Case Study 1

Expression Tree Evaluator

A XYZ company wants a program that implements the calculation of arithmetic expressions through data structures. They want to also implement complete modeling of the application domain.
e.g., splits data structures into modules that correspond to “objects” and relations in expression trees
Therefore construct the expression trees for the same reason

Hint:

Expression trees consist of nodes containing operators and operands
Operators have different precedence levels, different associativities, and different arities,
e.g.,
Multiplication takes precedence over addition
The multiplication operator has two arguments, whereas unary minus operator has only one
Operands are integers, doubles, variables, etc.

Case Study 2: Searching

A website called songs.com faces the problem of storing the data on the disk as the redundant copies of the data getting stored on the hard disk. For mp3 files in a large collection of MP3 files, there may be more than one copy of the same song, stored in different directories or with different file names. The goal of this exercise is to search for these duplicates.

Write a program that searches a directory and all of its subdirectories, recursively, and returns a list of complete paths for all files with a given suffix (like .mp3). Provides several useful functions for manipulating file and path names

Hint:

To recognize duplicates, you can use a hash function that reads the file and generates a short summary of the contents. For example, MD5 (Message-Digest algorithm 5) takes an arbitrarily long “message” and returns a 128-bit “checksum.” The probability is very small that two files with different contents will return the same checksum.
On a Unix system you can use the program md5sum and a pipe to compute checksums from Python.

Case Study 3: Hash Functions

Some company wants to introduce the searching capability to their word program.The features they want to implement are as follows:

1. If the amount of information associated with each key is relatively large (compared to the key itself), this information should not be stored in the hash table. Explain why and propose a scheme for representingsuch a set of data.

2. Consider the proposal to solve the clustering problem by constructing overflow trees instead of overflow lists, i.e., of organizing those keys that collided as tree structures. Hence, each entry of the scatter (hash) table can be considered as the root of a (possibly empty) tree. Compare the expected performance of this tree hashing method with that of open addressing.

3 Devise a scheme that performs insertions and deletions in a hash table using quadratic increments for collision resolution. Compare this scheme experimentally with the straight binary tree organization by applying random sequences of keys for insertion and deletion.

4 The primary drawback of the hash table technique is that the size of the table has to be fixed at a time when the actual number of entries is not known. Assume that your computer system incorporates a dynamic storage allocation mechanism that allows to obtain storage at any time. Hence, when the hash table H is full (or nearly full), a larger table H' is generated, and all keys in H are transferred to H', whereafter the store for H can be returned to the storage administration. This is called rehashing. Write a program that performs a rehash of a table H of size n.

5 Very often keys are not integers but sequences of letters. These words may greatly vary in length, and therefore they cannot conveniently and economically be stored in key fields of fixed size. Write a program that operates with a hash table and variable length keys.

Case study 4: Hospital Management System

Children's Memorial engaged ABS to provide a variety of consulting services. One of the assignments involved conducting an independent assessment of the hospital's existing help desk in order to improve service levels. ABS compared performance against industry standards and examined existing end-user support strategy and direction, tools and technology as well as staffing and skill sets. ABS'project team delivered a detailed analysis incorporating call metrics, conclusions, and recommendations. Having worked closely with ABS throughout the assessment process, the hospital's technology management team had an additional project take-away: firsthand knowledge of the ABS staff, its expertise and understanding of ITIL, and commitment to first-rate customer service. In 2006, when Children's Memorial decided to outsource, upgrade and expand its help desk services, they chose a partner they knew and trusted: ABS.

So develop a software that maintains the all features explained above and customer and patients records.