MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG315 Algorithms

Assignment 1: Workplace Emails' Stats

This assignment aims to help you utilize the heap sort and binary search algorithms in a practical manner. Your main task in this assignment is to organize emails sent and received in a workplace **using C programming language**.

Overview

A workplace manager wants to study his employees' activities based on their exchange of emails. Henceforth, your task will be to write a program that will get specific stats that the manager wants to find out. Heap sort and binary search are prominent algorithms which you have encountered in this course as well as previous courses, and you will be using them to fetch the stats the manager requests.

In this assignment, your program shall read emails from a group of text files and operate on them with the following functionalities:

1. Read a group of data files, each containing an email ID, a sender, a recipient, a date, and the content
2. Store the data in an array of email structures
3. Print a menu displaying the stats options that could be fetched
4. Sort emails based on id, number of words, or day
5. Search for emails by ID
6. Print the results with the respective information

Input

The program will two inputs:

- Data files with each one storing an email. Each data file will be structured as following:

  Email_ID
  From: Sender
  To: Recipient
  Date: Day_of_the_month
  Content

  - Email_ID: An integer value
  - Sender: A string value with maximum 50 characters
  - Receipt: A string value with maximum 50 characters
  - Day_of_the_month: For the sake of simplicity, all emails are assumed to be sent in the same month and year, therefore the date will only be an integer with range [1, 31].
  - Content: A string value – Please note that we are interested in the number of words in the content rather than its actual content.

- Manager input which will be a command that decides what to be displayed next.

Internal Processing

As the program reads the data about each email, it will create a corresponding data structure for each one as it reads the email's id, sender, recipient, date, and counts the number of words. Feel free to add attributes to the structure. The program will keep these structures in an array as illustrated below:

Id: 13 Sender: Molly Recipient: Ash Date: 21 Words: 32	Id: 16 Sender: Chris Recipient: Perla Date: 25 Words: 21	Id: 41 Sender: Perla Recipient: Melina Date: 16 Words: 9	Id: 7 Sender: Ash Recipient: Melina Date: 26 Words: 20	Id: 22 Sender: Melina Recipient: Carla Date: 5 Words: 10	Id: 10 Sender: Carla Recipient: Chris Date: 13 Words: 80

You need to implement the following functions and use them in your main program where appropriate.

- **readEmails:** This function takes the path to the directory which contains the data files, and the number of data files. It then reads the files that are named according to a number. For example, if there are 50 data files, then the files will be named "1.txt", "2.txt", all the way to "50.txt". The function creates an array of email structures and reads the data of each file into one of those structures. Finally, the function returns the emails array. **Be sure to add a few error checks to this function, such as when the user enters a path that does not exist, or a number of emails that is more than the one found in the entered directory.**
- **menu:** This function prints a menu in a loop which shows the manager the options he can choose.
- **heapSort:** This function takes the emails array, the size of the array, and the sorting criteria which is either 1: id, 2: number of words, or 3: date, and then returns the sorted Emails array. The function firstly applies the Build Heap algorithm to create a max-heap and then sorts the data based on the heap sort algorithm.
- **binSearch**: This function takes the emails array, the size of the array, and the search key (id). Binary search requires an array to be sorted, henceforth, you will first call the **heapSort** function you wrote, and then apply binary search.
- **printEmails**: This function takes the emails array and the size of the array to print the array.

Algorithm Analysis

- For each function mentioned above, you need to add a comment to your code to specify the time complexity in Big-O and explain the reason with your own words.

Output

The program will print the sorted email lists according to the manager's criteria or display the results of the search. An example output is shown below. **Please note that in the implementation, there are 50 emails in the data (check ODTUCLASS for the data files) that you need to process and sort, the following is just an example to show you how to display the output.**

```
Enter the path of the directory which contains data files: datafiles
Enter the number of data files: 6

6 emails have been read successfully!

Please choose one of the following options:
   (1)   Display emails sorted by id
   (2)   Display emails sorted by number of words
   (3)   Display emails sorted by date
   (4)   Search email by ID
   (5)   Exit

Command: 3
```

```
Id: 22
Sender: Melina
Recipient: Carla
Date: 5
Words: 10

Id: 10
Sender: Carla
Recipient: Chris
Date: 13
Words: 80

Id: 41
Sender: Perla
Recipient: Melina
Date: 16
Words: 9

Id: 13
Sender: Molly
Recipient: Ash
Date: 21
Words: 32

Id: 16
Sender: Chris
Recipient: Perla
Date: 25
Words: 21

Id: 7
Sender: Ash
Recipient: Melina
Date: 26
Words: 20

Please choose one of the following options:
   (1)   Display emails sorted by id
   (2)   Display emails sorted by number of words
   (3)   Display emails sorted by date
   (4)   Search email by ID
   (5)   Exit

Command: 4
Enter the search key: 16

Id: 16
Sender: Chris
Recipient: Perla
Date: 25
Words: 21


Please choose one of the following options:
```

```
   (1)   Display emails sorted by date.
   (2)   Display emails sorted by id.
   (3)   Search emails on a specific date.
   (4)   Search email by ID.
   (5)   Exit
Command: 5
Goodbye!
```

Incremental and Modular Development

You are expected to follow an incremental development approach. Each time you complete a function or module, you are expected to test it to make sure that it works properly. **You are also expected to follow modular programming which means that you are expected to divide your program into a set of meaningful functions.**

Professionalism and Ethics

You are expected to complete the assignments on your own. Sharing your work with others, uploading the assignment to online websites to seek solutions, and/or presenting someone else's work as your own work will be considered as cheating.

Rules

- You need to write your program by using C programming.
- You need to name your file with your student id, e.g. 1234567.c, and submit it to ODTUCLASS.
- Code quality, modularity, efficiency, maintainability, and appropriate comments will be part of the grading.
- **You need follow the function prototype specifications mentioned above, but feel free to add parameters to pass to the functions.**
- **You can add extra helper functions.**

Grading Scheme

Grading Item	**Mark (out of 100)**
Email Data Structure	5
Main Function	15
readEmails	30
menu	5
heapSort	25
binSearch	15
printEmails	5