자료구조 프로젝트

2015410124 허환

1. **소스코드**

**def** main():  
 *#유저를 읽음* f = open(**"C:\\test\\cp949\\user.txt"**, **"r"**)  
 a = {}  
 l = 0  
 **while True**:  
 user\_id = f.readline()  
 user\_id = user\_id[0:-1]  
 **if not** user\_id:  
 **break** f.readline()  
 name = f.readline()  
 name = name[0:-1]  
 tmp = user\_id  
 tmpn = name  
 a[tmp] = tmpn  
 l = l + 1  
 f.readline()  
  
  
 *#단어를 읽음 유저-트윗수 대응관계* f = open(**'C:\\test\\cp949\\word.txt'**, **'r'**)  
 wdic = {}  
 cntw = {}  
 j = 0  
 **while True**:  
 user\_id = f.readline()  
 user\_id = user\_id[0:-1]  
 **if not** user\_id:  
 **break** f.readline()  
 wd = f.readline()  
 wd = wd[0:-1]  
 tmpi = user\_id  
 tmpw = wd  
 **if** tmpw **in** wdic:  
 wdic[tmpw] = wdic[tmpw] + [tmpi]  
 **else**:  
 wdic[tmpw] = [tmpi]  
 j = j + 1  
 f.readline()  
  
 *#단어를 읽음2 단어-갯수 대응* f = open(**'C:\\test\\cp949\\word.txt'**, **'r'**)  
 wcdic = {}  
 cnti = {}  
 j = 0  
 **while True**:  
 user\_id = f.readline()  
 user\_id = user\_id[0:-1]  
 **if not** user\_id:  
 **break** f.readline()  
 wd = f.readline()  
 wd = wd[0:-1]  
 tmpi = user\_id  
 tmpw = wd  
 **if** tmpi **in** wcdic:  
 wcdic[tmpi] = wcdic[tmpi] + [tmpw]  
 **else**:  
 wcdic[tmpi] = [tmpw]  
 j = j + 1  
 f.readline()  
  
 *# 친구를 읽음* f = open(**'C:\\test\\cp949\\friend.txt'**, **'r'**)  
 fdic = {}  
 k = 0  
 **while True**:  
 user\_id = f.readline()  
 user\_id = user\_id[0:-1]  
 **if not** user\_id:  
 **break** fi = f.readline()  
 fi = fi[0:-1]  
 tmpi = user\_id  
 tmpf = fi  
 **if** tmpi **in** fdic:  
 **if not** tmpf **in** fdic[tmpi]:  
 fdic[tmpi] = fdic[tmpi] + [tmpf]  
 **else**:  
 k = k - 1  
 **else**:  
 fdic[tmpi] = [tmpf]  
 k = k + 1  
 f.readline()  
  
 **while True**:  
 print(**'0. Read data files'**)  
 print(**'1. display statistics'**)  
 print(**'2. Top 5 most tweeted words'**)  
 print(**'3. Top 5 most tweeted users'**)  
 print(**'4. Find all users who tweeted a word (e.g, 연세대)'**)  
 print(**'5. Find all people who are friends of the above users'**)  
 print(**'6. Delete all mentions of a word'**)  
 print(**'7. Delete all users who mentioned a word'**)  
 print(**'8. Find strongly connected components'**)  
 print(**'9. Find shortest path'**)  
 print(**'99. Quit'**)  
  
  
 *#워드에 저장된 정보를 횟수-그 횟수의 단어들 로 저장* wcnt = []  
 wc2= {}  
 **for** tmp **in** wdic:  
 tmpn = len(wdic[tmp])  
 wcnt = wcnt + [tmpn]  
 **if** tmpn **in** wc2:  
 wc2[tmpn]=wc2[tmpn]+[tmp]  
 **else**:  
 wc2[tmpn]=[tmp]  
 wcnt.sort()  
 wwmin = wcnt[0]  
 wwmax = wcnt[len(wcnt) - 1]  
  
 *#워드에 저장된 정보를 횟수-그 횟수의 유저들 로 저장* wccnt=[]  
 wc3={}  
 **for** tmp **in** wcdic:  
 tmpn = len(wcdic[tmp])  
 wccnt = wccnt + [tmpn]  
 **if** tmpn **in** wc3:  
 wc3[tmpn] = wc3[tmpn] + [tmp]  
 **else**:  
 wc3[tmpn] = [tmp]  
 wccnt.sort()  
 wmin = wccnt[0]  
 wmax = wccnt[len(wccnt) - 1]  
  
 *#친구에 저장된 정보를 저장* fcnt=[]  
 **for** tmp **in** fdic:  
 tmpn=len(fdic[tmp])  
 fcnt = fcnt +[tmpn]  
 fcnt.sort()  
 fmin=fcnt[0]  
 fmax=fcnt[len(fcnt)-1]  
  
  
 i=input()  
 **if** i==**'0'**:  
 print(**'Total users:%s'**%l)  
 print(**'Total friendship records:%s'**%k)  
 print(**'Total tweets:%s\n'**%j)  
 **elif** i==**'1'**:  
 print(**'Average number of friends:%f'**%(k/l))  
 print(**'Minimum friends:%s'**%fmin)  
 print(**'Maximum number of friends:%s\n'**%fmax)  
 print(**'Average tweets per user:%f'**%(j/l))  
 print(**'Minium tweets per user:%s'**%wmin)  
 print(**'Maximum tweets per user:%s\n'**%wmax)  
 **elif** i==**'2'**:  
 print(**'Top 5 most tweeted words is'**)  
 **for** i **in** range (1,6):  
 wmax1 = wcnt[len(wcnt) - i]  
 tmpn1=wmax1  
 print(wc2[tmpn1])  
 **elif** i==**'3'**:  
 tmp3 = []  
 tmpout = []  
 print(**'Top 5 most tweeted users is\n'**)  
 **for** i **in** range(1, 6):  
 wmax1 = wccnt[len(wccnt) - i]  
 tmpn1 = wmax1  
 **for** tmp **in** wc3[tmpn1]:  
 **if** len(tmpout) < 5:  
 tmpout=tmpout+[tmp]  
 tmp3 = tmp3 + [tmp]  
 **for** tmp **in** tmpout:  
 print(**'%s'**%a[tmp])  
 print(**''**)  
 **elif** i==**'4'**:  
 tmpword=input()  
 **if** tmpword **in** wdic:  
 print(**'Users who tweeted a word is'**)  
 **for** tmp **in** wdic[tmpword]:  
 print(**'%s'**%a[tmp])  
 tmp4=wdic[tmpword]  
 print(**''**)  
 **else**:  
 print(**'Users who tweeted %s not exist'** % tmpword)  
 **elif** i==**'5'**:  
 tmplist=[]  
 tmpout=[]  
 tmplist=tmp4+tmp3  
 **for** tmp **in** tmplist:  
 tmpout=tmpout+fdic[tmp]  
 print(**'People who is friend of the above user are'**)  
 **for** tmp **in** tmpout:  
 print(**'%s'**%a[tmp])  
 **elif** i==**'6'**:  
 tmpword=input()  
 **if** tmpword **in** wdic:  
 tmp=wdic[tmpword]  
 j=j-len(tmp)  
 **for** tmpw **in** tmp:  
 wcdic[tmpw].remove(tmpword)  
 print(**'Delete all mentions of a word %s complete.\n'**%tmpword)  
 **else**:  
 print(**'%s not exist'**%tmpword)  
 **elif** i==**'7'**:  
 tmpword = input()  
 tmp=wdic[tmpword]  
 tmpl=[]  
 **for** item **in** tmp:  
 **if** item **in** wcdic:  
 tmpl = tmpl + wcdic[item]  
 j=j-len(wcdic[item])  
 **del** wcdic[item]  
 tmpl2=list(set(tmpl))  
 **for** tmpw **in** tmpl2:  
 **for** tmpu **in** tmp:  
 **if** tmpu **in** wdic[tmpw]:  
 wdic[tmpw].remove(tmpu)  
 tmpl=[]  
 **for** tmpu **in** tmp:  
 tmpl=tmpl+fdic[tmpu]  
 k=k-len(fdic[tmpu])  
 tmpl2 = list(set(tmpl))  
 **for** tmpu **in** tmpl2:  
 **for** tmpu1 **in** tmp:  
 **if** tmpu1 **in** fdic[tmpu]:  
 fdic[tmpu].remove(tmpu1)  
 k=k-1  
 print(**'Delete all users who mentioned a word %s complete.\n'**%tmpword)  
 **elif** i==**'8'**:  
 print(**'Strongly connected component is\n'**)  
 **elif** i==**'9'**:  
 print(**'Shortest path is\n'**)  
 **elif** i==**'99'**:  
 **break**main()

1. Manual

0. Read data files

1. display statistics

2. Top 5 most tweeted words

3. Top 5 most tweeted users

4. Find all users who tweeted a word (e.g, 연세대)

5. Find all people who are friends of the above users

6. Delete all mentions of a word

7. Delete all users who mentioned a word

8. Find strongly connected components

9. Find shortest path

99. Quit

1. Report

Data structure : 헤쉬 테이블 ( 딕셔너리 )

키값을 통한 빠른 자료 검색과 정렬이 가능

또한 딕셔너리 안에 list를 넣음으로써 좀 더 용이한 자료 활용이 가능헀습니다.

(레드 블랙 트리를 이용하려 했으나 오류와 버그가 난무하여 황급히 바꿨습니다. ㅠㅠㅠ)

Expected perdormance:

딕셔너리를 이용했으므로, 거의 모든 탐색 과정은 O(n) 의 시간이 걸릴 것으로 수료되고,

삭제 과정에서 시간이 좀 걸리겠지만 O(nlogn)일 것으로 사료됩니다 .

Improve in the future

다른 자료구조형을 이용하여 8,9번도 가능한 코드로 만들겠습니다….

그래프 특성을 이용하므로 아마 링크드 리스트를 이용할 듯 합니다 .

Self evaluation form

• Submit a github account (10) : https://github.com/hwanhuh/hwhwhwan

• Commit source code displaying menu (10) 10

• Commit the first draft of manual (10) 10

• Read data files (20) • Statistics (20) 20/20

• Top 5 most tweeted words (10) 10/10

• Top 5 most tweeted users (5) 5/5

• Find all users who mentioned a word (10) 10/10

• Find all users who are friend of the above user (5) 5/5

• Top 5 strongly connected components (10) 0/10

• Find shortest path from a user (id) (10) 0/10

한 학기 동안 수고하셨습니다. 감사합니다