2020 Spring OOP Assignment Report

과제 번호 : ASSN2
학번 : 20190439
이름 : 오승훈
Povis ID : sho0927

명예서약 (Honor Code)

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

프로그램을 하다 보면 결정해야 할 세부 사항이 많은데, 이러한 세부 사항을 처리한 방법과 이유를 보고서에 쓰십시오.
독창적인 아이디어와 추가 기능은 보너스 점수를 받을 수 있으므로, 보고서에 명확히 기재하십시오.
제출 기한이 지나면 LMS를 통하여 과제를 제출할 수 없습니다. 제출 기한이 지난 후에는 담당 조교에게 email로 보내야 합니다. 늦게 제출된 과제에 대한 감점은 담당 조교에게 email이 도착한 시간을 기준으로 합니다.
문제가 여러 개인 경우, 각 문제별로 정리해서 작성합니다.

각 문항별 설명은 편의를 위한 것으로, 삭제하고 제출한다.

1. 프로그램 개요

a. ASSN2

A. 이 프로그램의 경우 미니 SNS를 구현하는 것을 목표로 하며 client를 추가하여 회원가입을 할 수 있는 기능, post를 해서 글을 쓰는 기능 , 친구 추가 기능, 친구인 경우 친구가 쓴 글을 볼 수 있고 그 글의 댓글을 볼 수 있는 기능 등을 구현하는 것을 목표로 한다.

B. 먼저 메뉴를 사용하는 방법은 main menu에서는 1이 회원가입, 2가 로그인, 3이 커맨드들을 불러오는 기능, 4번이 프로그램 종료에 해당한다. 2를 해서 로그인을 할 경우 My page가 나오고, 1번이 friend 메뉴로 이동, 2번이 feed 메뉴로 이동, 3번이 로그아웃, 4번이 내 계정 삭제가 된다. Friend 메뉴에서는 1번이 친구 추가 기능, 2번이 친구 삭제 기능, 3번이 내 친구목록을 보여주는 기능, 4번이 이전 메뉴로 이동하는 메뉴이다. Feed 메뉴에서는 1번이 친구들의 post를 보는 기능, 2번이 post를 추가하는 기능, 3번이 나의 post를 보는 기능, 4번이 이전 메뉴로 이

동하는 기능을 한다.

C. 이번 어싸인에서는 파일을 불러오는 기능이 필요한데 main menu에서 3번 Load command를 할 시 같은 디렉토리에 있는 command.txt파일을 불러온다. 그 이후 이 command.txt파일을 read하고 그 후는 사용자가 입력을 할 수 있도록 함수가 바뀌게 된다. 여기서 command.txt파일을 이용한 load command는 필수적으로 사용한다고 가정을 하고, 어싸인을 구현했다. 하지만 load command를 하지 않은 경우 command.txt파일 이후에 이어서 입력을 받은 값들을 출력하게 해 두었다.

D. 이번 어싸인에서는 좋아요와 글 쓰기 기능에서 좋아요의 경우 좋아요를 누르고, 다시 그 글을 들어간 경우 좋아요를 누르겠습니까?를 표시하지 않도록 구현을 해 두었고, 좋아요를 누른 사람의 경우 계정 삭제를 하면 좋아요를 눌러둔 댓글에도 좋아요 개수가 감소할 수 있도록 구현을 한다.

2. 프로그램의 구조 및 알고리즘

a. Member 클래스

Member 변수	int friends_num : 친구의 명수를 저장해주는 변수 std::string ID : 회원의 ID를 저장하는 변수 std::string name : 회원의 이름을 저장하는 변수 std::string birthday : 회원의 생일 정보를 저장하는 변수 std::string password : 회원의 password를 저장하는 변수 std::string friends[50] : 친구의 이름을 저장하는 변수
Member 함수	void add_friends(std::string tempt) : 친구 추가를 해주는 함수입니다. void del_friends(std::string tempt) : 친구 삭제를 해주는 함수입니다. void my_friends() : 친구 목록을 출력해주는 함수입니다. int check_friends(std::string tempt) : 이사람이 나의 친구인지 확인합니다. int check_client(std::string tempt) : 이사람이 client에 속해있는지 확인합니다. void pass_log_info(std::string& present_ID, std::string& : present_name, std::string& present_Birthday) : 로그인 정보를 main 함수에 전달해주는 역할을 합니다. void fill_info(std::string tempt_ID, std::string tempt_Name, std::string tempt_Birth, std::string tempt_password) : 회원가입시 받아온 정보를 저장해주는 역할을 합니다. int log_in(std::string tempt_ID, std::string tempt_password) : 로그인을 할 시 ID와 비밀번호가 일치하는지 확인합니다. std::string passing_friends_info(int index) : 친구의 정보를 main

	함수로 전달해주는 역할을 합니다.

A. 먼저 이번 어싸인에서 member 함수를 간단하게 보자면 private에 모든 변수들이 있고, public에 모든 함수들이 있는 구조이다. 함수를 크게 보자면 친구에 대한 기능과 client에 관한 기능으로 구분을 할 수 있다. 친구에 대한 기능은 추가, 삭제, 보여주기, 친구 정보 채워넣기 등과 같은 기능이 존재한다. Client에 대한 기능은 로그인, 정보 전달, 회원이 맞는지, 회원 탈퇴 등과 같은 기능을 수행하게 되어있다.

b. Feed 클래스

Member 변수	int like_num : 좋아요 개수를 저장하는 변수 int comment_num : 댓글의 개수를 저장하는 변수 std::string post_people : 포스팅한 사람의 ID를 저장하는 변수 std::string post_info : 포스팅한 내용을 저장합니다. std::string post_comment[50] : 댓글의 내용을 저장합니다. std::string post_comment_people[50] : 각 index에 맞게 댓글을저장한 사람의 이름을 저장하는 변수. std::string like_people[50] : 좋아요를 누른 사람의 이름을 저장하는 변수
Member 함수	int my_posting(std::string ID) : 내 포스팅인지 확인해주는 함수입니다. int check_friends(std::string ID) : 친구의 포스팅인지 확인해주는 함수입니다. void posting(std::string post, std::string ID) : 글을 쓰면 그 글을 저장해주는 함수입니다. void show() : 글들을 보여주는 함수입니다. void check_like(std::string ID) : 이 사람이 좋아요를 눌렀는지 확인해주는 함수입니다. void post_commenting(std::string ID, std::string comment) : 댓글을 적으면 그 댓글을 저장해주는 함수입니다. void comment_show() : 댓글을 보여주는 함수입니다. void deleting_comment(std::string ID) : 댓글을 삭제하는 함수입니다. void deleting_like(std::string ID) : 좋아요를 삭제해주는 함수입니다. int check_like_it(std::string ID) : 좋아요를 했는지 확인해주는 함수입니다. void deleting_my_post() : 내 글을 삭제해주는 함수입니다.

A. Feed 클래스 역시 private에는 모든 변수들이 public에는 모든 함수들이 구성이 되어있습니다. 이 함수들 역시 두 개의 부류로 나눌 수 있는데 댓글에 대한 기능

과 글에 대한 기능으로 나눌 수 있습니다. 댓글에 대한 기능은 댓글 달기, 댓글 지우기 등과 같은 기능들을 수행하고, 글에 대한 기능은 내 글인지 확인하기, 친구인지 확인하기, 포스팅하기 등과 같은 기능들을 수행합니다.

c. 클래스 함수에 대한 세세한 설명

A. Member 클래스

i. `void add_friends(std::string tempt)` : 이 함수의 경우 tempt에 이름을 받아와 이미 추가된 친구라면 추가를 하지 않고, 이미 추가가 되어있는 친구라고 출력해주고, 추가되어 있지 않는 user라면 그 사람을 사전순의 오름차순에 맞는 위치에 저장해준다. 그 후 친구 수를 저장해주는 friends_num의 값이 1 증가하게 구현이 되어있다. Add friend가 실행되기 전에는 main 함수에서 check_client가 선행되고 add_friend가 실행이 되게 된다. 클라이언트가 맞아야지만 친구로 저장할 수 있다는 특성을 반영했다.

ii. `void del_friends(std::string tempt)` : 이 함수 역시 tempt의 이름을 받아와 이미 추가된 친구라면 삭제를 하고, 추가되어 있지 않는 친구라면 오류를 출력해내는 함수이다. 또한 삭제를 하게 되면 friends_num이 1감소하게 된다. 이 역시 check_client가 우선 실행이 되고, 이 함수가 실행이 된다.

iii. `void my_friends()` : 이 함수는 내 친구 목록을 모두 보여주는 함수이다. For 문을 이용한 함수가 될 것이다.

iv. `int check_friends(std::string tempt)` : 내 친구인지 확인을 해주는 함수이다. 이 함수의 경우 내 친구 목록 배열에 이 사람이 저장되어 있으면 0을 반환하고, 저장되어 있지 않으면 1을 반환시켜준다.

v. `int check_client(std::string tempt)` : 이 사람이 회원가입이 되어있는지 확인하는 함수이다. client라면 1을 반환하고, 아니면 0을 반환하게 구현이 되어있다.

vi. `void pass_log_info(std::string& present_ID, std::string& present_name, std::string& present_Birthday)` : 로그인하는 사람의 정보를 보내주는 함수이다. 이 함수가 있는 이유는 main 함수에서 My - Page를 출력할 때 이 사람의 정보가 출력이 되어야하는데 main 함수에서 직접적으로 private 변수 영역에 접근을 할 수 없기 때문에 만들어진 함수이다.

vii. `void fill_info(std::string tempt_ID, std::string tempt_Name, std::string tempt_Birth, std::string tempt_password)` : 이 함수의 경우 정보를 채워주는 함수이다. 4개의 input을 받아서 class에 저장을 해준다.

viii. `int log_in(std::string tempt_ID, std::string tempt_password)` : 이 함수의

경우 main 함수에서 check_client가 선행이 되게 되는데 만약 회원가입이 되어 있지 않는 경우라면 회원가입이 되어 있지 않다고 출력을 해주고, 회원가입이 되어 있는데 비밀번호가 다르다면 2를 반환해서 비밀번호를 확인해보라는 메시지를 출력한다. 또한 ID 비밀번호가 일치한다면 1을 반환해서 로그인을 시켜주는 함수이다.

ix. std::string passing_friends_info(int index) : 친구에 대한 string 즉 ID를 보내주는 함수이다. 이 함수는 feed 클래스의 특정 부분을 위해 구현이 되어있는 함수이다.

x. void del_my_log() : 나의 회원가입을 한 것을 탈퇴시켜 주는 함수이다. 모든 private 변수들을 초기화를 시켜주는 함수이다.

xi. member() : 생성자인데 친구 수를 0으로 초기화를 시켜서 클래스를 만들어준다.

B. Feed 클래스

i. int my_posting(std::string ID) : 내가 쓴 포스팅인지 확인을 해주는 함수이다. 이 함수는 내 글을 보는 곳에서 필요하다. 내가 쓴 포스팅이라면 1을 반환해준다.

ii. int check_friends(std::string ID) : 친구가 쓴 포스팅인지 확인을 해주는 함수이다. 이 함수는 Feed all 기능에서 필요한 함수이다. 이 함수 역시 친구가 쓴 포스팅이면 1을 반환해준다.

iii. void posting(std::string post, std::string ID) : 포스팅함수는 내가 쓴 글을 post 클래스에 저장해주기 위해서 필요한 함수이다. 이때 다른 값들을 초기화 시켜준다.

iv. void show() : 이 함수는 글의 내용을 보여주기 위한 함수이다.

v. void check_like(std::string ID) : 이 함수는 좋아요를 눌렀는지 확인을 하고 누르지 않았다면 ID 내역을 저장을 해두고, 좋아요 수를 1 늘린다. 만약 좋아요를 누른 상태라면 이미 좋아요를 눌렀습니다를 출력해준다. 이는 원래 좋아요를 누른 사람은 좋아요를 눌렀다는 내용을 표시하려고 구현을 해둔것이나 어싸인 질문상에서 이미 좋아요를 누른 사람은 좋아요를 누르겠습니까? 를 출력을 하지 말고 바로 댓글달기로 넘어가라고 했기 때문에 필요하지 않는 부분이 일부 존재한다.

vi. void post_commenting(std::string ID, std::string comment) : 이 함수는 댓글을 달기 위한 함수로 댓글을 쓴 사람의 정보와 쓴 글의 정보를 클래스에 저장해준다.

vii. void comment_show() : 댓글을 보여주는 함수이다.

viii. `void deleting_comment(std::string ID)` : 회원탈퇴시 내가 쓴 댓글도 역시 모두 삭제를 해야하기 때문에 댓글을 쓴 아이디가 현재 받아온 ID와 일치하면 그 댓글을 지우고, 다른 댓글들을 다시 정렬을 해준다.

ix. `void deleting_like(std::string ID)` : 좋아요 역시 회원가입시 내가 누른 좋아요수를 모두 없애주고, 좋아요를 눌렀다는 ID 기록 역시 삭제가 되도록 조치를 취해두었다.

x. `int check_like_it(std::string ID)` : 이 함수는 좋아요를 누른 case 라면 1을 반환하고, 누르지 않았다면 0을 반환하는 함수이다. 이는 위에서 `void check_like(std::string ID)` 부분에서 나온 좋아요를 눌렀다는 멘트가 표시되지 않고, 그냥 좋아요를 누르겠습니까 부분을 뛰어 넘기 위해서 생겨난 함수이다.

xi. `void deleting_my_post()` : 내가 쓴 글을 지워주는 함수이다. 이 함수는 해당 클래스 내부 내용을 모두 초기화된 상태로 만들어주는 함수이다.

xii. `feed()` : 생성자로써 댓글의 수와 좋아요 수를 0으로 만들어서 클래스를 생성하게 만들어 준다.

C. 그 외 부연 설명

i. 현재 Main menu, My page, Friend, Feed 메뉴에서는 모두 While문을 무한 루프를 돌려서 구현을 해두었고, 그 전으로 나가는 커맨드를 입력할시 break 를 하여 나갈 수 있도록 구현을 해두었다.

ii. Main menu를 제외한 나머지 모든 부분에서 valid라는 변수를 이용하는데 이 변수는 로그인을 할 때 저장되는 회원의 클래스 index 값으로 이 index를 항상 valid라는 값에 저장을 해두어 모든 클래스 접근 또는 feed 클래스의 접근을 할 수 있도록 구현을 해두었다.

3. 토론 및 개선

□ 이번 과제를 통해서 가장 어려웠던 점은 아무래도 우리의 구현을 좀 더 편안하게 해주기 위해서 제한을 별로 걸어두지 않았던 점이 가장 어려웠던 것 같다. 확실한 방법으로 구현을 하는 것을 좋아하는데 모호한 case들이 몇 개 있었고, 그 case들에 대해서 lms 질문란에 올라온 것들을 확인을 해야하는 아쉬움이 있었던 것 같다. 또한 Load command 부분이 가장 어려웠던 것 같은데 이 load command 부분에서는 작년에 프로그래밍과 문제해결에서 사용했던 stdin, stdout을 이용하여 출력을 할 수 있을 것이라고 생각했는데 이것이 사용이 되지 않아서 당황을 하게 되었던 것 같다. 그래서 찾아낸 것이 rdbuf라는 멤버함수인데 이 함수가 정확히 어떻게 이용이 되고,

구동이 되는지 공부를 좀 더 해보면 좋지 않을까 라는 생각이 들었다.

- 또한 이번 ASSN에서 가장 아쉬운 점은 하나의 예외 케이스를 막지 못했다는 것이다. 댓글을 고르는 Select Number 이후 string 값을 입력하게 되면 받아들이는 변수가 int로 값을 처리를 해두었기 때문에 0이 input으로 들어가게되고, 자연스럽게 0번 index의 댓글로 들어가게 된다는 것을 볼 수 있었다. 이 case의 경우 cin을 해줄 때 string 값으로 값을 받아들이고, 그 값을 다시 int 값으로 변경해준 다른 변수로 넣어주게 된다면 해결이 되나? 라는 생각을 하게 만들었다. 하지만 이것은 string 값을 int 값으로 casting이 되지 않기 때문에 그 역시 불가능한 case라고 생각이 들었다. 하지만 왜 string 값을 int 값에 넣어주게 되면 0 값이 들어가는지는 아직까지 의문이 남아있다.

- 또한 이번에 클래스들은 총 두 가지로 구현을 하였는데 4개의 클래스로 구현을 했으면 어땟을까? 라는 생각이 들었다. 멤버에 대한 클래스, 친구에 대한 클래스, 글에 대한 클래스, 댓글에 대한 클래스 이렇게 총 네 가지로 구현을 했으면 더 깔끔한 코드가 나오지 않았을까? 라는 생각도 조금 들었던 것 같다.

- 이번 어싸인을 하면서 마지막으로 갈수록 마무리를 빨리 해야겠다는 생각으로 함수 선언을 하지도 않았고, 더 깔끔하게 처리할 수 있는 것들을 깔끔하게 처리를 못했다는 아쉬움이 남았다. 그리고 main함수에서 모든 것을 처리하는 것이 과연 객체지향 프로그래밍일까? 라는 의문도 남게 되었다. 클래스 메소드가 아닌 다른 함수들을 조금 더 선언을 해서 Main menu, My page, Friend page, Feed page를 구현을 하는 것도 가능하지 않았을까? 라는 의문점을 남기며 어싸인 2를 마무리한 것 같다.

- 마지막으로 이제 곧 나오게 될 어싸인 3는 클래스들의 hierarchy를 이용하는 것이라고 했는데 이번 어싸인을 더 발전시켜서 만들 수 있는 것들을 생각해보면 facebook의 개인 글 포스팅하기, 댓글, 좋아요, 친구추가와 같은 것들이 구현이 되어있으므로, 페이스북 그룹 만들기 또는 페이지 만들기와 같은 것들이 만들어지지 않을까? 하는 예상을 하며 이번 보고서를 마무리한다.

4. 참고 문헌

- https://www.geeksforgeeks.org/io-redirection-c/
- http://blog.naver.com/PostView.nhn?blogId=ruvendix&logNo=220980152324&categoryNo=0&parentCategoryNo=0
- https://bowbowbow.tistory.com/12