Date Structure Projcet 1

학　　과: 컴퓨터정공학부

담　당　교 수: 이기훈 교수님

학　　번:　2020202010

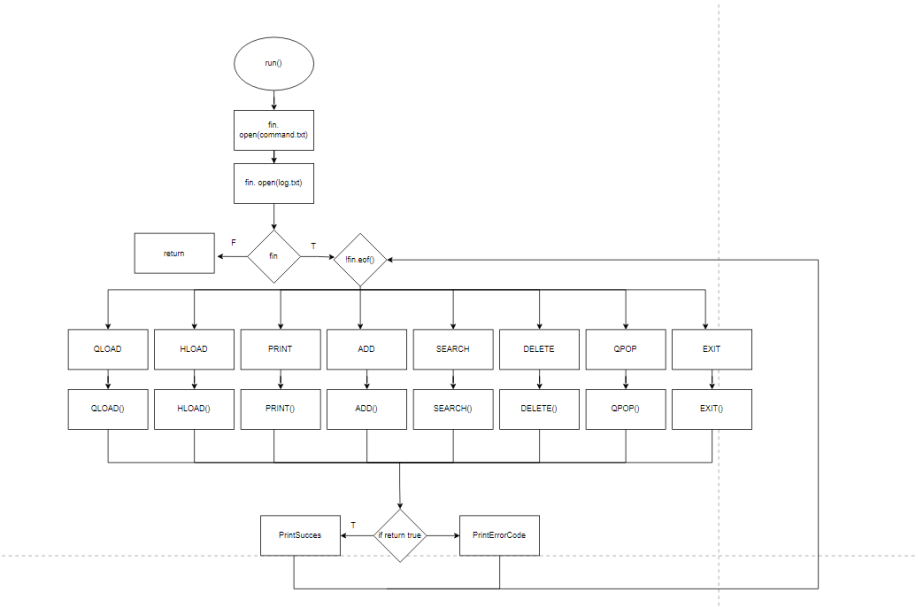
이　　름 : 배 재 용

Introduction

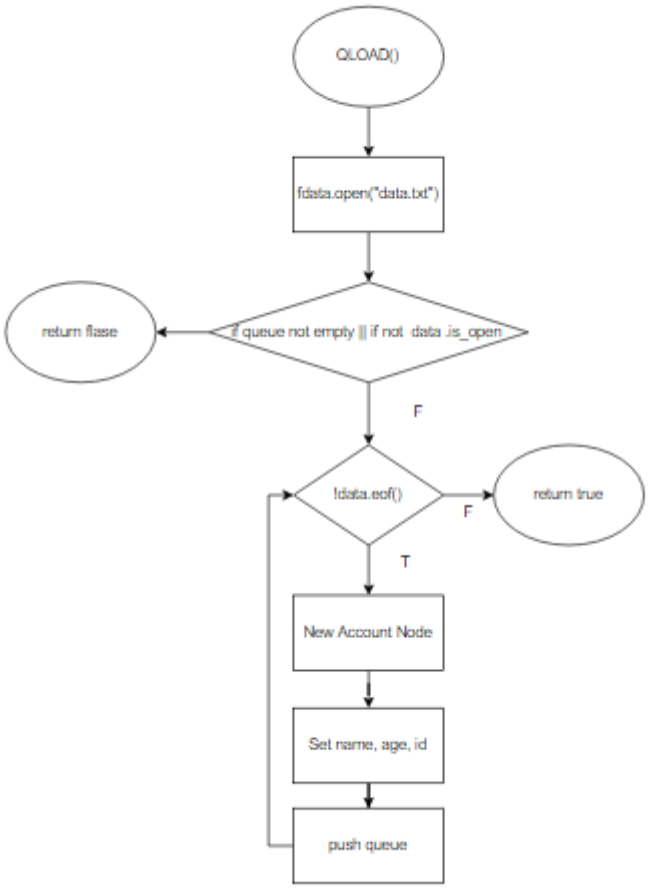
Project 1에서는 Queue, Binary Search Tree, Heap, Linkedlist를 사용하여 계정 관리 프로그램을 구현하여야 합니다. 우선 Queue에 Account Queue로 계정 정보를 저장합니다. Queue는 FIFO(first in first out)의 성질을 가집니다. 그래서 Queue에 먼저 들어간 정보가 먼저 나옵니다. BST에서는 사용자의 이름, 나이, ID가 저장되어 있습니다. 계정 ID와 사용자의 이름으로 노드를 구성합니다. 또한 계정 기준으로 BST를 연결합니다. 또한 고객의 정보는 Userlist에도 똑같이 저장됩니다. UserList에는 사용자의 이름, 나이, 보유한 계정수가 저장된 노드가 구성됩니다. 노드가 입력된 순서대로 List를 연결합니다. User_List의 노드는 Account Bst노드를 가르키는 포인터를 추가로 가져 Linkedlist로 연결합니다. Heap은 연령대와 연령별 사용자 수로 노드를 구성하며 연령대별 사용자 수를 기준으로 정렬 됩니다. 명령어를 수행하여 Queue와 BST, Userlist에 정보를 저장하고 Heap에 불러오도록 합니다.

Flowchart

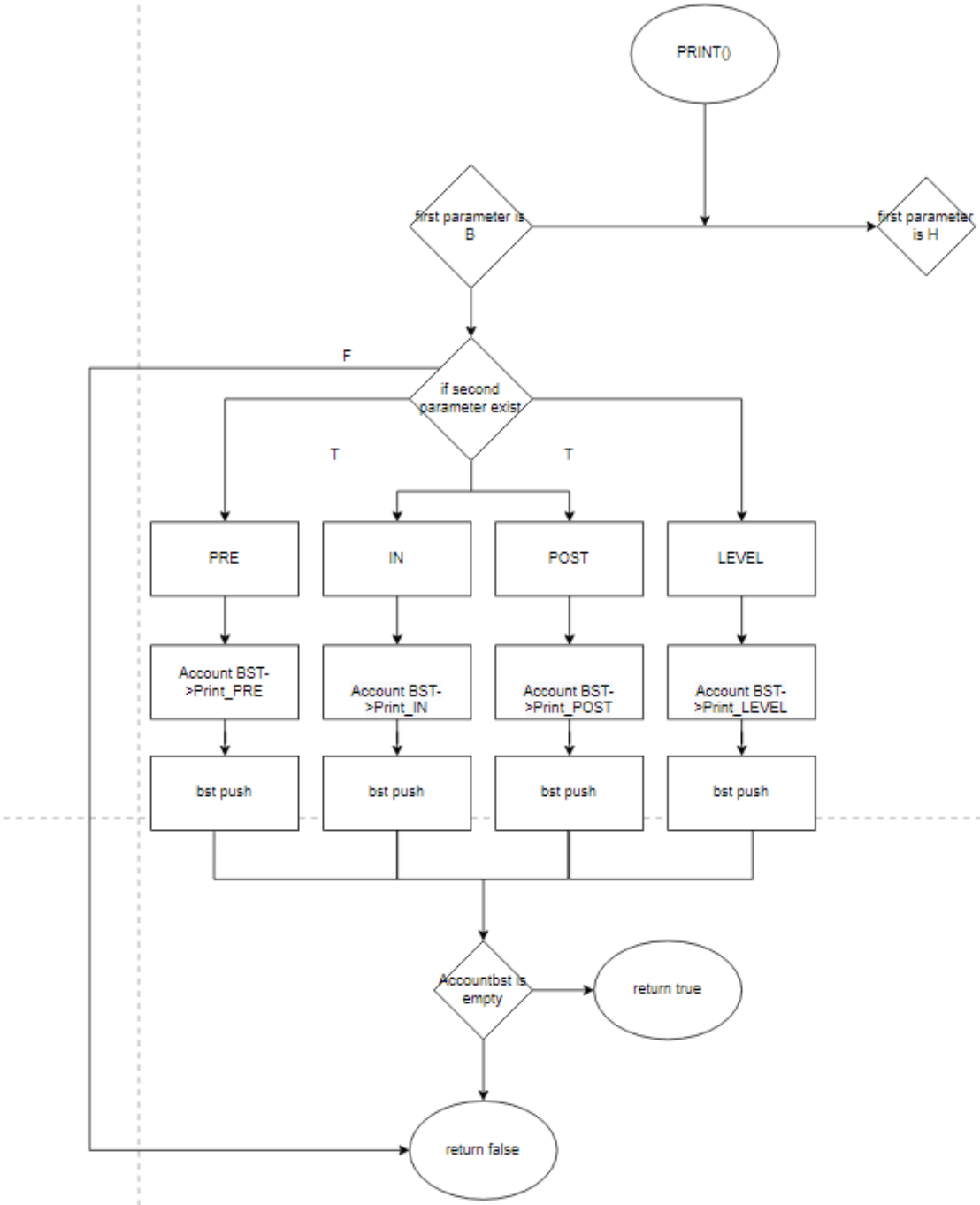
Run()

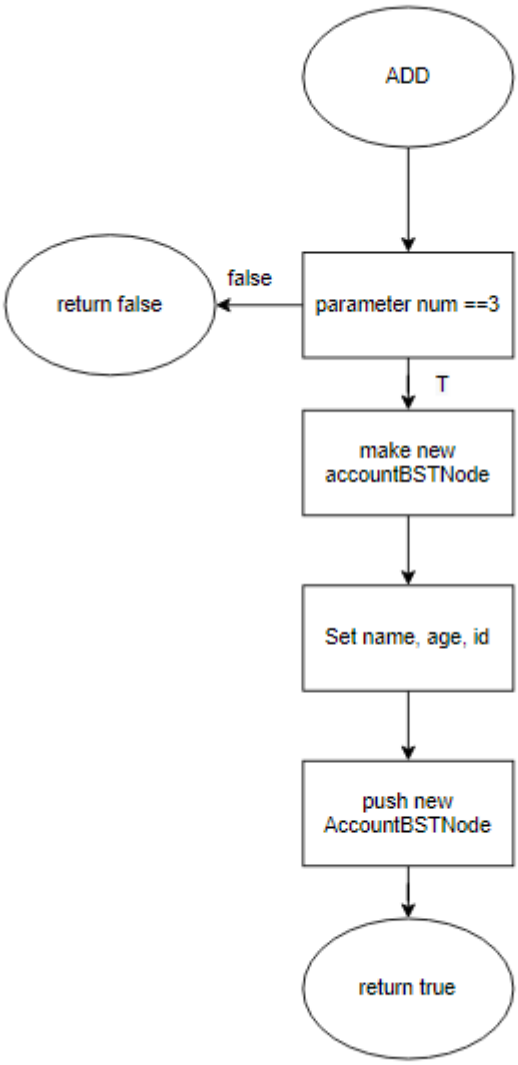


QLOAD()


QLOAD()
fdata.open("data.txt")
if queue not empty || if not data .is_open
return flase
F
!data.eof()
return true
F
T
New Account Node
Set name, age, id
push queue

PRINT()


PRINT()
first parameter is B
first parameter is H
F
if second parameter exist
T
T
PRE
IN
POST
LEVEL
Account BST->Print_PRE
Account BST->Print_IN
Account BST->Print_POST
Account BST->Print_LEVEL
bst push
bst push
bst push
bst push
Accountbst is empty
return true
return false



<Algorithm>

프로그램을 실행시키기 위해서는 Manager 객체를 사용하여야 합니다. 객체를 생성 후에 Queue, BST, Heap, Userlist 등을 명령어를 이용하여 호출하고 활용하기 위해 run()이 우선 호출됩니다. 호출되면 그 후 command.txt가 파일을 읽고 한 줄 씩 읽어와 각 줄에 쓰인 명령어를 실행합니다. 프로그램 내에서 실행 가능한 명령어는 QLOAD, ADD, QPOP, SEARCH, PRINT, DELETE, HLOAD, EXIT가 있습니다. Stork를 사용하여 명령어를 확인합니다. 명령어를 사용하기위한 조건을 충족 시키는 문장인지 확인합니다. 명령어는 공백 문자로 구분이되기에 stork 를 사용하여 구분합니다. 만약 명령어

가 조건에 맞지 않는다면 errorcode를 반환합니다.

AccountBST

AcocuntBST 생성 시에는 AccountBST Class의 생성자를 이용하며 초기에 구축 하며 초기화를 해줍니다. 각 노드에 유저의 정보를 저장합니다. 그리고 Insert를 통하여 BST를 구축합니다. 우선 Root가 Null이 아닌지 검사를 합니다. 만약 Null이면 insert하여야 할 Node를 Root에 연결합니다. 그 후 Root 노드가 존재하면 BST의 Root노드부터 insert를 하며 비교를 합니다. 계정의 ID의 사전적 순서가 작은 노드는 왼쪽 크면 오른쪽에 위치 하게합니다. 이 부분은 SetRight, SetLeft를 이용해서 하게합니다.

ADD()

이 명령어는 환자의 정보를 읽어와 추가 인자들을 읽어 QLOAD에서 수행한 것과 같은 것을 만들고 이용하여 환자 정보를 저장하고 Queue에 Push합니다. Add 명령어 수행 시 QLoad와 다르게 data.txt파일에서 정보를 읽어오는게 아니라 명령어 파일에 입력된 정보를 읽어와 새로 만들어 저장합니다. 인자의 개수가 맞지 않으면 (명령어) ErrorCode를 출력합니다. 마찬가지고 공백 문자로 strok를 이용하여 구분합니다. Push를 통해 queue의 queue에 넣습니다.

QPOP()

Queue의 데이터를 BST와 List로 옮깁니다. Queue의 front부터 첫번째 인자의 숫자만 큼 데이터가 방출되도록 하며 Queue에 저장된 데이터보다 높은 숫자가 입력되면 에러코드를 출력하도록 합니다. Queue에 저장된 환자의 정보를 인자가 나타내는 수 만큼 pop하도록 하여BST와 list로 옮깁니다. 만약 size보다 크거나 queue가 비어 있다면 error code를 출력합니다. 따라서 POP과PUSH를 이용하여 정보와 데이터를 꺼내어 BST와LIST에 저장합니다.

SEARCH()

이 명령어는 인자로 입력된 유저의 정보를 BST와 LIST에서 찾아 출력하는 명령어 입니다. 첫 번째 인자로 user나 id를 받으며 두 번째 인자로 검색할 정보를 입력 받습니다. 첫 번째 인자로 user사용시 BST에서 left와right로 이동하며 입력 받은 값과 일치할 때 까지 찾으며 일치하는 값을 출력하고 링크드 리스트인 유저 리스트를 통해 저장된 사용자가 보유한 모든 계정을 출력하도록 합니다. Id를 사용할 경우에는 찾고자 하는 계정을 보유한 사용자의 이름을 출력합니다. 노드를 선언하여 탐색하며 일치하는 정보를 bst와 user리스를 통해 발견하면 출력하도록 합니다. 만약 존재하지 않으면 에러코드를 print하며 strok로 인자를 공백 문자로 구분하여 명령어 규칙을 검사하도록 합니다.

PRINT()

이 명령어는 BST와 heap에 저장된 정보를 명령어에 저장된 인자에 따라 log.txt에 출력하는 명령어 입니다. 첫 번째 인자로 B 또는 H가 입력되고, 첫 인자가 H일 경우에는 두 번째 인자가 존재하지 않습니다. Strok를 통해 확인하며 알맞지 않으면 ErrorCode를 print합니다. 첫 번째 인자가 B이면 BST를 순회하고 환자 정보를 새롭게 담을 queue를 선언합니다. 이 queue에 저장된 저옵를 log.txt에 순차적으로 출력합니다. 두 번째 인자가 전위 중위 후위 와 같이 bst의 순회와 같은 명령어가 입력되면 순회 후 순서대로 log.txt파일에 정보를 출력합니다.

EXIT()

프로그램을 종료하고 메모리도 해제합니다.

실행결과

실행이 불가하여 작성하지 못하였습니다.

고찰

Project 1을 너무 늦게 시도한 것 같습니다. 또한 Queue, Heap, BST, Linkedlist와 같이 다시 공부하여야 할 것들도 많았습니다. 우선 처음 프로젝트 스켈레톤 코드를 보았을 때 너무 많은 것들이 있었고 큐, 힙, 리스트, 이진 탐색 트리 이것들을 어떻게 연결할지 정말 문제였습니다. 결과적으로 큐와 이진 탐색 트리를 구성을 하였지만 완벽히 하지 못하였습니다. 또한 처음 사용하는 리눅스를 통하여 컴파일이 불가하고 메모리가 누수되며 여러가지 문제점도 발생하였습니다.

큐를 만들 때 큐에 저장하고 어떻게 이걸 연결하는지 문제였지만 노드를 통하여 연결하고 이것을 이진 탐색 트리나 리스트와 힙과 연결하면 되겠다는 것을 알았고 큐는 쉽게 간단하게 빠르게 작성했던 것 같습니다. 하지만 이진 탐색 트리에서 search를하고 delete를 하는 부분에서 많이 생각을 했고 어렵게 작성하였지만 작동이 올바르게 되지 않아 아직까지도 잘 모르겠습니다. 공부가 부족하고 이러한 부분에 아직 흥미를 느끼지 못하여 하고싶은 공부만을 하기보다는 억지로라도 하면 기본적이 부분이 쌓이고 늘겠다라고 느꼈습니다. 아직 내가 알지 못하고 하지 못하는 것들이 많다는 것도 느꼈습니다. 앞으로 남은 기간 project2 열심히하여 부족한 부분을 채워보고 구현에 성공해야겠다고 생각하였습니다.