DATA STRUCTURES VOCAB LISTS

Module 1 - Vocabulary List

* Big O –WORST CASE OF COMPLEXITY
* Big Theta

**Big-Ο is used as a tight upper bound on the growth of an algorithm's effort** (this effort is described by the function f(n)), even though, as written, it can also be a loose upper bound. “Little-ο” (ο()) notation is used to describe an upper bound that cannot be tight.

* Big Omega – BEST CASE OF COMPLEXITY
* little omega

Little Omega (ω) is **a rough estimate of the order of the growth** whereas Big Omega (Ω) may represent exact order of growth. We use ω notation to denote a lower bound that is not asymptotically tight

* little o

**Big-Ο is used as a tight upper bound on the growth of an algorithm's effort** (this effort is described by the function f(n)), even though, as written, it can also be a loose upper bound. “Little-ο” (ο()) notation is used to describe an upper bound that cannot be tight.

* asymptotic
* functional

A functional data structure is (not surprisingly) **operated on using only pure functions**.

* correctness

Correctness. This property is related to the algorithm of data structures. It's important that the algorithm is correct. Correctness here means that **the algorithm always produces the expected output or follows the ground truth for the range of valid inputs, and eventually, it terminates**.

* pre-condition
* post-condition
* loop invariant

A loop invariant is a formal statement about the relationship between variables in your program which holds true just before the loop is ever run (establishing the invariant) and is true again at the bottom of the loop, each time through the loop (maintaining the invariant).

# Module 2 - Vocabulary List

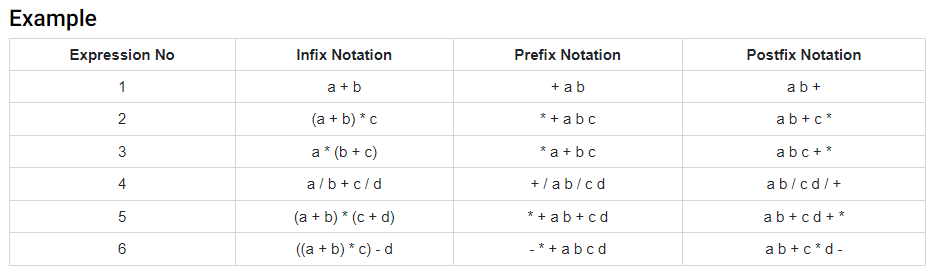
### **Vocabulary List**

* push
* pop
* empty test – if the DS is empty
* LIFO
* Prefix

Prefix Notation

In this notation, **operator is prefixed to operands, i.e. operator is written ahead of operands**. For example, +ab. This is equivalent to its infix notation a + b. Prefix notation is also known as Polish Notation

* postfix
* infix



# Module 3 - Vocabulary List

## **Vocabulary List**

Each module will include a vocabulary list – terms from the material and lecture with which you will need to be familiar. In some cases the terms will be new concepts from the lecture or text. In other cases they may be terms you are assumed to already know and will need for this module. Mastering the vocabulary in each module will help you progress more easily through the course, given the cumulative nature of the material.

* stopping case
* recursive characteristic
* redundancy
* recurrence relation
* closed form solution

Module 4 - Vocabulary List

Each module will include a vocabulary list – terms from the material and lecture with which you will need to be familiar. In some cases the terms will be new concepts from the lecture or text. In other cases they may be terms you are assumed to already know and will need for this module. Mastering the vocabulary in each module will help you progress more easily through the course, given the cumulative nature of the material.

* LIFO
* Header
* List Head
* List Tail
* ordered List
* general list

Module 5 - Vocabulary List

Each module will include a vocabulary list – terms from the material and lecture with which you will need to be familiar. In some cases the terms will be new concepts from the lecture or text. In other cases they may be terms you are assumed to already know and will need for this module. Mastering the vocabulary in each module will help you progress more easily through the course, given the cumulative nature of the material.

* linked list
* circular list
* doubly-linked list
* header
* tail
* multilinked list

Module 6 - Vocabulary List - Graphs

Each module will include a vocabulary list – terms from the material and lecture with which you will need to be familiar. In some cases the terms will be new concepts from the lecture or text. In other cases they may be terms you are assumed to already know and will need for this module. Mastering the vocabulary in each module will help you progress more easily through the course, given the cumulative nature of the material.

* rooted
* binary tree
* m-ary tree
* child
* parent
* ancestor
* descendent
* sibling
* degree
* in-degree
* out-degree

**Graphs**

* node
* vertex
* edge
* directed
* undirected
* parallel
* connected

Module 7 - Vocabulary List

Each module will include a vocabulary list – terms from the material and lecture with which you will need to be familiar. In some cases the terms will be new concepts from the lecture or text. In other cases they may be terms you are assumed to already know and will need for this module. Mastering the vocabulary in each module will help you progress more easily through the course, given the cumulative nature of the material.

**Huffman Encoding**

* Compression
* Encoding
* protocol
* path
* decoding
* frequency table

**Sorting**

* comparison
* best case
* average case
* worst case
* in-place sorting