Every user will have a unique username (email)

Eg: @Shashank01 → String.

or Shashank@gmail.com → string.

2 Structs
- → Users
  - → Name → string
  - → Email → string
  - → RSVP ENUM
    - → Yes
    - → No
    - → MayBe
    - → Not Answered.
- → Meeting
  - → Id → ~~const~~ long int. (Bcoz a lot of meeting can occur)
  - → Title → string.
  - → Participants → string [,] → email will go here.
  - → Start time → date format.
  - → end time → date format
  - → Creation Timestamp → = Now().

Create a constructor which

① validates the input

ⓐ start time < end time

ⓑ Participants != 0

ⓒ Title != Null.

~~ⓓ Participants.size()~~ Kz

Functions Logic

(1) Shedule a meeting ( )

→ if(meeting start time > meeting end time)
{
throw error
}

→ if (meeting title == null)
{
throw error
}

→ if (no. of participants == 0)
{
throw error
}

→ else
{
start the meeting
}

6. List of all meetings of a participant

```
( participant string C])
{
  Iterate through every meeting
  if (meeting.participant email
      == participant email)
  {
    meeting IDs append (meeting.id)
  }
```

36)

List of all meetings (start, end)

{

Iterate through all meetings

for every meeting check

if (start <= end time)

meeting. append (meeting.ID)

}

```
Get a meeting (ID, participants)
{
  Get (ID)
    if ID == nil
       return error:
    iterate through all participants
    if (participant.email == user email)
       return URL
    else
      Throw error
}
```