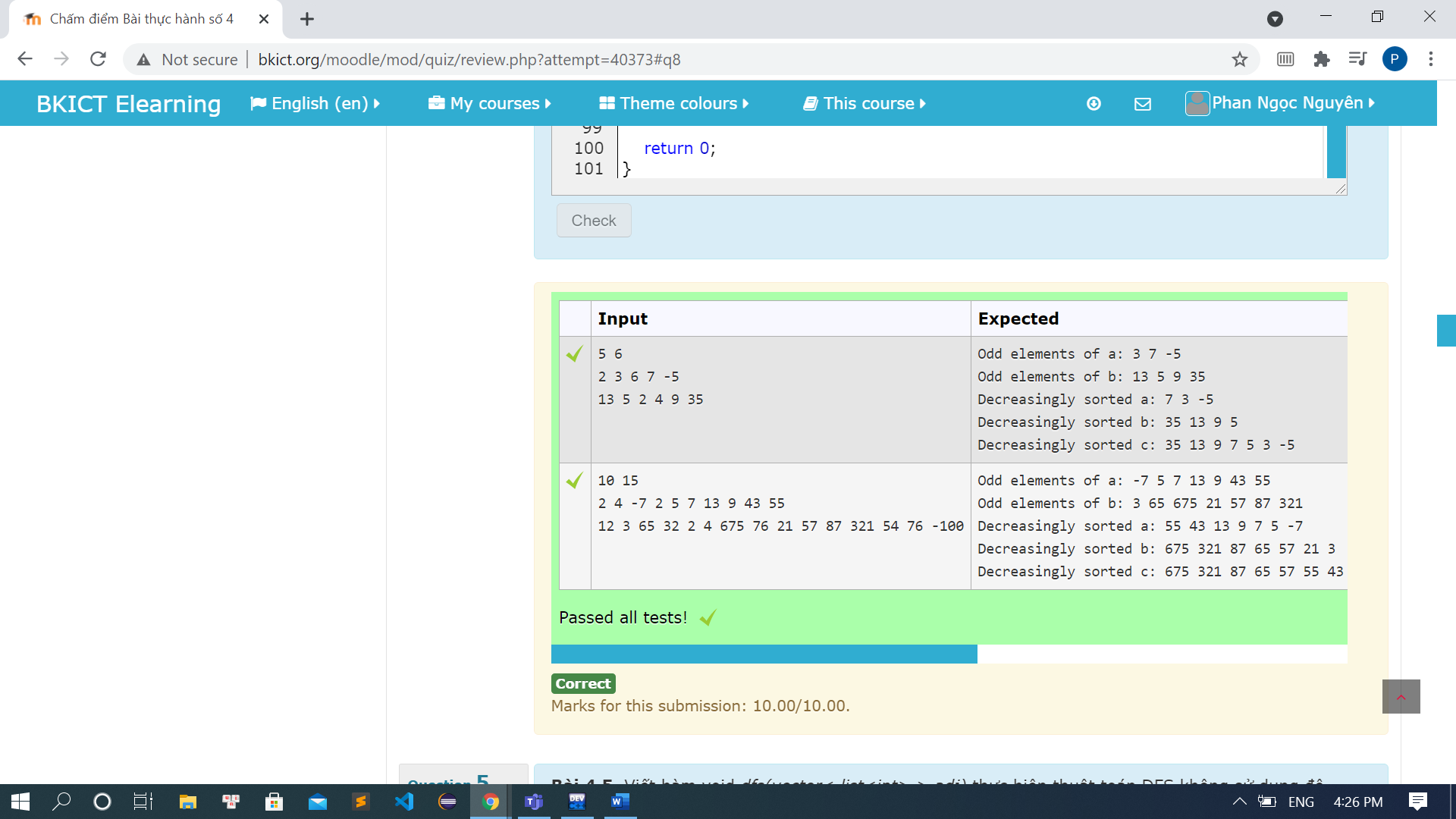
cout << get<0>(c) << ' ' << get<1>(c) << ' ' << get<2>(c) << endl;

return 0;

}

### Bài 4.4: Thao tác với vector





Code:

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

void print\_vector(const vector<int> &a) {

for (int v : a) cout << v << ' ';

cout << endl;

}

void delete\_even(vector<int> &a) {

a.erase(remove\_if(a.begin(), a.end(), [] (int x ){

return x % 2 == 0;

}), a.end());

}

void sort\_decrease(vector<int> &a) {

sort(a.rbegin(), a.rend());

}

vector<int> merge\_vectors(const vector<int> &a, const vector<int> &b) {

// int lena = a.size();

// int lenb = b.size();

// std::vector<int> c;

// int i = 0, j = 0;

// while(i < lena && j < lenb){

// if(a[i] > b[j]){

// c.push\_back(a[i]);

// i++;

// }else if(b[j] > a[i]){

// c.push\_back(b[j]);

// j++;

// }else{

// c.push\_back(a[i]);

// c.push\_back(b[j]);

// i++;

// j++;

// }

// }

// if(i < lena){

// while(i < lena){

// c.push\_back(a[i]);

// i++;

// }

//

// }else if(j < lenb){

// while(j < lenb){

// c.push\_back(b[j]);

// j++;

// }

// }

// return c;

vector<int> c;

merge(a.begin(), a.end(), b.begin(), b.end(), back\_inserter(c), greater<int>());

return c;

}

int main() {

cout << "Phan Ngoc Nguyen - 20194134" << endl;

int m, n, u;

std::vector<int> a, b;

std::cin >> m >> n;

for(int i = 0; i < m; i++){

std:: cin >> u;

a.push\_back(u);

}

for(int i = 0; i < n; i++){

std:: cin >> u;

b.push\_back(u);

}

delete\_even(a);

cout << "Odd elements of a: ";

print\_vector(a);

delete\_even(b);

cout << "Odd elements of b: ";

print\_vector(b);

sort\_decrease(a);

cout << "Decreasingly sorted a: ";

print\_vector(a);

sort\_decrease(b);

cout << "Decreasingly sorted b: ";

print\_vector(b);

vector<int> c = merge\_vectors(a, b);

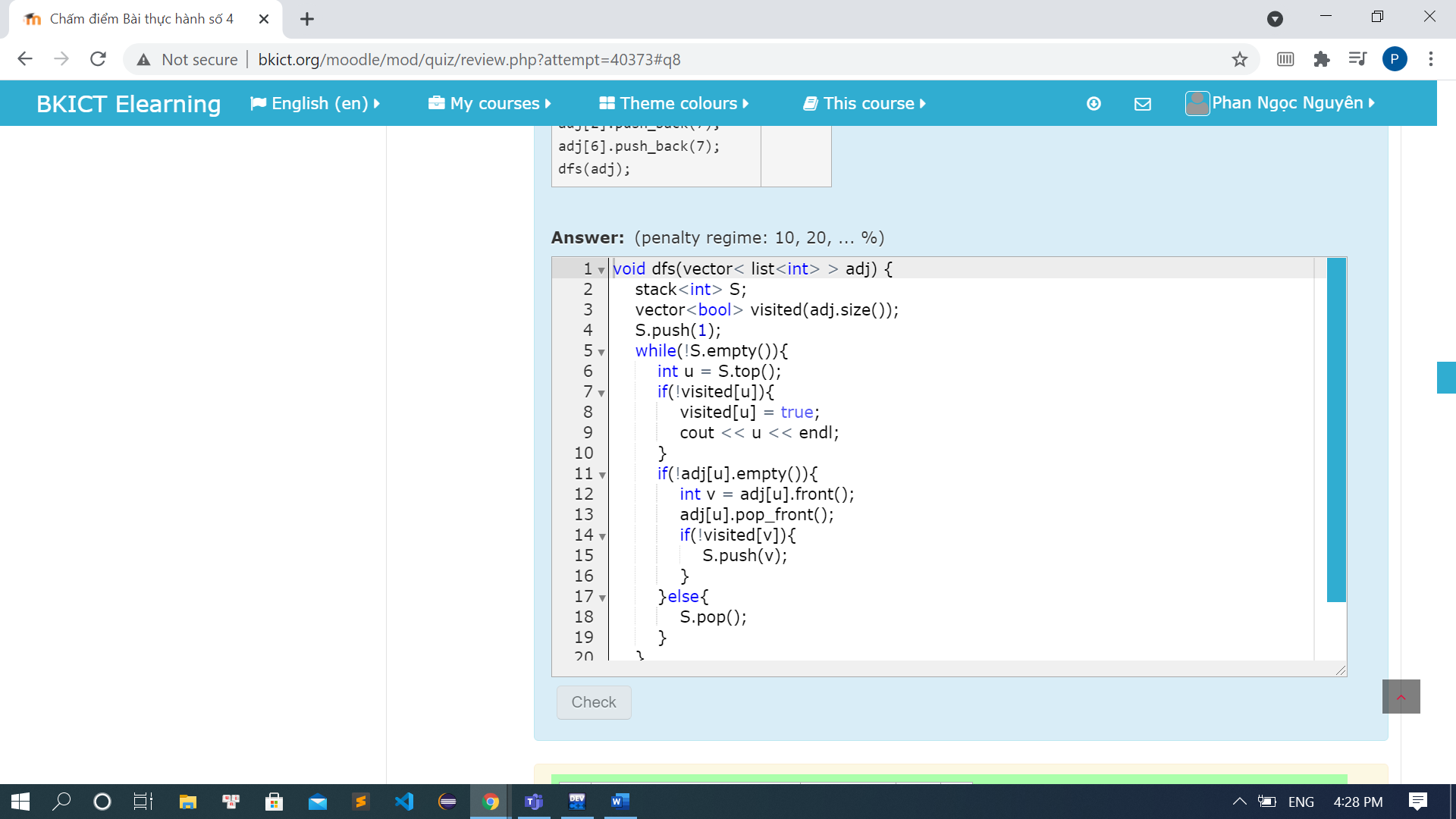
cout << "Decreasingly sorted c: ";

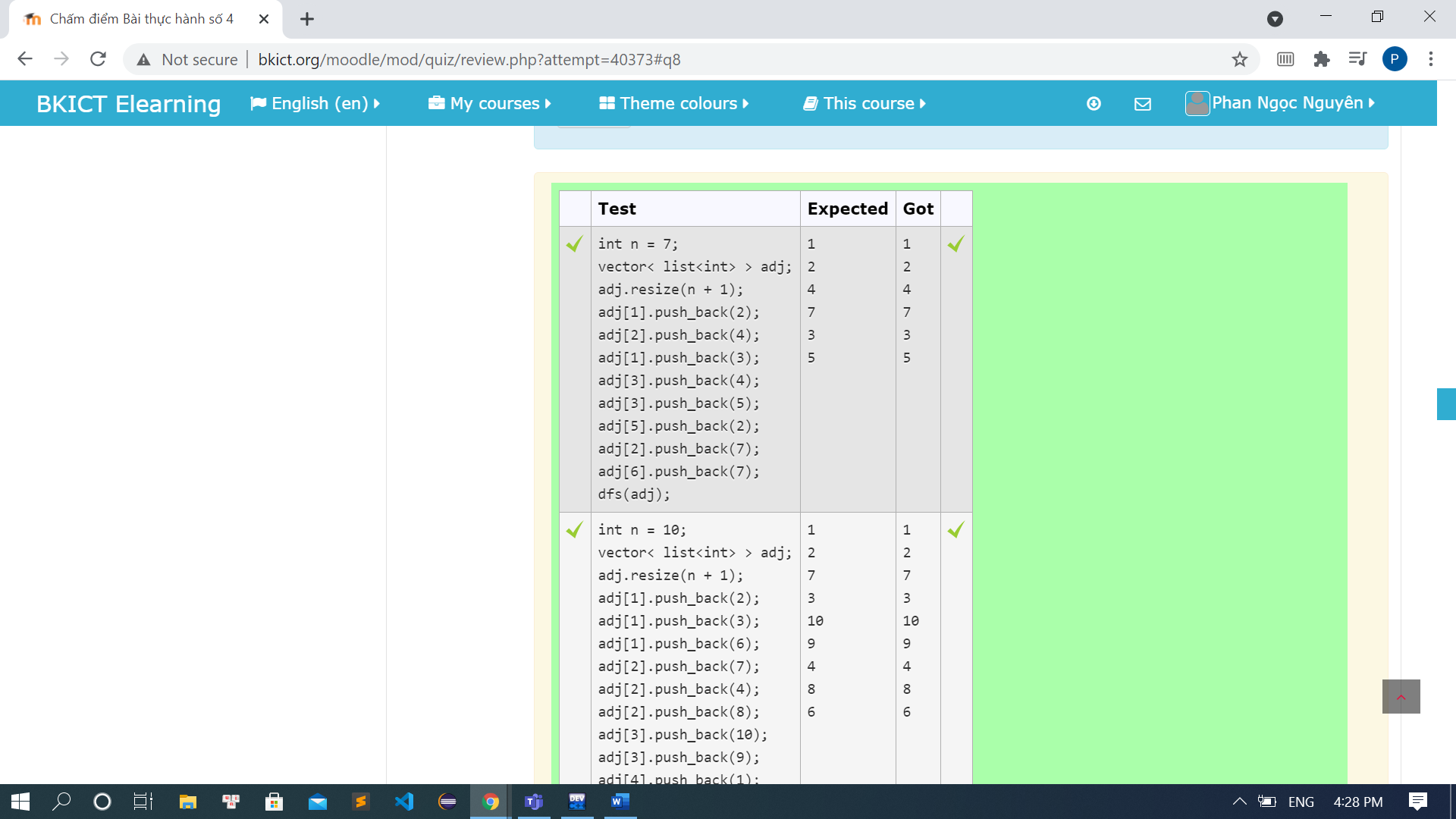
print\_vector(c);

return 0;

}

### Bài 4.5: DFS





Code:

#include<bits/stdc++.h>

using namespace std;

void dfs(vector< list<int> > adj) {

stack<int> S;

vector<bool> visited(adj.size());

S.push(1);

while(!S.empty()){

int u = S.top();

if(!visited[u]){

visited[u] = true;

cout << u << endl;

}

if(!adj[u].empty()){

int v = adj[u].front();

adj[u].pop\_front();

if(!visited[v]){

S.push(v);

}

}else{

S.pop();

}

}

}

int main(){

cout << "Phan Ngoc Nguyen - 20194134" << endl;

int n = 7;

vector< list<int> > adj;

adj.resize(n + 1);

adj[1].push\_back(2);

adj[2].push\_back(4);

adj[1].push\_back(3);

adj[3].push\_back(4);

adj[3].push\_back(5);

adj[5].push\_back(2);

adj[2].push\_back(7);

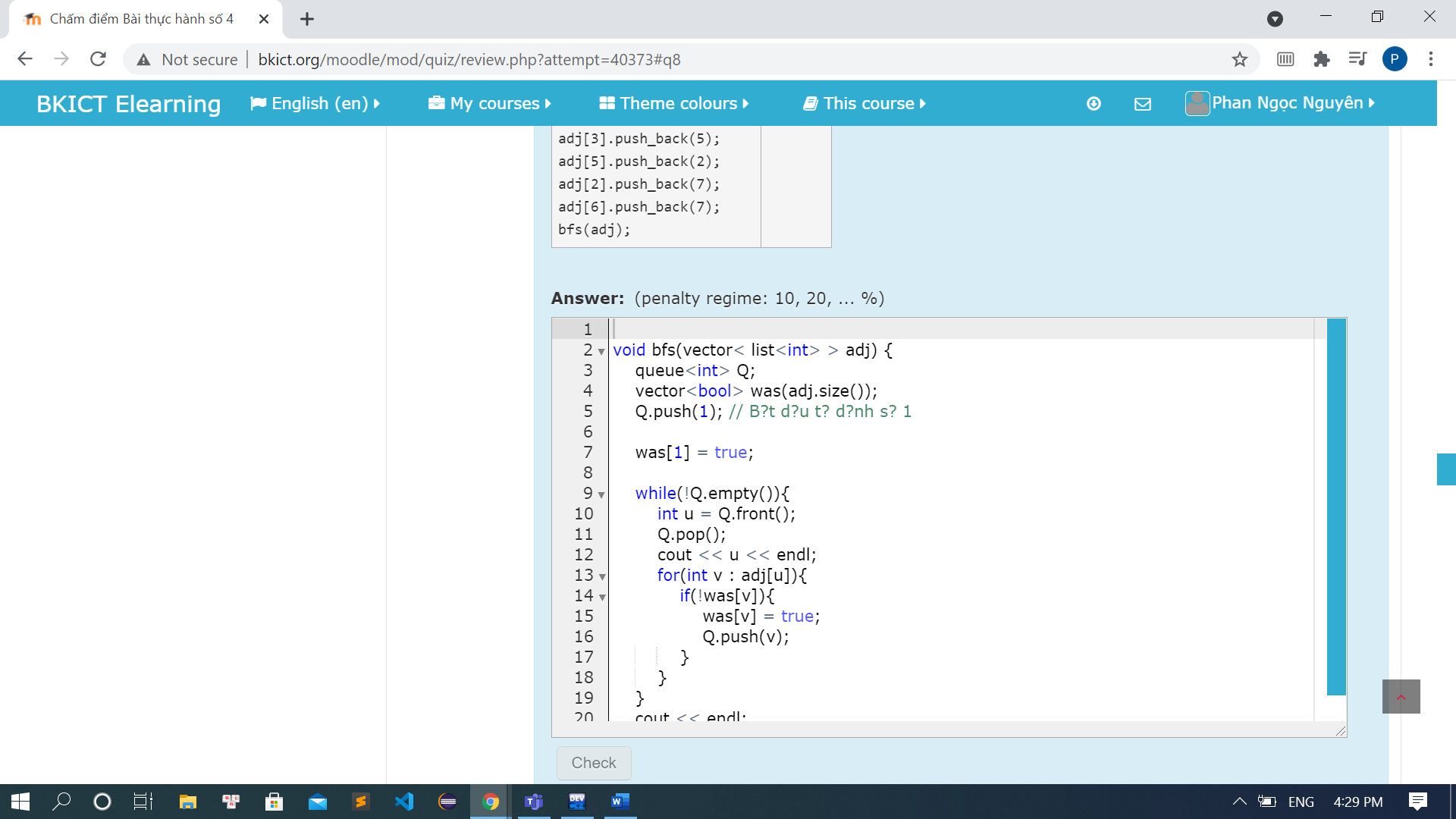
adj[6].push\_back(7);

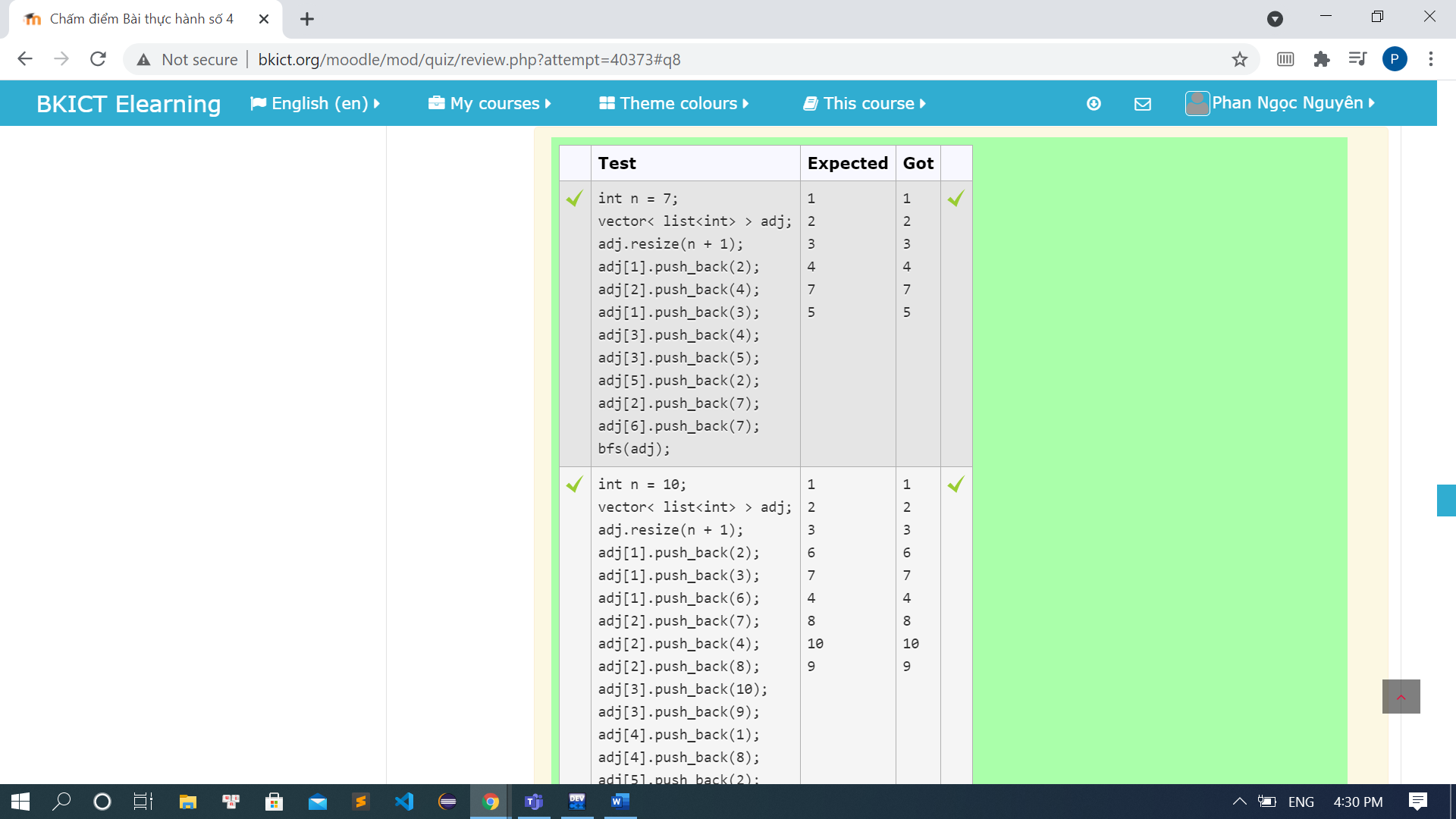
dfs(adj);

return 0;

}

### Bài 4.6: BFS





Code:

#include<bits/stdc++.h>

using namespace std;

void bfs(vector< list<int> > adj) {

queue<int> Q;

vector<bool> was(adj.size());

Q.push(1); // B?t d?u t? d?nh s? 1

was[1] = true;

while(!Q.empty()){

int u = Q.front();

Q.pop();

cout << u << endl;

for(int v : adj[u]){

if(!was[v]){

was[v] = true;

Q.push(v);

}

}

}

cout << endl;

}

int main(){

cout << "Phan Ngoc Nguyen - 20194134 " << endl;

int n = 7;

vector< list<int> > adj;

adj.resize(n + 1);

adj[1].push\_back(2);

adj[2].push\_back(4);

adj[1].push\_back(3);

adj[3].push\_back(4);

adj[3].push\_back(5);

adj[5].push\_back(2);

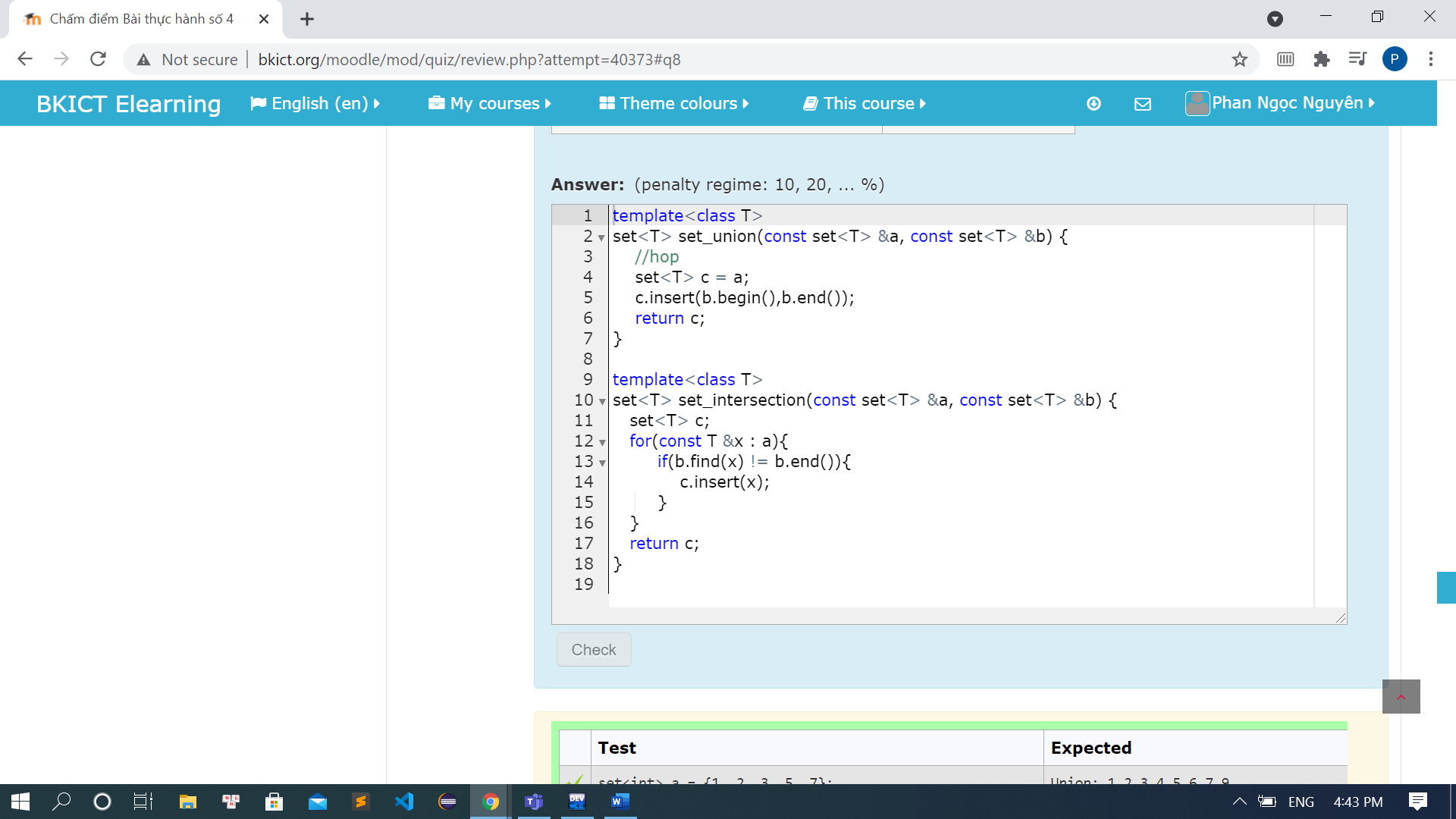
adj[2].push\_back(7);

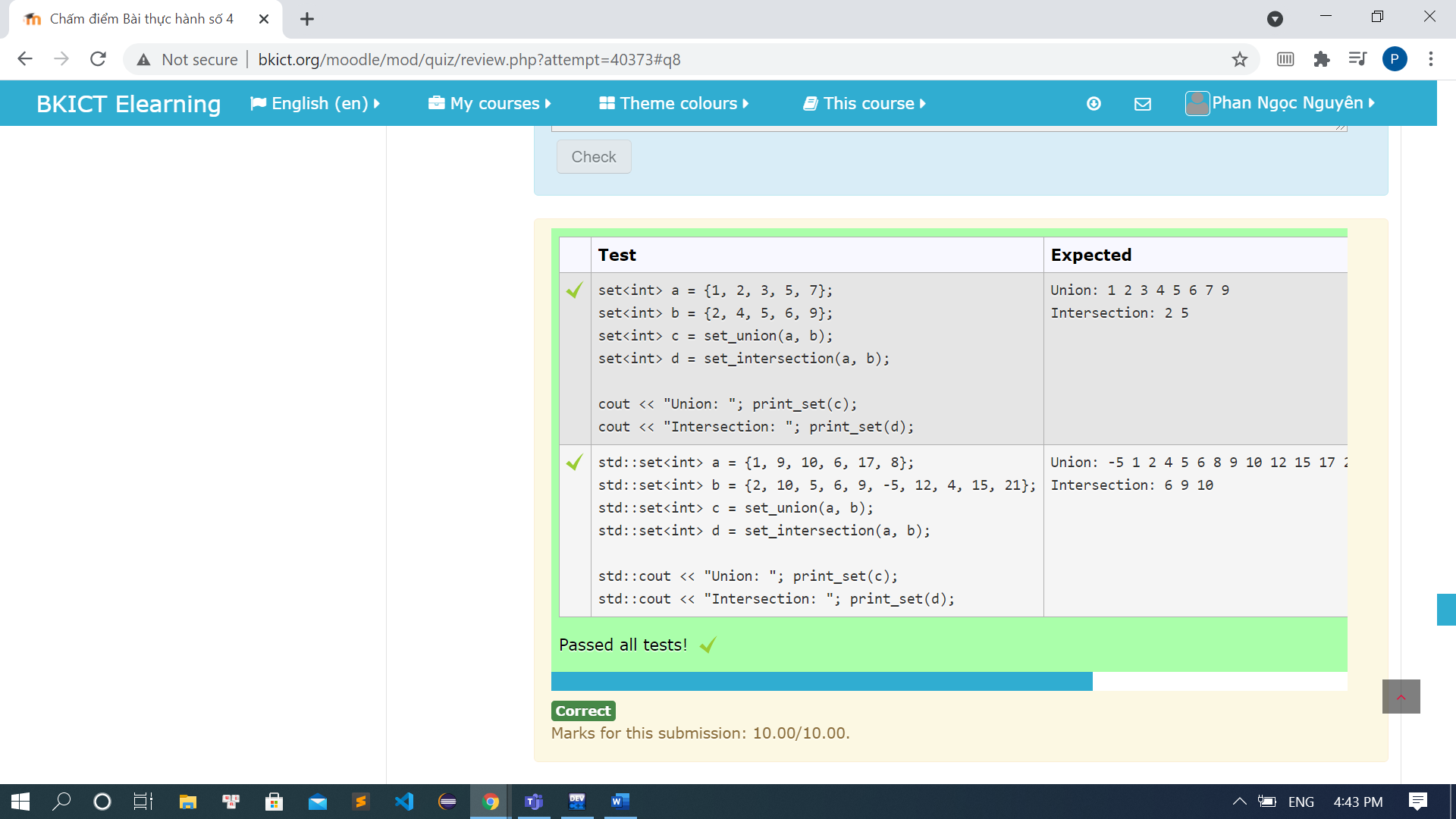
adj[6].push\_back(7);

bfs(adj);

}

### Bài 4.7: Set





Code:

#include <iostream>

#include <set>

using namespace std;

template<class T>

set<T> set\_union(const set<T> &a, const set<T> &b) {

//hop

set<T> c = a;

c.insert(b.begin(),b.end());

return c;

}

template<class T>

set<T> set\_intersection(const set<T> &a, const set<T> &b) {

set<T> c;

for(const T &x : a){

if(b.find(x) != b.end()){

c.insert(x);

}

}

return c;

}

template<class T>

void print\_set(const std::set<T> &a) {

for (const T &x : a) {

std::cout << x << ' ';

}

std::cout << std::endl;

}

int main() {

cout << "Phan Ngoc Nguyen" << endl;

std::set<int> a = {1, 2, 3, 5, 7};

std::set<int> b = {2, 4, 5, 6, 9};

std::set<int> c = set\_union(a, b);

std::set<int> d = set\_intersection(a, b);

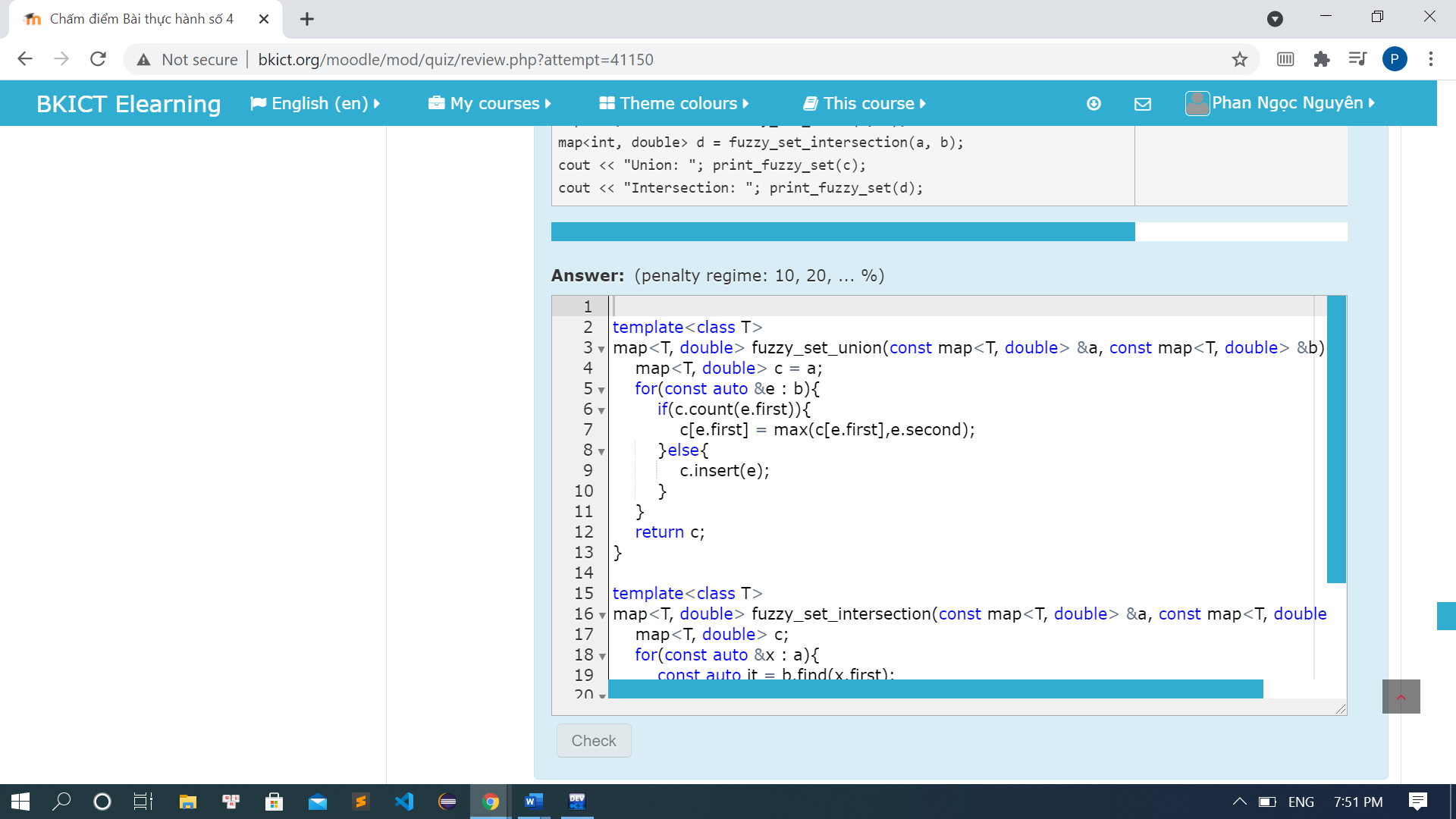
std::cout << "Union: "; print\_set(c);

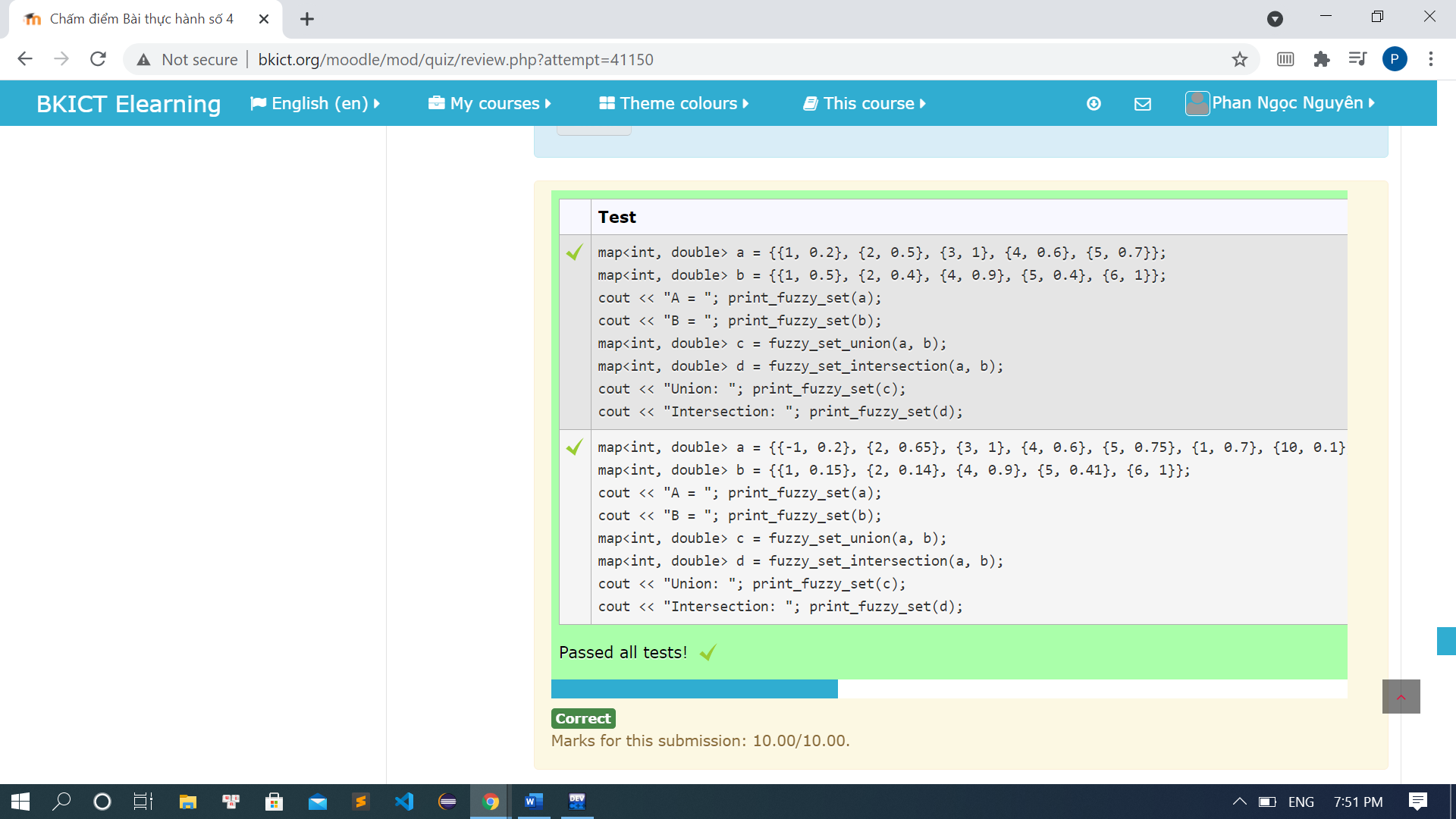
std::cout << "Intersection: "; print\_set(d);

return 0;

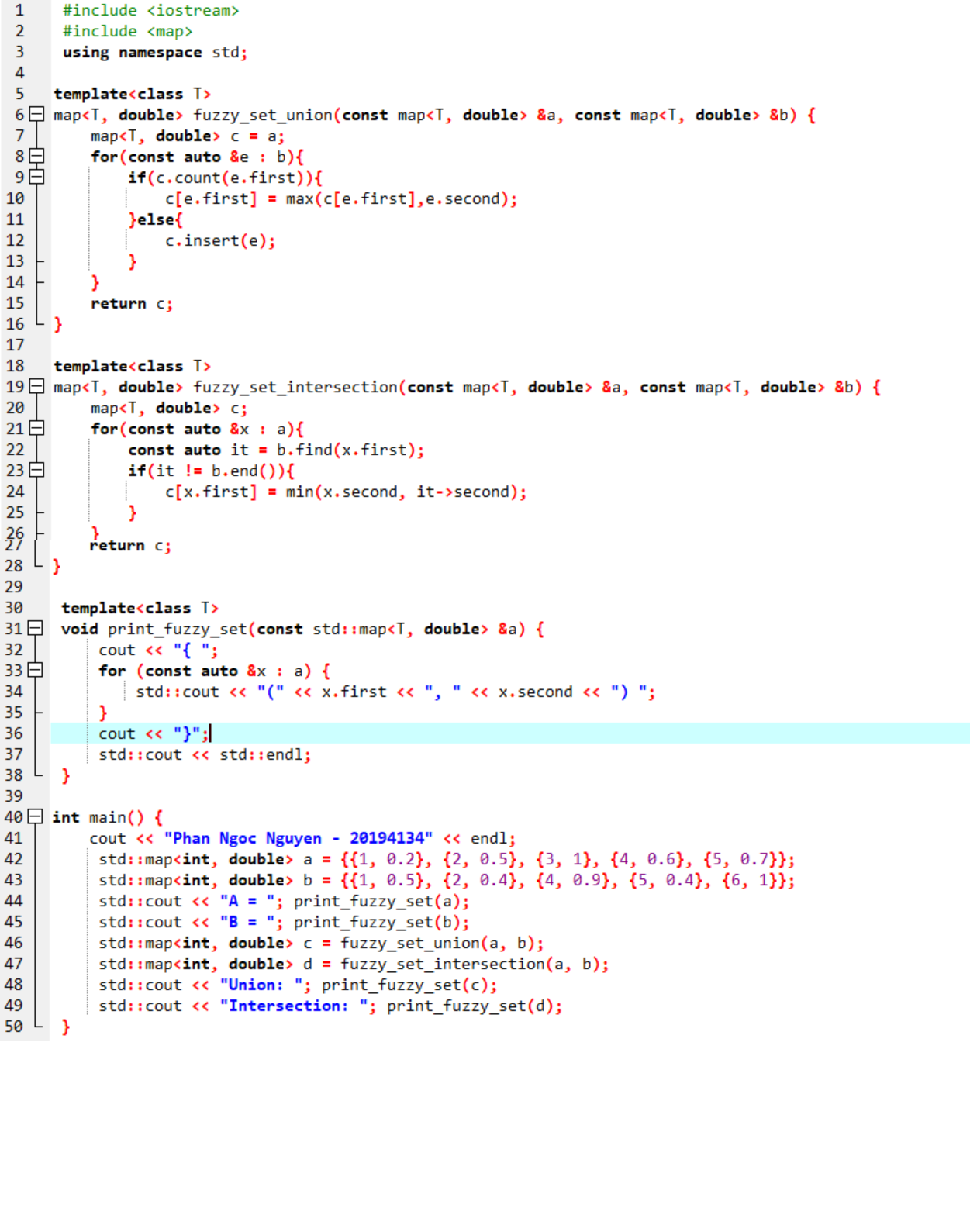
}

### 4.8 Map

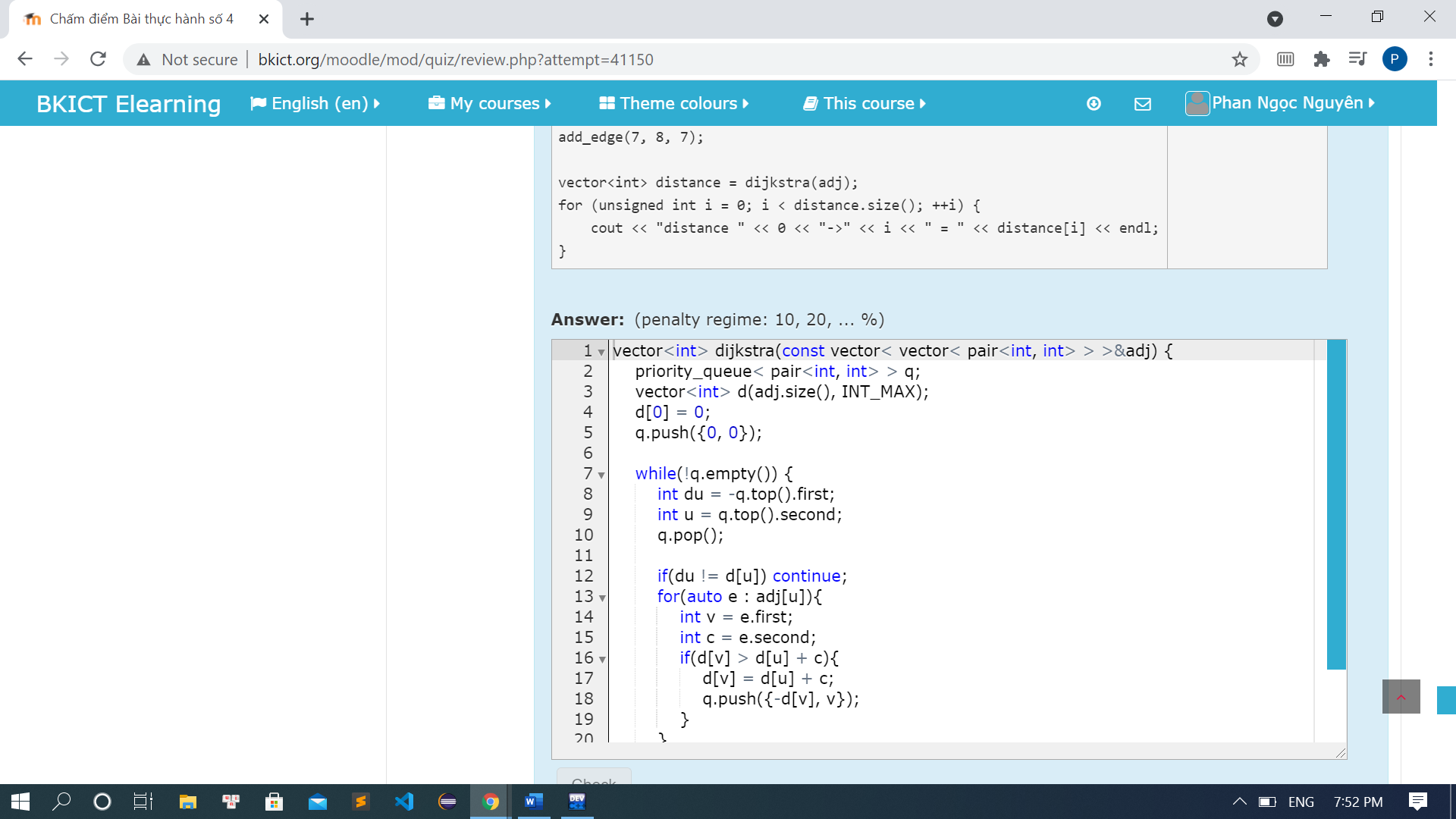


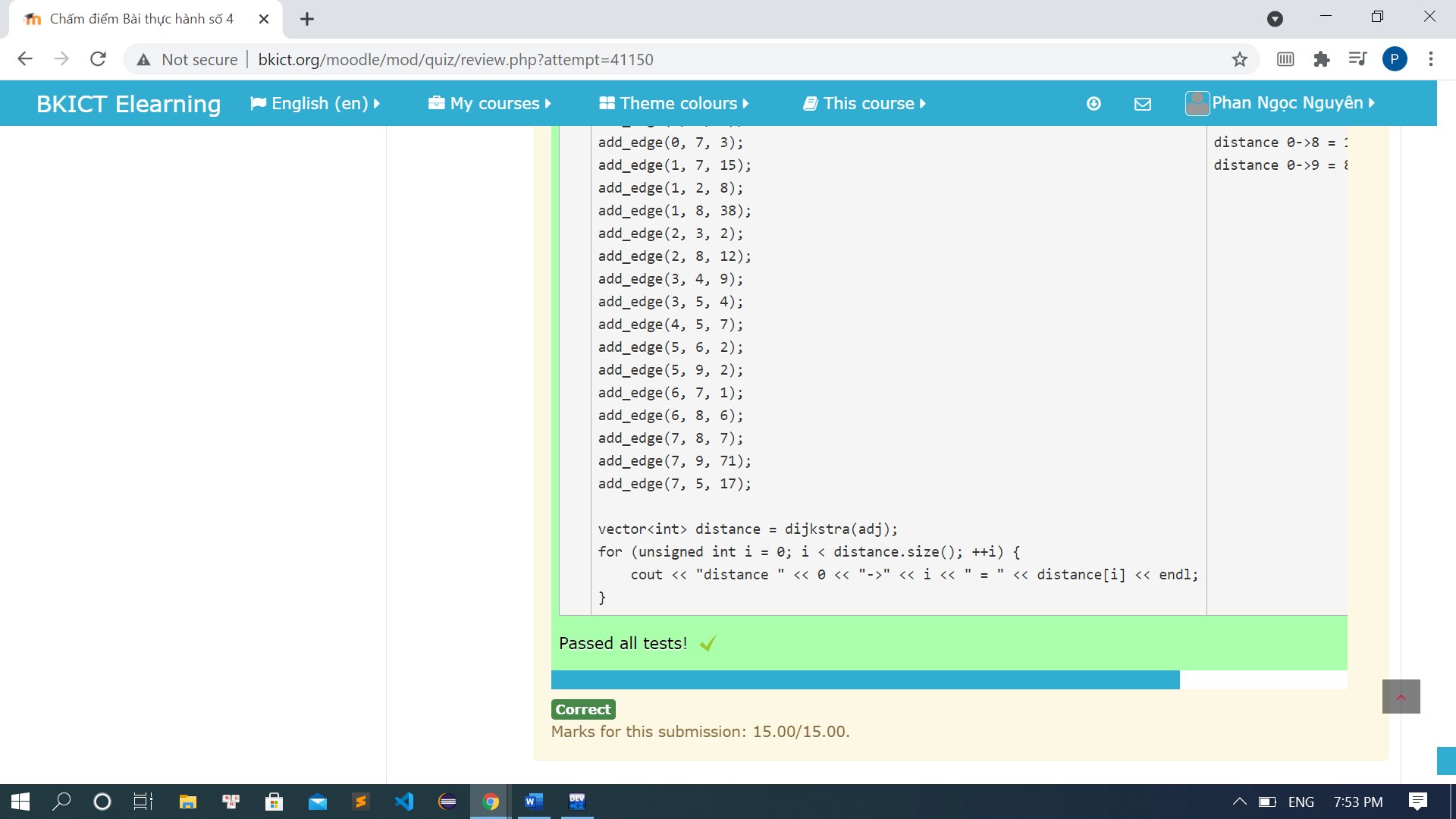


Code:



### 4.9 Dijitra

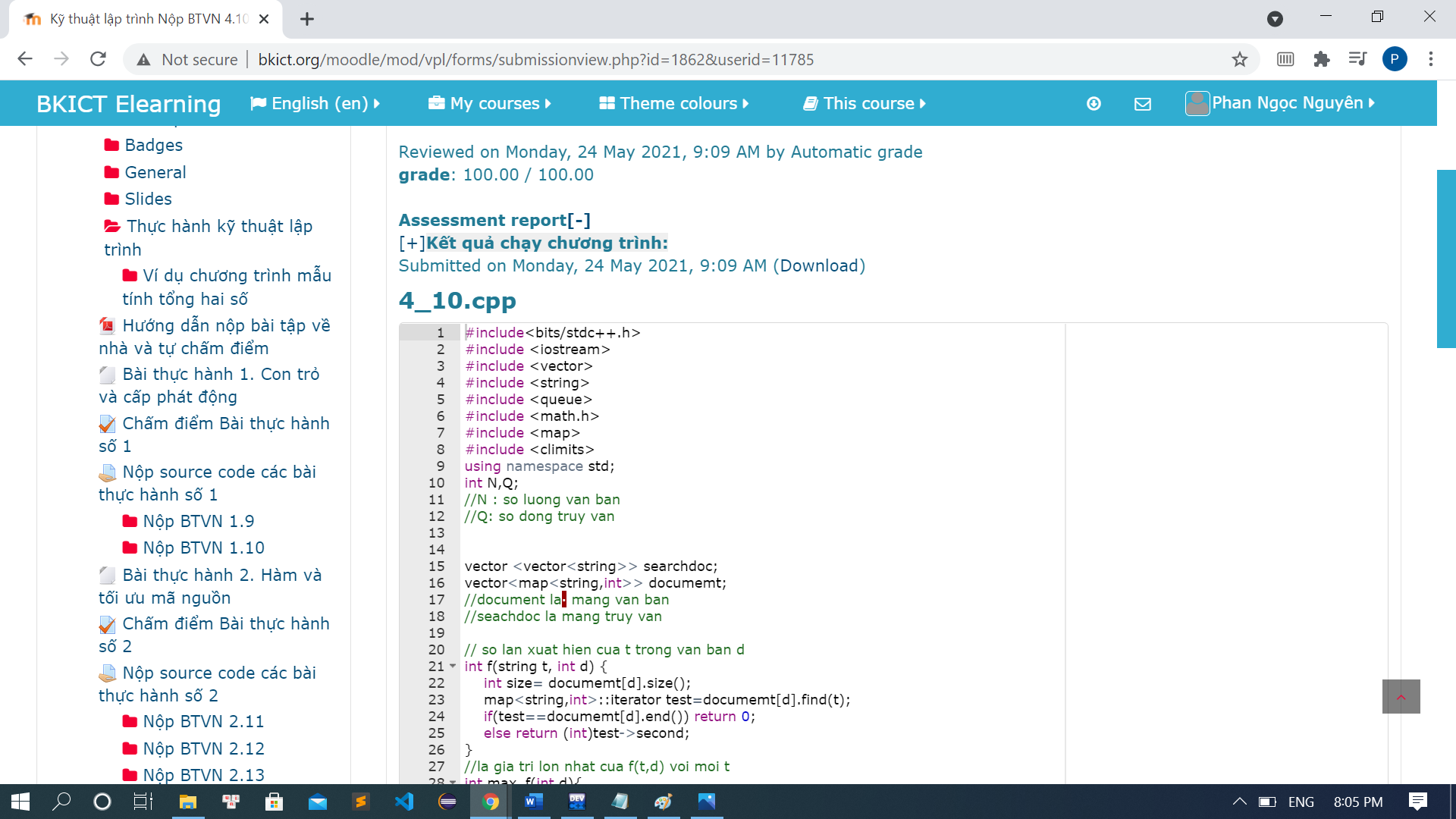




Code:



### 4.10 Search Engine



Code:

#include<bits/stdc++.h>

using namespace std;

int N,Q;

//N : so luong van ban

//Q: so dong truy van

vector <vector<string>> searchdoc;

vector<map<string,int>> documemt;

//document la  mang van ban

//seachdoc la mang truy van

// so lan xuat hien cua t trong van ban d

int f(string t, int d) {

int size= documemt[d].size();

map<string,int>::iterator test=documemt[d].find(t);

if(test==documemt[d].end()) return 0;

else return (int)test->second;

}

//la gia tri lon nhat cua f(t,d) voi moi t

int max\_f(int d){

int size= documemt[d].size();

priority\_queue<int> Max;

for (map<string,int>::iterator it=documemt[d].begin(); it!=documemt[d].end(); ++it){

Max.push(it->second);

}

return Max.top();

}

//so van ban chua tu t

int df(string t){

int count=0;

for(int d=0;d<N;++d){

if(f(t,d)>0) count++;

}

return count;

}

double TF(string t,int d){

return (double)0.5+0.5\*((f(t,d)\*1.0)/max\_f(d));

}

double IDF(string t){

return (double)log2f((N\*1.0)/df(t));

}

//diem so cua tu t xuat hien trong van ban d

double score(string t, int d){

if(f(t,d)!=0) return TF(t,d)\*IDF(t);

else return 0;

}

//diem so cua van ban d doi voi truy van gom cac tu (co the trung nhau) t1,t2,.....,tq

double finalScore(vector<string> doc1,int d){

int size=doc1.size();

double result=0;

for(int j=0;j<size;j++){

result+=score(doc1[j],d);

}

return result;

}

//chuyen chuoi dc nhap thanh vector

vector<string> getDoc(string str){

vector<string>doc1(0);

int size=str.length();

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

string t="";

while(str[i]!=','&&(i)<size){

t+=str[i];

i++;

}

if(t!="") doc1.push\_back(t);

}

return doc1;

}

//chuyen vector thanh map

map<string,int> getMap(vector<string>doc1){

map<string,int> map;

int size=doc1.size();

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

map[doc1[i]]++;

}

return map;

}

int main(){

cout << "Phan Ngoc Nguyen - 201941314" << endl;

vector<int> result;// mang ket qua

cin>>N;

//nhap van ban

for(int i=0;i<N;++i){

string str;

cin>>str;

vector<string> doc1=getDoc(str);

map<string,int>map=getMap(doc1);

documemt.push\_back(map);

}

//nhap truy van

cin>>Q;

for(int i=0;i<Q;++i){

//queue laf luu diem cua cua dong truy van doi voi van ban j

priority\_queue<pair<double,int>> que;

string str;

cin>>str;

vector<string> doc2=getDoc(str);

for(int j=0;j<N;++j){

double finalcso=finalScore(doc2,j);

que.push({finalcso,-j});

}

result.push\_back((-que.top().second)+1);

}

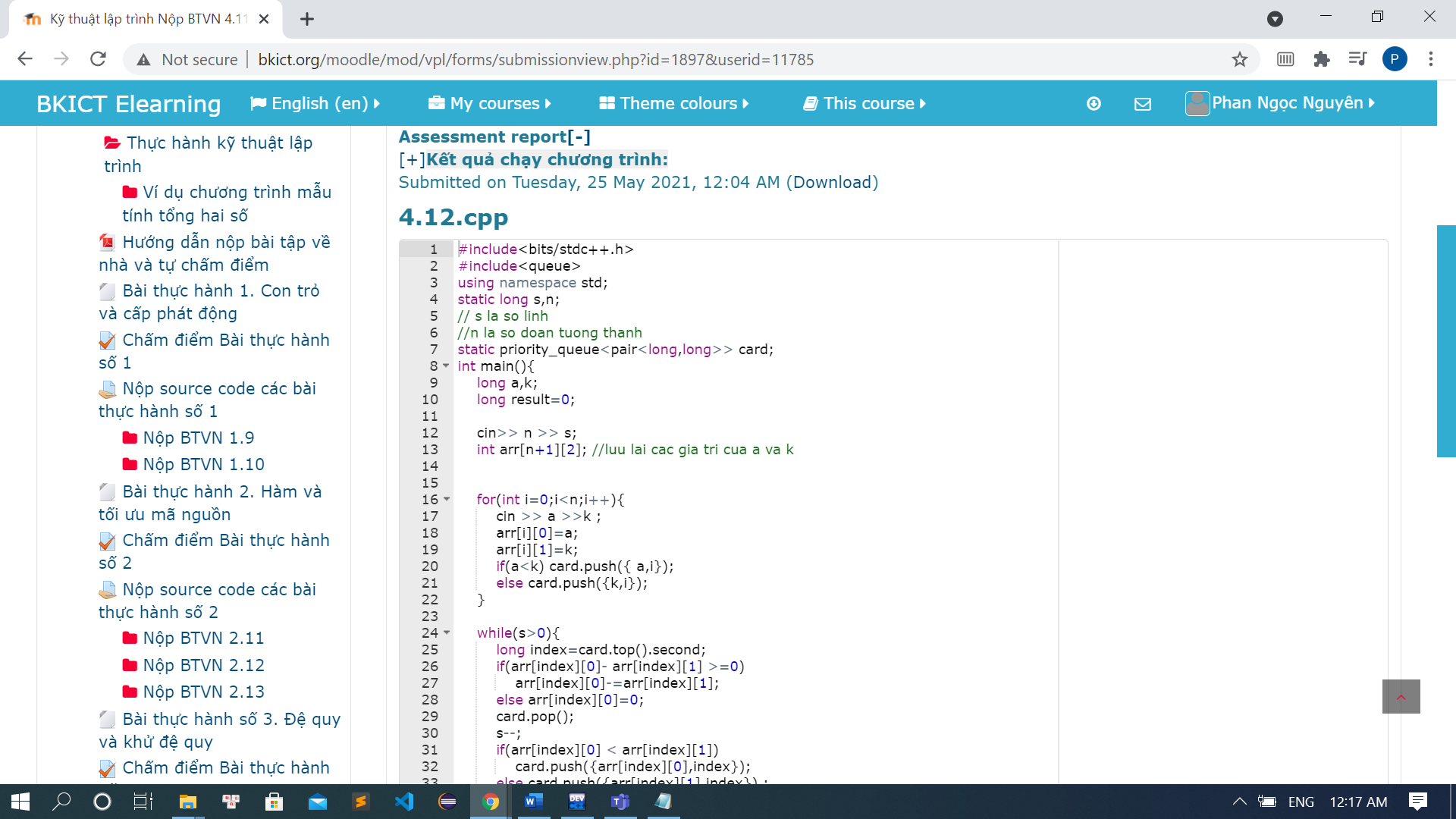
for(int i=0;i<Q;i++){

cout<<result[i]<<endl;

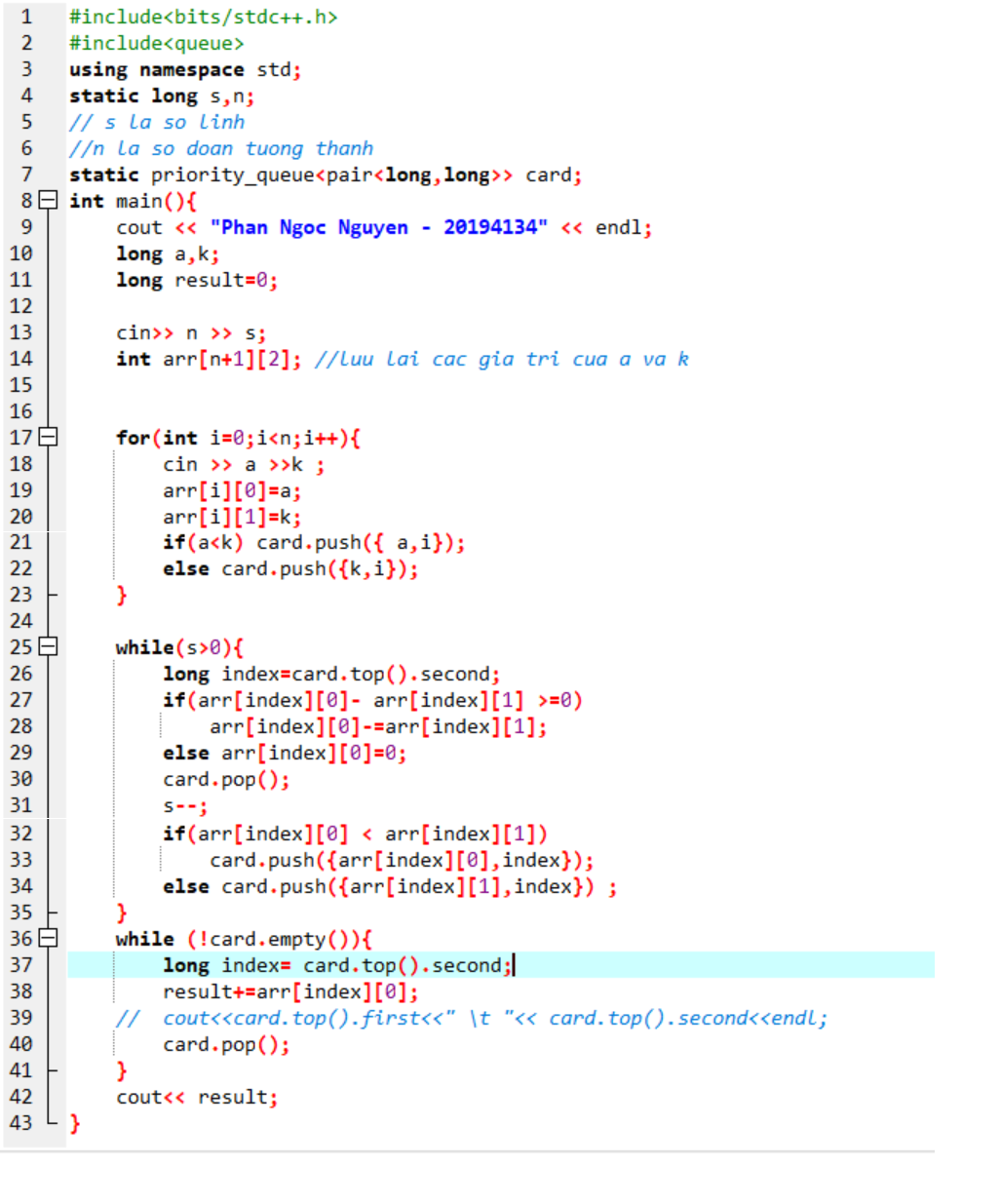
}

}

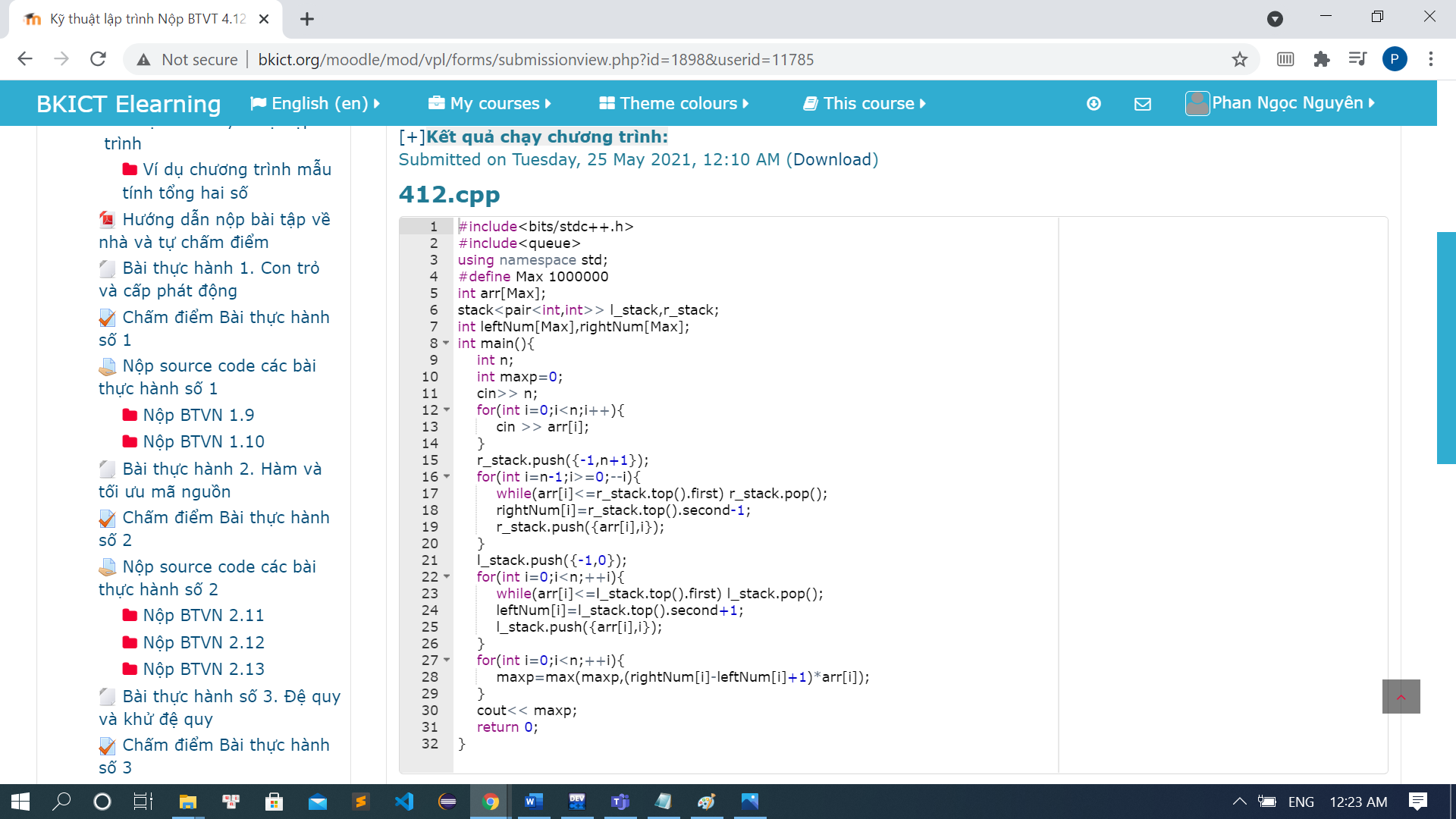
### 4.11 Bảo vệ lâu đài:



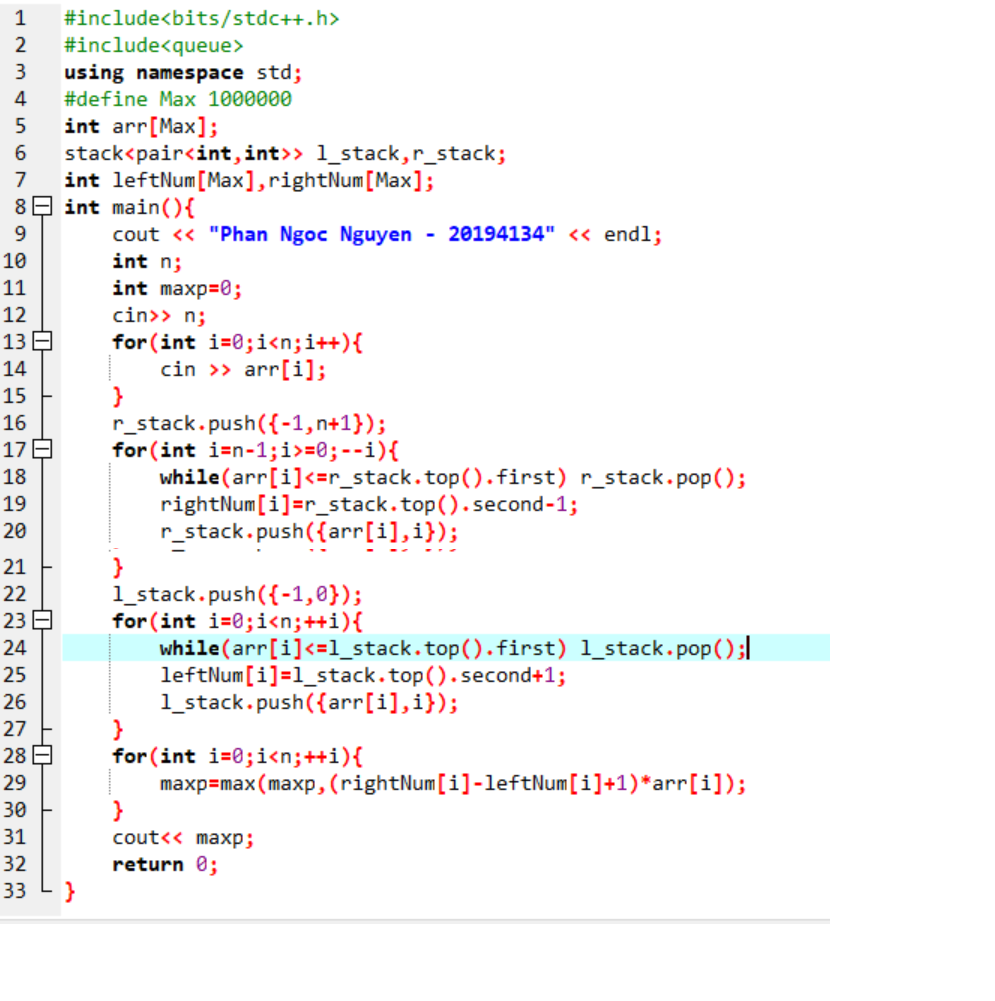
Code:



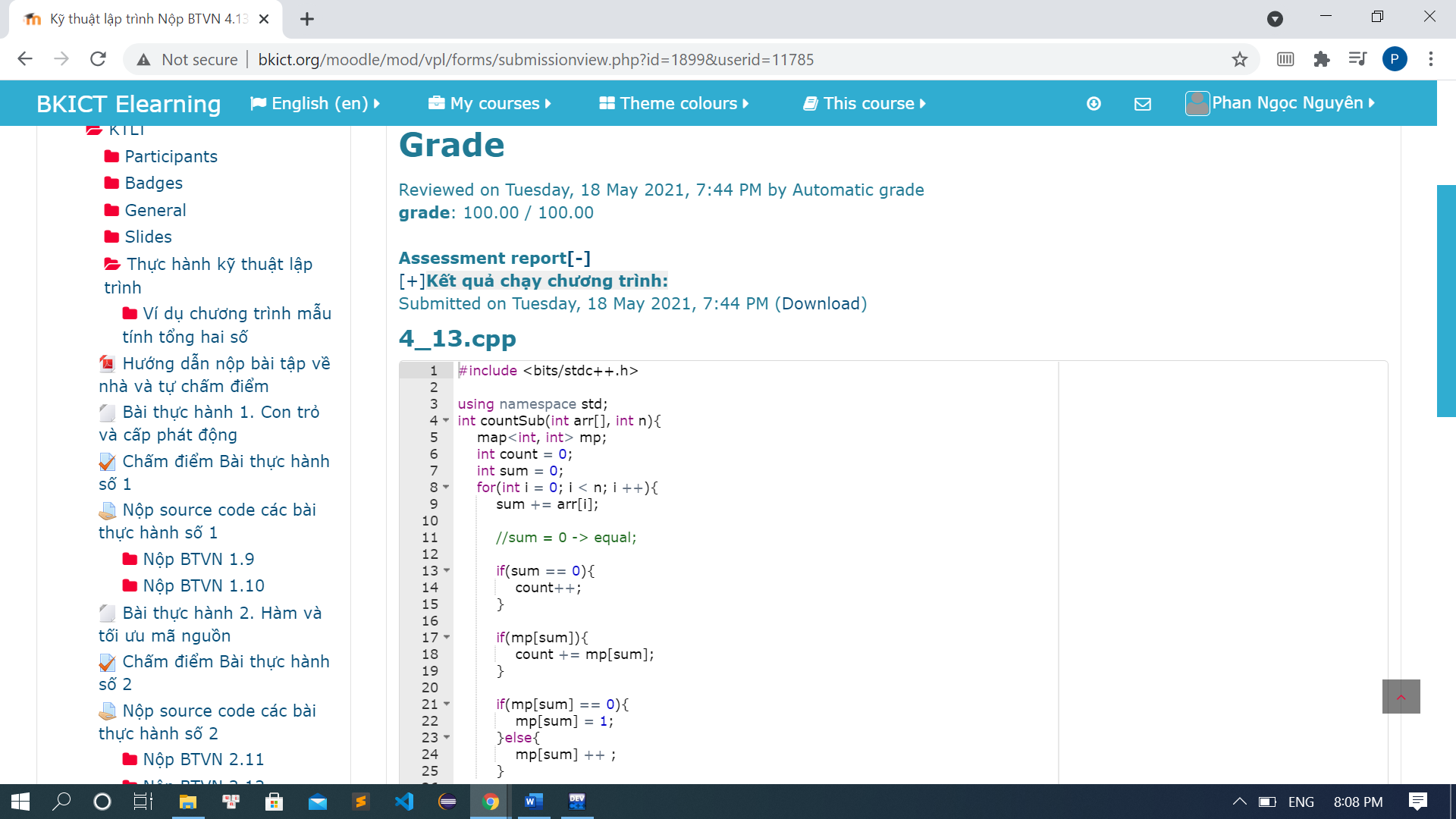
### 4.12 Lược đồ



Code:



### 4.13 Đếm xâu con



Code:

