CSED 232 Object-Oriented Programming (Spring 2023)

Programming Assignment #2

학생 데이터 피벗 변환

Due date : 4월 7일 23시 59분 59초

담당 조교 : 이규석 (gslee22@postech.ac.kr)

[안내사항]

1. 모든 문제는 C++의 standard 입출력(i.e., cout, cin)을 기본으로 합니다.
2. 잘못된 입력에 대해서는 예외처리를 기본적으로 수행합니다.
3. 과제에 관한 질문은 PLMS를 통해 문의하시길 바랍니다.

[감점]

1. 제출 기한이 지나면 얻은 총점의 20% 감점
2. 하루(24시간) 늦을 때마다 추가 20%씩 감점

   - 1일 이내: 20% 감점, 2일 이내: 40% 감점, 3일 이내: 60% 감점 4일 이내:80% 감점

   - 4일 이상 지연: 0점
3. 컴파일이 정상적으로 되지 않을 경우 프로그램 기능 점수 0점

[제출]

1. Code (e.g., cpp or hpp), executable file (exe), README.txt, report.pdf를 하나의 폴더에 담아서 압축하여 제출하시면 됩니다. 만약 MacOS를 이용하는 경우 executable file 대신 makefile을 제출하시면 됩니다.
2. 이때 makefile을 통해서는 executable file이 생성되어야 하며 README.txt에는 각 코드파일들에 대한 설명을 간단히 서술하시면 됩니다.
3. Report는 제시된 양식(PLMS 참고)을 따르되, 주어진 프로그램을 객체 지향 프로그래밍의 관점에서 어떻게 구현하였고 혹은 어떤 기능을 추가할 수 있을지에 대해 본인의 생각을 보고서 "3. 토론 및 개선"에 간략히 작성하면 됩니다.
4. 폴더 이름과 압축파일은 "Assign2_학번"으로 만드시면 됩니다 (e.g., Assign2_20229999).
5. 채점은 Windows Visual Studio 2022 환경에서 이루어집니다. 환경 세팅이 어려운 경우 Visual Studio Code를 이용해서 코드를 작성하셔도 됩니다.

6. 제출은 반드시 PLMS를 통해 제출해주시기 바랍니다. 이메일 제출은 인정되지 않습니다. 4일 이상 지연 제출할 경우 0점이므로 4일(4월 11일 23시 59분 59초)이 지난 이후는 PLMS를 통해 제출하실 수 없습니다.

[채점기준]

1. **프로그램 기능 – 50%**

- 프로그램이 요구 사항을 모두 만족하면서 올바로 실행되는가?

2. **프로그램 설계 및 구현 – 35%**

- 요구 사항을 만족하기 위한 변수 및 알고리즘 설계가 잘 되었는가?
- 각 문제에서 제시한 세부 조건의 유의사항을 모두 만족하였는가?
- 입력과 출력이 주어진 형식에 맞게 잘 나타나는가?
- 예외처리를 알맞게 하였는가?

3. **프로그램 가독성 – 5%**

- 프로그램이 읽기 쉽고 이해하기 쉽게 작성되었는가?
- 변수 명이 무엇을 의미하는지 파악하기 쉬운가?
- 프로그램의 소스 코드를 이해하기 쉽도록 주석을 잘 붙였는가?

4. **보고서 구성 및 내용, 양식 – 10%**

- 보고서는 적절한 내용으로 이해하기 쉽고 보기 좋게 잘 작성되었는가?
- 보고서의 양식을 잘 따랐는가?
- 각 문제에서 제시한 질문이 있다면, 그에 대한 답변이 충분한가?

[주의사항]

다른 사람의 프로그램이나 인터넷에 있는 프로그램을 단순히 복사(copy)하거나 수정해서 제출하면 부정행위로 간주됩니다. 부정행위 발견 시 'F' 학점을 받을 수 있으며 학과에서 정한 기준에 따라 추가적인 불이익이 있을 수 있습니다.

[문제]

1. Overview

본 과제에서는 학생 데이터를 입력 받은 후 피벗 변환을 수행하고, 수행 결과를 테이블 및 차트 로 시각화 하는 문제를 해결한다.

- **피벗 변환이란**?

주어진 데이터를 특정 열의 카테고리 기준으로 묶고 다른 열의 수치에 대해 합계, 평균, 최대값 등을 구하는 변환이다.

Dept	Gender	Name	Age
BIO	F	Hwang	19
BIO	F	John	23
BIO	M	Karim	27
CS	F	Kim	25
CS	M	Lee	18
CS	M	Park	30
RAW	F	Jung	25

Dept 및 나이(평균)로 피벗 변환

Dept	Average
BIO	23
CS	24.3
RAW	25

Dept, Gender 및 나이(최댓값)로 피벗 변환

Dept	Gender	Max
BIO	F	23
BIO	M	27
CS	F	25
CS	M	30
RAW	F	25

그림 1 피벗 변환 예시

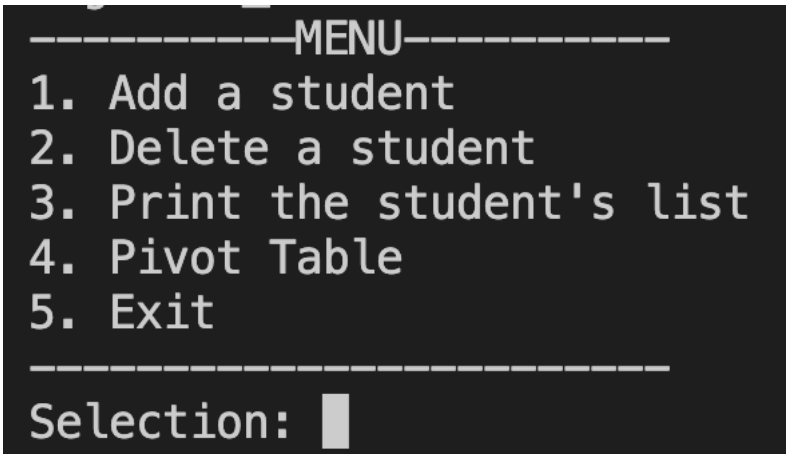
2. 학생 데이터 설명

1) Dept: 학과
2) Gender: 성별
3) Name: 이름
4) Age: 나이

- **데이터 세부 설명**
  - Dept, Gender, Name은 std::string type을 사용한다.
  - Dept, Gender, Name은 공백을 포함하지 않는다.
  - Age는 integer type을 사용한다.
  - Dept의 종류(e.g., CS, Bio)는 9개를 넘지 않는다. (9개까지 가능)
  - 각 Dept에 소속된 학생 수는 10,000을 넘지 않는다. (10000개까지 가능)
  - Gender는 M,F 만 입력이 가능하다. (M = male, F = Female)
  - Dept는 대문자만 입력으로 받는다.
  - Name은 대소문자 모두 입력으로 받을 수 있다. (e.g., Minsu, MinSu, minsu)
  - Age의 범위는 18 <= age <= 30으로 제한한다.

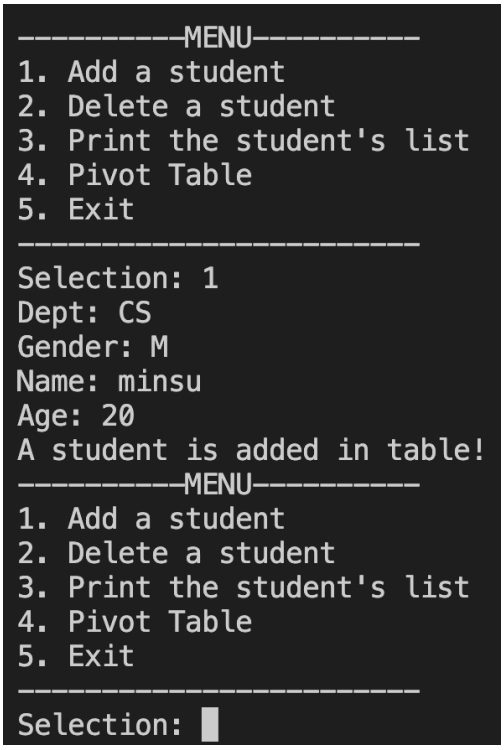
3. 프로그램 기능

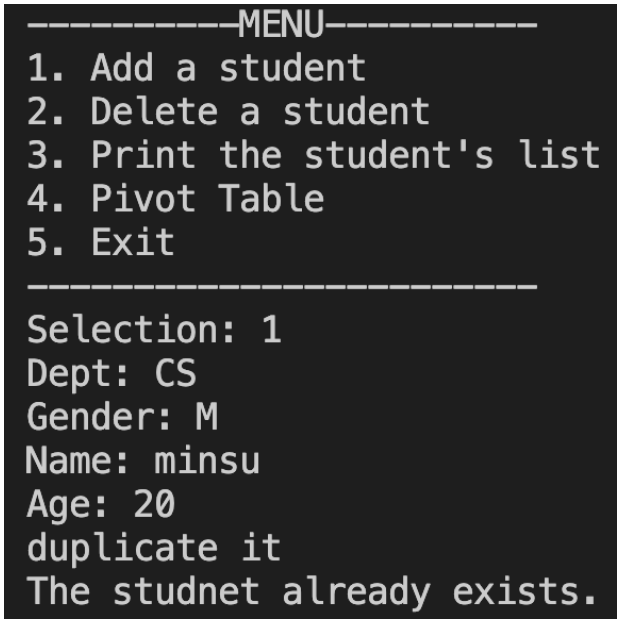
프로그램을 실행하면 아래 사진과 같은 메뉴가 출력되며 사용자로부터 메뉴 번호를 입력 받는다. 이때 선택 가능한 메뉴는 총 5개로 구성되고 각각의 메뉴에 따른 기능들은 아래에서 설명된다. 항상 메뉴에 따른 기능 수행이 끝난 후에는 메뉴를 다시 출력해주고 입력을 받도록 한다.



1) 학생 추가 (Add a student)

위에서 설명했던 학생 데이터를 사용자로부터 입력 받는다. 이때 입력된 (학과, 성별, 이름, 나이)가 이미 존재하면 중복으로 간주하여 추가하지 않고 "The student already exists"를 출력하도록 한다.
만약 중복이 아니라면 입력된 데이터를 추가한 뒤 "A student is added in table!"를 출력한다.





2) 학생 삭제 (Delete a student)

사용자로부터 (학과, 성별, 이름, 나이)를 순차적으로 입력 받은 후 주어진 입력이 이미 추가된 리스트 내에 존재한다면 삭제한다. 이때 성공적으로 삭제가 되었다면 "Deleted!"를 출력하면 된다. 만약 매칭이 되지 않거나 리스트가 비어 있는 상태라면 "Can't Delete it"을 출력하면 된다.

예시: 리스트내에 만약 (CS/M/Minsu/20)이 이미 존재한다고 가정한다면 아래 왼쪽 이미지는 정확하게 매칭이 되어 리스트 내에서 삭제되지만 오른쪽은 그렇지 아니하다)

```
----------MENU----------
1. Add a student
2. Delete a student
3. Print the student's list
4. Pivot Table
5. Exit
------------------------
Selection: 2
Dept: CS
Gender: M
Name: Minsu
Age: 20
Deleted!
```

```
----------MENU----------
1. Add a student
2. Delete a student
3. Print the student's list
4. Pivot Table
5. Exit
------------------------
Selection: 2
Dept: CS
Gender: M
Name: Minsu
Age: 21
Can't Delete it
```

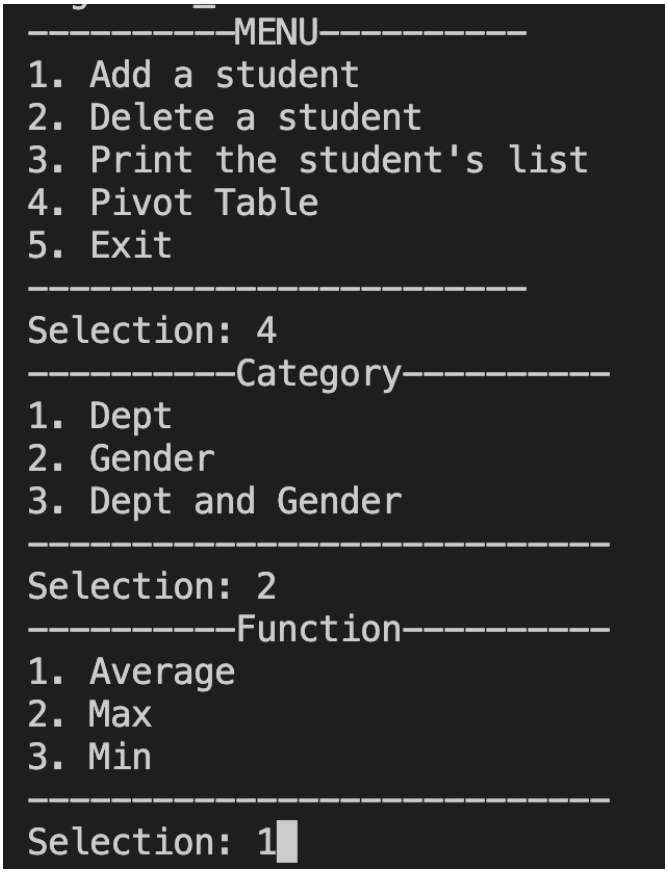
3) 학생 리스트 출력 (Print the student's list)

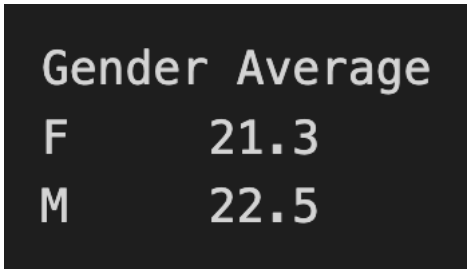
입력 받은 학생 데이터를 테이블 형태로 출력한다. 이때 첫번째 줄에는 4개 열의 이름을 Dept, Gender, Name, Age 출력한다. 다음 줄부터는 학생에 대한 정보를 출력하되 **정렬하여** 보여 주도록 한다. 이때 **문자열은 사전 순서대로 숫자는 오름차순**으로 정렬하도록 한다. 각 열은 탭 문자('₩t')로 구분되며 각 줄의 맨 뒤에는 탭이나 공백이 없다.

```
----------MENU----------
1. Add a student
2. Delete a student
3. Print the student's list
4. Pivot Table
5. Exit
------------------------
Selection: 3
Dept    Gender  Name    Age
CS      F       A       19
CS      F       B       18
CS      M       A       20
CS      M       C       21
```

4) 피벗 테이블 출력 (Pivot Table)

사용자는 카테고리와 함수를 선택할 수 있다. **카테고리에는 (Dept, Gender, Dept and Gender)**로 세 가지의 선택사항이 있다. **함수에는 (Average, Max, Min)**으로 세 가지 선택사항이 있다. 이때 Function의 대상이 되는 값으로 Age를 사용한다.
피벗테이블을 출력할 때, 첫째 줄에는 입력 받은 카테고리와 함수를 출력하고, 다음 줄부터는 카테고리별로 Age에 함수를 적용하여 테이블 형태로 출력한다. 이때 피벗 테이블의 출력은 카테고리를 사전 순서대로 정렬하여 나타낸다. 각 열은 탭 문자('₩t')로 구분되며 각 줄의 맨 뒤에는 탭이나 공백이 없다.
또한 함수로 적용된 값이 소수로 표현될 경우, **소수점 첫째자리까지 반올림하여** 나타낸다.
카테고리 또는 카테고리 조합에 속하는 학생이 없는 경우는 **해당 출력을 생략한다** (다음 페이지 이미지 참고).





```
----------MENU----------
1. Add a student
2. Delete a student
3. Print the student's list
4. Pivot Table
5. Exit
------------------------
Selection: 4
----------Category----------
1. Dept
2. Gender
3. Dept and Gender
----------------------------
Selection: 3
----------Function----------
1. Average
2. Max
3. Min
----------------------------
Selection: 1
```

Dept	Gender	Average
BIO	F	22.5
BIO	M	23.8
CS	F	21.4
CS	M	25.6
RAW	M	25.5

(Raw/F에 속한 학생이 없으므로 출력하지 않는다)

5) 프로그램 종료 (Exit)

"Exit!"를 출력하면서, 프로그램을 종료시킨다.

```
----------MENU----------
1. Add a student
2. Delete a student
3. Print the student's list
4. Pivot Table
5. Exit
------------------------
Selection: 5
Exit!
```

4. Class & Function

본 과제에서는 Class와 Function을 활용하여 문제를 해결하는 것을 목표로 한다. 따라서 아래와 같은 class를 구현하여 해당 문제를 해결한다. 각 Class는 hpp파일을 통해 정의하고, cpp파일을 통해 구체적인 기능을 구현하도록 한다**. 참고로 제시된 기능 이외에 추가적으로 새로운 class나 멤버 변수/함수가 필요하다고 판단되면 구현하여 제출할 수 있다.**

1) Student class

기본적인 학생 데이터를 표현하기 위한 class이다.

```
class student{
    public:
        string dept, name, gender;
        int age;

        void input_info();
};
```

(1) 멤버 변수

- string dept : 학과
- string name: 이름
- string gender: 성별
- int age: 나이

(2) 멤버 함수

- void Input_info(): 학생 정보(e.g., dept, name, gender, age)를 입력 받는다.

2) List class

전체 학생 정보를 저장하기 위해서는 하나의 List가 요구된다. 이때 List로 **Linked List**를 활용하도록 한다.

```
class list{
    public:
        int count, dept_cnt;
        string dept[9];
        node *head;

        void sort(string metric);
};
```

(1) 멤버 변수

- int count : List 내 존재하는 Node의 개수
- int dept_cnt: List 내 존재하는 학과의 개수
- string dept[9]: 학과를 저장하기 위한 string array (참고로 학과의 최대 개수는 9개 이므로 size를 9로 설정)
- node *head: Linked List의 head (node class 참고)

(2) 멤버 함수

- Void list_sort(string metric):

앞서 프로그래밍 기능 파트에서 설명한 것처럼, 모든 출력은 정렬이 된 형태로 나타나야 한다. 이를 위해서 주어진 linked list를 정렬하여 저장하도록 한다. 이때 metric으로는 student class의 멤버 변수인 dept, name, gender, age가 들어갈 수 있다. Metric이 주어지면 이를 기준으로 linked list를 정렬한다.

3) Node class

Linked List의 Node를 나타내기 위한 class이다.

```
class node{
    public:
        student data;
        node *next;
};
```

(1) 멤버 변수

- Student data: student의 정보를 저장한다.
- Node * next: 다음 노드를 연결하기 위한 node pointer를 의미한다.

4) Function

아래 function들은 linked list의 node를 추가 및 삭제하기 위한 function이다.

```
void save_node(list&, node *);
void delete_node(list&, node *);
```

- void save_node(list &, node *): linked list에 node를 저장한다.
- void delete_node(list &, node *): linked list에 node를 삭제한다.

5. 제약조건

Library로 iostream, string, cmath 만을 사용해서 구현하도록 한다.