# San José State University
# College of Science / Department of Computer Science
# CS 146 Data Structures and Algorithms
# Section 4/7, Spring 2015

Instructor: Dr. Angus Yeung

**Assignment 3**

Due Date: Wednesday, April 1, 11:59 pm

**Submission**:

1. Create a folder called `hw3` on your computer.
2. Create a WORD document (or any similar word processing document) and put all of your answers in the same document. Your word processing software must allow you to enter math equations.
3. Save the file as `hw3`, e.g., `hw3.doc` or `hw3.docx`. Export the Word document (or similar word processing document) to PDF file. For example, File -> Save As -> choose Format as "PDF".
4. Alternatively, you can write your answers on paper, scan and save your work as a PDF file.
5. Copy or move your PDF file, `hw3.pdf`, to folder `hw3`
6. For programming assignment, follow the instruction in the question to submit your `.java` files.
7. When you are ready for submitting your work, zip up the `hw3` folder and upload it to Assignment 3 on Canvas.

**Only softcopy is acceptable.** Do not hand in your solutions in hardcopy. Do not encrypt your zip file with a password.

**Problems (Total: 100 Points)**:

Part A contains written assignment questions. Follow the instruction in each question carefully and show all of your work. (50 Points)

Part B contains both written and programming assignment questions. Follow the guidelines in each assignment for programming and submission requirements. (50 Points)

PART A Non-programming Questions (50 Points)

3.1 (15 Points) Given input {4371, 1323, 6173, 4199, 4344, 9679, 1989} and a hash function h(x) = x mod 10, show the resulting:

a. Separate chaining hash table.
b. Hash table using linear probing.
c. Hash table using quadratic probing.
d. Hash table with second hash function h2(x) = 7 – (x mod 7). [Hint: double hashing.]

If you cannot insert an input into hash table, you shall provide an explanation for it.

3.2 (10 Points) Show the result of rehashing the hash tables in Question 3.1.

3.3 (10 Points) What are the advantages and disadvantages of the following collision resolution strategies?

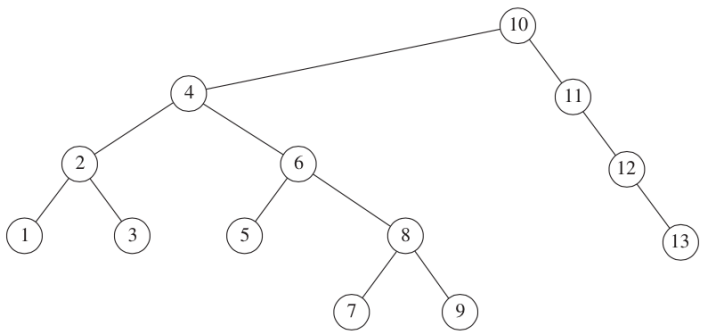
- Separate Chaining Hashing
- Linear Probing
- Quadratic Probing
- Double Hashing

And which one of the collision resolution strategies will be the fastest?

3.5 (5 Points)

a) Show the result of inserting 3, 1, 4, 6, 9, 2, 5, 7 into an initially empty binary search tree.
b) Show the result of deleting the root.

3.6 (10 Points) Show the result of accessing the keys 3, 9, 1, 5 in order in the following splay tree:



PART B Programming Questions. (50 Points)

3.7 (25 Points) Write an identifier search program using a Binary Search Tree. It reads a Java source code file and outputs a list of all identifiers (that is, variable names but not keywords, not found in comments or string constants) in alphabetical order. Each identifier should be output with a list of line numbers on which it occurs.

Your Java program must read in a text file and process the entire file. The input file name is supplied as command line argument. If the input file name is not provided, your program will process the source file of your own program.

**How to Submit:**

(1) Create the folder "`findIdentifiers`" that contains all required `.java` file(s).
(2) The folder should be part of the `hw3.zip` file that you upload to Canvas.
(3) Do not include any other file types inside the `findIdentifiers` folder except `.java` file(s).
(4) Do not declare and use any "`package`" in your `.java` file(s).
(5) Use "`findIdentifiers`" as the class name – so you can run as `%java findIdentifiers [input file].` It is unacceptable if your program can run using IDE (e.g., Eclipse) but not from command line as required here.

There is one point penalty for each requirement that a student fails to follow (total penalty: 5 points).

3.8 (25 Points) Implement a spelling checker by using a hash table with "quadratic probing" collision prevention strategy. Assume that the dictionary comes from two sources: an existing large dictionary and a second file containing a personal dictionary. Output all misspelled words and the line numbers in which they occur.

Your program must read in a text file and process the entire file. The input file name is supplied as your program's command line argument. If the input file name is not provided, your program will process a default text file. Create this default text file and place it in the same folder as your source file.

**How to Submit:**

(1) Create the folder "`spellChecker`" that contains all required `.java` file(s).
(2) The folder should be part of the `hw3.zip` file that you upload to Canvas.
(3) Do not include any other file types inside the `spellChecker` folder except `.java` file(s).
(4) Do not declare and use any "`package`" in your `.java` file(s).
(5) Use "`spellChecker`" as the class name – so you can run as `%java spellChecker [input file].` It is unacceptable if your program can run using IDE (e.g., Eclipse) but not from command line as required here.

There is one point penalty for each requirement that a student fails to follow (total penalty: 5 points).