Võ Mai Phương 2114485

Câu 1:

int countriesNum(*Node* \**pHead*)

{

*Node* \*temp = *pHead*->nextName;

    int count = 0;

    while (temp != nullptr)

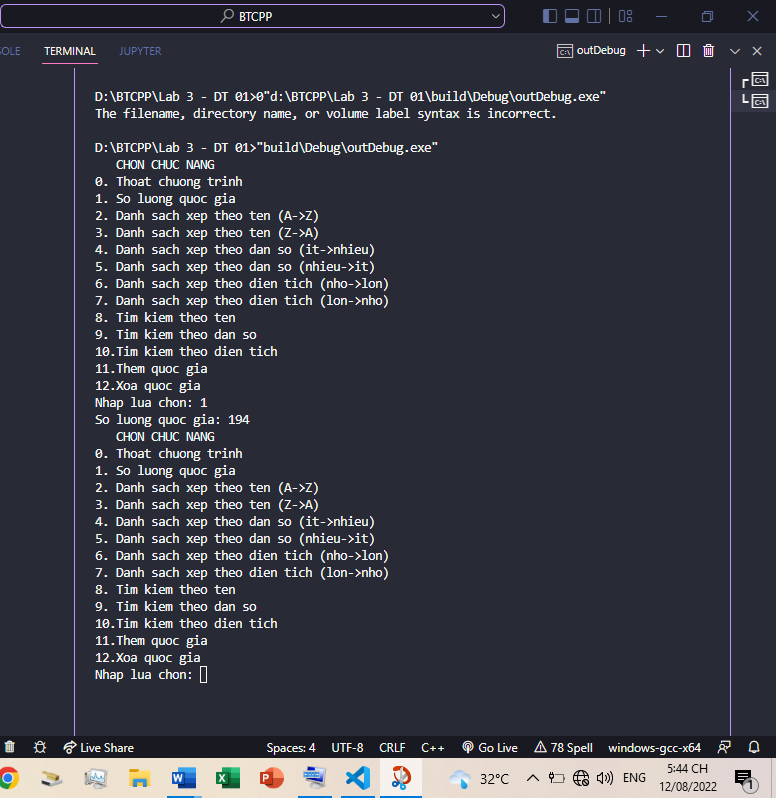
    {

        count++;

        temp = temp->nextName;

    }

    return count;

}

Câu 2:

void listbyNameAsc(*Node* \**pHead*)

{

*Node* \*temp = *pHead*->nextName;

    while (temp)

    {

        cout << setw(30) << left << temp->info.strName;

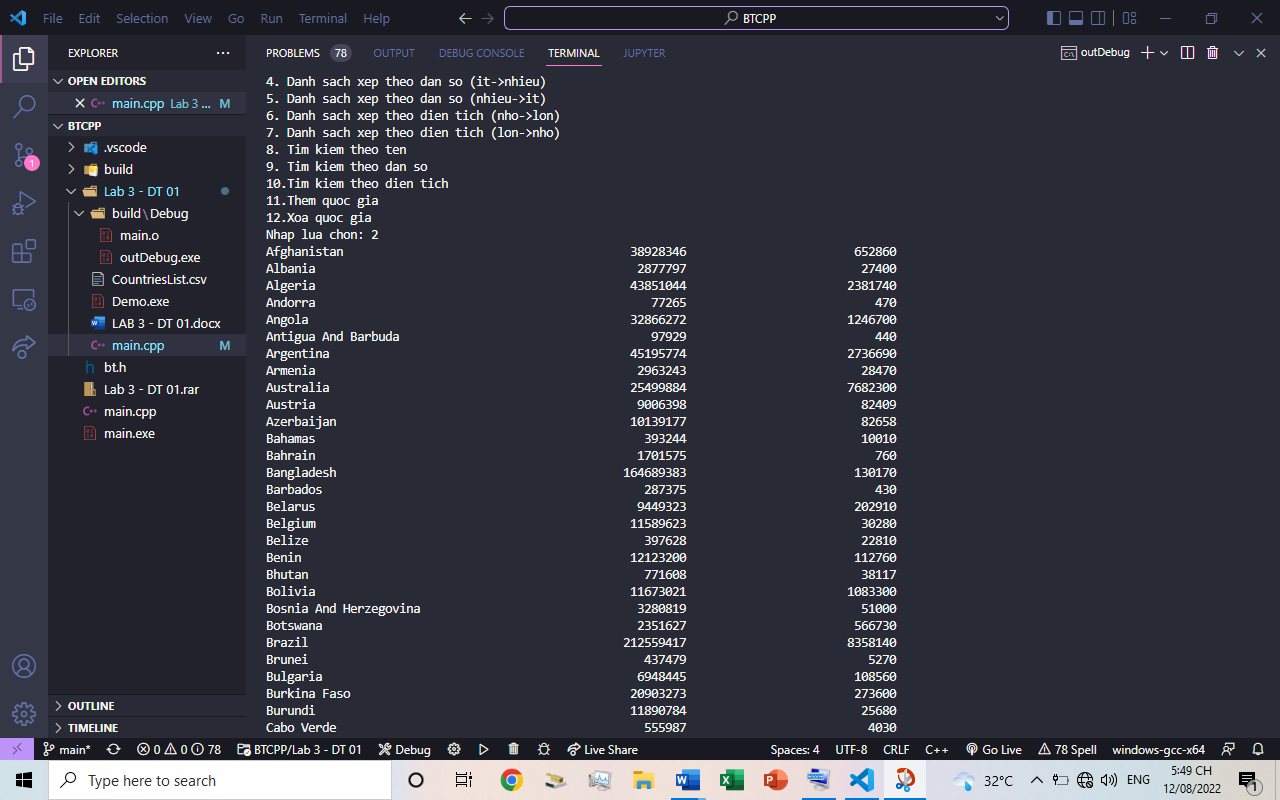
        cout << setw(30) << right << temp->info.nPopulation;

        cout << setw(30) << right << temp->info.nLandArea << endl;

        temp = temp->nextName;

    }

}

câu 3:

void listbyNameDes(*Node* \**pHead*)

{

    int sizeLinkedList = countriesNum(*pHead*);

*Node* \*\*arrNode = **new** *Node* \*[sizeLinkedList];

*Node* \*temp = *pHead*->nextName;

    for (int i = 0; i < sizeLinkedList; i++)

    {

        arrNode[i] = temp;

        temp = temp->nextName;

    }

    for (int i = sizeLinkedList - 1; i >= 0; i--)

    {

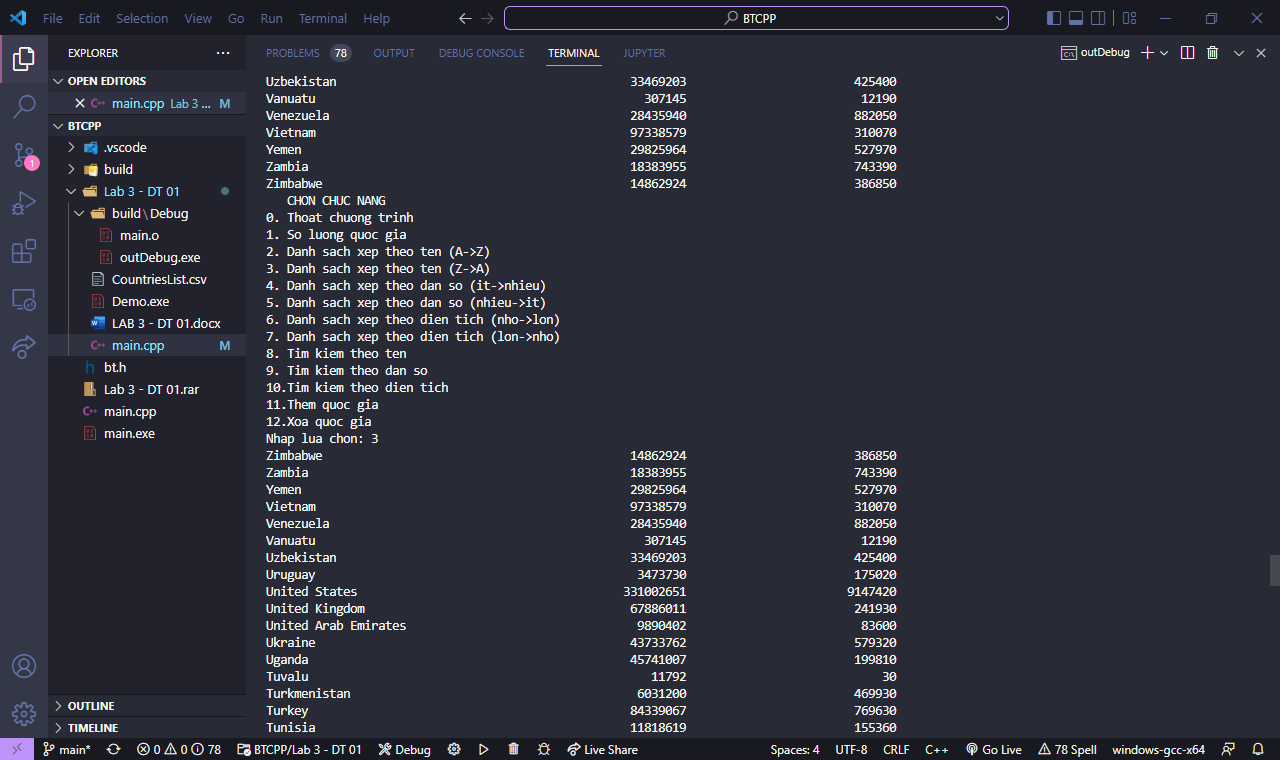
        cout << setw(30) << left << arrNode[i]->info.strName;

        cout << setw(30) << right << arrNode[i]->info.nPopulation;

        cout << setw(30) << right << arrNode[i]->info.nLandArea << endl;

    }

}

câu 4:

void listbyPopulationAsc(*Node* \**pHead*)

{

*Node* \*temp = *pHead*->nextPopulation;

    while (temp)

    {

        cout << setw(30) << left << temp->info.nPopulation;

        cout << setw(30) << right << temp->info.strName;

        cout << setw(30) << right << temp->info.nLandArea << endl;

        temp = temp->nextPopulation;

    }

}

câu 5:

void listbyPopulationDes(*Node* \**pHead*)

{

    int sizeLinkedList = countriesNum(*pHead*);

*Node* \*\*arrNode = **new** *Node* \*[sizeLinkedList];

*Node* \*temp = *pHead*->nextPopulation;

    for (int i = 0; i < sizeLinkedList; i++)

    {

        arrNode[i] = temp;

        temp = temp->nextPopulation;

    }

    for (int i = sizeLinkedList - 1; i >= 0; i--)

    {

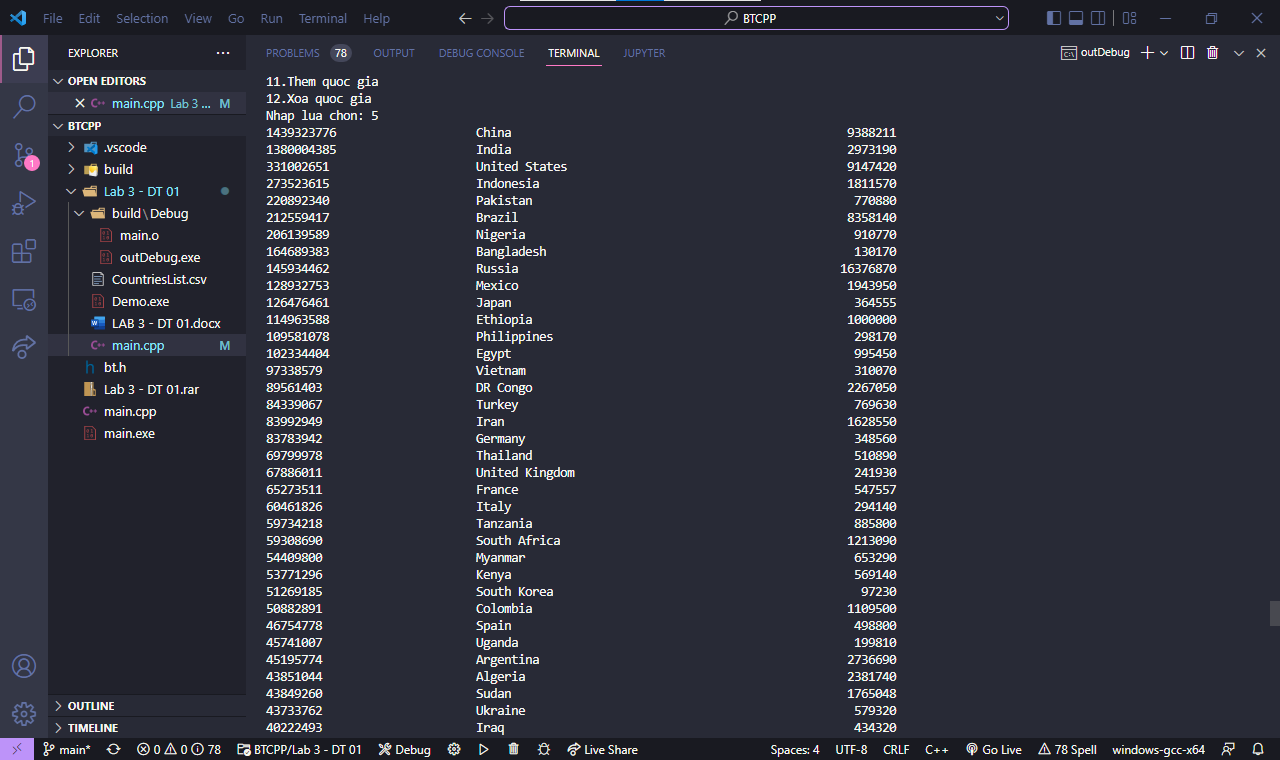
        cout << setw(30) << left << arrNode[i]->info.nPopulation;

        cout << setw(30) << left << arrNode[i]->info.strName;

        cout << setw(30) << right << arrNode[i]->info.nLandArea << endl;

    }

}

câu 6:

void listbyAreaAsc(*Node* \**pHead*)

{

*Node* \*temp = *pHead*->nextLandArea;

    while (temp)

    {

        cout << setw(20) << left << temp->info.nLandArea;

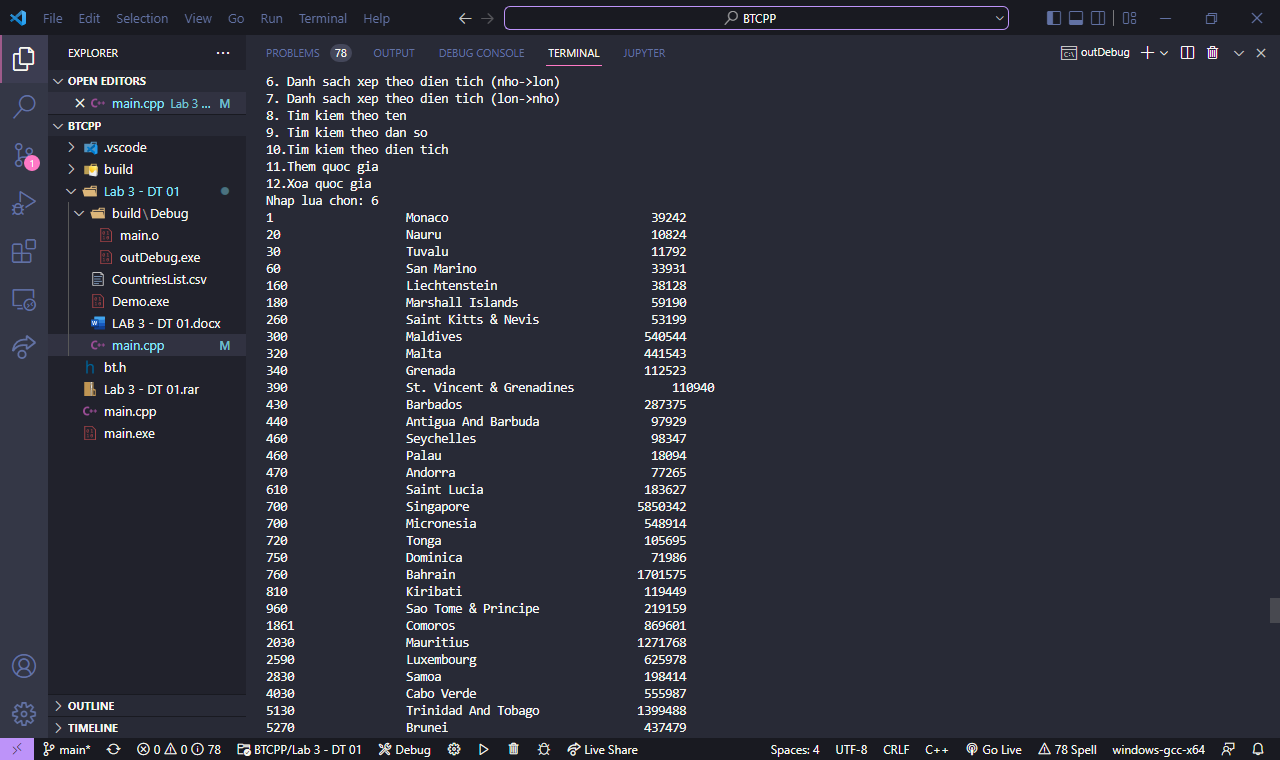
        cout << setw(20) << left << temp->info.strName;

        cout << setw(20) << right << temp->info.nPopulation << endl;

        temp = temp->nextLandArea;

    }

}

câu 7:

void listbyAreaDes(*Node* \**pHead*)

{

    int sizeLinkedList = countriesNum(*pHead*);

*Node* \*\*arrNode = **new** *Node* \*[sizeLinkedList];

*Node* \*temp = *pHead*->nextLandArea;

    for (int i = 0; i < sizeLinkedList; i++)

    {

        arrNode[i] = temp;

        temp = temp->nextLandArea;

    }

    for (int i = sizeLinkedList - 1; i >= 0; i--)

    {

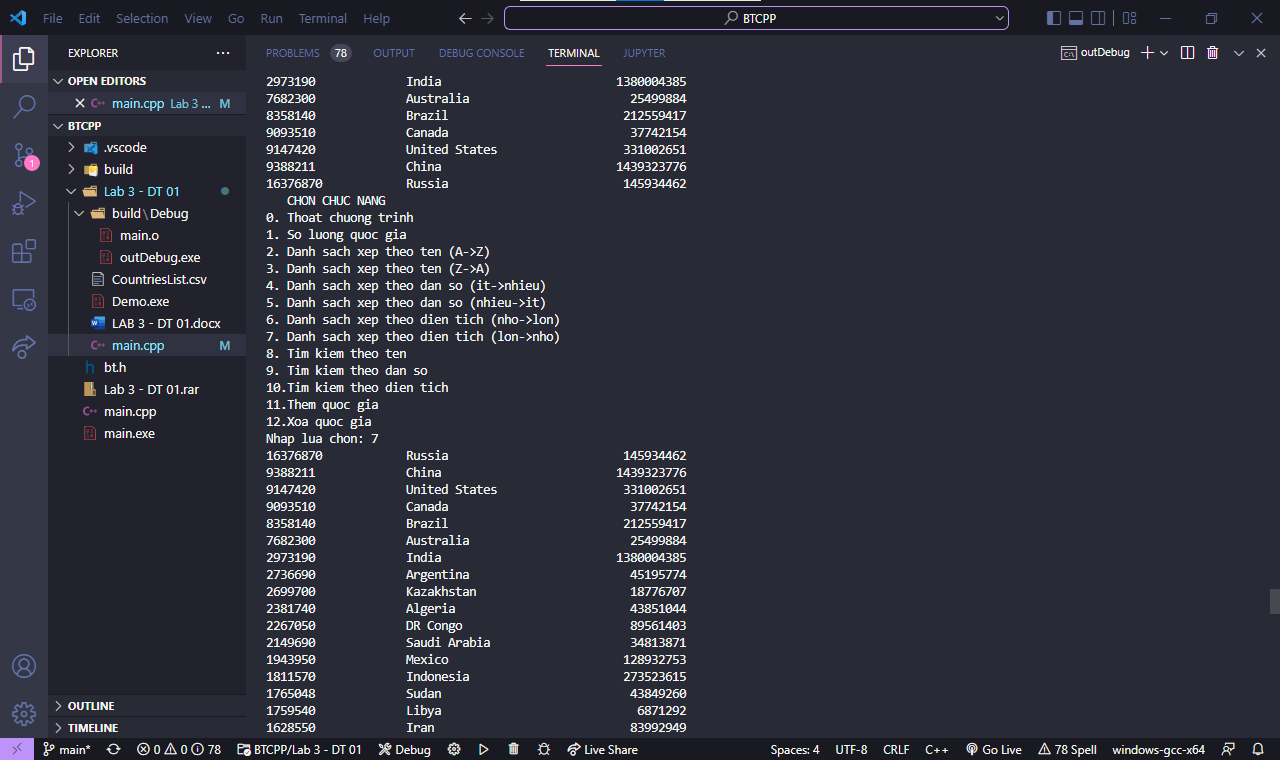
        cout << setw(20) << left << arrNode[i]->info.nLandArea;

        cout << setw(20) << left << arrNode[i]->info.strName;

        cout << setw(20) << right << arrNode[i]->info.nPopulation << endl;

    }

}

câu 8:

void normalizeName(*string* & *fullName*)

{

    char \*tmp = 0;

    int i = 0, j = 0, length = *fullName*.length();

    // Cấp phát bộ nhớ cho tmp

    tmp = (char\*)malloc((length+1)\*sizeof(char));

    // Thực hiện loại bỏ khoảng trắng

    while (i < length){

        // Nếu là chữ cái hoặc số

        if (isalnum(*fullName*[i])){

            tmp[j++] = *fullName*[i]; // đưa chữ cái hoặc số vào tmp

        }

        // nếu là dấu trắng và đằng sau phải là chữ cái hoặc số

        else if (j && (*fullName*[i] == ' ') && isalnum(*fullName*[i+1]))

        {

            tmp[j++] = *fullName*[i]; // đưa khoảng trắng vào tmp

        }

        i++;

    }

    tmp[j] = 0; // thêm kí tự NULL vào cuỗi chuỗi tmp

    // Thực hiện chuyển các kí tự đầu của từ thành IN HOA

    for (i = 0; i < j; i++){

        if (!i || ((tmp[i-1] == ' ') && isalpha(tmp[i]))){

            tmp[i] = toupper(tmp[i]);

        }

        else tmp[i] = tolower(tmp[i]);

    }

    // Cóp ngược trở lại theo yêu cầu của đề bài

*fullName* = "";

    for (int i = 0; i < length; i++)

    {

*fullName* += tmp[i];

    }

}

*Node*\* searchbyName(*Node* \**pHead*)

{

*string* name;

    cout << "Nhap ten quoc gia muon tim: ";

    std::cin.ignore(32767, '\n');

    getline(cin, name);

    normalizeName(name);

    // cout << name << endl;

*Node* \*p = *pHead*->nextName;

    while (p)

    {

        if (p->info.strName == name) return p;

        else p = p->nextName;

    }

    return NULL;

}

câu 9:

void searchbyPopulation(*Node* \**pHead*)

{

    int nMin = 0;

    int nMax = 0;

    cout << "nhap vao khoang gia tri ma ban muon tim: " << endl;

    cin >> nMin;

    cin >> nMax;

*Node* \*temp = *pHead*->nextPopulation;

    while (temp)

    {

        if (nMin <= temp->info.nPopulation && temp->info.nPopulation <= nMax)

        {

            cout << setw(30) << left << temp->info.strName;

            cout << setw(30) << right << temp->info.nPopulation;

            cout << setw(30) << right << temp->info.nLandArea << endl;

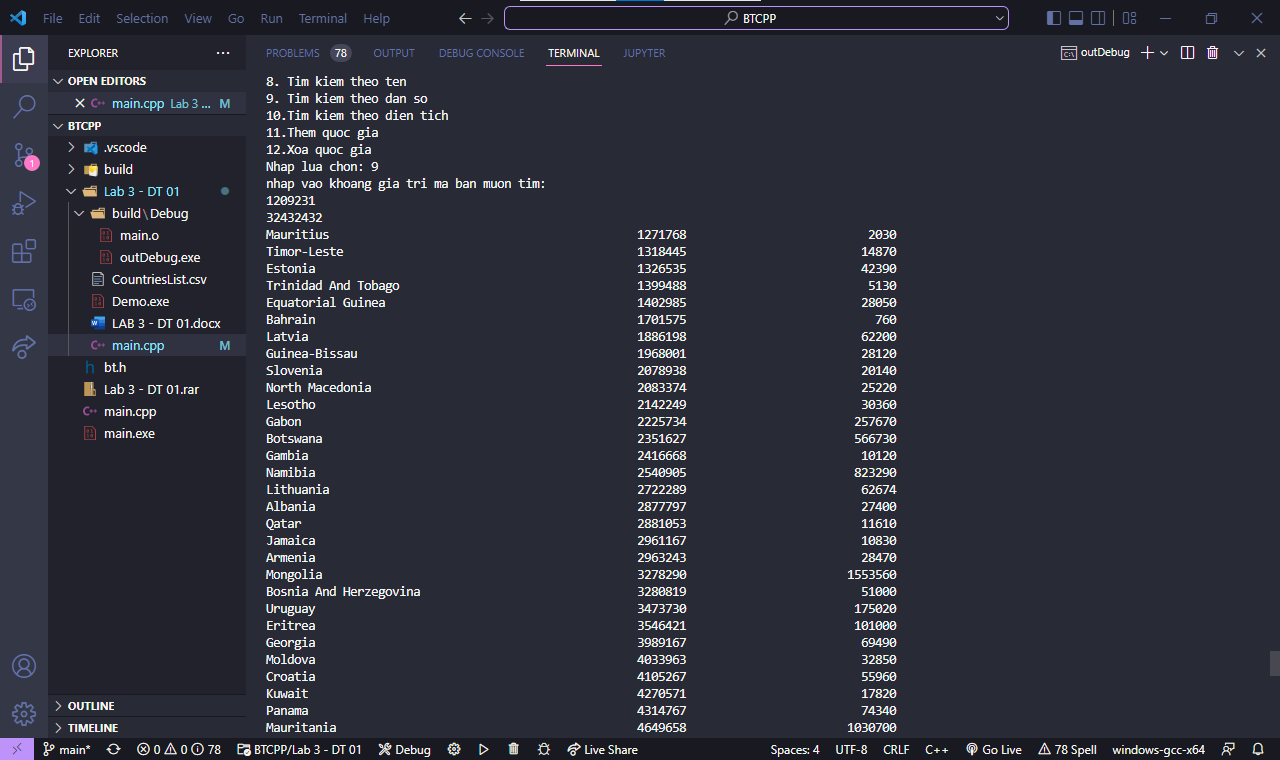
            temp = temp->nextPopulation;

        }

        else temp = temp->nextPopulation;

    }

}

câu 10:

void searchbyArea(*Node* \**pHead*)

{

    int nMin = 0;

    int nMax = 0;

    cout << "nhap vao khoang gia tri ma ban muon tim: " << endl;

    cin >> nMin;

    cin >> nMax;

*Node* \*temp = *pHead*->nextLandArea;

    while (temp)

    {

        if (nMin <= temp->info.nLandArea && temp->info.nLandArea <= nMax)

        {

            cout << setw(30) << left << temp->info.strName;

            cout << setw(30) << right << temp->info.nPopulation;

            cout << setw(30) << right << temp->info.nLandArea << endl;

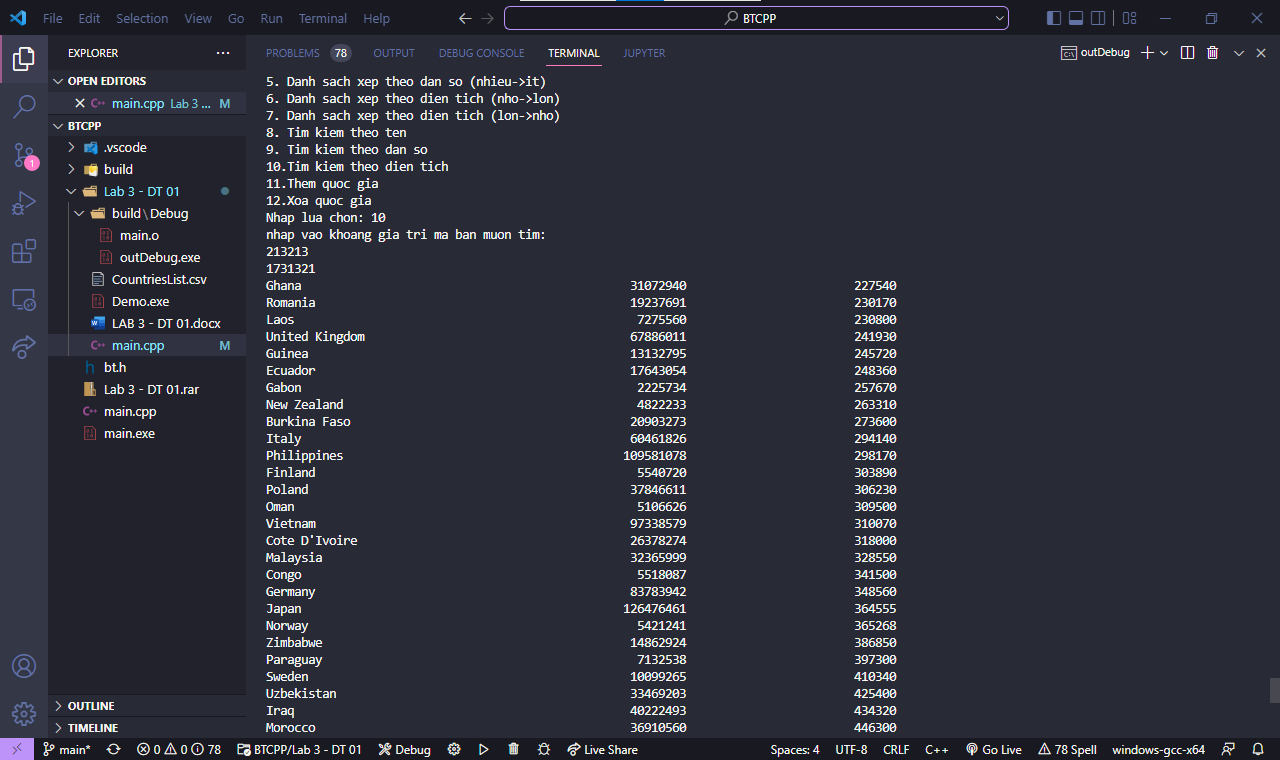
            temp = temp->nextLandArea;

        }

        else temp = temp->nextLandArea;

    }

}

câu 11:

void insertCountry(*Node* \*&*pHead*)

{

*string* name = "";

    int nPopulation = 0;

    int nLandArea = 0;

    cout << "Nhap ten quoc gia muon them: ";

    std::cin.ignore(32767, '\n');

    getline(cin, name);

    cout << "Nhap dan so quoc gia muon them: ";

    cin >> nPopulation;

    cout << "Nhap dien tich quoc gia muon them: ";

    cin >> nLandArea;

    normalizeName(name);

    // cout << name << endl;

*Node* \*p = *pHead*->nextName;

    bool country\_flag = false;

    while (p)

    {

        if (p->info.strName == name)

        {

            country\_flag = true;

            break;

        }

        else p = p->nextName;

    }

    if (!country\_flag)

    {

        Insert(*pHead*, name, nPopulation, nLandArea);

    }

}

cau 12:

void deleteCountry(*Node* \*&*pHead*)

{

*string* name;

    cout << "Nhap ten quoc gia muon xoa: ";

    std::cin.ignore(32767, '\n');

    getline(cin, name);

    normalizeName(name);

    // cout << name << endl;

*Node* \*p = *pHead*->nextName;

*Node* \*temp = *pHead*;

    // Node \*temp = pHead->nextName;

    while (p)

    {

        if (p->info.strName == name)

        {

            temp->nextName = p->nextName;

            temp->nextLandArea = p->nextLandArea;

            temp->nextPopulation = p->nextPopulation;

            p->nextLandArea = nullptr;

            p->nextName = nullptr;

            p->nextPopulation = nullptr;

            break;

        }

        else

        {

            temp = p;

            p = p->nextName;

        }

    }

}

