A. general overview

This program is designed to search the records according to the input, we have various ways to search. only need the user to input the keywords, the user can combine any number of keywords to continue the search.

B. detailed design

As for this program we divided into several functions, the interface is inside the main function to provide an interface for the user to use, it will prompt the users to input valid query according to the language grammar, the query part contains three functions: date function which retrieves the data by date and querybyterm function which retrieves the data by subject and body content and querybyemail function which retrieves the data by subject and body content. besides, there are two functions called evaluate and printfinalresult which takes the rowid and find the corresponding records. lastly, there are two strip utility functions which helps to eliminate the whitespace from the user input.

C.testing strategy:

we tested our program via multiple test cases strictly according to the question descriptions, first we ran the 1k.xml using phase 1 and use phase2 to produce a correct index file for later use and then input through the interface created by phase3 under the different queries according to the language grammar. Secondly, we also use plenty of ‘print’s in different places to test if it works properly along the way. Thirdly, we check the contents of the XML file compared with the output we got to see if relevant information is retrieved correctly and if the small tricky points have been performed successively.

There are some scenarios that we used to check. For instance, For the email query, when we input a single email query for

example,"*from:phillip.allen@enron.com*", the result must be all the records that have *from:phillip.allen@enron.com* in their "from" field and we also try to input multiple email query such as *from:phillip.allen@enron.com to:phillip.allen@enron.com* to check if the result row satisfy the requirement of both queries.

D.Group work strategy:

We breakdown this large project by assigning three phases（include functions handling the different types of query） to three of us, and work together whenever there are some bugs in the program, and integrate all of them together in the end.

Yanlin Li: I designed the phase2 and the query by terms function. I spent several (3 to 4) nights to deal with my own part and searched for some relevant information from the network. Indeed, I learned a lot from this mini-project 2. For instance, I learnt some extent of usages of berkley_db and several new methods in python when I want to deal with the database related problems. My strategy of work with partners also improved, and I realized how to distribute the whole work efficiently and how to communicate well with team members to keep a proper pace as a group.

Yuzhen Gao: I designed the phase 3 date function and fixed the main and evaluate the function, I spend 2 days on finishing that , i used stiptime to compare the date user input with the date stored in the records

Shiyu Xiu: I designed the phase 1 and query by email function, The time I spent on it is around two to three days. in the process of making such an interesting project with my teammates, I improved my skills of manipulating the berkley_db integrating with Python and had really good hands-on experience on how to build a real project that might benefit my future, I also improved the communication skill along the way via several meeting and solving bugs with my teammates face to face.

F.algorithm for efficiently evaluating queries:

there are few parts we did to improve the efficiency of our program. for example, we use the set() method for the exact term search to avoid redundant iteration. For multiple queries, we use the algorithm to find the row_id for each query and then find the intersection of them, and use this intersection list to find records in the re.idx.in the date part , since there is a situation for user to input a date which is not existed in the file , in this case, we used striptime to make the date are able to be compared , what is more, since the date in file is sorted, we just need to find the first one who is meet the situation then we will get all the dates which are met, according to weather user input = or not