CSED101. Programming & Problem solving
Fall 2021

Programming Assignment #4
(75 points)

유상우 (rswoo@postech.ac.kr)

■ **제출 마감일: 2021.12.14 23:59**

■ **개발 환경: Windows Visual Studio 2019**

■ **제출물**

- C 소스 코드 (assn4.c)
  - ➢ 프로그램의 소스 코드에 채점자의 이해를 돕기 위한 주석을 반드시 붙여주세요.
- 보고서 파일 (.docx, .hwp 또는 .pdf; assn4.docx, assn4.hwp 또는 assn4.pdf)
  - ➢ 보고서는 AssnReadMe.pdf를 참조하여 작성하시면 됩니다.
  - ➢ 명예 서약 (Honor code): 표지에 다음의 서약을 기입하여 제출해 주세요: "나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다." 보고서 표지에 명예 서약이 기입되어 있지 않은 과제는 제출되지 않은 것으로 처리됩니다.
  - ➢ 작성한 소스 코드와 보고서 파일은 PLMS를 통해 제출해 주세요.

■ **주의 사항**

- 컴파일이나 실행이 되지 않는 과제는 0점으로 채점됩니다.
- 제출 기한보다 하루 늦게 제출된 과제는 최종 20%, 이틀 늦게 제출된 과제는 최종 40% 감점됩니다. 제출 기한보다 사흘 이상 늦으면 제출 받지 않습니다 (0점 처리).
- 각 문제의 제한 조건과 요구 사항을 반드시 지켜 주시기 바랍니다.
- 모든 문제의 출력 형식은 채점을 위해 아래에 제시된 예시들과 최대한 비슷하게 작성해 주세요.
- 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 "POSTECH 전자컴퓨터공학부 부정행위 정의"를 따릅니다 (PLMS의 본 과목 공지사항에 등록된 글 중, 제목이 [document about cheating]인 글에 첨부되어 있는 disciplinary.pdf를 참조하세요).
- 이번 과제는 추가 기능 구현과 관련된 추가 점수가 따로 없습니다.

■ Problem: 화상 회의 관리 프로그램

[요약]

이 과제에서는 COVID-19 이후 많이 사용하는 Zoom, Webex와 같은 화상 회의 호스트가 되어 회의실 참여자들과 관련된 정보들을 관리하는 것을 목적으로 합니다.

[개요]

주요 기능은 아래와 같습니다.

1. 텍스트 파일로 저장된 유저 정보를 읽고 유저 리스트에 저장
2. 유저 리스트에 유저 추가/삭제, 전체 목록 보기
3. 현재 회의에 참여하고 있는 참여자 출력
4. 회의에 유저 추가/삭제
5. 특정 참여자 카메라/마이크 On/Off
6. 프로그램 종료

[목적]

이번 과제를 통하여 구조체(struct)와 연결리스트(linked list)의 사용법을 익힙니다.

[주의 사항]

1. 이번 과제는 총 9개의 명령어 (표1 참고)를 입력 받아 각 기능을 수행합니다. 적어도 각 명령어 별로 함수를 정의하여 사용해야 합니다. 그 외에 필요한 함수는 따로 정의해서 사용하실 수 있습니다. 기능적으로 독립됐거나 반복적으로 사용되는 기능은 사용자 함수를 정의해서 구현하도록 합니다.
2. 이번 과제는 구조체와 연결리스트를 활용하는 것이 목표이므로, 문제에 구조체와 연결리스트를 언급한 부분 (유저 리스트, 참여자 리스트 구현)을 배열을 이용하여 해결 시 감점 처리됩니다.
3. 전역 변수 및 goto 문은 사용할 수 없습니다.
4. string.h 에서 제공하는 라이브러리 함수를 사용해도 됩니다.
5. 명시된 에러 처리 외에는 고려하지 않아도 됩니다.
   [필수]가 표시된 것은 점수가 포함된 것으로 반드시 구현하고, [가이드라인] 표시된 것은 점수에는 포함되지 않으나 구현에 참고하도록 합니다.
6. 문제의 출력 형식은 채점을 위해 아래의 실행 예시와 최대한 비슷하게 작성해 주세요.
7. 과제 수행 시 도움이 되라고 [참고사항]을 12~13쪽에 작성을 해 두었으니 확인하시기 바랍니다.

[데이터 구조]

```
typedef struct User {
        char name[21];
        char email[41];
        struct User* next;
} USER;

typedef struct Participant {
        USER* user;
        bool camera;
        bool microphone;
        struct Participant* next;
} PARTICIPANT;

typedef struct Conference {
        char roomName[21];
        int numParticipant;
        int maxParticipant;
        PARTICIPANT* participants;
} CONFERENCE;
```

1) User
   - name: 사용자 이름
   - email: 사용자 이메일
   - next: 다음 사용자 주소 (연결리스트에서 사용)

2) Participant
   - user: 사용자를 가리키는 포인터
   - camera: 카메라 On/Off 여부 (On일 때 true, Off일 때 false)
   - microphone: 마이크 On/Off 여부 (On일 때 true, Off일 때 false)
   - bool 자료형은 <stdbool.h>에 포함되어 있습니다.

3) Conference
   - roomName: 회의실 이름
   - numParticipant: 현재 회의실에 참여하고 있는 참여자 수
   - maxParticipant: 회의실에 참여 가능한 최대 참여자 수
   - participants: 회의실에 참여하고 있는 참가자 연결 리스트 시작 노드

[프로그램 기능]

0. 프로그램 시작 화면

A. 프로그램을 실행하면 아래와 같이 읽어올 파일 이름을 입력 받습니다. 파일 이름에는 공백이 없으며, 확장자를 포함해 30자를 넘지 않는다고 가정합니다.

```
유저 리스트 파일 이름을 입력해주세요. >>
```

- 예외처리[필수]: 존재하지 않는 파일의 이름을 받은 경우, 아래와 같은 에러 메시지를 출력 후 다시 파일 이름을 입력 받습니다. (실행 예시의 노란색 글자는 사용자 입력에 해당)

```
유저 리스트 파일 이름을 입력해주세요. >> wrong.txt
유효하지 않은 입력입니다. 유저 리스트 파일 이름을 입력해주세요 >>
```

B. 파일을 열어 위에서부터 읽으며 유저 연결리스트(이하 유저 리스트)에 삽입합니다. 주어진 파일 user.txt의 내용과 구성은 아래와 같습니다.

```
Taeyeon taeyeon@postech.ac.kr
Sunny   sunny@postech.ac.kr
Tiffany tiffany@postech.ac.kr
Hyoyeon hyoyeon@postech.ac.kr
Yuri    yuri@postech.ac.kr
Sooyoung        sooyoung@postech.ac.kr
Yoona   yoona@postech.ac.kr
Seohyun seohyun@postech.ac.kr
```

- 한 줄에 한 유저의 정보가 들어가며, 각 유저 정보는 앞에서부터 차례로 이름, 이메일로 구성됩니다. 이름, 이메일 사이는 tab(‘ \t ’)으로 구분되어 있습니다.
- 이름은 20자, 이메일은 40자 이내의 문자열이며, 두 문자열 모두 공백을 포함하지 않으며, 중복된 이름, 이메일이 없다고 가정합니다.

C. 불러온 유저 리스트를 이름의 알파벳 순서로 정렬합니다. 유저 리스트를 생성하는 과정에서 정렬을 함께 수행하여도 되고, 유저 리스트를 완전히 만든 후 정렬을 수행해도 무관합니다. 정렬 알고리즘 또한 어떤 것을 사용하여도 무관합니다.

D. 불러온 파일 이름과 유저 수를 포함하는 메시지를 출력합니다.

E. 생성할 회의실의 이름과 최대 사용자 수를 입력 받아 Conference 구조체를 생성합니다. 회의실 이름과 최대 사용자 수를 포함하는 안내메시지 출력 후 명령어를 입력 받습니다.

```
유저 리스트 파일 이름을 입력해주세요. >> wrong.txt
유효하지 않은 입력입니다. 유저 리스트 파일 이름을 입력해주세요. >> user.txt

[INFO] user.txt에서 총 8명의 유저를 불러왔습니다.

회의실 이름을 입력해주세요. >> CSED101
최대 사용자 수를 입력해주세요. >> 5
[INFO] CSED101 회의실이 생성되었습니다!
```

```
명령어를 입력해주세요. >>
```

- 회의실 이름은 공백을 포함하지 않는 20자 이내의 문자열입니다.
- Conference 구조체는 정적 할당, 동적 할당 두 경우 모두 가능하며, 동적할당을 하는 경우 프로그램의 마지막에 할당 해제(free)를 해 주어야 합니다.

F. 명령어는 아래 명령어 리스트 중 하나만 가능하며, 명령어 문자열의 각 단어는 스페이스 1개로 구분됩니다. 명령어 문자열의 최대 길이는 25자로 제한합니다.

명령어	설명
user list	전체 유저를 출력합니다.
user add	유저 연결리스트에 유저를 추가합니다.
user delete	유저 연결리스트에서 유저를 제거합니다.
conf info	회의실 이름, 회의에 참여 중인 참여자 목록 등 회의실과 관련된 모든 정보들을 출력합니다.
conf join	회의에 특정 유저를 추가합니다.
conf hangup	회의에서 특정 유저를 제외합니다.
conf toggle camera	특정 참여자의 카메라 On/Off 상태를 바꿉니다.
conf toggle mic	특정 참여자의 마이크 On/Off 상태를 바꿉니다.
quit	프로그램을 종료합니다.

[표 1. 명령어 리스트]

- 예외처리[필수]: 유효하지 않은 명령어를 입력 받은 경우, 아래와 같이 에러 메시지를 출력 후, 명령어를 재입력 받습니다.

```
명령어를 입력해주세요. >> conf joinsung
유효하지 않은 입력입니다. 명령어를 입력해주세요. >>
```

1. 회의 관리(유저 관리)

A. *user list*: 전체 유저를 출력

- 유저 연결리스트에서 모든 유저의 정보를 순서대로 출력합니다. (출력 형식은 12 쪽 [참고사항] 1 을 참고)

```
명령어를 입력해주세요. >> user list

============================================================
번호              이름                                 이메일
------------------------------------------------------------
 1              Hyoyeon                  hyoyeon@postech.ac.kr
 2              Seohyun                  seohyun@postech.ac.kr
 3             Sooyoung                 sooyoung@postech.ac.kr
 4               Sunny                    sunny@postech.ac.kr
 5              Taeyeon                  taeyeon@postech.ac.kr
 6              Tiffany                  tiffany@postech.ac.kr
 7               Yoona                    yoona@postech.ac.kr
 8                Yuri                     yuri@postech.ac.kr
============================================================

명령어를 입력해주세요. >>
```

- 예외처리[가이드라인]: 유저 리스트에 유저가 존재하지 않을 경우에, 아래와 같이 틀만 출력합니다.

```
명령어를 입력해주세요. >> user list

============================================================
번호              이름                                 이메일
------------------------------------------------------------
============================================================
```

B. *user add*: 유저 연결리스트에 유저 추가

- 추가할 유저의 정보(이름, 이메일)를 순서대로 입력 받아 User 구조체를 할당해 유저 리스트에서 유저 이름 알파벳 순서에 맞는 올바른 위치에 추가합니다.
- 이름, 이메일을 포함하는 안내 메시지를 출력합니다.
- 예외처리[필수]: 이미 존재하는 유저를 입력한 경우, 아래의 안내 메시지를 출력하고 다시 명령어를 입력 받습니다

```
명령어를 입력해주세요. >> user add
추가할 유저의 이름을 입력해주세요. >> Jin
추가할 유저의 이메일을 입력해주세요. >> jin@postech.ac.kr
[INFO] Jin, jin@postech.ac.kr 유저가 추가되었습니다.

명령어를 입력해주세요. >> user add
추가할 유저의 이름을 입력해주세요. >> Jin
[INFO] Jin, jin@postech.ac.kr 유저가 이미 유저 리스트에 있습니다.
```

C. *use delete*: 유저 연결리스트에서 유저 제거

- 제거할 유저의 이름을 입력 받습니다.
- 유저 리스트에서 해당 유저를 찾아 이름, 이메일 정보와 함께 정말로 제거할 것인지 여부를 확인합니다.
- y 를 선택한 경우, 유저 리스트에서 해당 유저를 제거하고, 이름, 이메일을 포함하는 제거 안내 메시지를 출력합니다. 유저 리스트에서 삭제 시에는 할당 받은 메모리를 할당 해제(free)합니다.
- n 을 선택한 경우, 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> user delete
제거할 유저의 이름을 입력해주세요. >> Yuri
Yuri, yuri@postech.ac.kr 유저를 제거하시겠습니까? (y/n) >> y
[INFO] Yuri, yuri@postech.ac.kr 유저를 제거하였습니다.

명령어를 입력해주세요. >> user delete
제거할 유저의 이름을 입력해주세요. >> Sunny
Sunny, sunny@postech.ac.kr 유저를 제거하시겠습니까? (y/n) >> n

명령어를 입력해주세요. >> user list

============================================================
번호              이름                                 이메일
------------------------------------------------------------
 1             Hyoyeon                  hyoyeon@postech.ac.kr
 2                 Jin                      jin@postech.ac.kr
 3             Seohyun                  seohyun@postech.ac.kr
 4            Sooyoung                 sooyoung@postech.ac.kr
 5               Sunny                    sunny@postech.ac.kr
 6             Taeyeon                  taeyeon@postech.ac.kr
 7             Tiffany                  tiffany@postech.ac.kr
 8               Yoona                    yoona@postech.ac.kr
============================================================

명령어를 입력해주세요. >>
```

- 예외처리[필수]: 유저 리스트에 없는 이름을 입력하는 경우, 아래의 안내 메시지를 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> user delete
제거할 유저의 이름을 입력해주세요. >> Suga
[INFO] 해당 유저는 존재하지 않습니다!

명령어를 입력해주세요. >>
```

- 예외처리[가이드라인]: 유저 리스트가 비어 있는 경우, 아래의 안내 메시지를 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> user delete
[INFO] 유저 리스트가 비어 있습니다!

명령어를 입력해주세요. >>
```

2. 회의 관리(회의실 관리)

A. *conf info*: 회의에 참여중인 참여자 목록 출력

- 회의실 이름, 최대 사용자 수, 참여자 목록을 출력합니다
- 참여자 목록은 현재 회의에 참여중인 참여자들의 표시되는 이름, 이메일, 카메라 On/Off, 마이크 On/Off 정보를 모두 보여줍니다.
- 프로그램 실행 후, 'conf join' 명령어를 통해 회의 참여자를 추가하기 전까지는 참여자 목록은 비어 있습니다.

```
명령어를 입력해주세요. >> conf info

회의실 이름: CSED101
최대 사용자 수: 5
참여자 목록:
================================================================================
               이름                              이메일     카메라     마이크
--------------------------------------------------------------------------------
            Taeyeon                 taeyeon@postech.ac.kr         ON         ON
              Sunny                   sunny@postech.ac.kr         ON        OFF
            Tiffany                 tiffany@postech.ac.kr        OFF         ON
================================================================================

명령어를 입력해주세요. >>
```

- 예외처리[가이드라인]: 참여자 목록에 참여자가 존재하지 않을 경우에, 아래와 같이 틀만 출력합니다.

```
명령어를 입력해주세요. >> conf info

회의실 이름: CSED101
최대 사용자 수: 5
참여자 목록:
================================================================================
               이름                              이메일     카메라     마이크
--------------------------------------------------------------------------------
================================================================================

명령어를 입력해주세요. >>
```

B. *conf join*: 회의에 유저 추가

- 회의에 추가할 유저의 이름을 입력 받습니다.
- 카메라 On/Off 여부를 입력 받습니다.
- 마이크 On/Off 여부를 입력 받습니다.
- 해당 정보를 이용해 Participant 구조체를 할당해 Conference 구조체의 참여자 연결 리스트(이하 참여자 리스트)의 맨 뒤에 추가합니다.
- Participant 구조체를 만들 때, 구조체 멤버인 User* user 에 User 구조체를 새로 할당하지 않도록 주의합니다. 유저 리스트 생성시 할당한 User 구조체를 가리켜야 합니다. ([참고사항] 13 쪽 ‘ 4. 전체구조 ’ 를 참고)

- 예외처리[필수]:
  이미 회의에 참여한 유저를 입력한 경우에는 '회의에 이미 참여했습니다!'라는 메시지를 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf join
추가할 유저의 이름을 입력해주세요. >> Hyoyeon
카메라를 켠 상태로 시작하시겠습니까? (y/n) >> n
마이크를 켠 상태로 시작하시겠습니까? (y/n) >> n
[INFO] Hyoyeon 이(가) 회의에 참여했습니다!

명령어를 입력해주세요. >> conf join
추가할 유저의 이름을 입력해주세요. >> Hyoyeon
[INFO] Hyoyeon 이(가) 회의에 이미 참여했습니다!

명령어를 입력해주세요. >> conf join
추가할 유저의 이름을 입력해주세요. >> Seohyun
카메라를 켠 상태로 시작하시겠습니까? (y/n) >> y
마이크를 켠 상태로 시작하시겠습니까? (y/n) >> y
[INFO] Seohyun 이(가) 회의에 참여했습니다!

명령어를 입력해주세요. >> conf info

회의실 이름: CSED101
최대 사용자 수: 5
참여자 목록:
================================================================================
                이름                              이메일     카메라     마이크
--------------------------------------------------------------------------------
             Taeyeon              taeyeon@postech.ac.kr         ON         ON
               Sunny                sunny@postech.ac.kr         ON        OFF
             Tiffany             tiffany@postech.ac.kr        OFF         ON
             Hyoyeon             hyoyeon@postech.ac.kr        OFF        OFF
             Seohyun             seohyun@postech.ac.kr         ON         ON
================================================================================

명령어를 입력해주세요. >>
```

- 이미 최대 인원이 참여하고 있는 경우에는 회의 최대 참여자 수를 포함하는 메시지를 출력하고, 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf join
[INFO] 회의에 최대 인원 5 명이 참여하고 있습니다!

명령어를 입력해주세요. >>
```

- 예외처리[필수]: 해당 유저가 유저 리스트에 존재하지 않는 경우, 아래와 같은 안내 메시지 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf join
추가할 유저의 이름을 입력해주세요. >> Suga
[INFO] 해당 유저는 존재하지 않습니다!
```

```
명령어를 입력해주세요. >>
```

C. *conf hangup*: 회의에서 유저 제외

- 제외할 참여자의 이름을 입력 받습니다.
- 입력 받은 참여자를 참여자 리스트에서 찾아 삭제 후, 할당 해제를 합니다.
- 예외처리[필수]: 해당 참여자가 참여자 리스트에 존재하지 않는 경우 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf hangup
제외할 유저의 이름을 입력해주세요. >> Seohyun
정말로 회의에서 내보내시겠습니까? (y/n) >> y
[INFO] Seohyun 이(가) 회의에서 나갔습니다!

명령어를 입력해주세요. >> conf hangup
제외할 유저의 이름을 입력해주세요. >> Suga
[INFO] 해당 유저는 회의에 존재하지 않습니다!

명령어를 입력해주세요. >>
```

- 예외처리[가이드라인]: 참여자 리스트가 비어 있는 경우 아래와 같은 안내 메시지 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf hangup
[INFO] 회의 참석자가 없습니다.

명령어를 입력해주세요. >>
```

D. *conf toggle camera*: 참여자 카메라 On/Off

- 카메라 On/Off 여부를 변경할 참여자의 이름을 입력 받습니다.
- 카메라 On/Off 변경 여부를 포함하는 안내 메시지를 출력하고, 다시 명령어를 입력 받습니다.
- 예외처리[필수]: 해당 참여자가 참여자 리스트에 존재하지 않는 경우 아래와 같은 안내 메시지를 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf toggle camera
카메라 상태를 변경할 유저의 이름을 입력해주세요. >> Taeyeon
[INFO] Taeyeon 의 카메라가 ON -> OFF 되었습니다!

명령어를 입력해주세요. >> conf toggle camera
카메라 상태를 변경할 유저의 이름을 입력해주세요. >> Suga
[INFO] 해당 유저는 회의에 존재하지 않습니다!

명령어를 입력해주세요. >>
```

- 예외처리[가이드라인]: 참여자 리스트가 비어 있는 경우 아래와 같은 안내 메시지 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf toggle camera
[INFO] 회의 참석자가 없습니다.
```

```
명령어를 입력해주세요. >>
```

E. *conf toggle mic*: 참여자 마이크 On/Off

- 마이크 On/Off 여부를 변경할 참여자의 이름을 입력 받습니다.
- 마이크 On/Off 변경 여부를 포함하는 안내 메시지를 출력하고, 다시 명령어를 입력 받습니다.
- 예외처리[필수]: 해당 참여자가 참여자 리스트에 존재하지 않는 경우 아래와 같은 안내 메시지를 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >> conf toggle mic
마이크 상태를 변경할 유저의 이름을 입력해주세요. >> Taeyeon
[INFO] Taeyeon 의 마이크가 ON -> OFF 되었습니다!

명령어를 입력해주세요. >> conf toggle mic
마이크 상태를 변경할 유저의 이름을 입력해주세요. >> Suga
[INFO] 해당 유저는 회의에 존재하지 않습니다!

명령어를 입력해주세요. >>
```

- 예외처리[가이드라인]: 참여자 리스트가 비어 있는 경우 아래와 같은 안내 메시지 출력하고 다시 명령어를 입력 받습니다.

```
명령어를 입력해주세요. >>  conf toggle mic
[INFO] 회의 참석자가 없습니다.

명령어를 입력해주세요. >>
```

F. *quit*: 프로그램 종료

- 종료 시, 동적 할당 받은 메모리를 모두 할당 해제한 후 종료합니다.

```
명령어를 입력해주세요. >> quit
[INFO] 회의를 종료합니다!

계속하려면 아무 키나 누르십시오 . . .
```

[참고사항]

과제 수행 시 다음 내용을 참고하세요.

1. user list, conf info 출력 형식
   A. User 구조체의 경우 name은 20자, email은 40자로 제한했기 때문에, 아래와 같이 출력 형식을 입력하면 예시처럼 빈 공간이 있고, 오른쪽 정렬이 되어있는 결과를 얻을 수 있습니다.

```
printf("%20s %40s\n", name, email);
```

2. 공백을 포함한 문자열 입력
   A. 공백이 있는 문자열을 입력 받을 때는 <stdio.h>의 fgets() 함수를 사용합니다.

   char *fgets(char *str, int num, FILE *stream)

   - 스트림에서 문자열을 최대 (num - 1)개만큼 받아서 str이 가리키는 메모리에 저장합니다. \n 이 포함된 경우 그 문자열의 길이가 num보다 작더라도 더 읽지 않으며, 결과 문자열에 \n까지 저장합니다.

```
char str[10];
fgets(str, 10, stdin); // stdin은 키보드로부터 입력 받는 것을 의미
printf("%s\n", str); // 입력 받은 문자열을 출력

FILE *fp = fopen("file.txt", "r");
fgets(str, 10, fp);// 파일에서부터 한 라인을 읽음
printf("%s\n", str);// 파일로부터 읽은 문자열을 출력
```

3. 특정 문자를 기준으로 문자열 분리
   A. 문자열을 특정 문자를 기준으로 나눌 때는 <string.h>의 strtok()함수를 사용합니다.

   char *strtok(char *str, const char *delim)

   - 문자열 str을 delim에 포함된 문자들로 분리(tokenize)합니다. 예를 들어 “Hello/world!”라는 문자열에 대해 “/”을 기준으로 분리하면 “Hello”와 “world!”로 나누어집니다. 이 때 기준의 되는 문자(“/”)를 delimiter, 나누어진 문자들을 token이라고 합니다.

```
char str[100];
char *token1, *token2, *token3;
strcpy(str, "Tokenization/Test/String");
token1= strtok(str, "/"); // token1 = "Tokenization"
token2 = strtok(NULL, "/"); // token2 = "Test"
token3 = strtok(NULL, "/"); // token3 = "String"
```

4. 전체 구조

A. 프로그램은 USER 구조체로 이루어진 유저 연결리스트와 하나의 CONFERENCE 구조체, 그리고 CONFERENCE 구조체 내에 PARTICIPANT 구조체로 이루어진 참여자 연결리스트가 존재합니다. 이해를 돕기 위해 그림을 첨부하면 아래와 같습니다.