**Ma’lumotlarning abstrak turlari va ma’lumotlar strukturalari. (Stek, navbat, vektor, ro’yxat)**

**2 – Topshiriq**

*Iqbolshoh Ilhomjonov*

2. 100 ta elementdan iborat stek hosil qiling va ularni ixtiyoriy sonlar generatori bilan to’ldiring. Eng boshida joylashgan 5 ta elementini ko’paymasini toping.

#include <bits/stdc++.h>

using namespace std;

int main()

{

    stack<int> s; // int tipida stek hosil qilish

    // 100 ta elementni stekka to'ldirish

    for (int i = 0; i < 100; i++)

    {

        int a = rand() % 100 + 1; // ixtiyoriy sonni olish

        s.push(a); // stekka qo'shish

    }

    // Eng boshida joylashgan 5 ta elementni ko'paytirish

    int K = 1;

    for (int i = 0; i < 5; i++)

    {

        K \*= s.top(); // elementni ko'paytirish

        s.pop(); // stekdan o'chirish

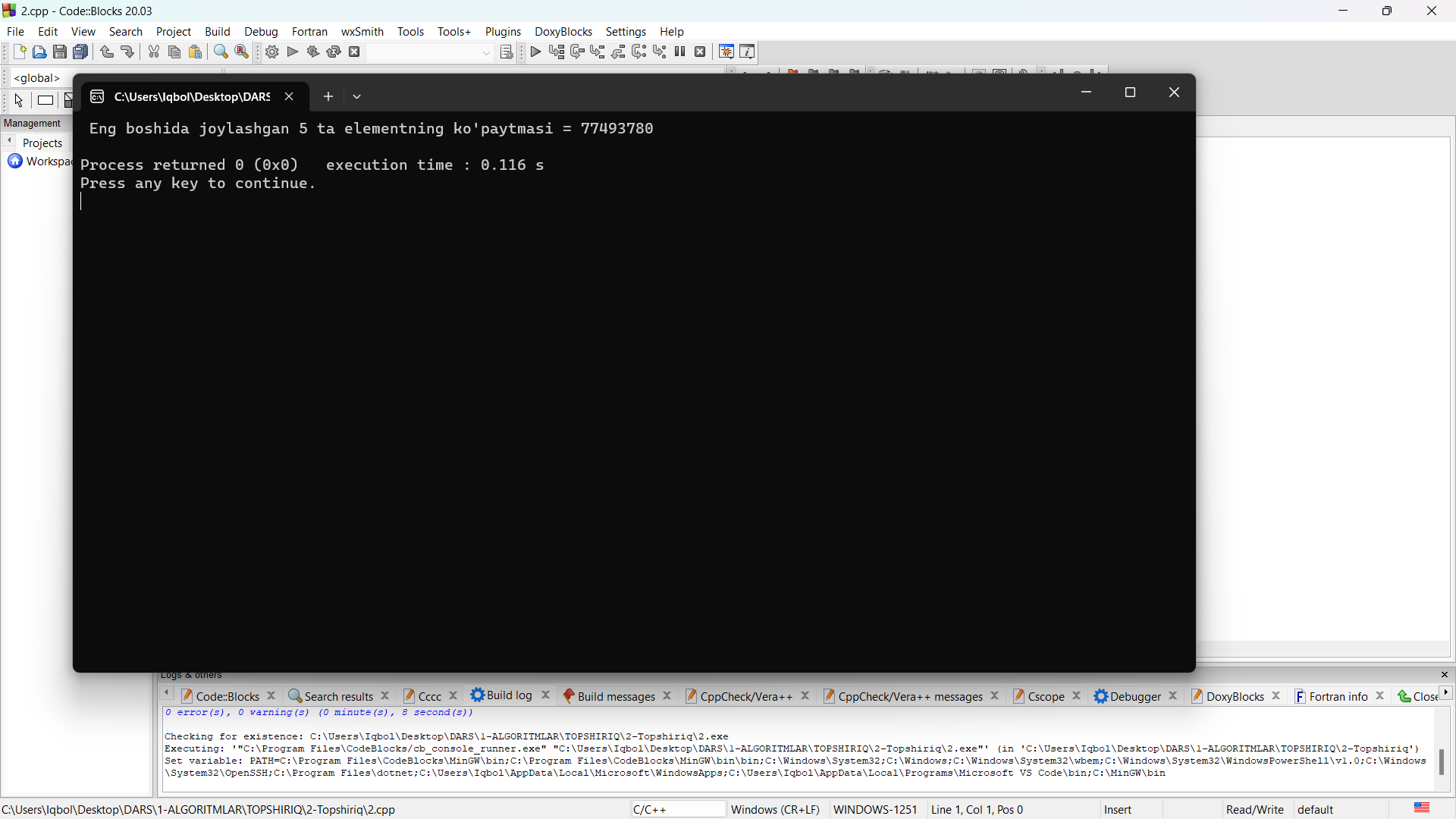
    }

    cout << " Eng boshida joylashgan 5 ta elementning ko'paytmasi = " << K << endl;

    return 0;

}

//NATIJA:



4 . 5 elementli stek hosil qiling. Bu stekka 3 element qo’shib, eng yuqori elementini aniqlang.

#include <bits/stdc++.h>

using namespace std;

int main() {

    stack<int> s;

    //5 ta elementli stek hosil qilindi

    for (int i = 0; i < 5; i++)

    {

        int a = rand() % 100 + 1; // 100 gacha ixtiyoriy sonni olish

    }

    // 3 ta element qo'shish

    s.push(10);

    s.push(20);

    s.push(30);

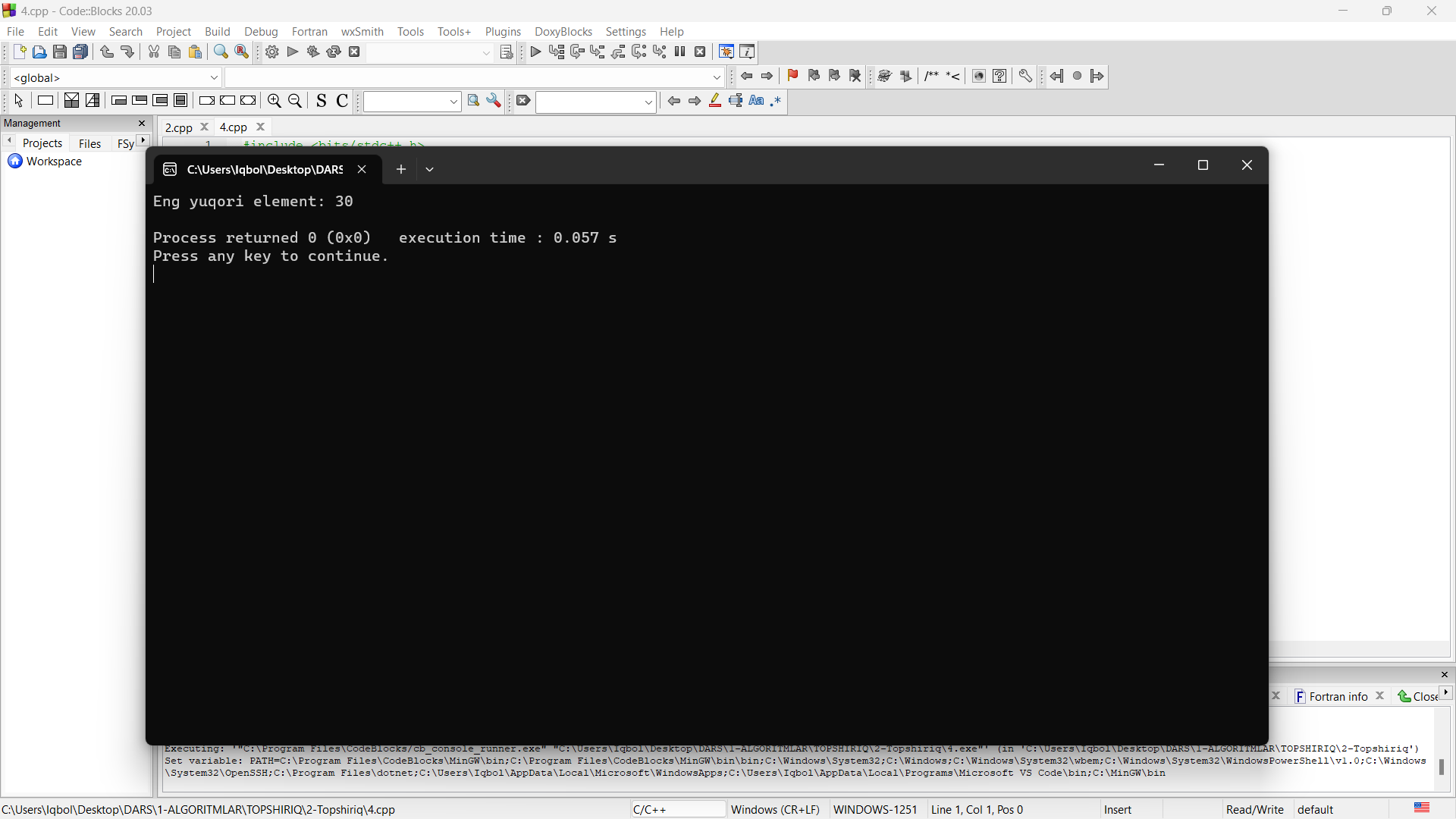
    // Eng yuqori (top) elementni ekranga chiqarish

    cout << "Eng yuqori element: " << s.top() << endl;

    return 0;

}

//NATIJA:



6. Stek hosil qiling va pop(), peek(), push(),top(), empty() metodlaridan foydalanib dasturini tuzing.

#include <iostream>

#include <stack>

using namespace std;

int main() {

    // Stek (stack) hosil qilish

    stack<int> s;

    // Push() metodidan foydalanib stekka element qo'shish

    s.push(5);

    s.push(3);

    s.push(8);

    // Top() metodidan foydalanib stekning eng yuqori elementini aniqlash

    cout << "Eng yuqori element: " << s.top() << endl;

    // Pop() metodidan foydalanib stekning eng yuqori elementini olib tashlash

    s.pop();

    // Empty() metodidan foydalanib stekning bo'shligini tekshirish

    if (s.empty()) {

        cout << "Stek bo'sh" << endl;

    } else {

        cout << "Stek bo'sh emas" << endl;

    }

    // Peek() metodidan foydalanib stekning eng yuqori elementini chiqarish (bitta Pop() qilishga o'xshash)

    cout << "Eng yuqori element: " << s.top() << endl;

    cout << "Stekdagi elementlar: ";

    while (!s.empty()) {

        cout << s.top() << " ";

        s.pop();

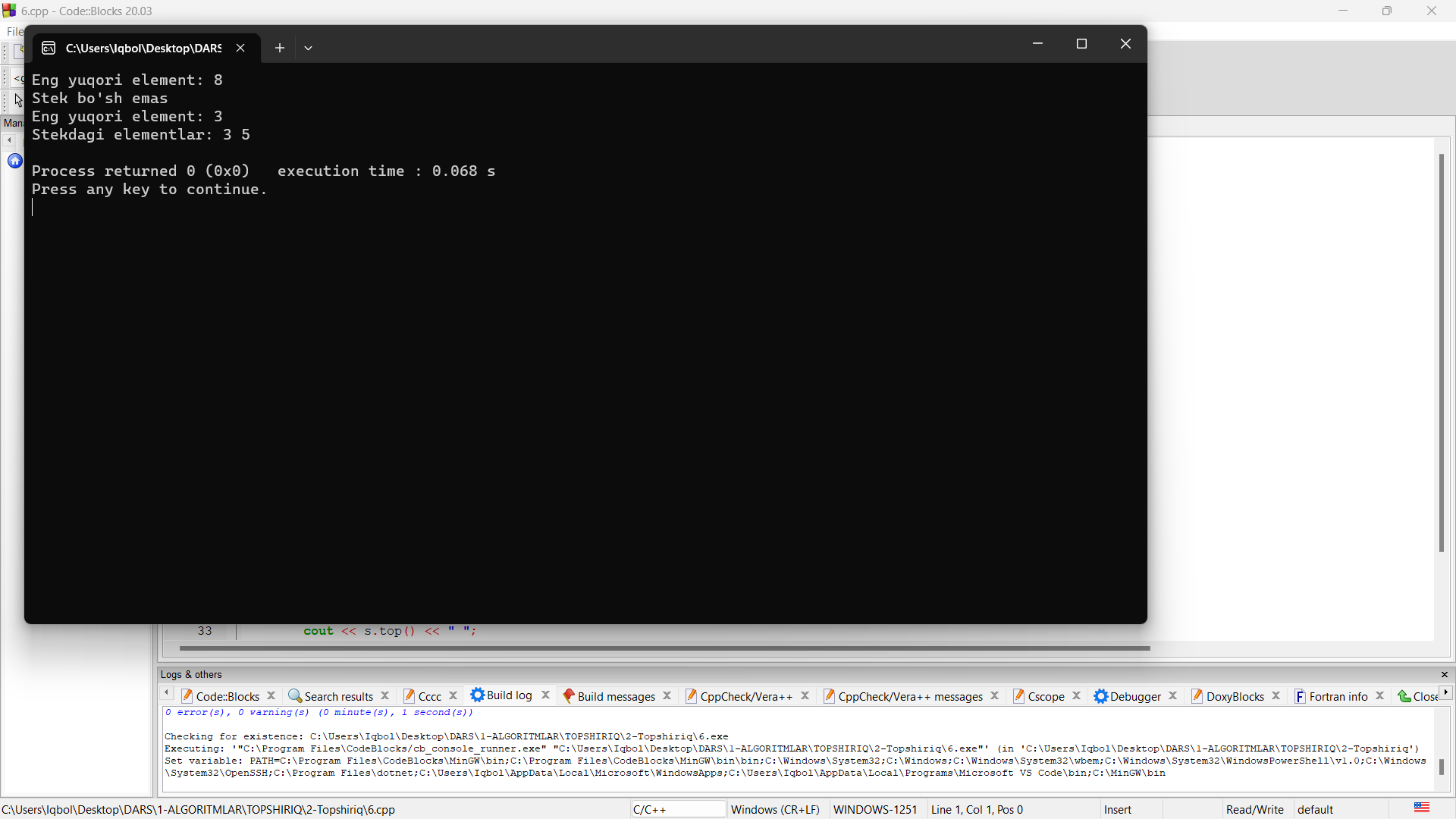
    }

    cout << endl;

    return 0;

}

//NATIJA:



8. Navbat hosil qiling va push() pop(), font(), back(), empty() metodlaridan foydalanib dasturini tuzing. 10. 100 elementdan iborat navbat hosil qiling va ularni ixtiyoriy sonlar generatori bilan to’ldiring, elementlarni chop eting va navbatni to’liqligicha bo’shating.

#include <iostream>

#include <queue>

using namespace std;

int main() {

    queue<int> navbat; // Navbat hosil qilish

    // Navbatga 3 ta son qo'shamiz

    navbat.push(10);

    navbat.push(20);

    navbat.push(30);

    // Navbatning boshidan (eng avvalgi son) o'chirib tashlaymiz

    navbat.pop();

    // Navbatning boshidagi (eng avvalgi son) qiymatini chiqaramiz

    cout << "Birinchi son: " << navbat.front() << endl;

    // Navbatning oxiridagi (eng so'nggi son) qiymatini chiqaramiz

    cout << "Oxirgi son: " << navbat.back() << endl;

    // Navbatning bo'shligini tekshirish

    if (navbat.empty()) {

        cout << "Navbat bo'sh" << endl;

    } else {

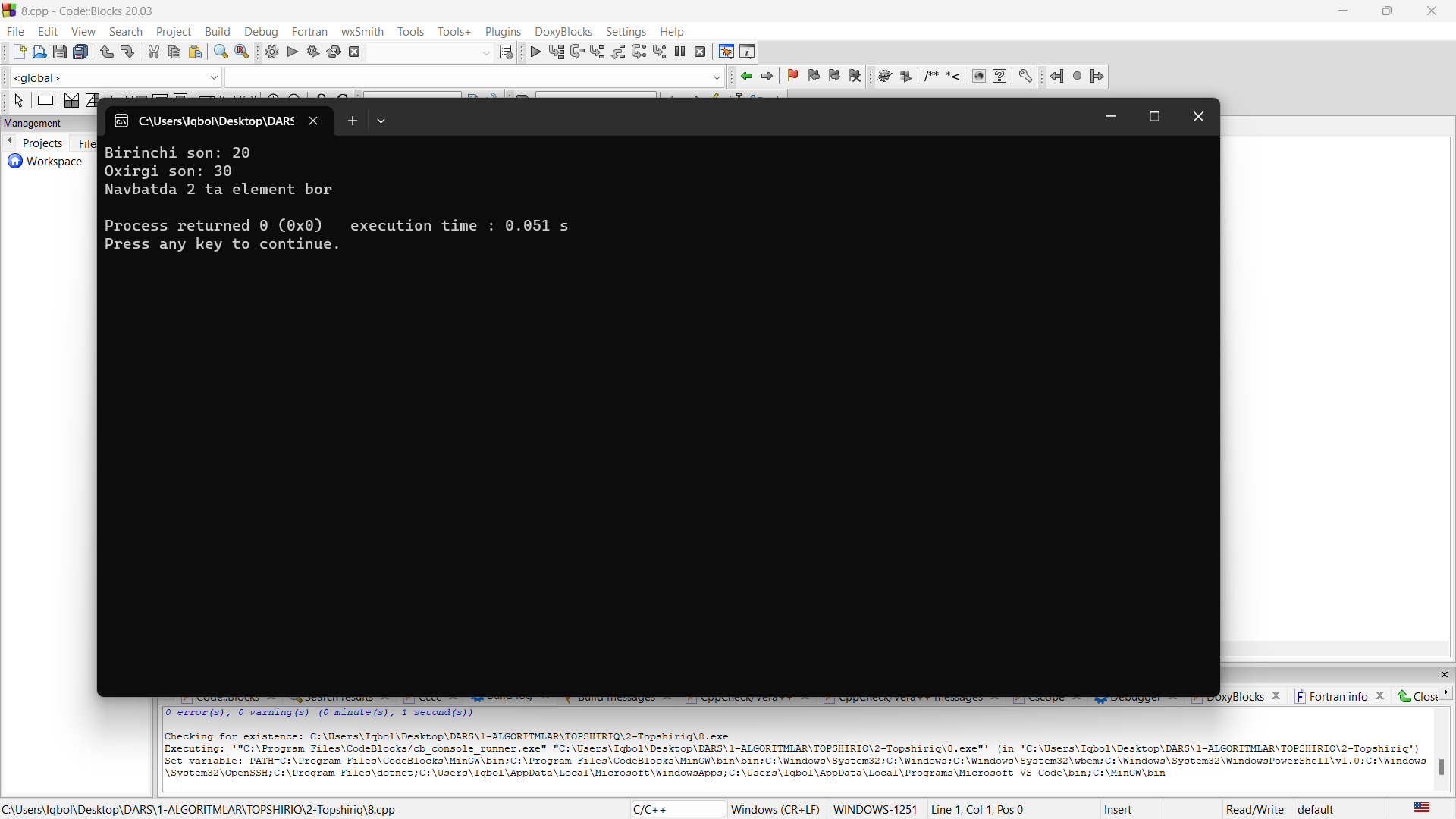
        cout << "Navbatda " << navbat.size() << " ta element bor" << endl;

    }

    return 0;

}

//NATIJA:



10. 100 elementdan iborat navbat hosil qiling va ularni ixtiyoriy sonlar generatori bilan to’ldiring, elementlarni chop eting va navbatni to’liqligicha bo’shating.

#include <bits/stdc++.h>

using namespace std;

int main() {

    queue<int> q;

    for (int i = 0; i < 100; i++)

    {

        int num = rand() % 100 + 1;

        q.push(num);

    }

    while (!q.empty())

    {

        cout << q.front() << " ";

        q.pop();

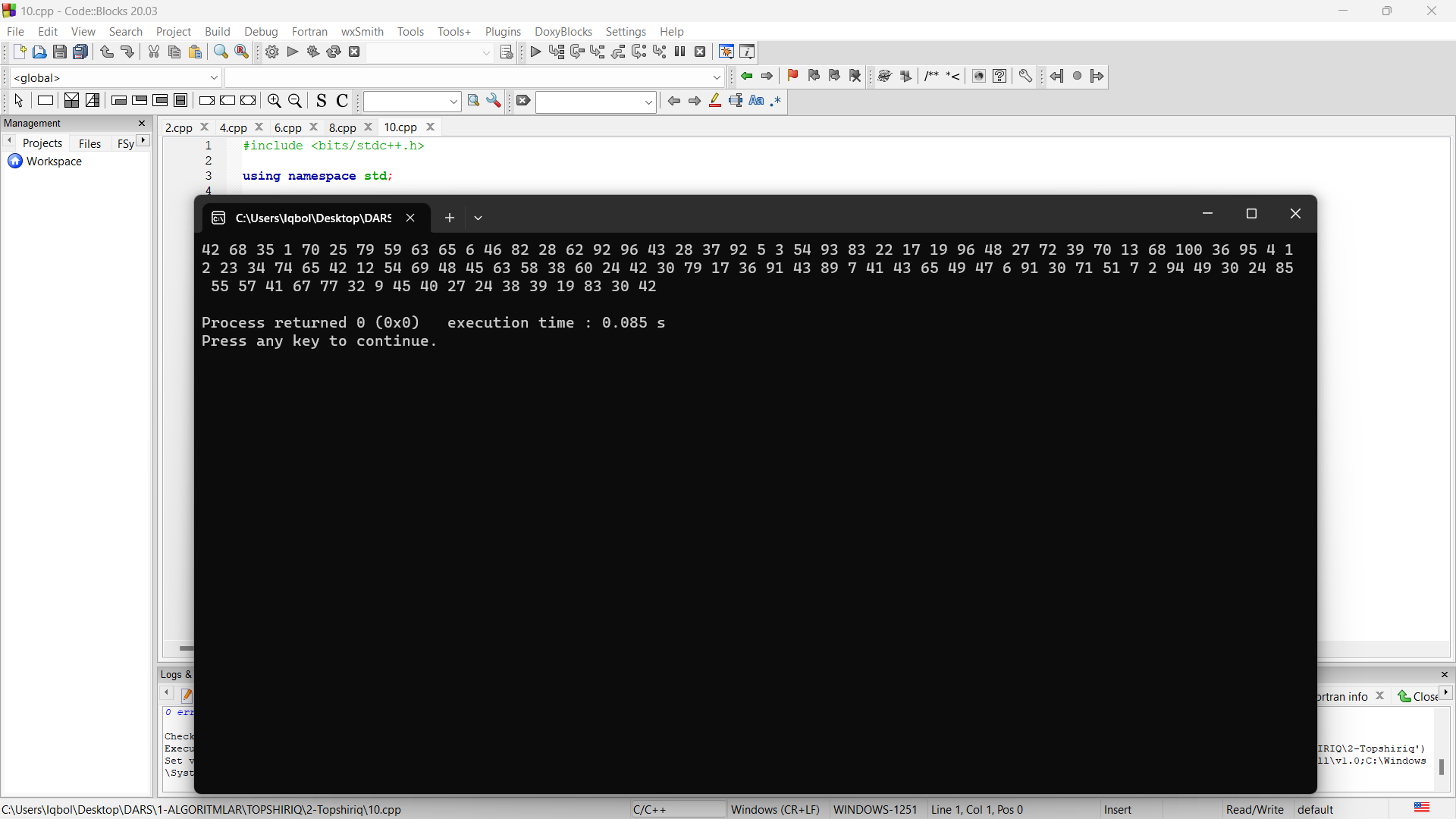
    }

    cout << endl;

    return 0;

}

//NATIJA:



12. 20 ta elementdan iborat Navbat hosil qiling. Uning 10 ta elementini o’chiring. Uning oxirgi va birinchi elementlarni qo’shing.

#include <bits/stdc++.h>

using namespace std;

int main() {

    queue<int> q;

    // 20 ta tasodifiy son qo'shish

    for (int i = 0; i < 20; i++) {

        q.push(rand() % 100);

    }

    // 10 ta elementni o'chirish

    for (int i = 0; i < 10; i++) {

        q.pop();

    }

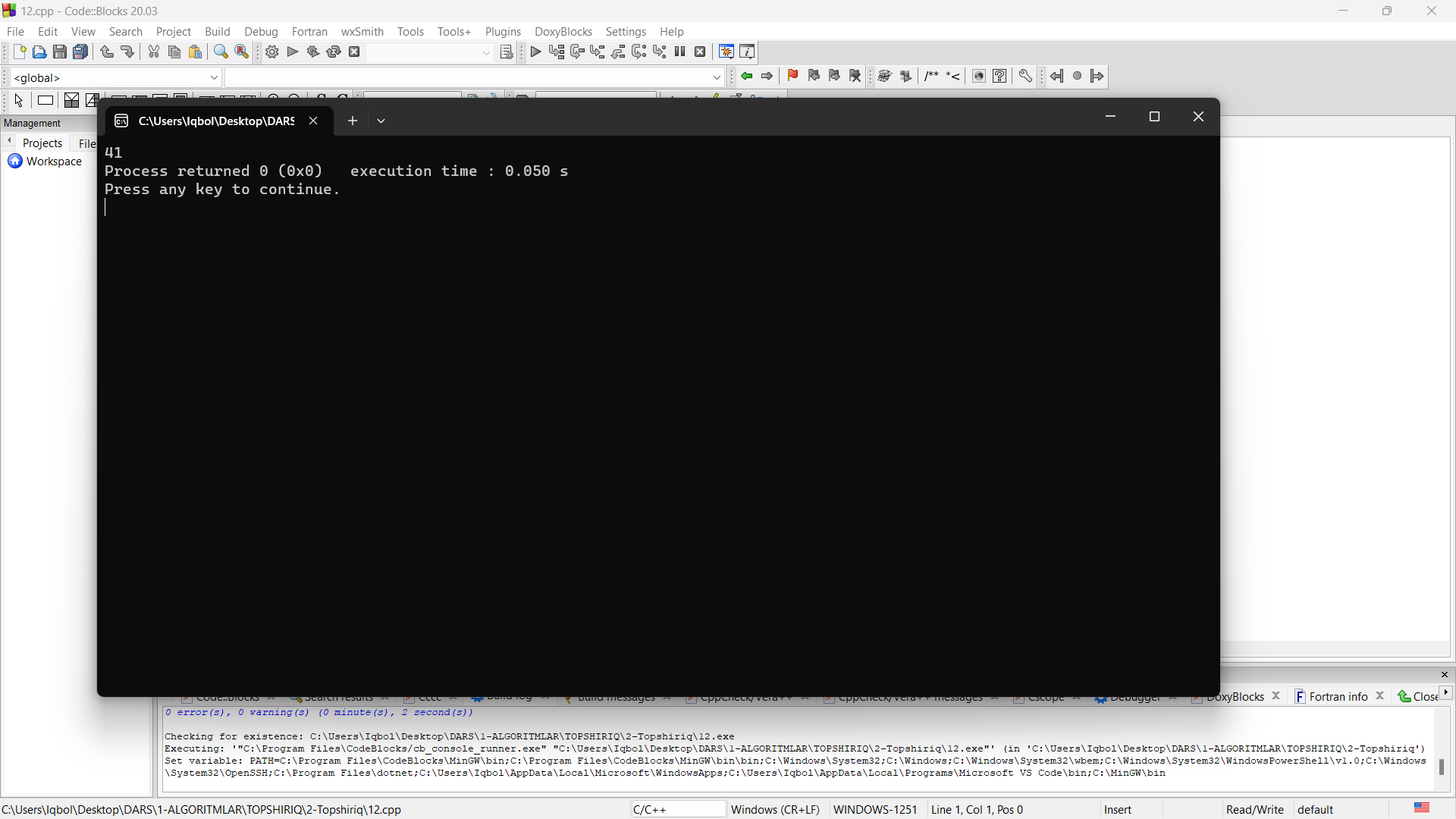
    // Navbatning birinchi va oxirgi qo'shish

    cout << q.back() + q.front();

    return 0;

}

//NATIJA:



14. m ta elementdan iborat navbat hosil qiling va unga n ta element qo’shib, navbat uzunligini aniqlang.

#include <bits/stdc++.h>

using namespace std;

int main() {

    int m, n;

    queue<int> q;

    // Navbatga m ta elementni qo'shish

    cout << "Navbat uzunligini kiriting: ";

    cin >> m;

    for (int i = 0; i < m; i++) {

        q.push(rand() % 100);

    }

    // Navbatga n ta elementni qo'shish

    cout << "Navbatga qo'shmoqchi bo'lgan elementlar sonini kiriting: ";

    cin >> n;

    for (int i = 0; i < n; i++) {

        q.push(rand() % 100);

    }

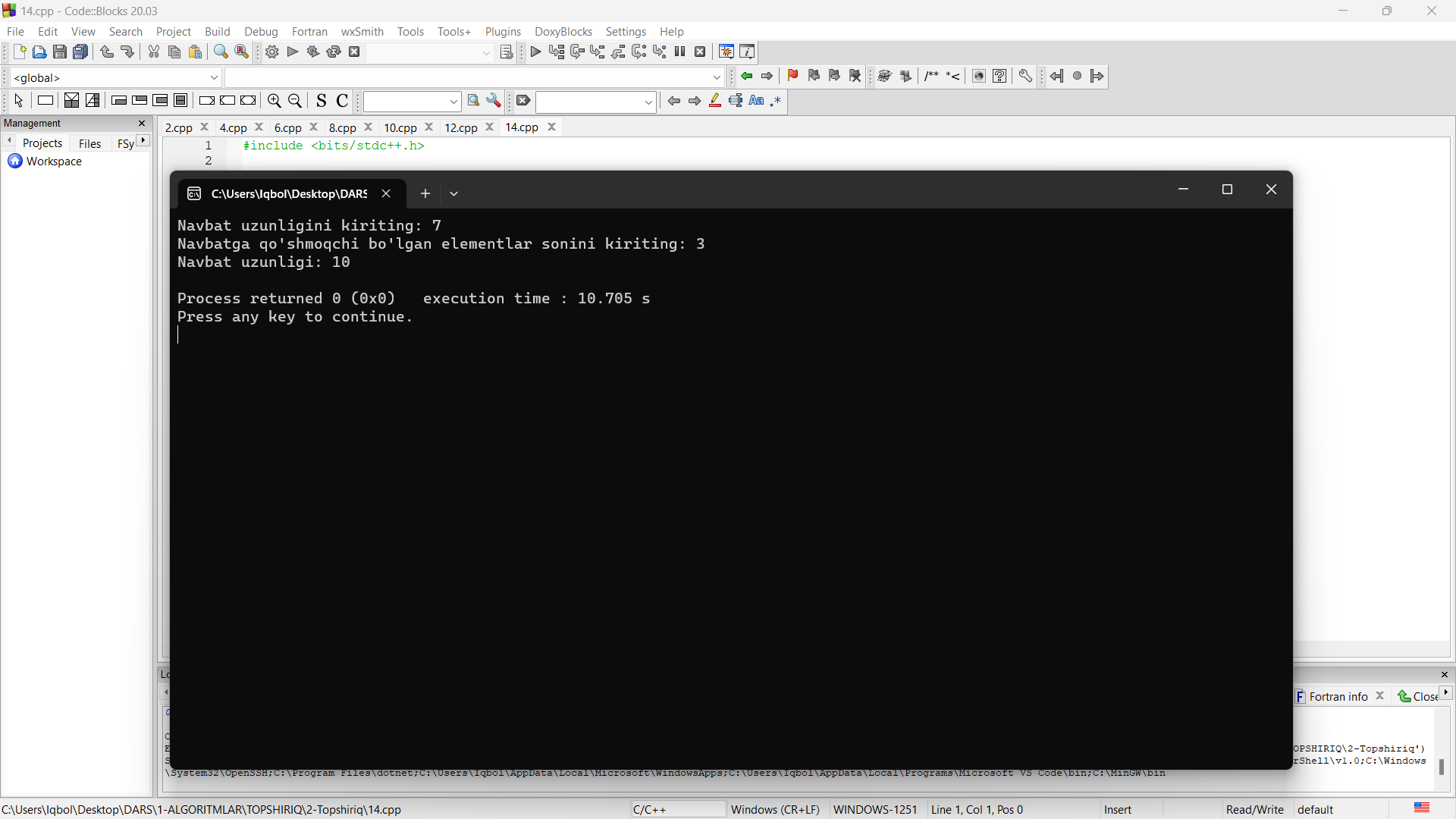
    // Navbat uzunligini aniqlash

    cout << "Navbat uzunligi: " << q.size() << endl;

    return 0;

}

//NATIJA:



16. N ta elementdan iborat stek hosil qiling. Oxirgi elementiga teng elementlar sonini toping.

#include <iostream>

#include <stack>

using namespace std;

int main() {

    int N;

    cout << "Stek uzunligini kiriting: ";

    cin >> N;

    stack<int> s;

    // N ta elementni stekka qo'shish

    for (int i = 1; i <= N; i++) {

        int a;

        cin >> a;

        s.push(a);

    }

    int oxirgi = s.top();

    int c = 0;

    // oxirgi elementga teng bo'lgan elementlarni hisoblash

    while (!s.empty() && s.top() == oxirgi) {

        c++;

        s.pop();

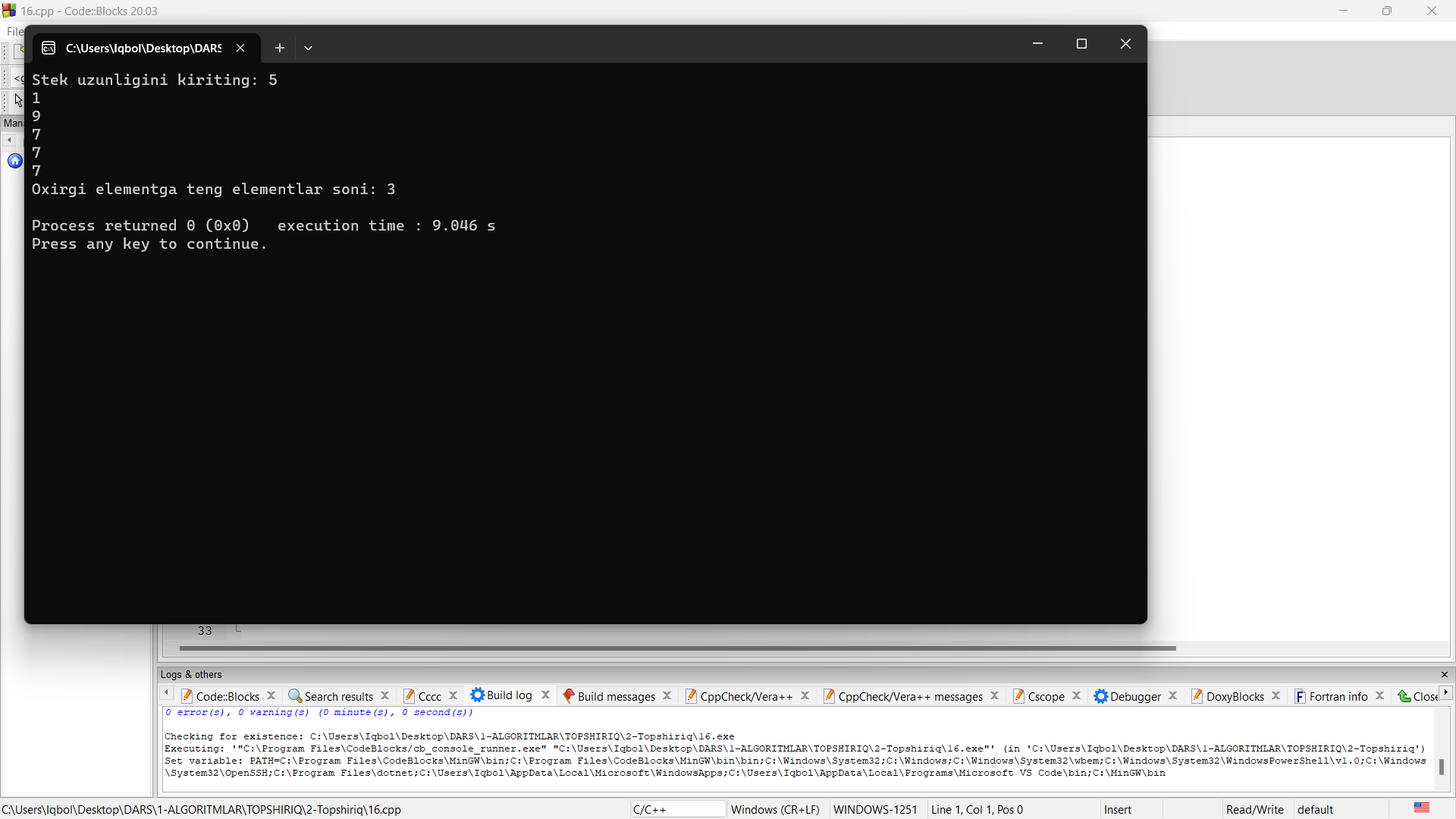
    }

    cout << "Oxirgi elementga teng elementlar soni: " << c << endl;

    return 0;

}

//NATIJA:



18. N elementdan iborat stek va navbat hosil, ularning eng katta va eng kichik qiymatlarining yig’indisini hiosoblang.

#include <bits/stdc++.h>

using namespace std;

int main() {

    stack<int> s;

    queue<int> q;

    // n ta element qo'shish

    int n = 10;

    for (int i = 1; i <= n; i++) {

        s.push(rand() % 100);

        q.push(rand() % 100);

    }

    // eng katta qiymat hisoblanishi

    int maxS = INT\_MIN, maxQ = INT\_MIN;

    while (!s.empty()) {

        int a = s.top();

        maxS = max(maxS, a);

        s.pop();

    }

    while (!q.empty()) {

        int a = q.front();

        maxQ = max(maxQ, a);

        q.pop();

    }

    // eng kichik qiymat hisoblanishi

    int minS = INT\_MAX, minQ = INT\_MAX;

    for (int i = 1; i <= n; i++) {

        int a = rand() % 100;

        s.push(a);

        q.push(a);

        minS = min(minS, a);

        minQ = min(minQ, a);

    }

    cout << "Stack: " << endl;

    cout << "Eng katta qiymat: " << maxS << endl;

    cout << "Eng kichik qiymat: " << minS << endl;

    cout << "Queue: " << endl;

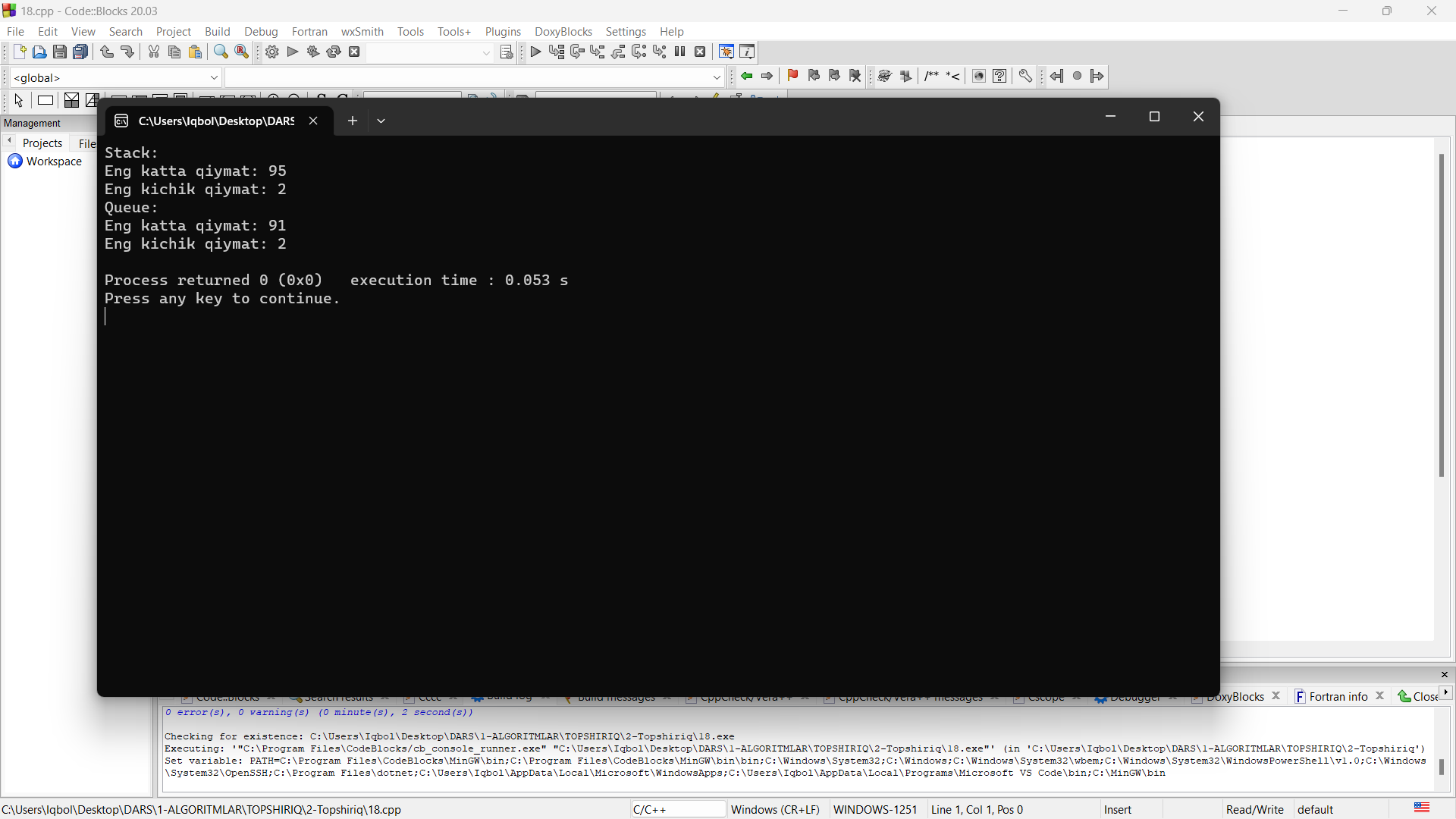
    cout << "Eng katta qiymat: " << maxQ << endl;

    cout << "Eng kichik qiymat: " << minQ << endl;

    return 0;

}

//NATIJA:



20. Funksiya yordamida navbatning toq va juft elementlarni sonini aniqlang va konteynerlarni to’liqligicha bo’shating.

#include <iostream>

#include <queue>

using namespace std;

void func(queue<int> &q, int &toq, int &juft) {

    while (!q.empty()) {

        int num = q.front();

        q.pop();

        if (num % 2 == 0) {

            juft++;

        } else {

            toq++;

        }

    }

}

int main() {

    queue<int> q;

    int n, a;

    cin >> n;

    // Navbatga elementlar qo'shish

    for(int i=0; i<n; i++){

        cin >> a;

        q.push(a);

    }

    int toq = 0, juft = 0;

    func(q, toq, juft);

    cout << "Toq sonlar soni: " << toq << endl;

    cout << "Juft sonlar soni: " << juft << endl;

    while (!q.empty()) {

        q.pop();

    }

    if (q.empty()) {

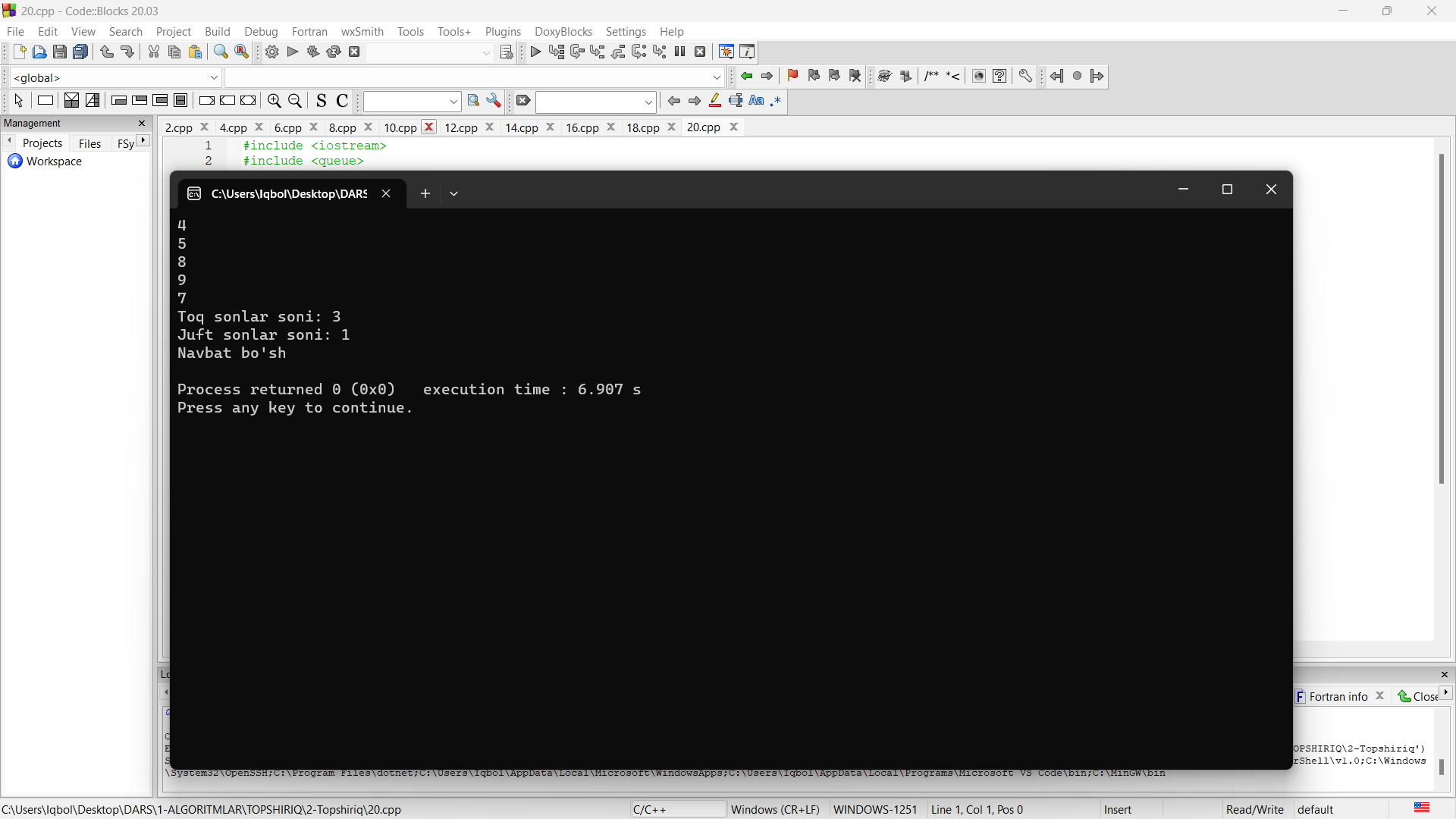
        cout << "Navbat bo'sh" << endl;

    }

    return 0;

}

//NATIJA:



22. Stek va navbat hosil qiling. Ulardan funksiyalarda foydalaning.

#include <iostream>

#include <stack>

#include <queue>

using namespace std;

// Stek va navbatni

void stek() {

  stack<int> s;

  s.push(10);

  s.push(20);

  s.push(30);

  cout << "Stekdagi elementlar: ";

  while (!s.empty()) {

    cout << s.top() << " ";

    s.pop();

  }

  cout << endl;

}

void navbat() {

  queue<int> q;

  q.push(10);

  q.push(20);

  q.push(30);

  cout << "Navbatdagi elementlar: ";

  while (!q.empty()) {

    cout << q.front() << " ";

    q.pop();

  }

  cout << endl;

}

int main() {

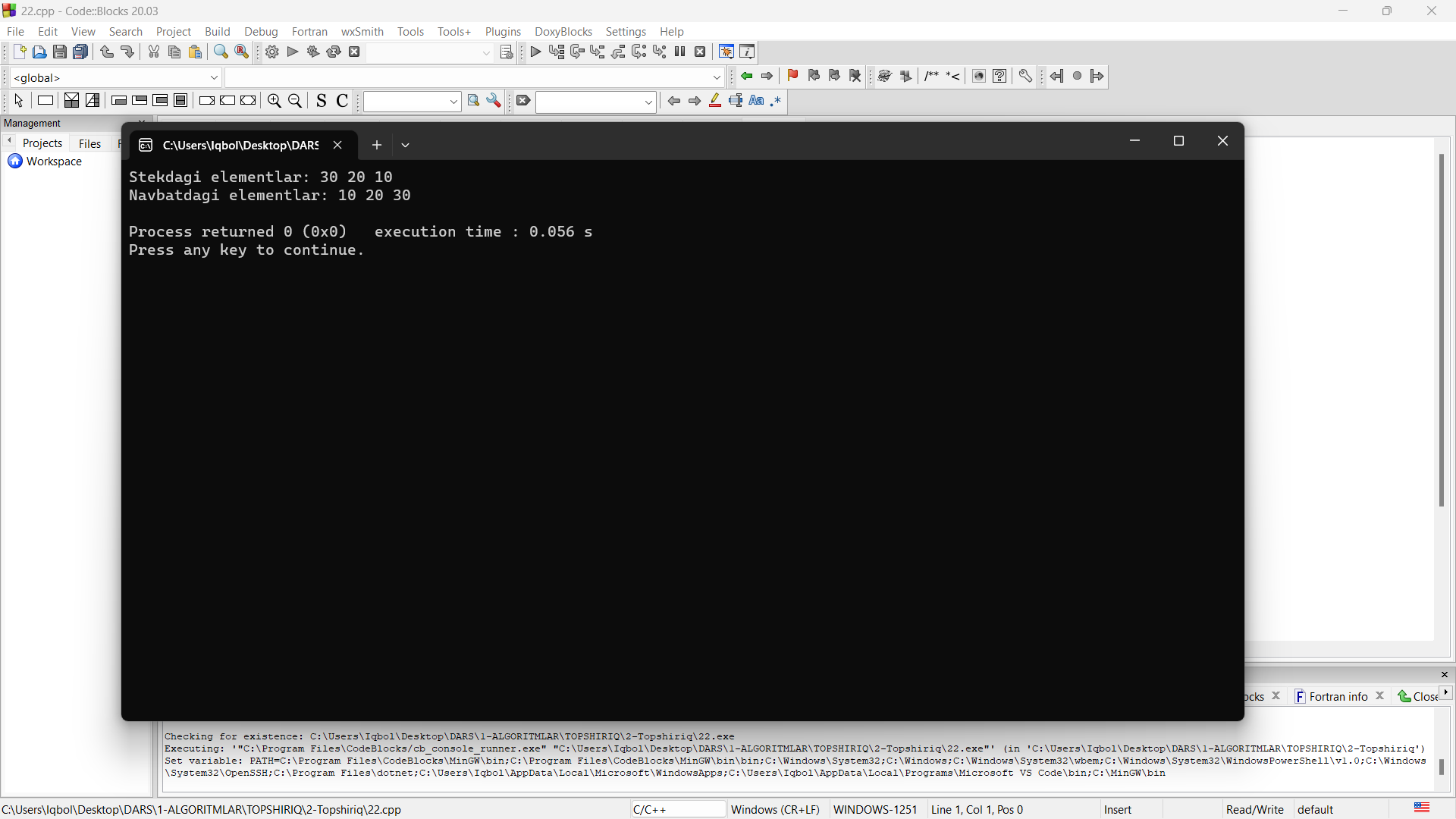
  stek();

  navbat();

  return 0;

}

//NATIJA:



24. Stek hosil qiling. Stekni birinchi va oxirgi elemenlar yigindisini hisoblang.

#include <iostream>

#include <stack>

using namespace std;

int main() {

    stack<int> s;

    int n, a;

    cin >> n;

    // Navbatga elementlar qo'shish

    for(int i=0; i<n; i++){

        cin >> a;

        s.push(a);

    }

    // Birinchi va oxirgi elementlar yig'indisini hisoblaymiz

    int first = s.top(); // Birinchi elementni olish

    for(int i=0; i<n-1; i++){

        s.pop();

    }

    int last = s.top(); // Oxirgi elementni olish

    s.pop();

    // Hisoblangan yig'indisni ekranga chiqaramiz

    cout << "Birinchi va oxirgi elementlar yig'indisi: " << first + last << endl;

    return 0;

}

//NATIJA:

