자료구조 프로젝트 2 보고서

카카오톡 서버와 클라이언트를 위한 자료구조 설계 및 구현.

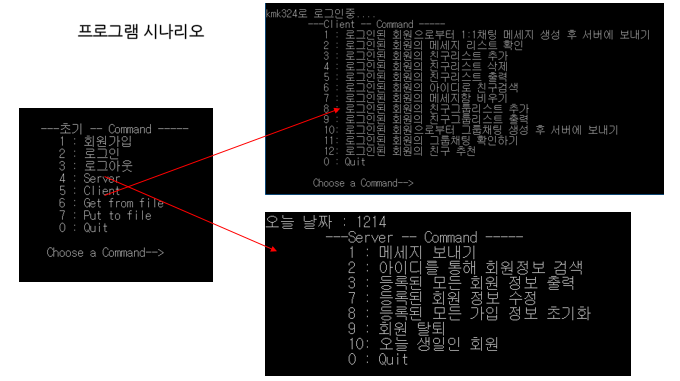
2013101775 김민규

제출일: 2017.12.14

1. **프로젝트 주제 및 목표**

현재 우리나라에서 가장 대중적으로 사용되고 있는 메신저 카카오톡의 기능들을 분석하고 최적의 자료구조를 이용하여 설계하고 구현한다. 필요한 새로운 기능을 분석해보고 최적의 자료구조를 이용하여 설계 및 구현한다.

**2. 프로그램 구조 및 기능 요약도.**



1. 회원가입 : 회원의 정보를 키보드로 입력 받아 회원리스트에 추가해준다.

2. 로그인 : 서버에있는 아이디와 비밀번호 정보를 기반으로 로그인 한다. 로그인이 완료되면 CurrentMember유저타입 포인터 변수가 로그인된 회원을 가리킨다. LogonStateE부울 변수가 True로 설정되어 5. Client 메뉴에 입장이 가능해진다 3. 로그아웃 기능도 활성화 된다.

3. 로그아웃 : 로그인 된 회원을 로그아웃 한다. LogonStateE변수를 False로 바꾼다. 5번 Client메뉴에 접속할 수 없게 된다.

4. Server : 서버에서 처리할 수 있는 기능들의 목록을 보여준다.

4.1 메시지 보내기

서버에 올라와 있는 메시지들을 메시지 큐가 빌 때 까지 각 클라이언트(리시버)들에게 보낸다.

메세지가 그룹 메세지이면 메시지를 받는 회원이 그 그룹을 그룹리스트에 두고 있는지 확인한 후

그렇지 않다면 그룹리스트에 그룹을 추가해준다.

4.2 아이디를 통해 회원정보 검색

아이디를 입력 받고 해당하는 정보의 회원이 서버에 있으면 회원 정보를 출력해준다.

4.3 등록된 모든 회원 정보 출력

서버에 등록된 모든 유저의 정보를 출력해 준다.

4.7 등록된 회원 정보 수정

아이디를 입력 받아 해당하는 정보의 회원이 서버에 있으면 정보를 수정해준다.

4.8 등록된 모든 가입 정보 초기화

서버에 등록된 회원리스트를 비운다.

4.9 회원 탈퇴

아이디를 입력 받고 해당하는 정보의 회원이 서버에 있으면 삭제한다.

4.10 오늘 생일인 회원

오늘의 날짜와 회원의 생일정보를 비교하여 일치하면 그 회원의 정보를 출력해준다.

5. Client : 회원이 처리할 수 있는 기능들을 보여준다.(LogonStateE변수가 True이면 가능)

5.1 로그인된 회원으로부터 1:1채팅 메시지 생성 후 서버에 보내기

센더를 로그인된 회원아이디로 자동적으로 설정하고, 리시버와 메시지를 키보드로 입력 받아서 저장된 메시지 정보를 로그인회원의 채트리스트에 넣고, 서버의 메시지큐에 인큐한다.

5.2 로그인된 회원의 메시지 리스트 확인

로그인 회원의 메시지들을 상세히 보여준다.

5.3 로그인된 회원의 친구리스트 추가

로그인 회원의 임시 친구리스트에 아이디를 입력 받아서 해당하는 정보의 회원이 있으면 회원의 정보를 받아와 임시 친구리스트에 추가한다.

5.4 로그인된 회원의 친구리스트 삭제

친구아이디를 입력 받고, 로그인된 회원으로부터 해당하는 친구의 정보가 존재하면 삭제한다.

5.5 로그인된 회원의 친구리스트 출력

로그인회원의 친구리스트정보를 출력해준다.

5.6 로그인된 회원의 아이디로 친구검색

친구아이디를 입력 받고, 로그인된 회원으로부터 해당하는 친구의 정보가 존재하면 출력한다.

5.7 로그인된 회원의 메시지함 비우기

로그인회원의 채트리스트를 모두 비운다.

5.8 로그인된 회원의 친구그룹리스트 추가

로그인 회원의 친구리스트목록을 출력해주고, 새로운 친구그룹리스트의 이름을 설정, 그룹에 추가할 친구들의 아이디를 입력 받아 새로운 친구 그룹리스트를 추가해준다. (그룹 채팅을 위해 로그인 회원 또한 그룹에 추가해주어야 한다.)

5.9 로그인된 회원의 친구그룹리스트 출력

로그인 회원의 친구그룹리스트 정보를 출력해준다.

5.10 로그인된 회원으로부터 그룹채팅 생성 후 서버에 보내기

로그인 회원의 친구그룹리스트를 입력해서 메시지 정보를 입력 받고, 해당하는 친구들 모두를 리시버로 설정하여 메시지를 서버로 보내준다.

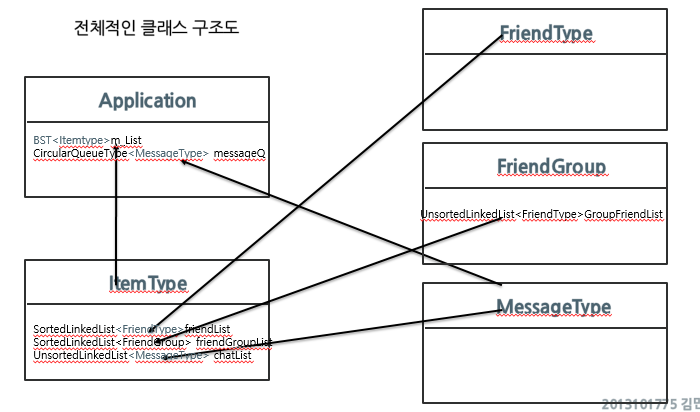
5.11 로그인된 회원의 그룹채팅 확인하기

로그인 회원의 친구그룹리스트를 선택하여 그룹채팅 목록을 출력한다.

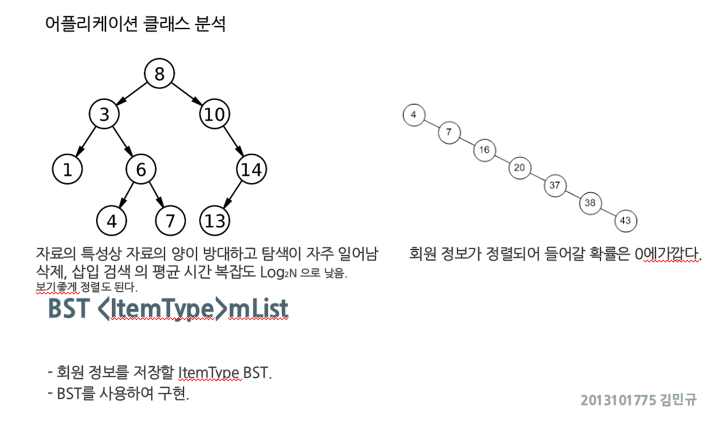
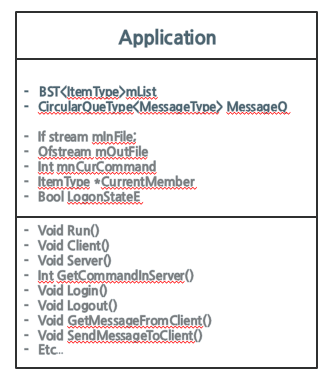
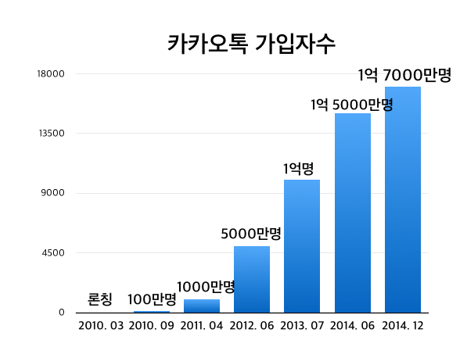
5.12 로그인된 회원의 친구 추천

오늘 생일인 회원 한명이 로그인된 회원과 친구가 아니라면 친구로 추천해준다. (서로 모르는 사이에서 생일을 축하해주면서 친해지는 것이 자연스럽게 친해지기 쉽다고 생각)

**3. 프로그램에서 사용된 자료구조들과 해당 자료구조가 선택된 이유.**



**3.1 Application클래스**



서버에 저장될 회원리스트 mList를 BinarySearchTree자료구조를 이용하여 구현하였다.

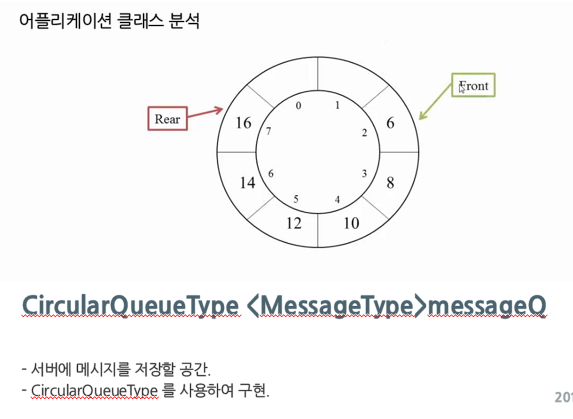
2014년 12월 기준 카카오톡의 가입자수는 1억7000만명을 넘었다.

방대한 회원정보에 많은 검색이 일어난다.

한번 가입한 회원이 탈퇴를 거의 하지 않기 때문에 추가와 삭제에 대한 시간복잡도보다, 잦게 일어나는 **검색에 관한 시간복잡도가 중요하다.**

따라서 기존에 LinkedList(추가와 삭제에 대한 시간복잡도는 O(1)이지만 검색시간복잡도 O(N)으로 상당히 높음)로 구현한 회원리스트를 BinarySearchTree로 변경, 구현 하였다.

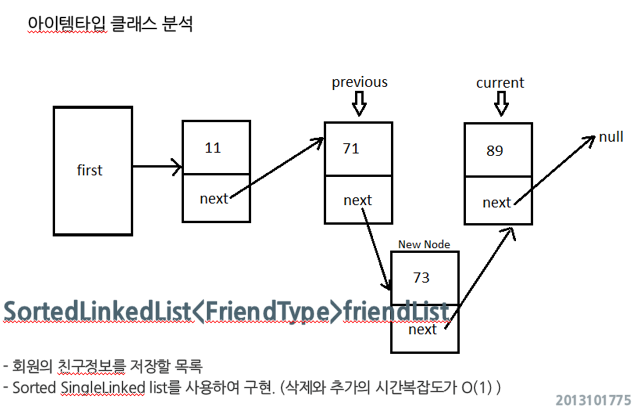
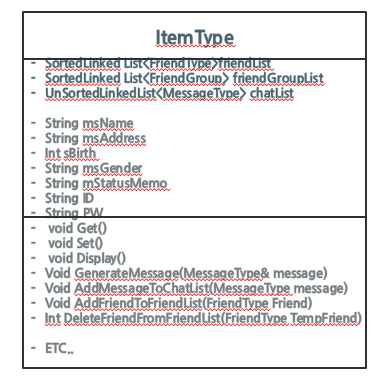
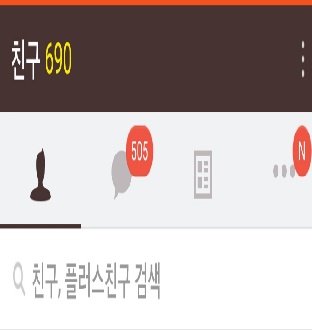
BinarySearchTree는 이번 자료구조 **교과과정에서 배운** 자료구조중 검색의 평균 시간 복잡도(O(logN))가 가장 빠른 자료구조이다. 회원의 정보가 무작위의 키 값으로 들어가기 때문에 BinarySearchTree의 삽입에서 최악의 상황을 일으키는 정렬된 구조로 회원정보가 들어갈 확률이 매우 적다. 또한 삽입시 보기 좋게 정렬이 된다는 장점이 있다.

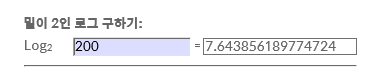
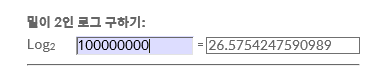
 서버에 저장될 메세지리스트 messageQ를 CircularQueueType의 자료구조로 구현하였다.

메시지는 선입선출 기능이 필요하기에 큐를 선택, 그냥 큐를 사용하면 디큐 할때 프론트가 하나씩 밀려나기 때문에 front앞에 사용할 수 없는 빈 인덱스가 하나씩 생긴다.

원형큐는 이를 막는다. 따라서 원형큐를 사용, 구현하였다.

**3.2 ItemType클래스**

[](http://cafe.naver.com/lolkor/15263090)



**N이 ‘방대해야’ LogN의 효율성이 기하급수적으로 증대 된다.**

N이 1억일 경우(서버의 회원 수 가정) N과 LogN의 차이는 약 1억으로 엄청나다.

하지만 N이 200일 경우 LogN과의 차이는 190가량으로 컴퓨터의연산속도를 보았을때 매우 적다.

링크드리스트의 시간복잡도를 구해보자.

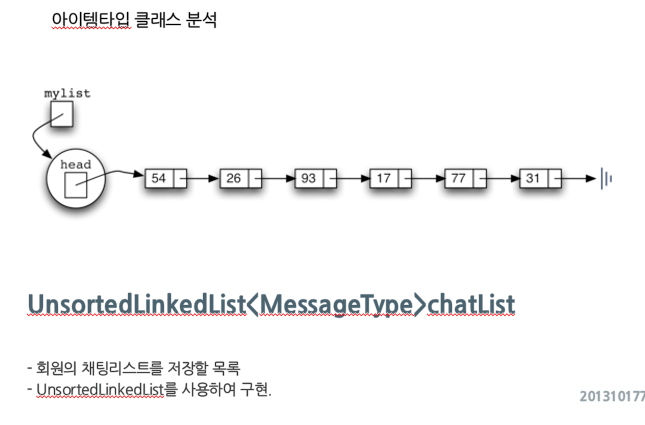
N이 200일 경우, 링크드 리스트의 검색 O(200) **삭제O(1) 추가 O(1)**

반면 BinarySearchTree는

검색 O(7.6) **삭제 O(7.6) 추가 O(7.6)**이다.

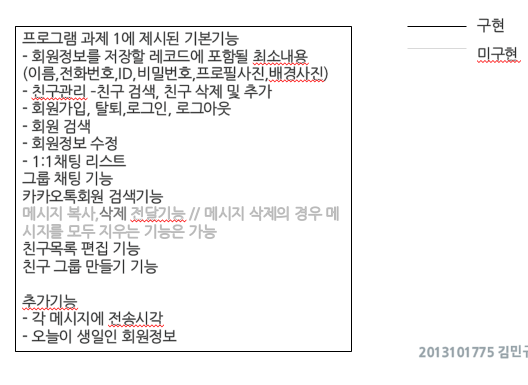
카카오톡의 친구의 수는 서버의 회원 수에 비해 현저히 적다. 따라서 **검색에 대한 시간복잡도O(N)과 O(LogN)의 체감상 차이가 미비하다.** 또한 추가 삭제 기능이 많이 일어난다. 따라서 검색시간복잡도가 낮은 BST보다 추가와 삭제의 시간복잡도가 O(1)로 빠른 링크드리스트가 효율적이다. 검색의 경우 **링크드리스트가** BinarySearchTree보다 26배정도 비효율적이지만 **삭제와 추가의 경우 7.6배 효율적**이기에 삭제와 추가가 각각 2번씩만 일어나도 30배정도 효율적이며 이는 26배보다 높은 효율성을 갖는다. 또한 실질적으로 카카오톡 친구 특성상 검색보다는 추가와 삭제가 이처럼 빈번하게 일어난다. 따라서 회원의 친구리스트를 SortedLinkedList로 구현 하였다.(정렬을 위해 LinkedList중에서도 SortedLinkedList사용)

또한 FriendGroup은 Friend중에 뽑아서 그룹을 만드는 것이기 때문에 위와 같은 이유는 더욱 극대화 된다. (N이 훨씬 더 적어진다.) 따라서 회원의 FriendGroupList를 SortedLinkedList로 구현하였다. (FriendGroup클래스의 GroupFriendList 또한 동일. 위와 같은 이유가 더욱더 극대화.)



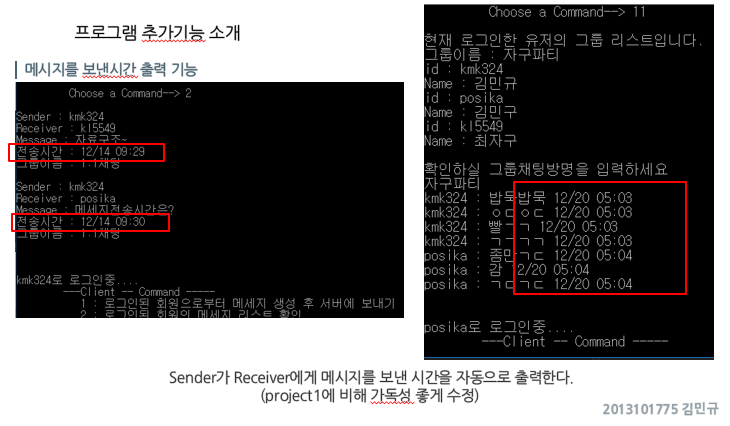
자료구조 프로젝트1때 ArrayList로 구현하였던 회원의 채팅리스트가 저장될 ChatList를UnsortedLinkedList 자료구조를 사용하여 구현하였다. 채팅 특성상 어느 정도의 공간이 필요할 지 가늠할 수 없으므로 메모리를 동적으로 할당 받아 가변적으로 가능해야 한다. 따라서 ArrayList가 아닌 메모리 효율이 높은 LinkedList를 선택하였다. 또한 키값을 바탕으로 정렬을 하면 채팅의 순서가 뒤죽박죽이 되기 때문에 ‘Unsorted’LinkedList를 사용하여 구현하였다.

**4. 체크리스트**



**5. 추가기능 소개**

5.1 메시지를 보낸 시간 출력기능



5.2 오늘이 생일인 유저 알림 기능

