2023 Spring OOP Assignment Report

과제 번호 : 2
학번 : 20220302
이름 : 김지현
Povis ID : jihyunk

명예서약 (Honor Code)

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.
I completed this programming task without the improper help of others.

1. 프로그램 개요

□ 이번 과제는 class 를 이용해 학생 리스트를 저장하고 pivot table 을 구현하는 것이다.

□ 새로운 학생을 등록하고 dept > gender > name > age 순서에 따라 alphabetical 하게 정렬할 수 있다. 뿐만 아니라 등록된 학생을 삭제할 수도 있다. 사용자가 print student 를 원하면 위의 순서대로 정렬된 학생 리스트가 출력된다.

□ 만약 새로 저장하려는 학생이 이미 존재하는 경우 새로 저장하지 않고 이미 같은 학생이 존재 한다는 문구를 띄운 후 다시 메뉴 화면으로 돌아간다. 만약 삭제하려는 학생이 학생 리스트에 존재하지 않는 경우 학생이 존재하지 않는다는 문구를 띄우고 메뉴 화면으로 돌아간다.

□ Dept/ Gender/ Dept and Gender 로 pivot table 을 만들어서 학생들을 각각에 저장하고 Average/ Max/ Min 나이를 나타낼 수 있도록 한다. 이를 구현하기 위해서 array 를 사용하였다.

2. 프로그램의 구조 및 알고리즘

Class: student
Public member variable : string dept string name string gender int age
Public member function : void input_info (bool*)

Class: Node
Public member variable : student data Node* next

Class: List
Public member variable : int count int dept_num string dept[9] Node* head
Public member function : void menu() void add_student(List*) void delete_student(List*) void print_student(List) void pivot(List) void del_node(List*)
Private member functions: int check_repeat(Node*, List) Node* previous(int, Node*) void print_category_menu() void print_function_munu() void category_dept(List, int) void category_gender(List, int) void category_dept_gender(List, int)

- 이번 어싸인에서는 총 3개의 Class 를 구현하였다. Student class 는 학생의 정보를 저장하고, Node class 는 student 변수와 포인터 변수를 통해 각 학생들은 연결해주고, 마지막으로 List class 에서는 총 학생수, 학생 리스트의 시작, 그리고 다른 함수들을 구현하고 있다.
- 먼저 Student class 에 있는 변수들은 각각 학과 (string dept), 이름 (string name), 성별(string gender), 그리고 나이(int age)를 뜻한다. 그리고 멤버 함수인 void input_info(bool*) 은 cin 함수를 이용해 사용자가 입력한 학생의 정보를 가져와 저장하고 만약 잘못된 값이 입력되었다면 bool 변수를 이용해 나타낸다.
- 잘못된 값이 입력되는 상황은 1) 학과를 입력할 때, 띄어쓰기가 포함되거나 소문자가

포함된 경우 2) 성별이 'M' 또는 'F' 를 제외한 다른 성일 경우 3) 이름에 띄어쓰기가 포함된 경우 4) 나이가 18세 미만이거나 30세 초과일 경우를 말한다. 각각의 경우 Boolean variable 인 exception 이 false 를 가지게 한다. Void add_student 에서 while 문을 사용해 exception 이 true 일 경우에만 학생을 더하게 해주었다.

- List class 에 있는 변수들은 각각 총 학생의 수 (int count), 총 학과의 개수 (int dept_cnt), 학과를 저장한 array (string dept[9]), 그리고 Node 포인터 head 가 있다. Node 포인터 head 는 가장 앞에 오는 학생의 정보를 가르킨다.

- List class 의 void add_student 함수에서 1) if 문을 사용해 학과가 9개가 초과했는지를 확인해준다. 만약 학과가 9개를 넘어간다면 더이상 학생을 추가할 수 없다는 문구를 띄우고 메뉴 화면으로 돌아간다. 2) Student 타입의 input_student 변수를 만들고 student class 안의 input_info 함수를 사용해 학생의 정보를 받아온다. 3) exception 이 true 인지 확인 한 후, 4) Node pointer 변수를 동적할당 한 뒤, 학생의 정보를 저장한다. 5) 새로 저장하려는 학생이 처음 저장되는 학생인지 확인한 뒤, 맞다면 s_list 의 head 포인터에 저장하도록 한다. 6) List::check_repeat 함수를 이용해 추가하려는 학생이 이미 리스트에 존재하는지 확인하고 만약 존재하지 않는다면 리스트에 이미 저장된 학생들의 학과>성별>이름>나이를 순서대로 비교하여 정렬과 동시에 저장을 할 것이다. 이때, walker 의 전에 들어와야 한다면 previous 함수를 사용해 walker 의 위치를 전으로 바꿔줄 것이다. 7) 만약 새로운 학과를 가진 학생이 저장된다면 학과만을 알파벳 순서로 비교한 뒤에 적절한 자리에 넣어준다. 이때 s_list 의 학과 어레이와 수를 더해줘야한다. 8) 만약 더하려고 하는 학생이 이미 리스트에 존재한다면 이미 학생이 존재한다는 문구를 띄우고 메뉴화면으로 돌아간다.

- List::previous 함수는 prev 라는 Node 포인터를 prev_num-1 만큼 움직이는 것이다. Prev 는 초기에 리스트의 head 를 가르키고 prev_num 은 walker 가 움직일 때 마다 1씩 늘어난다. 결과적으로 prev pointer 는 walker 의 전(before) node 를 가르키게 된다. 그리고 이 함수의 return type 은 Node* 이며, prev 포인터를 반환하게 된다.

- List::check_repeat 함수는 기존의 list 에 입력받은 학생이 존재하는지를 확인해준다. 존재하면 0을 반환하고 존재하지 않는다면 1을 반환한다.

- List::delete_student() 에서는 add_student 와 비슷한 꼴의 함수를 구현한다. 그러나 if 문을 이용해 입력받은 학생이 리스트에 존재한다면 그 학생을 지우고 그 학생의 정보를 저장하는 walker 포인터를 할당해제 한다. 이에 따라 총 학생명 수 도 줄어들 것이고, 만약 지워진 학생이 그 학과에서 유일하게 존재하던 학생이라면 s_list 의 학과 array 에서 그 학생의 학과는 삭제 되고 총 학과의 수도 줄어들 것이다. 만약 입력받은 학생이 존재하지 않는다면 적절한 오류 문구를 프린트하고 다시 메뉴로 돌아온다.

- List::print student 는 리스트에 정렬되어 저장된 학생을 차례대로 출력한다.
- List::menu()는 첫 메뉴화면을 프린트 하는 함수이고, List::print_category_menu 와 List::print_function_menu 함수는 각각 pivot 메뉴 화면을 프린트 하는 함수이다.
- List::pivot 은 각각의 category (dept/gender/dept&gender)와 function (average/max/min) 에 따라 switch case 를 이용해 List class 의 private 에 정의된 함수들을 불러온다.
- List::category_dept 는 위의 카테로리 중 dept 를 선택했을 때 function call 이 된다. Switch case 를 이용해 function 마다 arr[2][9] 또는 answer[9] 에 값을 저장하게 된다. Average 를 계산해야 하는 경우, 총 학과는 9개까지 저장이 가능하기 때문에 arr[2][9] 를 사용해 arr[0][dept] 에는 나이를 더하고 arr[1][dept] 에는 더해진 명수를 저장한다. 그리고 학과가 변하는 경우 dept 변수의 값을 더해서 어레이를 옮겨준다. 그리고 결과를 출력하기 전, answer[9] 에 평균값을 저장한다. Max 와 Min 을 찾는 경우에 먼저 answer[x] 가 0의 값을 가지면 그 dept 의 첫 학생의 나이로 initialization 을 한 뒤, 같은 dept 내에서 값을 비교하며 max/min 을 찾는다.
- List::category_gender은 위와 같은 과정을 거친다. 다른 점으로는 category_gender 에는 array가 아닌 int famle, male, f_cnt, m_cnt 를 사용해 값을 저장할 것이다. 왜냐하면 성별은 M 과 F 가 아닌 모든 다른 성별은 에러처리를 했기 때문이다.
- 와 List::category_dept_gender 에서는 위의 두 케이스와 달리 먼저 값을 저장한 뒤, switch case 를 이용해 출력한다. 평균값을 저장할 때는 arr[2][18]를 이용하고 min/max 를 저장할 때는 각각 min[18] 과 max[18] 을 생성해 할당시켰다. 각 학과마다 여학생과 남학생이 모두 있는 상황에서는 최대 18개의 element 가 할당 될 것이다. 비교 과정 자체는 위의 두 케이스와 같다. 참고로, 여학생은 짝수번, 그리고 남학생은 홀수번의 순서에 오게 된다.
- 마지막으로 프로그램을 종료하기 전에, 학생을 저장할 때 사용한 linked list 를 동적 할당 해제 해야한다. 이는 List::del_node 에서 구현되어있다. Temp 에 각각의 노드를 저장한 뒤, walker 포인터를 옮겨주고, temp 를 할당해제 해준다. 마지막 노드까지 할당해제 해준뒤, 프로그램을 종료한다.

3. 토론 및 개선

- 이번 과제를 통하여 class와 객체를 사용해 프로그램을 구현하는 방법을 알게되었다. 기존의 C 언어 보다 사용자의 관점에서 보는 프로그램을 구현할 수 있었다. 특히 수업 ppt 에서 나온 것 처럼 구체적은 기능을 구현하는 함수는 private 에 선언하고

interface 에 사용되는 함수는 public 에서 선언하려고 노력했다.

- 그러나 아직 객체에 대한 이해도가 높지 않아 public 에서 선언한 함수가 많이 있는 것 같다. Data usage 를 컨트롤하고 안전하게 데이터를 저장하기 위해서는 private 하게 변수를 선언한뒤, 함수를 이용해 public 하게 사용할 수 있도록 하는 것이 객체지향 프로그램에 더 맞는 접근일 것 같다.
- 이번 과제에서 pivot table 을 만들 때, 학과가 9개로 정해져 있기 때문에 array 를 이용해 값을 저장했지만 학과가 9개 보다 많을 경우를 생각해 linked list 등을 이용해 구현하면 더 좋은 프로그램을 만들 수 있을 것이라 생각한다.
- 그리고 using namespace std; 를 사용하는 것 보다는 using std::cout 등을 사용해 cout과 같은 자주 사용하는 함수에서는 std:: 를 사용하지 않되, 다른 함수를 사용할 때는 각 헤더파일에 맞는 함수를 사용할 수 있도록 해야 할 것 같다.

4. 참고 문헌

- https://cplusplus.com/reference/istream/istream/ignore/ - std::cin.ignore();
  - Getline 을 이용한 입력값을 받을 때, 버퍼를 비워주는 용도이다.
- https://cplusplus.com/reference/string/string/getline/ - std::getline(std::cin, var);
  - 띄어쓰기를 포함해 Enter kep 전의 모든 입력값을 받아오는 함수이다.
- https://cplusplus.com/reference/cmath/round/ - round()
  - 소수로 저장된 값을 반올림해주는 함수이다.