Glossary

abstract data type (ADT)—a set of values (the **carrier set**), and operations on those values.

abstract function—a function with a signature but no body.

adjacency—vertices v_1 and v_2 are **adjacent** in an undirected graph $G=<V, E>$ if $\{v_1, v_2\}$ is a member of E.

adjacency list—a linked list of vertices adjacent to a given vertex.

adjacency matrix—an $n \times n$ Boolean matrix m that represents a graph with n vertices by storing true at location $m[v,w]$ if and only if there is an edge between v and w.

ADT assertion—a statement that must be true of the carrier set values or method set operations of the type.

acyclic graph—a graph with no cycles.

algorithm—a finite sequence of steps for accomplishing some computational task. An algorithm must have steps that are simple and definite enough to be done by a computer and terminate after finitely many steps.

algorithm analysis—the process of determining as precisely as possible how much of various resources (such as time and memory) an algorithm consumes when it executes.

array—a fixed length, ordered collection of values of the same type stored in contiguous memory locations; the collection may be ordered in several dimensions.

assertion—a statement that must be true at a designated point in a program.

average case complexity $A(n)$—the average number of basic operations performed by an algorithm for all inputs of size n given assumptions about the characteristics of inputs of size n.

best case complexity $B(n)$—the minimum number of basic operations performed by an algorithm for any input of size n.

binary search tree—a binary tree whose every vertex is such that the value at the vertex is greater than the values in its left sub-tree and less than the values in its right sub-tree.

binary tree—an ordered tree whose vertices have at most two children. The children are distinguished as the *left child* and the *right child*. The sub-tree whose root is the left (right) child of a vertex is the *left (right) sub-tree* of that vertex.

breadth-first search—a search that begins by visiting vertex v, then visits the vertices adjacent to v, then visits the vertices adjacent to each of v's adjacent vertices, and so on.

carrier set—in an abstract data type, the set of values of the type; in a data type, the set of representations of values in the carrier set of the corresponding ADT.

chaining—a hashing collision resolution scheme in which key-value pairs whose keys collide are formed into a linked list or chain whose head is in the hash table.

class invariant—an assertion that must be true of any class instance before and after calls of its exported operation.

collection—a traversable container.

collision—the event that occurs when two or more keys are transformed to the same hash table location.

complete binary tree—a full binary tree whose missing vertices are all at the right of its bottom level.

complexity $C(n)$—the number of basic operations performed by an algorithm as a function of the size of its input n when this value is the same for any input of size n.

computational complexity—the time (and perhaps the space) requirements of an algorithm.

connected components—in a graph that is not connected, the sub-graphs that are connected.

connected graph—a graph in which every pair of vertices is connected.

connected vertices—two vertices in a graph with a path between them.

container—an entity that holds finitely many other entities.

contents-of—the operation of following the address held by a pointer to obtain a value of its base type; also called the *dereferencing* operation.

cursor—a value marking a location in a data structure.

cycle—a path $<v_1, v_2, \ldots, v_n>$ in a graph in which $v_1 = v_n$.

data structure—an arrangement of data in memory locations to represent values of the carrier set of an abstract data type.

data type—an implementation of an abstract data type on a computer.

depth-first search—a search that begins by visiting vertex v, and then recursively searches the unvisited vertices adjacent to v.

dequeue—a dispenser whose elements can be accessed, inserted, or removed only at its ends.

dereferencing—the operation of following the address held by a pointer to obtain a value of its base type; also called the *contents-of* operation.

digraph—a directed graph.

directed graph—a graph in which the edges are ordered pairs of vertices; the edges have direction and are represented by arrows.

dispenser—a non-traversable container.

divide and conquer algorithm—an algorithm that solves a large problem by dividing it into parts, solving the resulting smaller problems, and then combining these solutions into a solution to the original problem.

double hashing—in open addressing collision resolution, a probe sequence using an increment generated by applying a second hash function to the key.

doubly linked list—a linked structure whose nodes each have two reference or pointer fields used to form the nodes into a sequence. Each node but the first has a predecessor link field containing a reference or pointer to the previous node in the list, and each node but the last has a successor link containing a reference or pointer to the next node in the list.

dynamic array—an array whose size is established at run-time and can change during execution.

element—a value stored in an array or a traversable container (a collection).

factory method—an operation of a class that returns a new instance of some class.

fixed array—an array whose size is established when space for the array is allocated and cannot change thereafter.

forest—a set of trees with no vertices in common.

full binary tree—a binary tree whose every level is full except possibly the last.

graph—a collection of *vertices* (or *nodes*) and *edges* connecting the vertices. An edge may be thought of as a pair of vertices. Formally, a graph is an ordered pair $<V,E>$ where V is a set of vertices and E is a set of pairs of elements of V.

graph search—a systematic traversal of a graph along its edges.

hash function—a function that transforms a key into a value in the range of indices of a hash table.

hash table—an array holding key-value pairs mapped to array locations by a hash function applied to keys.

heap—a complete binary tree whose every vertex has the heap-order property.

heap-order property—a vertex has the heap order property when the value stored at the vertex is at least as large as the values stored at its descendents.

infix expression—an expression in which the operators appear between their operands.

iteration—the process of accessing each element of a collection in turn; the process of traversing a collection.

iterator—an entity that provides serial access to each member of an associated collection.

linear probing—in open addressing collision resolution, a probe sequence that begins with the hash table index and increments it by a constant value modulo the table size.

linked data structure—a collection of nodes formed into a whole through its constituent node link fields.

linked tree—a linked structure whose nodes form a tree.

list—an ordered collection.

load factor—in hashing, the value $\lambda = n/t$, where n is the number of elements in the hash table and t is the table size.

map—an unordered collection of elements, which are values of an *element type*, accessible using a key, which is a value of a *key type*; also called a **table**, **dictionary**, or **associative array**.

open addressing—a hashing collision resolution scheme in which records with colliding keys are stored at other free locations in the hash table found by probing the table for open locations.

node—an aggregate variable with data and link (reference or pointer) fields; in a graph, a vertex.

open addressing—a hashing collision resolution scheme in which key-value pairs with colliding keys are stored at other free locations in the hash table found by probing the table for open locations.

path—in a graph, a sequence $p = <v_1, v_2, \ldots, v_n>$, where $n \geq 2$, such that every pair of vertices v_i and v_{i+1} in p are adjacent.

path length—the number of edges in a path.

pointer—a type whose carrier set contains addresses of values of an associated base type.

post condition—an assertion that must be true at the completion of an operation.

postfix expression—an expression in which the operators appear after their operands.

precondition—an assertion that must be true at the initiation of an operation.

prefix expression—an expression in which the operators appear before their operands.

priority queue—a queue whose elements each have a non-negative integer **priority** used to order the elements of the priority queue such that the highest priority elements are at the front and the lowest priority elements are at the back.

queue—a dispenser holding a sequence of elements that allows insertions only at one end, called the **back** or **rear**, and deletions and access to elements at the other end, called the **front**.

record—a finite collection of named values of arbitrary type called **fields** or **members**; a record is also called a **struct**.

recurrence—a recurrence relation plus one or more initial conditions that together recursively define a function.

recurrence relation—an equation that expresses the value of a function in terms of its value at another point.

recursive operation—an operation that either calls itself directly, or calls other operations that call it.

reference type—a type whose variables hold references to locations where data structures representing the values of the carrier set of the type are stored; compare to **value type**.

sentinel value—a special value placed in a data structure to mark a boundary.

sequential search—an algorithm that looks through a list from beginning to end for a key, stopping when it finds the key.

set—an unordered collection in which an element may appear at most once.

simple cycle—a cycle in a graph with no repeated edges or vertices (except the first and the last vertices).

simple path—a list of distinct vertices such that successive vertices are connected by edges.

simple type—a type in which the values of the carrier set are atomic, that is, they cannot be divided into parts.

singly linked list—a linked data structure whose nodes each have a single link field used to form the nodes into a sequence. Each link but the last contains a reference or pointer to the next node in the list; the link field of the last node contains nil.

slice—a reference to a contiguous segment of an associated array.

software design pattern—a model proposed for imitation in solving a software design problem.

sorting algorithm—an algorithm that rearranges records in lists so that they follow some well-defined ordering relation on values of key fields in each record.

spanning tree—any sub-graph of a connected graph G that is a tree and contains every vertex of G.

stack—a dispenser holding a sequence of elements that can be accessed, inserted, or removed at only one end, called the **top**.

static array—see fixed array.

string—a finite sequence of characters drawn from some alphabet.

struct—a record consisting of named fields of various types.

structured type—a type whose carrier set values are composed of some arrangement of atomic values.

sub-graph—a graph $H=<W, F>$ is a **sub-graph** of graph $G=<V, E>$ if $W \subseteq V$ and $F \subseteq E$.

tail recursive algorithm—an algorithm in which at most one recursive call is made as the last step of each execution of the algorithm's body.

traversable—a container is traversable iff all the elements it holds are accessible to clients.

tree—a graph with a distinguished vertex r, called the *root*, such that there is exactly one simple path between each vertex in the tree and r; alternatively, an acyclic connected graph.

undirected graph—a graph in which the edges are sets of two vertices; the edges have no direction and are represented by line segments in pictures.

unreachable code assertion—an assertion that is placed at a point in a program where execution should not occur under any circumstances.

value type—a type whose variables hold data structures representing the values of the carrier set of the type; compare to **references type**.

variable-length encoding—a representation of a set of values that uses bit strings of different lengths to save space: more frequently occurring values are represented by shorter bit strings and less frequently occurring values by longer bit strings.

worst case complexity $W(n)$—the maximum number of basic operations performed by an algorithm for any input of size n.