This is a live document. Please keep checking it for updates.

Mini Google

There are 2 parts to the search engine: Crawl Web and Search. Each of these parts also called as “software components” will be described below.

The crawl web software component is responsible for fetching web pages from the World Wide Web and adding each web page to a data structure called Search Index. You can read more about the Crawler. <https://en.wikipedia.org/wiki/Web_crawler>

In this project we begin building a bare bones web crawler. This means it won’t have all the features of an industry scale web crawler but will have enough for us to understand how to program one. All of the crawl web features can be implemented as a set of java classes which are described below. Consider placing all these java classes in a package. Packages are not very difficult to understand. Read about how to use packages <https://docs.oracle.com/javase/tutorial/java/package/packages.html>

**The WebPage class**

The WebPage class holds the web page content and it has the following methods.

* getAllLinks() - returns an ArrayList of links in the webpage.
* getWords() - returns an ArrayList of words in the page, after removing the stopwords.
* getContent() - fetches the web page content identified by its URL. Use the example from <https://docs.oracle.com/javase/tutorial/networking/urls/readingURL.html> to fetch the web page content from the Internet. In the C project we have given you a web page in the folder. In Java project you can make you of the Java API to directly fetch a real web page from the Internet
* getKeywordFrequency() - given a keyword as input returns the number of times the keyword appears in the web page.

**The WebCrawler Class**

The WebCrawler class has the functions must have the following methods.

* A static method Crawl with one parameter seed page. Seed page is the first page for the web crawler to process. The goal is to start with the seed page, extract all the links from it and repeat crawling for each link that is extracted, until all the web pages connected to the seed page are fetched. The data type for the seed page can be a String.

**The SearchIndex Class**

The SearchIndex class as the name suggests represents the search index. SearchIndex is a data structure that holds keywords with a list of WebPages with at least one occurrence of the keyword in the page. The Hashtable class in the Java API is very useful to implement the search index. <http://docs.oracle.com/javase/7/docs/api/java/util/Hashtable.html>  The Hashtable class provides methods to add a keyword and web page to it and to look up for the web pages by providing the keyword as a parameter. The SearchIndex should be save to a file when a web page is added to it. On creating the object of the Search Index its data should be retrieved from a file.

Hotel Reservations

Your task is to implement a simple hotel reservation system. This system will allow customers to request any available room, request a specific room, and cancel reservations. Since the staff is optimistic about future economic growth the reservation system also has to accommodate the growth of the hotel. In other words, the proprietor should be able to add additional rooms without disturbing existing reservations.

**The *Reservation* class**

The *Reservation* class is a simple class that models customer reservations, including the customer's name and the room number. As you might expect, the *Reservation* class provides some simple methods for maintaining an individual reservation.

Please consult the comments in the class skeleton for the details of each method's parameters and behaviors. Please also note that you may or may not actually use all of the methods of this class in your implementation of the reservation system, but should implement and test them, anyway.

**The *Hotel* class**

The *Hotel* class models the hotel as a collection of rooms, each of which is represented as a *Reservation*, either actual, or *null*.

To maintain the *Reservation*s associated with each room, the *Hotel* class will contain a *ArrayList*. If you don't know how to use *ArrayLists*, please don't worry. They are a very simple indexed collection, much like an *Array*. They have a much richer collection of behaviors. Among their additional capabilities is their ability to grow *on the fly*.

Be sure to familiarize yourself with the *ArrayList* API before beginning. Please pay close attention to the difference between the size() and ensureCapacity() methods.

You will need to complete the following methods of the *Hotel* class. For more detail, please see the comments within the skeleton.

* public int reserveRoom(String person)
* This method should instantiate a Reservation and insert it into a empty room. This function should then return the room number where the room number of the Reservation and the index of the ArrayList in which the Reservation is placed match. In the case where the hotel is full, you should return -1.
* public boolean reserveRoom(String person, int roomNum)
* This method reserves the specified room for the customer if possible. It returns true on success, or false if the room could not be reserved.
* public void cancelReservations(String person)
* This method cancels all the reservations for the specified customer.
* public void printReservations()
* This method prints out all current reservations. Also, you should print how many reservations and free rooms there are.

**The *Simulation* Class**

This class contains the main() method for the simulation. It creates a new *Hotel* and provides a menu system and work loop that yield a very simple reservation system. If you give careful thought to your test cases, this system will allow you to test your implementation of the other classes.