c++ 프로그래밍 및 실습

대학생 대상 금융 정보 제공 시스템 개발 프로젝트

진척 보고서 #2

제출일자: 2023.12.17(일)
제출자명: 김채흔
제출자학번: 223445

1. 프로젝트 목표

1) 배경 및 필요성

대학생들이 자취 등으로 돈이 필요해 대출을 하거나, 적금/예금 등의 경제활동을 시작할 때 정보를 알지 못해 신청하지 못하는 경우가 많음. 또한 지원해주는 정책을 알지 못해 혜택이 있지만, 이를 알지도 못해 신청하지 못하는 경우가 많음. 이를 방지하기 위해 각자 상황에 맞는 정책 정보나 금융 상품 등의 금융 정보를 제공해주는 프로그램이 필요함.

2) 프로젝트 목표

사용자가 입력한 문제에 따른 금융 및 정책 정보를 자동적으로 제공.

3) 차별점

기존 사이트는 대출, 적금 등 다양한 정보와 정책 관련 정보가 여기저기에 있어서 경제활동을 이제 처음 시작하는 대학생들이 정보에 접근하기에 쉽지 않음. 또한, 은행 별로 사이트가 전부 달라 하나하나 비교해보아야 함. 이를 보완하여 사용자의 상황에 맞는 금융 정보를 제공하여 대학생들이 손쉽게 정보를 얻을 수 있도록 함.

2. 기능 계획

1) 기능 1 대학생 대상 정책 추천 기능

- 설명

사용자가 입력한 상황에 따라 정책을 추천해 주는 기능

(1) 세부 기능 1 정부 지원 정책 데이터 수집

- 설명

현재 정부에서 지원하고 있는 청년 관련 정책들 정보를 수집하여 지역별, 분야별로 세분화함.

(2) 세부 기능 2 추천 시스템

- 설명

지역/분야의 정보를 입력받고 조건에 해당하는 정책 정보를 출력

(3) 세부 기능 3 자세한 정보 제공

- 설명

사용자가 정보를 자세히 알고자 하면 세부사항 출력(사이트 정보 포함)

2) 기능 2 청년 대상 금융 정보 제공 기능

- 설명

대학생들이 받을 수 있는 대출 정보와 청년 대상 금융 상품을 함께 추천해준다.

(1) 세부 기능 1. 은행 별 금융 상품 수집

- 설명

은행 별 청년 대상 금융 상품과 대출 정보 데이터를 수집한다.

(2) 세부기능 2. 사용자 상황 맞춤 금융 정보 제공

- 설명

소득 수준/나이대/ 구직자 or 학생 등 사용자의 상황과 대출/적금 등 사용자의 목적을 입력받고 이에 해닿하는 정보를 제공한다

(3) 세부기능 3. 구체적인 정보 제공

- 설명

사용자가 특정 정보에 대해 더 알고 싶어 할 경우, 관련 사이트 정보를 통해 구체적인 정보를 제공한다.

(4) 세부기능 4. 사용자 맞춤 추천

- 설명

사용자에게 입력받은 정보들을 바탕으로 가장 많이 검색한 순으로 상단에 띄워 추천해준다.

3. 진척사항

1) 기능 구현

(1) 정부 지원 정책 데이터 수집

- 입출력

<financial.cpp>

- class YouthPolicy

Fields = 분야 카테고리 벡터

Regions = 정책 카테고리 벡터

- isValid함수

userinput = 사용자 입력 변수

ValidOptions = 유효성 검증할 벡터 변수

<csv.h>

- class CSVReader

string filename_ = CSV 파일의 경로를 저장하는 변수
char delimiter_ = CSV 파일에서 열을 구분하는 데 사용되는 구분자를 저장하는 변수
- CSVReader 생성자

filename = CSV 파일의 경로를 나타내는 문자열

char delimiter = CSV 파일에서 열을 구분하는 데 사용되는 구분자(',' 쉼표)
- readData함수

line = 파일에서 한 행을 저장하는 변수

data = CSV 파일의 데이터가 저장된 벡터

row = 한 행의 데이터가 저장된 벡터

- 설명

엑셀 데이터를 가져와 지역과 분야를 카테고리에 맞게 나눈 후, 분야와 정책 카테고리를 벡터로 설정해 사용자가 입력한 지역이나 분야가 카테고리에 있는지 검증한다.

- 적용된 배운 내용

클래스 : 정책 클래스를 만들어 정책 관련된 함수를 쉽게 쓸 수 있도록 한다.

벡터 : 벡터를 활용하여 카테고리를 세분화한다.

함수 : bool isValid(~) 함수를 활용하여 사용자 입력의 유효성을 검증한다.

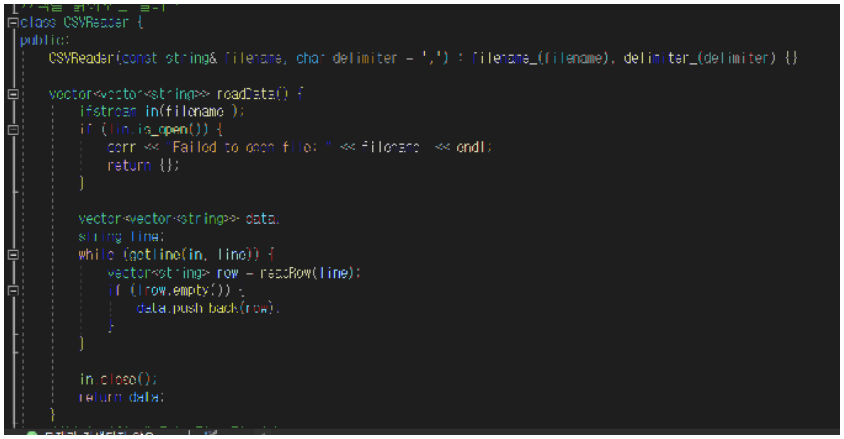
클래스, 생성자 : 클래스와 생성자를 활용하여 financial.cpp에서 엑셀파일 데이터를 읽어올 수 있도록 한다.

파일 입출력 : 엑셀 데이터를 불러와서 정보를 저장하여 출력할 수 있도록 한다.

- 코드 스크린샷

<financial.cpp>

<csv.h>



(2) 추천 시스템

- 입출력

<financial.cpp>

-Field, Region, PolicyShow 함수

FieldPolicy = 사용자에게 분야 입력받는 변수

RegionPolicy = 사용자에게 지역 입력받는 변수

<csv.h>

- PrintValue 함수

columnName1 = 첫 번째 열의 열 이름

value1 = 첫 번째 열에서 찾고자 하는 값

columnName2 = 검색하려는 두 번째 열의 열 이름
value2 = 두 번째 열에서 찾고자 하는 값
outputColumn = 출력하려는 열의 열 이름
data = CSV 파일에서 읽은 전체 데이터 벡터
columnIndex1, columnIndex2 = 각각 첫 번째 및 두 번째 검색 열의 인덱스
outputIndex = 출력하려는 열의 인덱스

- 설명

사용자가 분야와 지역을 입력하면 그에 해당하는 행을 찾고, 해당 행에서 정책 정보 열

의 값을 출력

- 적용된 배운 내용

반복문(while) : 사용자가 올바른 입력을 할 때까지 오류 출력

반복문(for) : 사용자가 입력한 값에 해당하는 행 찾고 출력하는데 활용

함수 : 사용자에게 분야와 지역을 입력받는 코드 void Field(), void Region()로

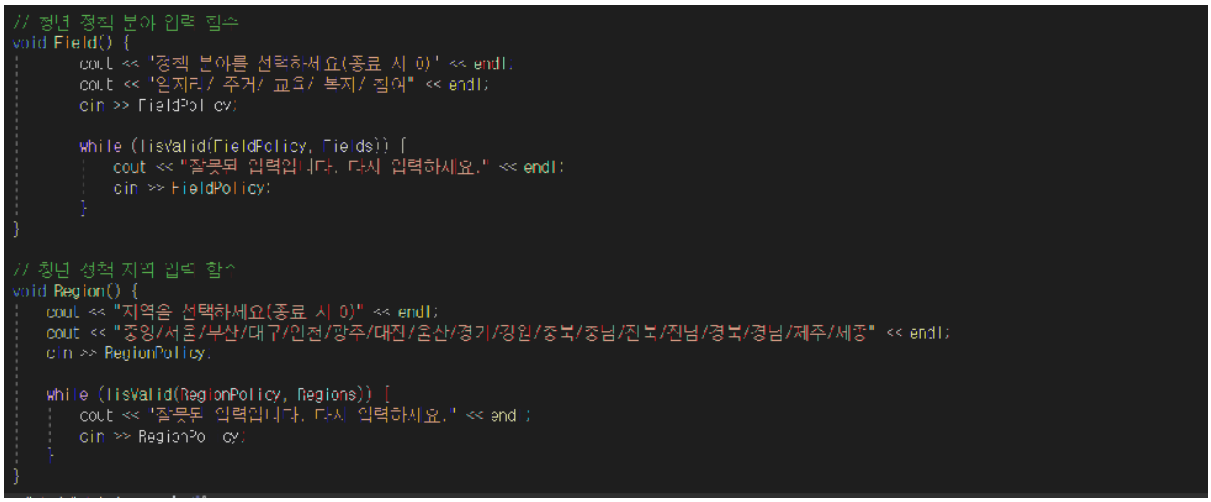
함수화하여 활용

: void PolicyShow()로 사용자가 입력한 조건에 맞는 정책 출력

헤더 파일, 함수 : 엑셀 데이터에서 사용자가 입력한 조건에 맞는 정책을 가져오는 코드를 csv.h 의 PrintValue() 함수로 사용

- 코드 스크린샷

financial.cpp

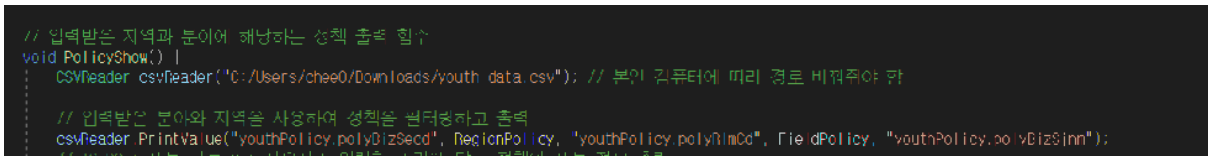


```
// 청년 정책 분야 입력 함수
void Field() {
        cout << '정책 분야를 선택하세요(종료 시 0)' << endl;
        cout << '일자리/ 주거/ 교육/ 복지/ 참여" << endl;
        cin >> FieldPolicy;

        while (!isValid(FieldPolicy, Fields)) {
            cout << "잘못된 입력입니다. 다시 입력하세요." << endl;
            cin >> FieldPolicy;
        }
}

// 청년 정책 지역 입력 함수
void Region() {
    cout << "지역을 선택하세요(종료 시 0)" << endl;
    cout << "중앙/서울/부산/대구/인천/광주/대전/울산/경기/강원/충북/충남/전북/전남/경북/경남/제주/세종" << endl;
    cin >> RegionPolicy;

    while (!isValid(RegionPolicy, Regions)) {
        cout << '잘못된 입력입니다. 다시 입력하세요." << end ;
        cin >> RegionPo  cy;
    }
}
```



```
// 입력받은 지역과 분야에 해당하는 정책 출력 함수
void PolicyShow() {
    CSVReader csvReader("C:/Users/chee0/Downloads/youth data.csv"); // 본인 컴퓨터에 따라 경로 바꿔주어야 함

    // 입력받은 분야와 지역을 사용하여 정책을 필터링하고 출력
    csvReader.PrintValue("youthPolicy.polyBizSecd", RegionPolicy, "youthPolicy.polyRlmCd", FieldPolicy, "youthPolicy.polyBizSjnm");
```

csv.h

```cpp
//분야,지역 에 맞는 값 출력 함수
void PrintValue(const string& columnName1, const string& value1,
    const string& columnName2, const string& value2,
    const string& outputColumn) {
    vector<vector<string>> data = readData();
    if (data.empty()) {
        cerr << "Empty data" << endl;
        return;
    }

    // 특정 열 찾기
    int columnIndex1 = -1;
    int columnIndex2 = -1;
    for (size_t i = 0; i < data[0].size(); ++i) {
        if (data[0][i] == columnName1) {
            columnIndex1 = static_cast<int>(i);
        }
        if (data[0][i] == columnName2) {
            columnIndex2 = static_cast<int>(i);
        }
    }

    if (columnIndex1 == -1 || columnIndex2 == -1) {
        cerr << "One or both columns not found" << endl;
```

```cpp
    // 특정 열에 특정 값 갖는 행 출력
    for (size_t i = 1; i < data.size(); ++i) {
        if (columnIndex1 < data[i].size() && columnIndex2 < data[i].size() &&
            data[i][columnIndex1] == value1 && data[i][columnIndex2] == value2) {
            if (outputColumn == "") {
                for (const auto& cell : data[i]) {
                    cout << "[" << cell << "]" << "₩t";
                }
            }
            else {
                int outputIndex = -1;
                for (size_t j = 0; j < data[0].size(); ++j) {
                    if (data[0][j] == outputColumn) {
                        outputIndex = static_cast<int>(j);
                        break;
                    }
                }

                if (outputIndex != -1) {
                    cout << "[" << data[i][outputIndex] << "]";
                }
                else {
                    cerr << "Output column not found" << endl;
```

(3) 자세한 정보 제공

- 입출력

<financial.cpp>

FieldPolicy = 사용자에게 분야 입력받는 변수

RegionPolicy = 사용자에게 지역 입력받는 변수

input = 사용자에게 자세한 정보 출력 여부 입력받기 위한 변수

PrintInfo() = 특정 정책 자세한 정보 출력 함수

<csv.h>

- getInfo함수

Region: 검색하려는 지역
Field: 검색하려는 분야
filename_ = CSV 파일의 경로를 저장하는 변수

file = 파일을 열 때 사용되는 변수

line = 한 행의 데이터를 저장하는 문자열

iss = 한 행의 csv 데이터를 , 로 구분하기 위한 변수

token = 구분한 각 열 값 저장 변수

tokens = 열 값을 모으기 위한 벡터

result = 조건 만족하는 행을 저장 벡터

- PrintInfo 함수

info = 조건 만족하는 행들 저장 벡터

row = info의 행을 하나씩 읽기 위한 변수

cell = 행에서 열 값을 하나씩 읽기 위한 변수

- 설명

사용자가 1을 입력하면 사용자가 입력한 조건에 해당하는 행을 찾아 열 정보 값을

모두 출력

- 적용된 배운 내용

조건문 : 사용자에게 1을 입력받았을 때 자세한 정보 출력

: csv.h 에서 파일이 열리지 않았을 때 오류 출력

헤더 파일, 함수 : 정책의 세부정보를 출력하는 코드를 헤더파일 csv의 PrintInfo 함수를 호출하여 활용

벡터 : 특정 조건 행을 찾고 이를 반환하는 함수에서 여러 변수를 벡터로 설정

반복문(while) : 데이터의 행과 열을 읽어 저장할 때 활용

- 코드 스크린샷

financial.cpp

```
// TO DO : 세부 기능 3 : 사용자가 입력한 조건에 맞는 정책의 세부 정보 출력
cout << "자세한 정보를 보고 싶다면 1을 입력하세요" << endl;
cin >> input;
if (input == "1")
    csvReader.PrintInfo( RegionPolicy, FieldPolicy);
```

csv.h

```
// CSV 파일에서 특정 조건을 만족하는 행을 std::vector에 저장하는 함수
vector<vector<string>> getInfo(const string& Region, const string& Field) {
    vector<vector<string>> result;

    ifstream file(filename_);
    if (!file.is_open()) {
        cerr << "파일을 열 수 없습니다." << endl;
        return result;
    }

    std::string line;
    while (getline(file, line)) {
        istringstream iss(line);
        string token;
        vector<string> tokens;

        while (getline(iss, token, ',')) {
            tokens.push_back(token);
        }

        if (tokens.size() >= 33 && tokens[2] == Region && tokens[32] == Field) {
            result.push_back(tokens);
        }
```

```cpp
// CSV 파일에서 특정 조건을 만족하는 행을 출력하는 함수
void PrintInfo(const string& Region, const string& Field) {
    vector<vector<string>> info = getInfo(Region, Field);

    // 저장된 행들 출력
    for (const auto& row : info) {
        for (const auto& cell : row) {
            cout << "[" << cell << "]" << "₩n";
        }
        cout << endl;
    }
}
```

(4) 자세한 정보 제공 수정

- 입출력

<financial.cpp>

- PolicyShow 함수

RegionPolicy = 사용자 지역 입력 변수

FieldPolicy = 사용자 분야 입력 변수

<csv.h>

- PrintInfo 함수

Region = 지역 필터링 변수
Field= 분야 필터링 변수

input = 정책 번호 변수

info = 필터링한 정책 저장 변수

- 설명

기존의 사용자가 입력한 조건에 맞는 모든 정책의 자세한 정보를 출력했던 것에서,

사용자에게 자세한 정보를 제공할 정책 번호를 입력받고 해당 정책의 세부 사항만을 출력한다.

- 적용된 배운 내용

벡터 : 필터링된 정책을 벡터로 저장한다.

함수 : PrintInfo, PolicyShow 함수를 활용하여 자세한 정보를 출력할때 필요한 코드를 함수화하여 사용하였다.

반복문, 조건문 : 반복문과 조건문으로 사용자가 0을 입력하기 전까지 입력받은 번호의 정책의 열들을 출력하도록 하였고, switch문을 사용해 사용자가 더 편하게 볼 수 있게끔 해당 열이 나타내는 것을 출력하였다.

- 코드 스크린샷

<financial.cpp>

```
//TO DO: 세부 기능 3 : 자세한 정보 출력
// 입력받은 지역과 분야에 해당하는 정책 출력 함수
void PolicyShow() {
    CSVReader csvReader("C:/Users/chee0/Downloads/download/YouthData.csv"); // 본인 컴퓨터에 따라 경로 바꿔줘야 함

    // 입력받은 분야와 지역을 사용하여 정책을 필터링하고 출력
    csvReader.PrintValue("youthPolicy.polyBizSecd", RegionPolicy, "youthPolicy.polyRlmCd", FieldPolicy, "youthPolicy.polyBizSjnm");
    csvReader.PrintInfo(RegionPolicy, FieldPolicy);
}
```

<csv.h>

```
// 필터링 된 정책 자세한 정보들 출력하는 함수
void PrintInfo(const string& Region, const string& Field) {
    vector<vector<string>> info = getInfo(Region, Field);
    int input=0;
    cout << "자세한 정보를 보고 싶은 정책의 번호를 입력하세요(없으면 0)" << endl;
    cin >> input;
    // 저장된 행들 출력
    while (input != 0) {
        // 출력할 행이 있는지 확인
        if (input >= 1 && input <= Policy.size()) {
            // 입력받은 행의 열들 출력
            for (const auto& cell : info[input - 1]) {
                // 일어서 필요한 줄바꿈 출력
                if (cell != "-") {
                    if ((&cell - &info[input - 1][0] > 2) or (&cell - &info[input - 1][0] < 9) or (&cell - &info[input - 1][0] > 32))
                        switch (&cell - &info[input - 1][0]) {
                        case 2:
                            cout << "지역: [" << cell << "]" << endl;
                            break;
                        case 3:
                            cout << "기관 및 지자체 구분: [" << cell << "]" << endl;
                            break;
                        case 4:
```

(5) 은행별 금융상품 수집

- 입출력

<financial.cpp>

class YouthFinancial = 금융정보 관련 기능 클래스

FinancialType = 사용자에게 금융상품 종류 입력 받는 변수

BankType = 사용자에게 은행명 입력받는 변수

FinancialItem = 금융상품 종류 벡터 변수

Banks = 은행명 벡터 변수

- isValie 함수

userInput = 유효성 체크할 사용자 입력 변수

ValidOptions = 유효성 체크할 벡터 변수

<csv.h>

CSVReader = 엑셀 데이터 불러오는 클래스(앞선 정책클래스에서 활용한 것과 동일)

- 설명

사용자에게 입력받은 은행명과 금융 상품 종류를 세분화하고, 사용자가 해당 카테고리 안에서 입력했는 지 유효성 검증을 한다.

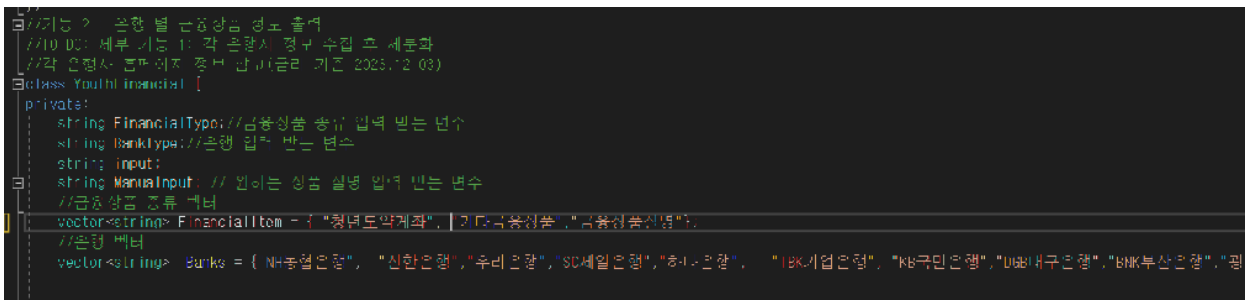
- 적용된 배운 내용

함수 : 유효성 검증을 함수화하여 지속적으로 사용할 수 있도록 했다.

벡터 : 은행 명과 금융 상품 종류를 벡터로 설정해 후에 기능 구현에 유용하도록 한다.

- 코드 스크린샷

<financial.cpp>



<csv.h>

```
class CSVReader {
public:
    CSVReader(const string& filename, char delimiter = ',') : filename_(filename), delimiter_(delimiter) {}
    // 필터링 된 정책 벡터
    vector <string> Policy = {};
    //엑셀 파일 읽어오기
    vector<vector<string>> readData() {
        ifstream in(filename_);
        if (!in.is_open()) {
            cerr << "Failed to open file: " << filename_ << endl;
            return {};
        }

        vector<vector<string>> data;
        string line;
        string currentRow;
        while (getline(in, line)) {
            currentRow += line;

            // 큰 텍스트 블록을 묶은 경우 처리
```

(6) 사용자 맞춤 금융정보 제공

- 입출력

<financial.cpp>

- Bank 함수

BankType = 사용자 입력 은행명

- Item 함수

FinancialType = 사용자 입력 금융상품 정보

- FinancialShow 함수

YouthAccount = 사용자가 입력한 은행에 맞는 청년도약계좌 정보 출력 함수

-ManualShow 함수

ManualInput = 사용자가 보고자 하는 금융상품 입력 변수

<csv.h>

- GetInfo함수

bank = 출력하려는 은행명
filename_ = CSV 파일의 경로를 저장하는 변수

file = 파일을 열 때 사용되는 변수

line = 한 행의 데이터를 저장하는 문자열

iss = 한 행의 csv 데이터를 , 로 구분하기 위한 변수

token = 구분한 각 열 값 저장 변수

tokens = 열 값을 모으기 위한 벡터

result = 조건 만족하는 행을 저장 벡터

- YouthAccount, FinancialManual 함수

info = 조건 만족하는 행들 저장

cell = 행에서 첫번째 열 값을 하나씩 읽기 위한 변수

- 설명

사용자가 금융 상품 종류(청년도약계좌)와 은행 명을 입력하면 그에 맞는

자세한 정보를 출력해준다.

사용자가 금융상품(청년도약계좌)의 설명을 얻고자 하면 이에 대한 정보를 출력해준다.

- 적용된 배운 내용

반복문 : 사용자에게서 입력받을 때 유효성 검증함수 확인

: 사용자가 보기 편하도록 각 열이 뜻하는 정보 출력(switch 문)

- 코드 스크린샷

financial.cpp

```
// 은행명 입력 함수
void Bank() {
    cout << "은행을 선택하세요 ?(종료 시 0)" << endl;
    cout << "-------------------------------------------------------------------------------------------------------------------" << endl;
    cout << "NH농협은행 /신한은행 /우리은행 /SC제일은행 /하나은행 /IBK기업은행 /KB국민은행 /DGB대구은행 /BNK부산은행 /JB우리은행 /전북은행 /BNK경남은행
    cout << "------------------------------------------------------------------------------------------------------------------- " << endl;
    cin >> BankType;

    while (!isValid(BankType, Banks)) {
        cout << "잘못된 입력입니다. 다시 입력하세요." << endl;
        cin >> BankType;
    }
```

```
// 금융 상품 입력 함수
void Item() {
    cout << "금융 상품을 선택하세요(종료 시 0)" << endl;
    cout << "----------------------------------------" << endl;
    cout << "청년도약계좌 /기타금융상품/금융상품설명" << endl;
    cout << "----------------------------------------"<< endl;
    cin >> FinancialType;

    while (!isValid(FinancialType, FinancialItem)) {
        cout << "잘못된 입력입니다. 다시 입력하세요." << endl;
        cin >> FinancialType;
    }
}
```

```
void ManualShow() {
    cout << "설명을 원하는 상품을 적어주세요." << endl;
    cout << "-------------------------" << endl;
    cout << "청년도약계좌/기타금융상품" << endl;
    cout << "-------------------------" << endl;
    cin >> ManualInput;
    CSVReader csvReader("C:/Users/chee0/Downloads/Info2.csv"); // 본인 컴퓨터에 따라 경로 바꿔줘야 함
    csvReader.FinancialManual(ManualInput);
}
```

csv.h

```
//사용자가 입력한 은행에 따른 청년도약계좌 정보 출력 함수
void YouthAccount( const string& value) {
    vector<vector<string>> info = GetInfo(value);
    // 저장된 행들 출력
    for (const auto& cell : info[0]) {
        //     // 각 열의 정보 출력
     switch (&cell - &info[0][0]){
                case 0:
                    cout << "은행명: [" << cell << "]" << endl;
                    break;
                case 1:
                    cout << "기본 금리(3년 고정): [" << cell << "]" << endl;
                    break;
                case 2:
                    cout << "소득 우대금리: [" << cell << "]" << endl;
                    break;
                case 3:
                    cout << "적금담보대가산금리(만기일시상환대출): [" << cell << "]" << endl;
                    break;
```

```
//금융상품 정보 출력 함수
void FinancialManual(const string& name) {
    vector<vector<string>> info = GetInfo(name);
    // 저장된 행들 출력
    for (const auto& cell : info[0]) {
        // 각 열의 정보 출력
        switch (&cell - &info[0][0]) {
        case 1:
            cout << cell << endl;
            break;
        case 2:
            cout << "연령 조건: [" << cell << "]" << endl;
            break;
        case 3:
            cout << "소득 조건: [" << cell << "]" << endl;
            break;
        case 4:
            cout << "기간: [" << cell << "]" << endl;
            break;
        case 5:
            cout << "월 납입금: [" << cell << "]" << endl;
            break;
        case 6:
            cout << "정부 지원금: [" << cell << "]" << endl;
```

(7) main 함수 수정

- 입출력

<financial.cpp>

- getFieldPolicy 함수

사용자가 입력한 분야 반환

- getRegionPolicy 함수

사용자가 입력한 지역 반환

userinput = 정책 정보와 금융상품 정보 중 출력할 정보 입력 받는 변수

- 설명

사용자가 금융 상품 정보/ 정책 정보를 선택하면 그에 맞는 함수가 호출되며

사용자가 원하는 정보를 출력하게 끔 한다.

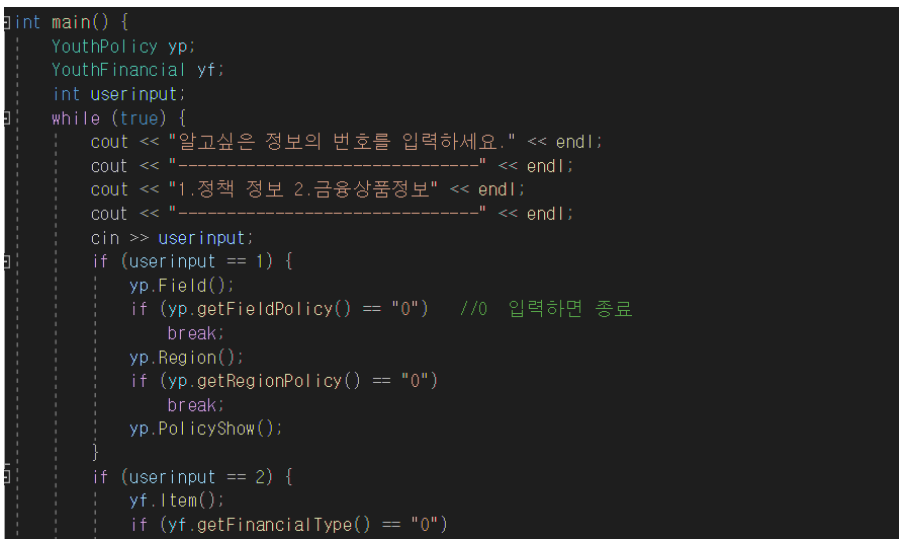
사용자가 0을 입력하면 종료 한다.

- 적용된 배운 내용

클래스 : 앞서서 만든 정보 출력 클래스를 활용해 함수를 호출한다.

조건문 : 사용자가 0을 입력하면 프로그램을 종료 하도록 한다.

- 코드 스크린샷



```
int main() {
    YouthPolicy yp;
    YouthFinancial yf;
    int userinput;
    while (true) {
        cout << "알고싶은 정보의 번호를 입력하세요." << endl;
        cout << "------------------------------" << endl;
        cout << "1.정책 정보 2.금융상품정보" << endl;
        cout << "------------------------------" << endl;
        cin >> userinput;
        if (userinput == 1) {
            yp.Field();
            if (yp.getFieldPolicy() == "0")   //0  입력하면 종료
                break;
            yp.Region();
            if (yp.getRegionPolicy() == "0")
                break;
            yp.PolicyShow();
        }
        if (userinput == 2) {
            yf.Item();
            if (yf.getFinancialType() == "0")
```

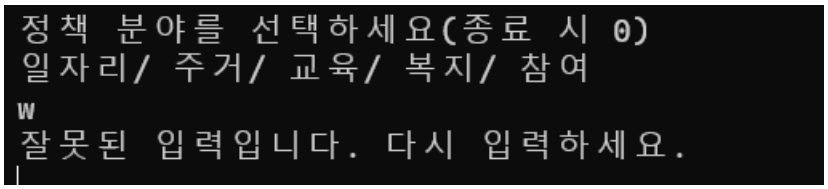
2) 테스트 결과

(1) 정부 지원 정책 데이터 수집

- 설명

사용자가 분야와 정책을 입력할 수 있는 카테고리 세분화하고 유효성 검증하여 출력

- 테스트 결과 스크린샷

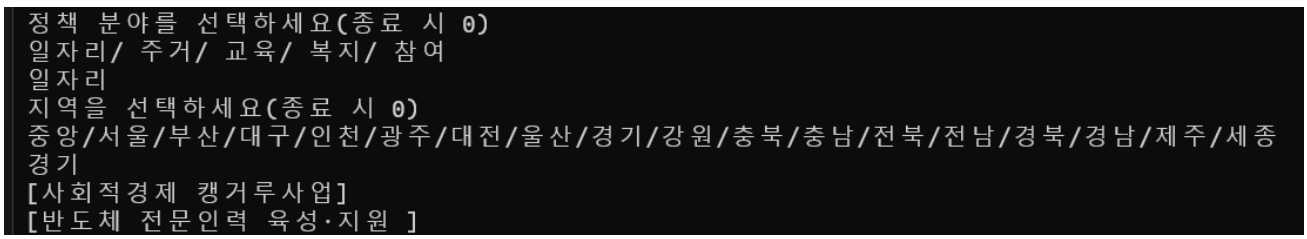


(2) 추천 시스템

- 설명

사용자가 입력한 분야와 지역에 맞는 정책 출력

- 테스트 결과 스크린샷

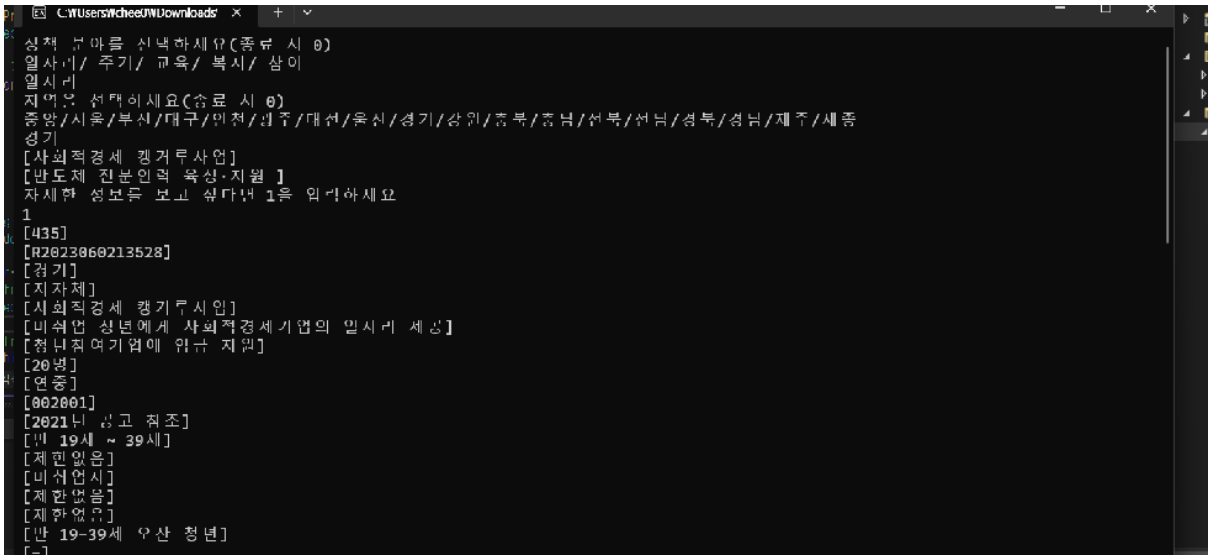


(3) 자세한 정보 출력

- 설명

사용자가 정보를 자세히 알고자 하면 세부사항 출력(사이트 정보 포함)

- 테스트 결과 스크린샷

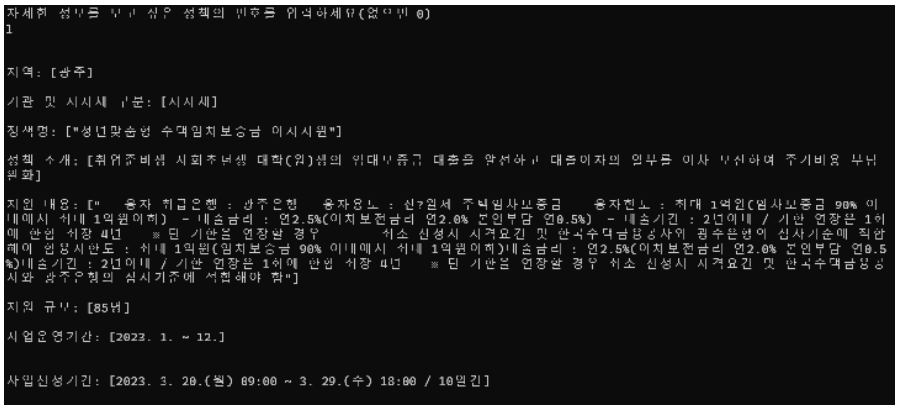


(4) 자세한 정보 제공 수정

- 설명

사용자가 입력한 정책 번호에 맞는 세부 정보 출력

- 테스트 결과 스크린샷

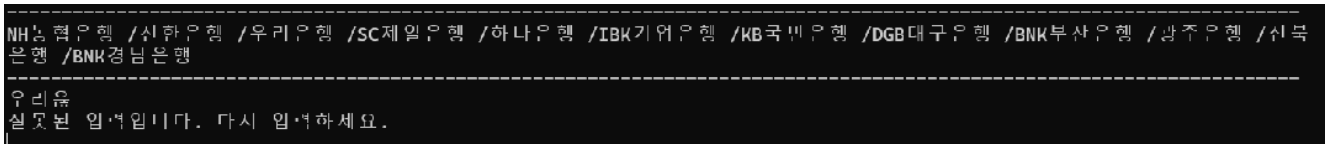


(5) 은행별 금융상품 수집

- 설명

사용자가 은행 명과 금융 상품 종류를 입력할 수 있는 카테고리 세분화하고 유효성 검증 하여 출력

- 테스트 결과 스크린샷



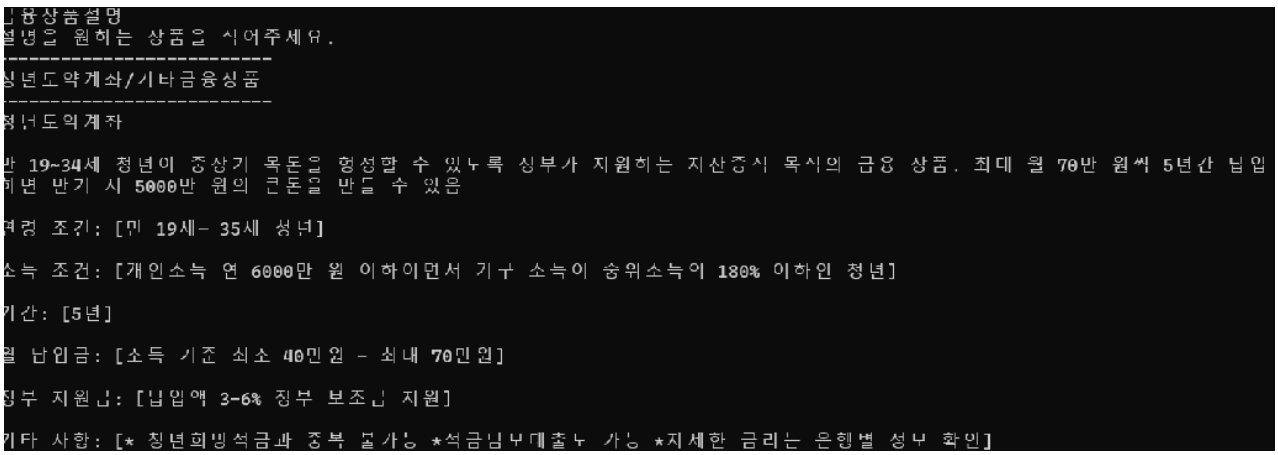
(6) 사용자 맞춤 금융정보 제공

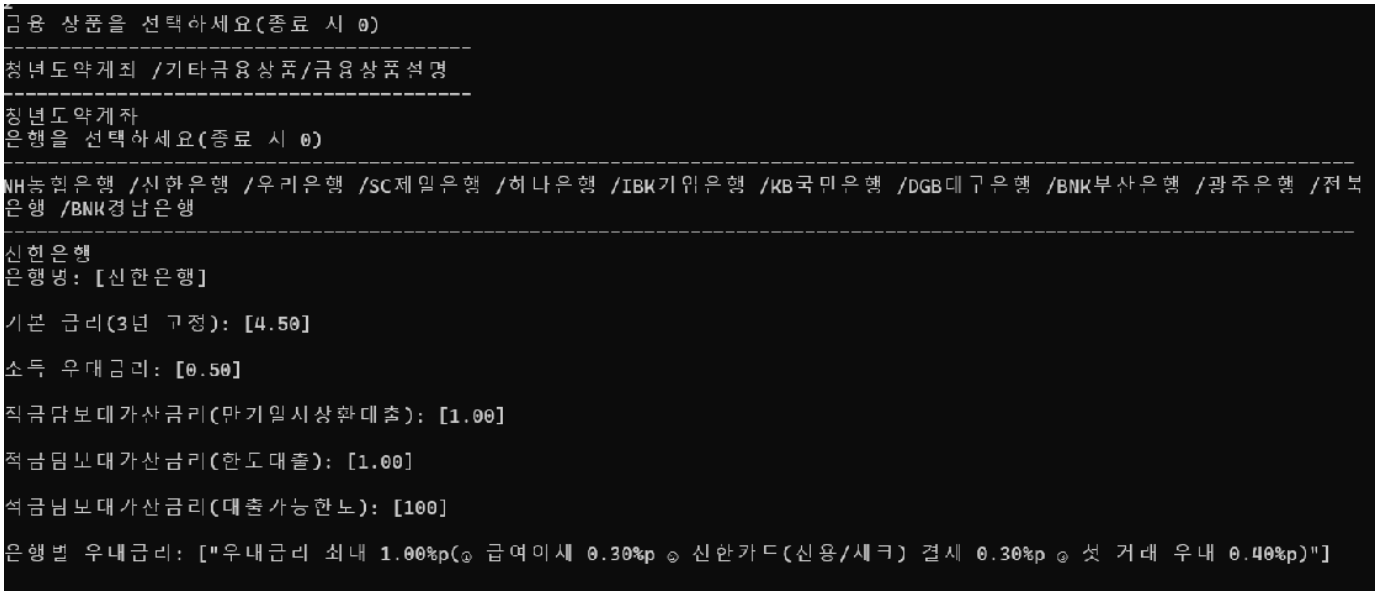
- 설명

사용자가 알고자하는 은행의 청년도약계좌 세부 정보 제공

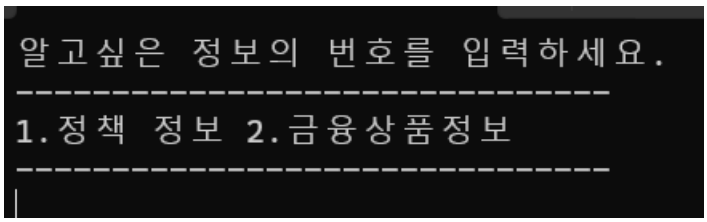
사용자가 알고자하는 금융상품 설명 제공 (*기타 금융상품은 구현 중)

- 테스트 결과 스크린샷





(7) main 함수 수정



4. 계획 대비 변경 사항

1) 정책 정보 세분화 카테고리 변경

- 이전

현재 정부에서 지원하고 있는 청년 관련 정책들 정보를 수집하여 지역별, 나이대별로 세분화함.

- 이후

현재 정부에서 지원하고 있는 청년 관련 정책들 정보를 수집하여 지역별, 분야별로 세분화함.

- 사유

본래 지역별, 나이대별을 계획했지만 청년 대상인 프로그램이기에 나이대는 중요한

정보가 아니라고 판단, 나이대별을 분야별로 대체

2) 정보 출력 방식 변경

- 이전

사용자가 더 알고자하는 정보를 클릭하면 자세한 정보와 함께 관련 사이트 연결

- 이후

사용자가 정보를 자세히 알고자 하면 세부사항 출력(사이트 정보 포함)

- 사유

본래 정보를 클릭하면 사이트가 연결될 수 있도록 하려 했으나 이는 다른 기술이 더 필요하므로 불가능하다고 판단. 사이트 정보를 출력하여 사용자로 하여금 들어갈 수 있도록 하는 형식으로 변경

3) 금융 정보 세분화 카테고리 설정

- 이전

금융 정보의 카테고리를 은행 별로 다르게 두어, 사용자가 은행을 입력하면 해당 은행의 카테고리를 나올 수 있게끔 하였음.

- 이후

청년도약계좌는 따로 둠.

- 사유

가장 대표적이고 많이 찾는 두 가지는 더 쉽게 비교할 수 있게끔 따로 빼는 게 낫다고 판단. 나머지 상품은 기타에서 은행별로 출력 예정

5. 프로젝트 일정

(진행한 작업과 진행 중인 작업 등을 표기)

업무		11/3	11/22	11/25	11/26
제안서 작성		완료			
기능1	세부기능1		완료		
	세부기능2			완료	
	세부기능3				완료

업무		11/30	12/1	12/3	12/17
기능1	세부기능3	완료			
기능2	세부기능1		완료		
	세부기능2			----------> (진행 중)	

업무		12/22	12/23	12/3	12/16
기능2	세부기능3	----------> (진행 중)			
	세부기능4		(예정)		