


Spring Quarter Course Project Description

To-Do Date: May 30 at 11:59pm

Course Project Information

Project Overview


As a team of 5-6 programmers, you will assist in the implementation of a data processing system that includes its own search engine. **You must sign up to become part of a team and select a topic on [this sign up](https://docs.google.com/spreadsheets/d/1zPf9K2Cb4UEd5VxMkHv_DjhYhFPth0RZ9aqCb8jnerw/edit?gid=0#gid=0)  (https://docs.google.com/spreadsheets/d/1zPf9K2Cb4UEd5VxMkHv_DjhYhFPth0RZ9aqCb8jnerw/edit?gid=0#gid=0) sheet by Monday, June 2 at 11:59pm.**

Each member of the team is to code one or more functional areas of the project culminating in a video presentation at the end of the quarter. **The project source code, presentation slides, and presentation video will be due on Tuesday, June 21 at 11:59pm.**

Data and Topic Requirements

Your team must select one of the below four topics as the focus of your project. Under special circumstances, and with the permission of your instructor, you may select a different topic:

- Songs
- Poetry
- Blog posts
- Recipes

Within your topic, you will need to select a focus or **genre**  (<https://www.merriam-webster.com/dictionary/genre>). Some examples of genres are as follows:

- Songs: Rap, or 1940s hits, or Pink Floyd songs, etc
- Poetry: Haiku, or Love Poems, or Elizabethan poetry, etc...

- Blog posts: Travel blogs, or tech blogs, or cooking blogs, etc.
- Recipes: French cooking, or vegan cooking, or desserts, etc...

Note that no other team should have your topic AND genre.

Please Post Your Course Project Topic on the Sign Up Sheet to Reserve Your Chosen Topic. Topics May be Changed Later in the Quarter by Notifying the Instructor.

Once you have selected a topic and genre, your team will be responsible for finding a collection of data that falls within the specifications of your topic. The data collection is of your own choosing, with the requirement that each record in the system must contain a unique key (must be a string), and at least three non-key fields, one of which must be the full text of the entry for that particular record. For instance, your team might select a collection of poems with the following attributes:

Title + Author– unique key
Year
Country of origin
Full Text of poem

Your collection must have a minimum of 15 records with which to seed your project. Users of your system must be able to upload additional records contained in additional files.

Once you have gathered your collection of data, you will be required to create a class whose fields are the attributes of this data set. For example:

```
public class Poem {  
    private String title;  
    private String author;  
    private int year;  
    private String genre;  
    private String text;  
}
```

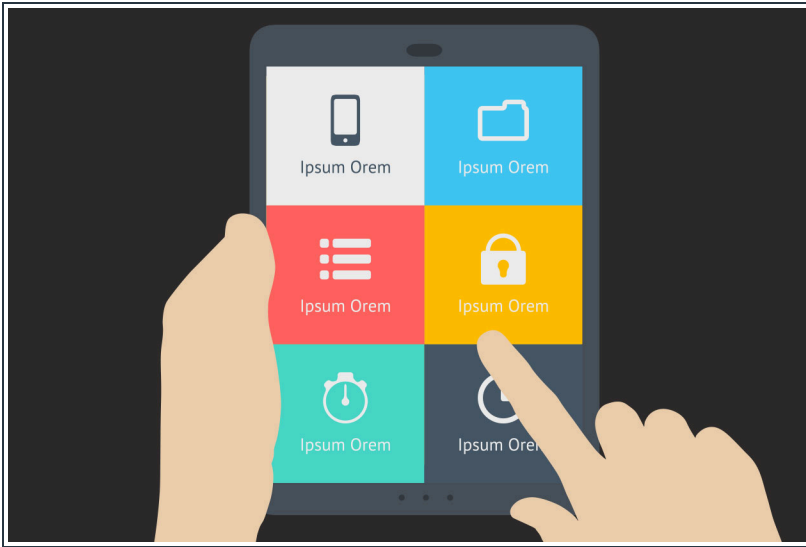
Each record within your data set should be an instance of this class.

Menu Requirements

The goal of this project is to create an interactive data processing system, with a menu of options provided to the user. Your project ***must*** display the following options for the user:

- Upload a new record
- Delete a record
- Search for a record (has a sub-menu)
 - Find and display one record using the primary key
 - Find and display records using keywords (your search engine)
- Modify or update a record
- Statistics (When this option is selected, your program must display at least 3 different statistics of your choice about the data)
- Quit
 - Write all records to a file of user choice and exit the program

At the end of the program, the data is to be automatically written to a text file whose name is typed in by the user.



(<https://sites.google.com/site/deanzacollegecis/cis22c/courseproject2/web-usability.jpg?attredirects=0>)

image source  (https://web.archive.org/web/20160501200748if_/http://webusabilitytalk.com/wp-content/uploads/2014/02/web-usability.jpg)

Required Data Structures and Additional Project Specifications

The system's data structure must contain a Hash Table of Objects of your class, sorted according to the primary key.


The system must also contain an ArrayList of BSTs and an additional Hash Table, both of which will be used for the purposes of the search engine:

- The ArrayList of BSTs will be used to store the inverted index (see discussion below) sorted by the title field. See the below section on the search engine for more information.

- The second Hash Table will be used to store a one-to-one mapping of words to numerical ids (see discussion below).



(<https://sites.google.com/site/deanzacollegecis/cis22c/courseproject2/Search%20Engines.png?attredirects=0>)

image source  (https://web.archive.org/web/20250506143032if_/https://3.bp.blogspot.com/-rUZMrrfOryc/Vi3yu_y4HWI/AAAAAAAAApU/xswpEk8UTEM/s1600/Search%2BEngines.png)

Search Engine Requirements

The heart of this project is to create a search engine. The purpose of your search engine is to allow users to search via keyword for entries in your system. For example, if the user types in the word "love" to the search engine, the title of each record where the word "love" appears in the text should be displayed to the user. The user should then be able to select the entry of his or her choice to view additional information.

For example, if the given topic is Pink Floyd Songs and the user types in the word "time" to your search engine, your system should display an **alphabetically sorted** list of song titles. The screen might display an output like the following:

The following songs contain the word "time":

1. Breathe
2. Money
3. The Great Gig in the Sky
4. Time

To view more information about any of these songs, enter the number next to the song title:

The search engine will build what is known as an ***inverted index*** - a well-known technique from a subfield of computer science known as **information retrieval** [↗\(https://en.wikipedia.org/wiki/Information_retrieval\)](https://en.wikipedia.org/wiki/Information_retrieval). You will doubtless hear more about concept of the inverted index as you continue your study of computer science. In fact, most modern search engines, including Google, build some form of inverted index in order to display results to a user.

What is an index? If you flip to the back of any textbook, you will see an example of an index, where the keywords in the textbook are mapped to page numbers where you can find those words.

What is an inverted index? According to Wikipedia: "In computer science, an **inverted index** ... is an **index data structure** [↗\(https://en.wikipedia.org/wiki/Index_%28database%29\)](https://en.wikipedia.org/wiki/Index_%28database%29) storing a mapping from content, such as words or numbers, to its locations in a **database file** [↗\(https://en.wikipedia.org/wiki/Table_%28database%29\)](https://en.wikipedia.org/wiki/Table_%28database%29), or in a document or a set of documents (named in contrast to a **Forward Index** [↗\(https://en.wikipedia.org/wiki/Forward_Index\)](https://en.wikipedia.org/wiki/Forward_Index), which maps from documents to content)."

In the example above, your textbook provides a forward index in the back of the book, where topics are mapped to page numbers. In contrast, we will be creating an inverted index for our search engine by mapping words to Objects (Songs, Poems, BlogPosts, or Recipes depending on your topic). In the example above using Pink Floyd, the word "time" was mapped to four Songs.

Building an inverted index like this one allows for greater speed when searching for an entry: Once the index has been built, the search engine does not have to scan through all of the text entries looking for matches each time a user makes a **query** [↗\(http://www.merriam-webster.com/dictionary/query\)](http://www.merriam-webster.com/dictionary/query).

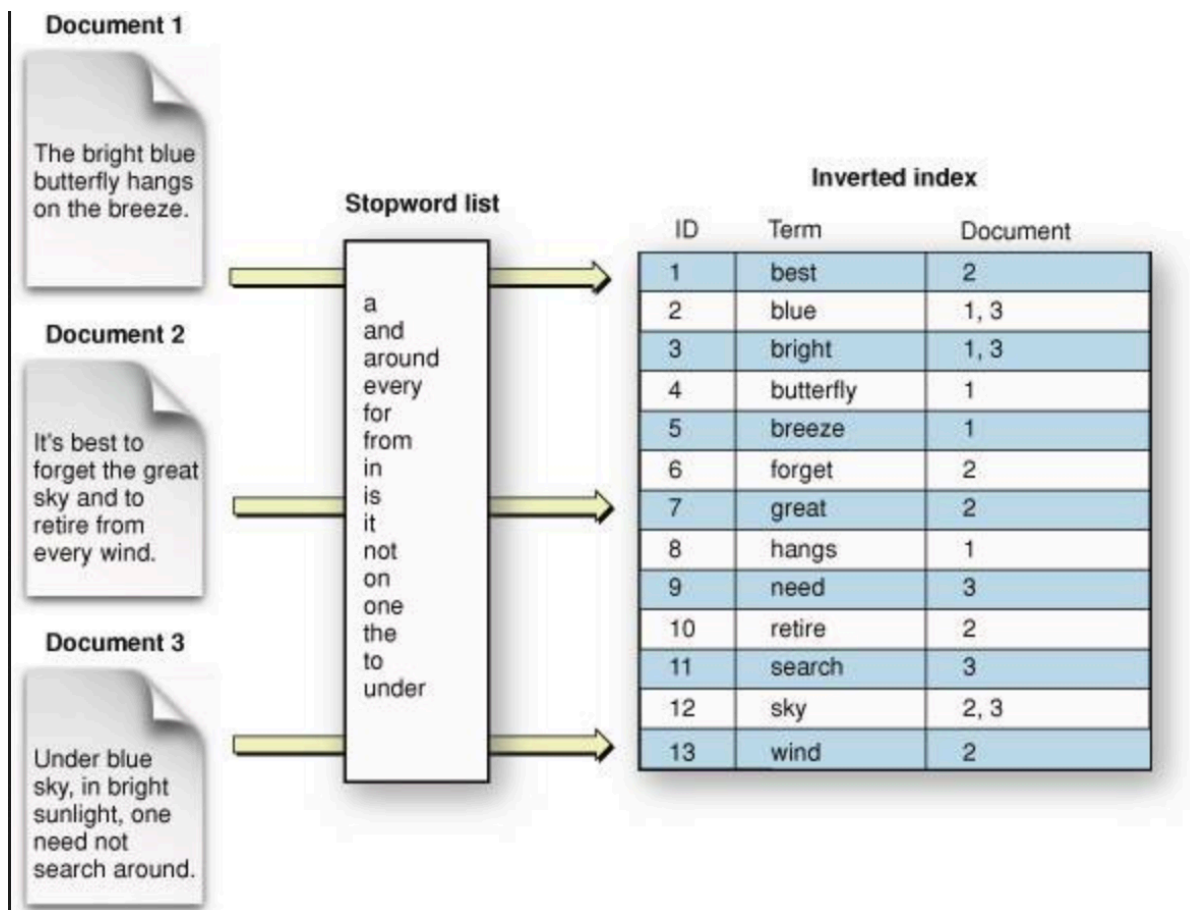


image source https://4.bp.blogspot.com/-xOLgE5ZxTC8/TI3WDq_i7UI/AAAAAAAAAB8/7sA8ptfATIU/s1600/InvertedIndex.jpg

Your search engine is **required** to follow these steps:

Step 1: Continuing with our example, the lyrics of the songs Breathe, Time, and Money, as well as the other Pink Floyd songs in the collection, should each be placed in text file or individual text files (your choice). You will carefully structure each entry. For example, perhaps the first line will always have the name of the song, the second line will always be the artist, the third line will contain the year and the rest of the lines will contain the lyrics.

Step 2: Remove any words that would not be considered useful, such as any articles (a, the), prepositions (of, to), conjunctions (and, but, nor) and common adjectives such as some, any, very. It is up to you which words you remove from your collection. As part of your video demonstration, you should test your search engine by using some common

words such in the examples above to see demonstrate you have removed them. You should remove a minimum of 20 non-useful words from these files.

Step 3: Create an ArrayList of BSTs (i.e. each index stores one BST), where the *index* should represent a *unique word* in your files. The basic idea is as follows.

For ALL the unique words 1 through n contained in your data files:

```
ArrayList[word1]
ArrayList[word2]
ArrayList[word3]
...
ArrayList["love"]
....
ArrayList["time"]
...
ArrayList[wordn]
...
```

Thus, each unique word contained in one or more of your files, should be one index of your ArrayList. For more information, see the below discussion about using Strings as array indices.

Step 4: Go through your files word-by-word, and, for each word, insert the Object (the song, poem, recipe, or post object) into the BST at that word's index in the ArrayList. Each chain should be a binary search tree.

```
ArrayList[word1]-> bst containing Song4 and Song6
ArrayList[word2]-> bst containing Song2, Song4, and Song5
ArrayList[word3]-> bst containing Song1
...
ArrayList["love"]-> bst containing "Love Scene (Version 6)"
....
ArrayList["time"] -> bst "Breathe", "Money", "The Great Gig in the Sky", and "Time"
...
ArrayList[word1000]-> bst containing Song1, Song6, Song8, Song24, etc.
```


...

Important: You should not store duplicate titles within one bucket even if the same word appears more than once inside of the same text. Therefore, you may wish to rewrite your BST class to disallow duplicates in insertion.

More Information About How to Create an Index from a Word

Part of the complexity of this project will be to decide how to use a **word** as an **index of an array**, which is not possible to do directly in Java, as Strings cannot be used as the indices of arrays in this language. My suggestion is to assign each word an integer id, and then use the id as the index your ArrayList. You will, therefore, assign an ID to each word from 0 to n-1, where the id is also the index in the ArrayList.

Using this approach, you will also need a way to store the information about which word has been assigned which integer id so that you do not accidentally assign the same word two different ids. It is important to keep in mind that each time you retrieve a word from one of your files, you will need to determine first whether it has been assigned an id yet, and then proceed accordingly. Thus, you will need to use an **additional hash table** to store the information about which id has been assigned to which word.

I strongly encourage you to make use of your knowledge of classes and objects here. You should consider creating a class like the following to bundle words and their ids together:

```
public class WordID {
    private String word;
    private int id;

    //constructor(s), getters and setters
}
```

Then, each time you assign an id to a word, you can create a new instance of the WordID class and store this WordID Object in your hash table. The primary key of this hash table should be the word.

This project is intended to be challenging. If the ideas contained in this section seem confusing, I recommend that you do more research into the concepts behind an "inverted index" and then come to your professor's office hours with any questions.

Additional Resources

- A good definition of inverted index can be found here ➞ (<https://www.igi-global.com/dictionary/inverted-index/15654>)
- A conceptual description of an inverted index, with examples can be found here ➞ (<https://www.elastic.co/guide/en/elasticsearch/guide/current/inverted-index.html>)
- You can find out more about indexing by watching the video series starting here ➞ (https://www.youtube.com/watch?v=QA_vuzx9mt4)

Indexing 1: what makes google fast



[Minimize Video](#)

Expectations for Team Members

In addition to writing and debugging code, each member of the team will:

1. Give suggestions for the application data: think about original/interesting/educational data that matches the program requirements (you may not use the Book example or the other example topics listed above – come up with something else).
2. Participate in designing a solution to the problem: UML Diagrams, sketches, pseudocode, etc
3. Provide relevant test cases for the walk-throughs and final presentation.
4. Individually test his/her part of the program (as much as possible). This implies writing test files (code that will not be included in the final project, but it will be used to test parts of the project).
5. Participate in all of the team meetings. Come prepared to the meetings. If a team member must be absent from a meeting, he/she should inform the other team members in advance; also he/she could keep in touch by phone or email.
6. Submit the project on the scheduled date.
7. Work with the team coordinator and other members in the team to integrate his/her part of the code.
8. Prepare for and participate during his/her team presentation.
9. Actively attend all of the presentation sessions. (Ask questions/provide answers).



image source  ([https://2.bp.blogspot.com/-](https://2.bp.blogspot.com/-ye0saPfdkkY/UxdHRWUUn2I/AAAAAAAAAOsU/Q4NvfycemyE/s1600/TEAM++.bmp)

[ye0saPfdkkY/UxdHRWUUn2I/AAAAAAAAAOsU/Q4NvfycemyE/s1600/TEAM++.bmp](https://2.bp.blogspot.com/-ye0saPfdkkY/UxdHRWUUn2I/AAAAAAAAAOsU/Q4NvfycemyE/s1600/TEAM++.bmp))

Suggested Team Assignments

Each member of the team must have a predefined role on the team. If the team is 5 members or smaller, some teammates will take on more than one role.

Team Member 1: Pro Team Coordinator: *Data structure design coordination, write main() class, Integration, Testing, keeping everyone on track, interfacing with the instructor about any issues that arise.*

Team Member 2: HashTable Specialist: *Implement HashTable Insert, Delete, Search by Primary Key.*

Team Member 3: Search Engine Whiz 1: *Create the ArrayList of BSTs for the Search Engine and the HashTable of WordID Objects.*

Team Member 4: Search Engine Whiz 2: *Create the ArrayList of BSTs for the Search Engine and the HashTable of WordID Objects.*

Team Member 5: Screen Output Expert: *Create user interface. Connect menu of options to the data structures. File I/O. Work with Team Coordinator to integrate individual data structures into a unified system.*

(Optional) Team Member 6: Statistician and Presentation Design Guru: *Write all code for statistics menu option, create presentation, manage presentation time, use checklist during presentation to ensure that all features demonstrated and team gets highest score possible.*

What to Do If Someone on Your Team is Not Doing His or Her Part

The main purpose of this project is to practice teamwork skills. If someone on the team is not doing his or her part, it is the responsibility of the team leader to talk to that person and encourage him/her to take the project more seriously. If this teammate is unresponsive, the next step is to meet with the instructor to explain the situation (please provide email or other evidence). The instructor will intervene and talk to the student and issue a written warning. Ultimately, however, if your teammate refuses to work, his/her portion of the project should be reassigned to another member(s) so that the team doesn't fall behind or fail. In this case, the other members of the team should include this student in the presentation. Any student who does not present will be given a 0 on the project.

What to Do If Your Teammates Are Excluding You From the Project

The main purpose of this project is to practice teamwork skills. If your teammates are excluding you from decision-making and team meetings, please contact me ASAP and provide email or other evidence. I will follow up with those students with a written warning. Any student who continues to demonstrate poor teamwork skills by excluding others following this warning will receive a 0 on the project.

Project Score

A team score will be calculated as shown below.

Completeness (60%). The project contains all of the required data structures and functionality.

Accuracy (20%). The system demonstration runs without errors. Also, the system loads and executes without errors in a customer (instructor) test.

Presentation (10%). An approximately 15 minute video presentation and demonstration of the project

Team Document (10%).

Note that I reserve the right to lower the score of any student who is not demonstrating teamwork skills. If you are not participating on your team or you are not excluding team members from participating, then I reserve the right to assign you a 0 on this project.

Required Project Specifications

The system's data structure is to contain a hashed table of **at least 15 records read from a file or multiple files.**

Collisions will be resolved using separate chaining within each bucket.

In addition to the hash table, build a search engine using an inverted index as described above.

Draw a UML Class Diagram to show **all** classes used in your project (including inner classes) and how they interrelate. All methods and variables from all classes should be represented. Please see section below on UML Class Diagrams.

Provide a menu of options for users as described earlier - note that you must offer the exact menu options as shown above, though wording can vary slightly. If in doubt, contact the instructor.

UML Class Diagrams

As part of your presentation, you will be required to show a UML class diagram representing your project. **All team members** should contribute to the creation of the UML class diagram.

- Please have one box for each class used in your project
- The box should be titled with the name of the class, bolded and centered in the upper compartment

- Variables should be defined in the middle compartment of the box and marked as private (-). They should be left justified.
- Methods should be defined in the bottom compartment of the box and marked as public (+) or private (-). They should be left justified.
- Connectors should be used between boxes to demonstrate the interrelationships among the classes.

Example Class Diagram:

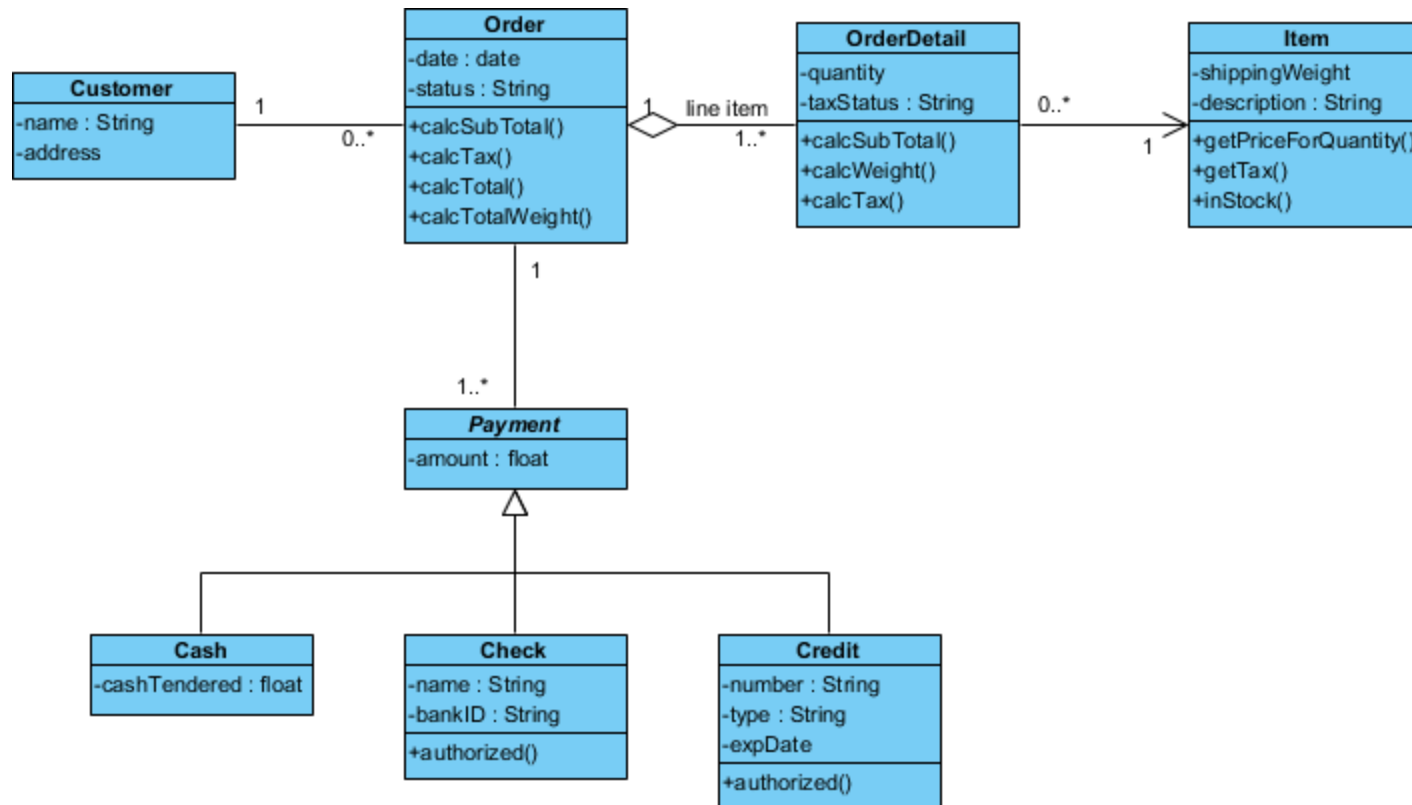
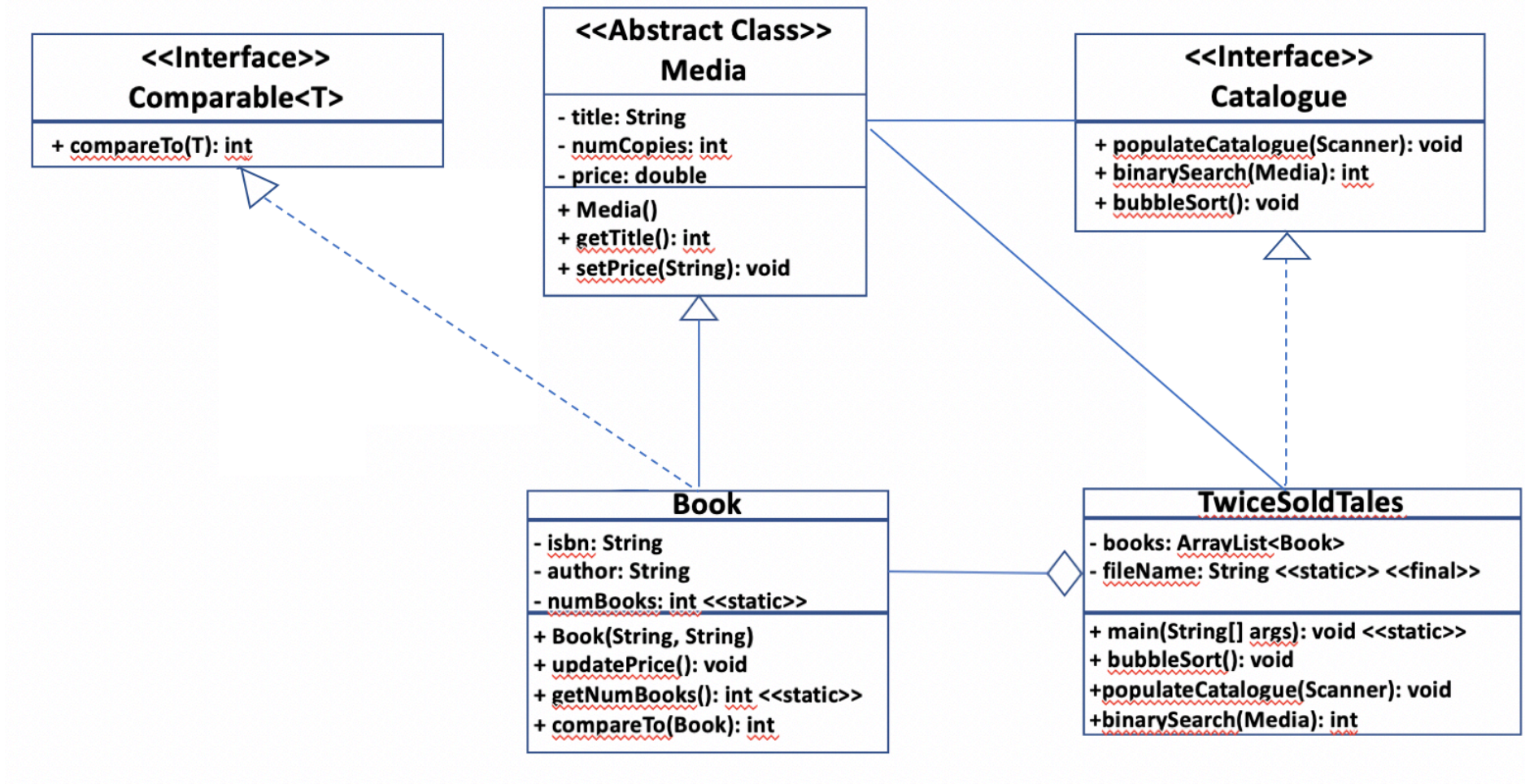


image source https://1.bp.blogspot.com/-62OFIjfR8mU/Ws8HyDCMzZI/AAAAAAAAA8/8YKQsksTSREYc-DE5KCzy_qa07NZ7MzYwCLcBGAs/s1600/class-diagram.png

A second example:



In the diagram above, classes are represented with boxes that contain three compartments:

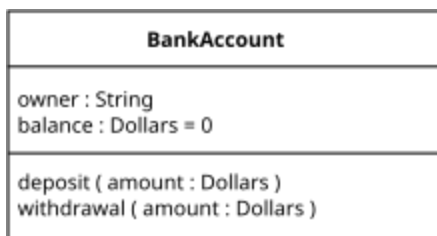


image source https://en.wikipedia.org/wiki/Class_diagram#/media/File:Uml_class_relation_arrows_en.svg.png

- The top compartment contains the name of the class. The name is displayed in bold and centered, and the first letter is capitalized (just like the name of a class should be in your project).
- The middle compartment contains the attributes (member variables) of the class. They are left-aligned and the first letter is lowercase.
- The bottom compartment contains the operations the class can execute (the member methods). They are also left-aligned and the first letter is lowercase.

There are different types of connectors between the boxes:

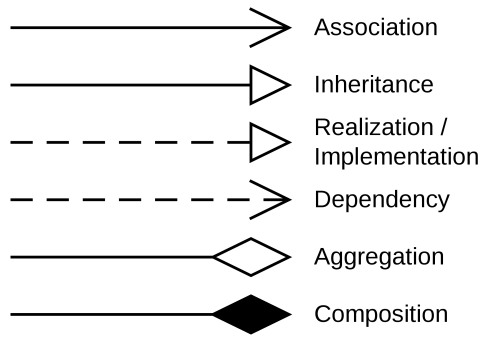


image source (https://en.wikipedia.org/wiki/Class_diagram#/media/File:Uml_class_relation_arrows_en.svg.png)

Visibility

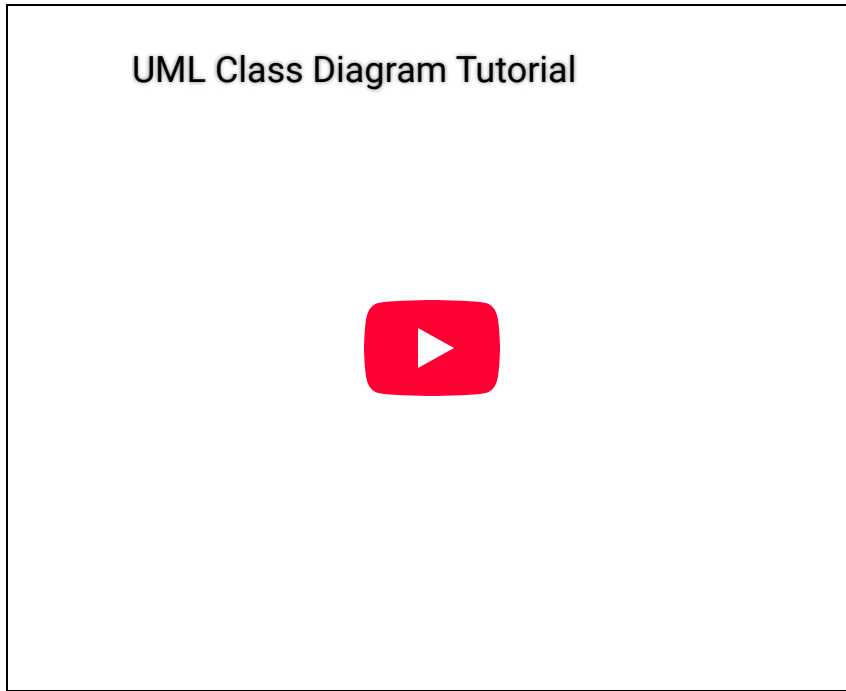
To specify the visibility of a class member (i.e. any variable or method), these notations must be placed before the member's name:

UML diagram visibility notation

- Public
- Private
- Protected
- Derived (can be combined with one of the others)
- Package

Source and More Information (https://en.wikipedia.org/wiki/Class_diagram)

Video  (<https://www.youtube.com/watch?v=UI6lqHOVHic>)



Minimize Video

(Ignore the advertisement in the middle - use Draw.io or Powerpoint instead)

Recommended creating UML diagrams: **Draw.io**  (<http://Draw.io>)

Walk-Throughs

Select 2 friends to test your project. Give these friends precise goals to accomplish, for example: “Scenario 1: Create a new account, log in, and search for a book.” List the three scenarios you asked your test participants to complete here.

Scenario 1:

Scenario 2:

Scenario 3:

List the feedback that you received from each of your participants.

Feedback from Participant 1:

Feedback from Participant 2:

List any problems you noted while the participants were testing your project or any changes you wanted to make after observing the tests:

Please present this information as part of the the project presentation (described below).

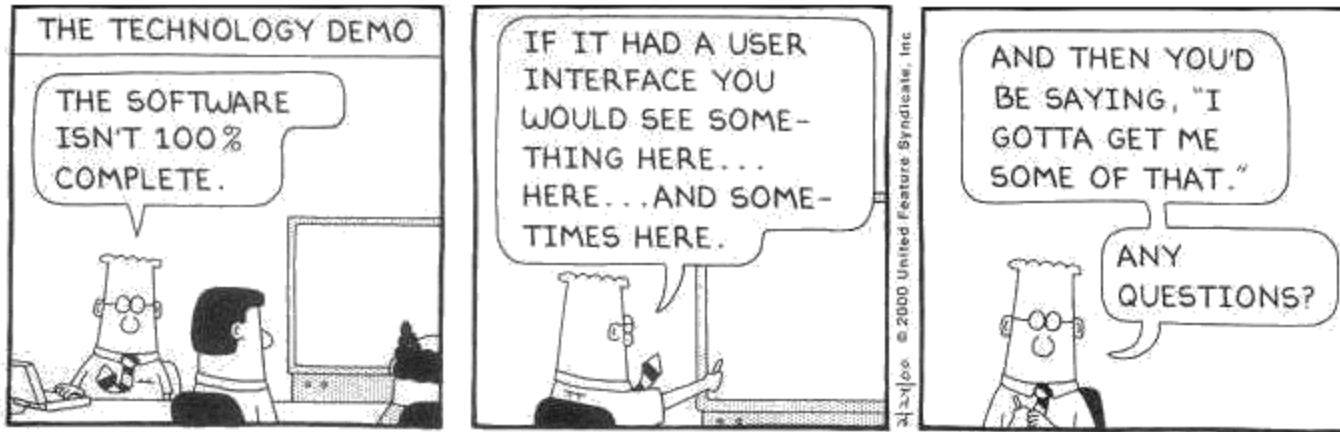


image source. http://class-editor.sourceforge.net/under_con_uuml.html

Presentation

At the end of the quarter, you will need to present the final version of your project to the instructor.

This presentation and demonstration will occur via video and should be ~15 minutes in length - no more than 20 maximum, no less than 10 minimum.

Suggested software for screen capture to record your video:

- **Camtasia** <https://www.google.com/url?q=https%3A%2F%2Fwww.techsmith.com%2Fvideo-editor.html&sa=D&sntz=1&usg=AFQjCNEjBtOEdJK5CIni2lfe8vkPkOrhUg> (free 30 day trial)
- QuickTime (comes standard with a Mac)
- **Zoom** <https://support.zoom.us/hc/en-us/articles/201362473-Local-Recording>

The video should have two parts:

1. A PowerPoint presentation (or similar) as described below
2. A demonstration of your project, where each team member shows the functionality of his or her portion of the project.

Each team member should speak as part of the presentation.

When discussing the UML diagrams, the teammate responsible for that portion of the project should explain the diagram

for his or her component.

Please show a continuous recording of the project demo.

Please do not edit out any component of the project or the results of entering a key.

- Please do not break the project into pieces and record each component separately.

As no extensions are allowed, make sure that you at least have a running version of your project when you record the video (even if it is incomplete) or you will receive a 0 on your project (no exceptions!).

Please read all directions below to maximize your presentation score.

Important: Your project will also be evaluated by the instructor during this video presentation. If a component is not included on the video, I will assume that it does not work or was not completed, and will not bother to test it. Please be careful to demonstrate all components of the project, as your grade will depend on the thoroughness of your presentation. Use the checklist below as you are recording to ensure that you demonstrate all aspects of the project:

-

- Upload a new record (10 pts)
- Delete a record (10 pts)
- Search for a record (has a sub-menu)
 - Find and display one record using the primary key (10 pts)
 - Find and display records using keywords (your search engine) (50 pts)
- Modify or update a record (10 pts)
- Statistics (When this option is selected, your program must display at least 3 different statistics of your choice about the data) (5 pts)
- Quit (5 pts)
 - Write all records to a file of user choice and exit the program

Total: 100pts

I will be using the above checklist as I am watching your video. Please do not skip any item on the checklist or you will receive no credit for that component of your project

The presentation itself will be graded on a 20 point scale as follows:

Presentation requirements

Requirement	Description	Point Value
Slide 1	<p>Your first slide must display your topic and the full names of all presenters.</p> <p>At this point in the presentation, each team member should introduce him or herself</p> <p>And, one person should state the topic of the presentation.</p>	1
Slide 2	<p>Your second slide must provide an outline of what you will cover in your presentation. This outline must be a list in either bullet point or numerical form.</p> <p>One member of the team should briefly go over what will be discussed during the presentation.</p>	1

<p>Slide 3</p>	<p>Your third slide must give an overview or brief summary of what your chosen topic is, by doing the following:</p> <ol style="list-style-type: none"> 1. defining your topic - what it is 2. explaining why you chose this topic 	<p>2</p>
<p>Slide 4</p>	<p>In the fourth slide, you will need to give an overview of your data set, including</p> <ol style="list-style-type: none"> 1. where you found it (give me a link to a website or websites) 2. what are the fields contained in the data set, e.g. title, author, isbn, year, price? 	<p>4</p>
<p>Slide 5-6 or 7</p>	<p>In the next 2-3 slides, give an overview of the data structures used in your presentation. You must present these by using the UML class diagrams that you created. You may also show me some of your code.</p> <p>Important: In this part of the presentation, you need to describe your search engine in detail.</p>	<p>4</p>

<p>Slide 7 or 8</p>	<p>In the next slide, list the results from your Walk-Throughs. You must present the information as described in the above section on Walk-Throughs, including the 3 scenarios you provided to your participants, the feedback of the two participants, any errors you noted while observing the participants testing your code, and any changes you made to your project as a result of the Walk-Throughs.</p>	<p>4</p>
<p>Live Demo of Project</p>	<p>During the live demo, you should display the main functionality of your system, including: Insertion, deletion, search, display, statistics, and writing to a file, as well as any special features you would like to show off.</p> <p>Important: You will earn the bulk of your project grade in this part of the presentation. Therefore I would recommend that one person keep track of which features have been demo'd - using the above checklist - while the presentation is happening. Don't end the live demo unless you are 100% certain you have shown off all project features.</p>	<p>3</p>

Final Slide	In the final slide, thank the audience	1
Presentation Aesthetics	Is your presentation laid out nicely with lots of images and clear, readable text?	1
Professionalism	Is the presentation delivered in a practiced and professional manner?	1