Glossary for Core Structures

A

- abstract class [Abstract and concrete classes]
  - A class that exists purely for the sake of being inherited by other classes to provide some shared behavior
- abstraction [Abstraction]
  - Focusing on the essential qualities of something rather than one specific example
- access control [Access control, compliance, and injection]
  - Rules specifying who can access which parts of a database
- accessor [Creating class diagrams: Behaviors]
  - See getter method.
- ACID [ACID and transactions]
  - An acronym that refers to the requirement that database transactions be atomic, consistent, isolated, and durable
- actor (use case) [Identifying the actors]
  - Anything that lives outside the application but interacts with it to accomplish some goal
- aggregate functions [Aggregate functions]
  - SQL functions that use more than one piece of data to generate a value
- aggregation [Aggregation]
  - An object relationship in which one object is built of other objects, often referred to as a has-a relationship; the subsidiary objects have an existence independent of the containing object.
- analysis (object-oriented) [Analysis, design, and programming]
  - Figuring out what a program should do; what problem you are trying to solve
- ANSI [Basic SQL]
  - American National Standards Institute. Their web site says ANSI is "a private, non-profit organization that administers and coordinates the U.S. voluntary standards and conformity assessment system."
- approximate algorithm [What are algorithms?]
  - An algorithm that tries to find an answer that might or might not be exact (such as a facial recognition algorithm, which might not give the same answer every single time)
- array [Arrays; What are arrays?]
  - A collection that groups pieces of data in a certain order and assigns the collection a name, also called an array or list in some programming languages; the position of each element is defined by an index.
- array index out of bounds error [What are arrays?]
  - An error that happens when a program tries to access an item at an index position outside an array's limit
- array indexing [Arrays]
  - A constant-time operation to access an element of an array
- array insertion/deletion [Arrays]
  - A linear time operation to insert or delete an element at an arbitrary location in an array; at the end of an array, this is usually a constant time operation.
- array, jagged [Jagged arrays]
  - A multidimensional array where each row can have a different length
- array, ragged [Jagged arrays]
  - A multidimensional array where each row can have a different length
- array, resizable [Resizable arrays and language support]
  - An array whose length can be modified; programs can add additional elements to such an array.
- ASCII [Understanding hash functions]
  - American Standard Code for Information Interchange—a format for text where each character has a single numerical representation; this encoding has only 256 possible entries.

- associative table (database) [Relationships]
  - A table that contains columns for foreign keys from the tables being associated
- atomic transaction [ACID and transactions]
  - A transaction must be indivisible; pieces of it can't be separated out.
- attribute [Objects]
  - Data that an object has, also called a field or property
- auto-increment [Primary and foreign keys]
  - A database field whose value increases by one every time a row is added; usually used for a primary key

B

- behavior [Objects]
  - An action that an object can do, also called a method
- behavioral patterns [Design patterns]
  - Design patterns concerned with communication between objects as a program is running
- big O notation [Measuring algorithm performance; Big O notation]
  - A notation used to describe how a particular algorithm performs as the size of the set of inputs grows over time; the O stands for orderof operation.
- binary search [Ordered list search; Search arrays]
  - An algorithm that determines whether an item is in a sorted list; it works by starting in the middle of the list and, depending on whether the value sought is less or greater than the item, discards the "wrong half" of the list. It repeats this process until the item is found or there are no elements left to search. Its time complexity is logarithmic, O(log n).
- binary search tree [Understand binary search trees]
  - A binary tree in which the left child must be less than its parent and the right child must be greater than its parent
- bit [Numerical data types]
  - A single zero or one
- black boxing [Encapsulation]
  - The principle that an object should not make anything about itself available except what is absolutely necessary for other parts of an application to work
- Boolean data type [Booleans and characters]
  - A data type with two possible values: True or False; usually used to perform logical operations, most commonly to determine whether some condition is true
- BST [Understand binary search trees]
  - A binary tree in which the left child must be less than its parent and the right child must be greater than its parent
- bubble sort [The bubble sort]
  - A sorting algorithm that works by repeatedly comparing each item to its neighbor and swapping them if needed; its time performance is quadratic, O(n2).
- bucket [Understanding hash tables]
  - A place for a key-value pair to go in a hash table

C

- camel case format [Creating class diagrams: Attributes]
  - Naming convention that starts with a lowercase letter and uses uppercase for additional words, such as priceAfterDiscount; Java tends to use this convention.
- cascading delete [Relationship rules and referential integrity]
  - A process in which associated records are removed in order to maintain consistency; for example, deleting a restaurant customer would also remove all their orders.
- chaining [Pros and cons of hash-based structures]
  - Creating a linked list with additional values to handle collisions in a hash table
- CHAR [Columns and data types]
  - A database data type that contains a string with a fixed length of characters
- character data type [Booleans and characters]
  - A data type that can store a single symbol or letter
- child class [Inheritance; Identifying inheritance situations]
  - See subclass.
- child node [Introduction to tree data structures]
  - A node in a tree that is referred to by another node
- class [Classes]

- A detailed description; the definition or template of what an object will be
- class, child [Inheritance; Identifying inheritance situations]
  - See subclass.
- class, parent [Inheritance; Identifying inheritance situations]
  - See superclass.
- class relationships [Identifying class relationships]
  - A diagram showing how objects in an application are related to one another
- class responsibilities [Identifying class responsibilities]
  - Determination of the actions that a class must perform and which classes are responsible for which actions
- clause, SQL [Basic SQL]
  - A part of an SQL statement that has a keyword specifying some action to take and something to act on or use
- code smell [General development principles]
  - Valid code that somehow doesn't seem right; it doesn't pass the "smell test"
- collaborator [CRC cards]
  - A class that will interact with a class being described in a CRC card
- collection algorithms [Common algorithms in programming]
  - Algorithms that involve manipulating or navigating among sets of data that are stored within a particular structure
- collision [Hash Tables; Understanding hash functions]
  - An event that occurs when two separate keys in a hash table map to the same slot in the table
- column (database) [Relational databases; Understanding databases: Benefits of spreadsheets]
  - Columns represent attributes of each row. For example, each customer's row might contain information about their name, email address, and phone number.
- compile-time polymorphism [Polymorphism]
  - Choosing which of several overloaded methods to call by looking at the method signature, also called static polymorphism
- composite key [Keys and unique values]
  - A key that is made up of two or more fields in the data
- composition [Composition]
  - An object relationship in which one object is built of other objects; the subsidiary objects are owned by the containing object.
- computational algorithms [Common algorithms in programming]
  - Algorithms that take one set of data and derive another set of data from it
- conceptual model [Identifying the objects]
  - Model that identifies the most important objects in an application and the relationship between them
- concrete class [Abstract and concrete classes]
  - A class that implements any missing functionality from an abstract class
- consistent transaction [ACID and transactions]
  - Whatever a transaction does, it must leave the database in a valid or consistent state.
- constant time [Measuring algorithm performance; Big O notation]
  - A big O of one; an operation that doesn't depend on the number of elements in a dataset
- CRC cards [CRC cards]
  - Class, responsibility, collaboration—index cards containing the same information as a conceptual object diagram
- creational patterns [Design patterns]
  - Design patterns focused on the instantiation of objects
- CRUD [Basic SQL]
  - An acronym for common database operations: Create, Read, Update, and Delete

D

- data structure [Introduction to data structures]
  - A container that allows programmers to combine several pieces of data into a single structure; this helps connect and group data.
- data type [Introduction to data and data types]
  - A category for values; for example, a value can be a number, or it can be a string of text characters. A data type also defines how we can operate on those values.
- data type (database) [Columns and data types]
  - The kind of information stored in each column

- database [Why use a database?]
  - A structure that stores information in an organized, consistent, reliable, and searchable way
- database column [Relational databases; Understanding databases: Benefits of spreadsheets]
  - Columns represent attributes of each row. For example, each customer's row might contain information about their name, email address, and phone number.
- database field [Understanding databases: Benefits of spreadsheets]
  - See database column.
- database index [Indexes, transactions, and stored procedures]
  - A reference to each value in a field and where it's located for quick access; this makes retrieval faster but can increase time for some operations like inserting a record.
- database key [Keys and unique values]
  - A unique value used to refer to only one specific row or record
- database, relational [Relational databases]
  - A database in which data are organized into relations—tables of related data
- database row [Relational databases; Understanding databases: Benefits of spreadsheets]
  - An instance of a given entity; for example, each customer in a database has their information in a table row.
- database table [Relational databases]
  - A collection of rows (representing individual entities) and columns (the attributes of those entities)
- database transaction [Indexes, transactions, and stored procedures]
  - A group of queries or statements treated as a block of activities; if one of the components fails for any reason, the whole group of statements is not executed, and anything that is partially completed is rolled back.
- DATETIME [Columns and data types]
  - A database type that stores a time along with a date
- DBMS [What you should know]
  - Database management system—this is the software we use to interact with a database.
- DCL [Basic SQL]
  - Data control language—commands for controlling access to database tables
- DDL [Basic SQL]
  - Data definition language—commands for defining a database
- delete, cascading [Relationship rules and referential integrity]
  - A process in which associated records are removed in order to maintain consistency; for example, deleting a restaurant customer would also remove all their orders.
- DELETE statement [Modifying data]
  - An SQL statement that lets you remove specific rows in a database table
- denormalization [Denormalization]
  - The process of intentionally duplicating information in database tables in violation of normalization rules
- deque [Stacks and queues; Specialized queues]
  - Double-ended queue—a data structure that is optimized for adding and removing elements from both ends of the collection
- dequeue [Stacks and queues; Implement queues in Swift]
  - Remove an element from a queue; Note: this is not the same as a deque, although the spellings are similar.
- design (object-oriented) [Analysis, design, and programming]
  - Figuring out how a program should accomplish its task
- design pattern [Design patterns]
  - A common, repeatable solution for creating software programs; these patterns define code architectures and best practices for common problems that occur across all kinds of applications.
- desktop databases [Software options]
  - Database tools intended for small solutions with just a few users, usually hosted on a workstation rather than a dedicated server
- destructor [Classes with multiple constructors]
  - A method that is called when an object is no longer needed and is being disposed of, also called a finalizer
- deterministic algorithm [What are algorithms?]
  - An algorithm where each step has an exact decision
- dictionary (collection) [Hash Tables; What are associative arrays?; Using dictionaries in

Python]
    - A collection that lets you store related information with a label for each item, also called a map, hash map, table, or associative array in some programming languages
- difference (set) [Sets in Python 3]
    - Given two sets, the difference is a new set that contains all the items in the first set that are not in the second set; for example, the difference of A, B, C and B, D, E is A, C.
- divide-and-conquer algorithm [The merge sort]
    - An algorithm that works by dividing a problem into smaller parts and solving those
- DML [Basic SQL]
    - Data manipulation language—commands for interacting with database data
- doubly linked list [Singly vs. doubly linked lists]
    - A linked list in which each node has both a pointer to the next node and a pointer to the previous node in the list
- DRY [General development principles]
    - "Don't repeat yourself"—a principle that says you should avoid copying and pasting large sections of code without any changes
- durable transaction [ACID and transactions]
    - The information changed in a transaction must actually get written to the database.
- dynamically typed language [OOP support in different languages]
    - A language where a variable's type does not need to be specified; Python and Ruby are dynamically typed.

E

- encapsulation [Encapsulation]
    - Bundling an object's attributes and methods within the same class; this is often done to restrict access to some of an object's components.
- enqueue [Stacks and queues; Implement queues in Swift]
    - Add an element to a queue
- enterprise database management systems [Software options]
    - Database systems intended to be used by huge numbers of people and run on infrastructure that can serve millions of interactions simultaneously
- ER diagram [Modeling and planning a database]
    - Entity-relationship diagram—shows a table's fields and its relationships to other tables
- exact algorithm [What are algorithms?]
    - An algorithm that produces a known predictable value
- execution flow [Use cases]
    - In a use case, the steps needed to accomplish a goal
- expression, SQL [Basic SQL]
    - Part of an SQL statement that sets parameters within which to operate

F

- factorial [Power and factorial]
    - A mathematical operator that is the product of a number and all the numbers before it; thus, 5 factorial (written as 5!) is 5 × 4 × 3 × 2 × 1, or 120.
- factory method [Design patterns]
    - A design pattern providing a structured way to instantiate different types of objects
- field [Objects]
    - Data that an object has, also called an attribute or property
- field (database) [Understanding databases: Benefits of spreadsheets]
    - See database column.
- FIFO [What are queues?]
    - First in, first out—the policy that a queue follows
- filter [Unique filtering with hash table]
    - A filtering algorithm goes through a data structure, retaining only those elements that satisfy some condition and discarding the others; for example, you might want to go through a list of items and keep only the unique items, discarding all the duplicates.
- final class [Abstract and concrete classes]
    - In Java, a class that cannot be extended or inherited from
- finalizer [Classes with multiple constructors]
    - See destructor.
- finite set [Sets in Python 3]

    - A set with a specific set of elements that cannot be added to
- first normal form [First normal form]
    - A database requirement that values in each field in each table have only one value in them and there are no columns representing repeated kinds of data for each row
- foreign key [Relationships]
    - A key that is a primary key in one table but not in another
- frozen set [Sets in Python 3]
    - A set with a specific set of elements that cannot be added to
- function, hash [Hash Tables; Understanding hash functions]
    - A function that uses a key to compute an index to the slots in the hash table and map the key to the value
- functionality [FURPS+ requirements]
    - The capabilities and features of an app
- functions, aggregate [Sorting results]
    - SQL functions that use more than one piece of data to generate a value
- FURPS [FURPS+ requirements]
    - Functionality, usability, reliability, performance, and supportability—a checklist of key qualities to consider when determining requirements
- FURPS Plus [FURPS+ requirements]
    - FURPS with four more categories: design constraints, implementation, standards, and physical requirements

G

- garbage collection [Classes with multiple constructors]
    - A process by which the runtime system keeps track of which items in memory are no longer needed and deletes them
- getter method [Creating class diagrams: Behaviors]
    - A method that retrieves the value of an object's attribute, also called an accessor
- "god object" [Identifying class responsibilities]
    - In object-oriented design, an object that knows too much or does too much

H

- hash function [Hash Tables; Understanding hash functions]
    - A function that uses a key to compute an index to the slots in the hash table and map the key to the value
- head (linked list) [What are linked lists?]
    - The first node in a linked list
- heap (using trees) [Understand heaps]
    - A data structure implemented as a binary tree to hold a collection of objects; these can be used for a priority queue.

I

- identity [Objects]
    - An object's existence; an object's identity is different from every other object's identity.
- index [What are arrays?]
    - A number that gives an item's position in a list; in most programming languages, the first item in a list has an index of zero.
- index, database [Indexes, transactions, and stored procedures]
    - A reference to each value in a field and where it's located for quick access; this makes retrieval faster but can increase time for some operations like inserting a record.
- index number [What are arrays?]
    - A number that gives an item's position in a list; in most programming languages, the first item in a list has an index of zero.
- infinite set [Sets in Python 3]
    - A set with an infinite number of elements, such as a set containing all negative numbers
- inheritance [Inheritance; Identifying inheritance situations]
    - A process for passing on attributes and methods from other existing classes to a new class
- initialize [Use arrays in Swift]
    - To give a piece of data or data structure a starting (initial) value

- To give a piece of data or data structure a starting (initial) value
- instance [Classes]
  - An individual object; different instances can have different values for their attributes even if they share the same class.
- instantiation [Classes; Instantiating classes]
  - Creating an object based on a class
- interface (object-oriented) [Interfaces; Lists in other languages]
  - A structure that declares a set of methods for a class to implement; the methods are not implemented—the method signatures specify a capability for a class to implement. Every class that implements the interface must provide a body for all the methods.
- intersection (set) [Sets in Python 3]
  - Given two sets, the intersection is a new set that contains the items in the first set that also belong to the second set; for example, the intersection of A, B, C and B, D, E is B (it is the only item in both sets).
- isolated transaction [ACID and transactions]
  - While activities in a transaction are being completed, nothing else can make changes to the data involved.

J

- jagged array [Jagged arrays]
  - A multidimensional array where each row can have a different length
- JOIN keyword [Joining tables]
  - An SQL keyword that tells the database to consider two or more tables when making a selection

K

- key, composite [Keys and unique values]
  - A key that is made up of two or more fields in the data
- key, database [Keys and unique values]
  - A unique value used to refer to only one specific row or record
- key, foreign [Relationships]
  - A key that is a primary key in one table but not in another
- key, surrogate [Keys and unique values]
  - A field that is added to a table and has no natural key so that each row can have a unique value
- key, synthetic [Keys and unique values]
  - A field that is added to a table and has no natural key so that each row can have a unique value
- key-value pair [What are associative arrays?]
  - Entries in a dictionary; the key is used for indexing the dictionary to find the corresponding value.

L

- leaf node [Introduction to tree data structures]
  - A node in a tree that has no child nodes
- LIFO [What are stacks?]
  - Last in, first out—the policy that a stack follows
- linear time [Measuring algorithm performance; Search arrays; Big O notation]
  - A big O of n; the operation takes time proportional to the number of elements in the dataset.
- linked list [Linked lists; What are linked lists?]
  - A linear collection of data elements, sometimes called nodes; the data items in a linked list need not be contiguous—they are linked using pointers.
- linking table (database) [Relationships]
  - A table that contains columns for foreign keys from the tables being associated
- list (collection) [Arrays; What are arrays?]
  - A collection that groups pieces of data in a certain order and assigns the collection a name, also called an array or list in some programming languages; the position of each element is defined by an index.
- log-linear time [Measuring algorithm performance]
  - A big O of n log n; the operation takes time proportional to the number of elements in a dataset times the logarithm of the number of elements.

- logarithmic time [Measuring algorithm performance]
  - A big O of log n; the operation takes time proportional to the logarithm of the number of items in the dataset.

M

- many-to-many relationship [Relationships; Many-to-many relationships]
  - A relation that associates many records in one table with multiple records in another table; for example, associating many customers with many purchases if you want to keep a record of each purchase each customer has ever ordered
- master object [Identifying class responsibilities]
  - An object filled with many unrelated behaviors, seemingly existing to control everything else around it; this is something to avoid.
- max heap [Understand heaps]
  - A heap (using trees) in which the root node always contains the largest value in the entire heap
- memento design pattern [Design patterns]
  - A design pattern outlining a proven approach for restoring an object to a previous state
- merge sort [The merge sort]
  - A sorting algorithm that recursively divides a set of data into smaller parts, sorts them, and then recombines the parts; in general, its time performance is log linear, O(n log n). This sort requires additional memory.
- method [Classes]
  - Also called a behavior, a function that specifies what actions an object can perform; it is a block of code that can be called to perform some action, and it may return a value.
- method signature [Polymorphism]
  - In Java, the method's name and the number and types of its parameters
- method, static [Static attributes and methods]
  - A variable or method that is shared across all objects in the same class, also referred to as a class-level or shared variable/method
- min heap [Understand heaps]
  - A heap (using trees) in which the root node always contains the lowest value in the entire heap
- minimum viable product [Defining requirements]
  - The bare necessities required to have a usable product
- multidimensional array [Multidimensional arrays]
  - An array that has other arrays embedded in it, also called a multidimensional list in some programming languages
- multiple inheritance [OOP support in different languages]
  - The condition where a derived class can inherit from more than one base class; C++ and Python support this.
- multiplicity [Identifying class relationships]
  - In a UML diagram, a representation of one or more of something
- mutator [Creating class diagrams: Behaviors]
  - See setter method.
- MVP [Defining requirements]
  - The bare necessities required to have a usable product

N

- node [Linked lists; What are linked lists?]
  - An element in a linked list
- node, child [Introduction to tree data structures]
  - A node in a tree that is referred to by another node
- node, leaf [Introduction to tree data structures]
  - A node in a tree that has no child nodes
- node, parent [Introduction to tree data structures]
  - A node in a tree that refers to another node; the root node in a tree does not have a parent node—no other nodes refer to it.
- node, root [Introduction to tree data structures]
  - The specific starting node in a tree
- node, sibling [Introduction to tree data structures]
  - Two child nodes in a tree with the same parent node
- non-deterministic algorithm [What are algorithms?]

- An algorithm that attempts to produce a solution using successive guesses that become more accurate over time
- non-functional requirements [Defining requirements]
  - Requirements that place constraints on how an application should function; for example, a banking transaction may have to comply with regulations.
- normalization rules [Normalization]
  - Rules that help reduce redundancy and improve the integrity of data in a database
- null [Numbers and other types]
  - A database value that indicates that a value is missing
- numerical data type [Numerical data types]
  - Different ways numbers are classified, for example whole numbers and decimal numbers.

O

- object [Primitive vs. reference types in memory]
  - A value in memory referenced by an identifier
- one-to-many relationship [Relationships; One-to-many relationships]
  - A relation that associates one record in one table with multiple records in another table; for example, one item on a menu might be associated with multiple customers who choose it as their favorite.
- one-to-one relationship [Relationships; One-to-one relationships]
  - A relation that associates only one record on one table with one—and only one—record on another table
- ORDER BY clause [Sorting results]
  - An SQL clause that lets you specify the order in which results should be sorted; Ascending order is abbreviated as ASC and descending order as DESC.
- ordered list search [Ordered list search; Search arrays]
  - An algorithm that determines whether an item is in a sorted list; it works by starting in the middle of the list and, depending on whether the value sought is less or greater than the item, discards the "wrong half" of the list. It repeats this process until the item is found or there are no elements left to search. Its time complexity is logarithmic, O(log n).
- overloaded methods [Polymorphism; Classes with multiple constructors]
  - Several methods in the same class with the same name but a different set of input parameters

P

- parallel algorithm [What are algorithms?]
  - An algorithm that can split up its data into parts and work on the parts simultaneously
- parent class [Inheritance; Identifying inheritance situations]
  - See superclass.
- parent node [Introduction to tree data structures]
  - A node in a tree that refers to another node; the root node in a tree does not have a parent node—no other nodes refer to it.
- performance [FURPS+ requirements]
  - Measurement based on factors such as an app's speed, efficiency, memory usage, and response time throughput
- performance requirements [Defining requirements; FURPS+ requirements]
  - Constraints on an application's performance, such as response time or number of simultaneous users
- pointer [Primitive vs. reference types in memory]
  - An address that points to where a data structure is in memory
- polymorphism [Polymorphism]
  - From the Greek words for "many forms"; see run-time polymorphism and compile-time polymorphism.
- pop [Resizable arrays and language support; Implement stacks in Swift]
  - An operation that removes items from the end of an array or the top of a stack
- precision (numerical) [Numerical data types]
  - The range of numerical values that a data type can store
- precondition (use case) [Use cases]
  - A condition that must be true to begin the use case
- predicate, SQL [Basic SQL]
  - Part of an SQL statement that sets parameters within which to operate

- primary actor [Use cases]
  - In a use case, the person who will interact with the application
- primary actor (use case) [Identifying the actors]
  - The person who initiates a scenario; this person might not be the most important actor.
- primary key (database) [Keys and unique values]
  - The most important key in a table
- primitive data type [Primitive types in memory]
  - Basic or value types; they have a fixed size that does not depend on the data inside them.
- priority queue [Specialized queues]
  - A queue in which each element has a priority associated with it; the next item to be dequeued is the item with the highest priority.
- procedural programming [Object-oriented thinking]
  - Writing a program as a long series of operations to execute
- programming paradigm [Object-oriented thinking]
  - A set of ideas that is supported by many languages
- property [Objects]
  - Data that an object has, also called a field or attribute
- push [Resizable arrays and language support; Implement stacks in Swift]
  - An operation that adds items to the end of an array or the top of a stack

Q

- quadratic time [Measuring algorithm performance]
  - A big O of n2; the operation takes time proportional to the square of the number of elements in a dataset.
- queue [Stacks and queues; What are queues?]
  - A data structure where the first entry stored is also the first item removed; you can think of a queue like the line of customers in a store—the first person in line is the first person to be served.
- quicksort [The quicksort]
  - A sorting algorithm that recursively divides a set of data into parts that are greater and less than a pivot point, sorts the subparts, and recombines them; its time performance is log linear, O(n log n). This sort does not require additional memory.

R

- ragged array [Jagged arrays]
  - A multidimensional array where each row can have a different length
- random access [Pros and cons of lists]
  - Access to a data structure in which you give an index and get the value at that slot immediately, as in an array
- recursion [Understanding recursion]
  - When a function calls itself from within its own code
- reference type [Primitive vs. reference types in memory]
  - A data type that uses a reference (pointer) to its specific value from an address where the item is stored rather than giving direct access to the data itself
- referential integrity [Relationship rules and referential integrity]
  - A design principle in which ta database is aware of a relationship and will not let you modify data in a way that violates that relationship
- relational database [Relational databases]
  - A database in which data are organized into relations—tables of related data
- relationship [Relationships]
  - How records should be connected to one another
- relationship, many to many [Relationships; Many-to-many relationships]
  - A relation that associates many records in one table with multiple records in another table; for example, associating many customers with many purchases if you want to keep a record of each purchase each customer has ever ordered
- relationship, one to many [Relationships; One-to-many relationships]
  - A relation that associates one record in one table with multiple records in another table; for example, one item on a menu might be associated with multiple customers who choose it as their favorite.
- relationship, one to one [Relationships; One-to-one relationships]
  - A relation that associates only one record on one table with one—and only one—record on another table

- reliability [FURPS+ requirements]
  - Usually expressed in terms of how much system downtime is acceptable and how the system can be recovered
- requirements [Defining requirements]
  - What an application or product needs to do
- resizable array [Resizable arrays and language support]
  - An array whose length can be modified; programs can add additional elements to such an array.
- root node [Introduction to tree data structures]
  - The specific starting node in a tree
- row (database) [Relational databases; Understanding databases: Benefits of spreadsheets]
  - An instance of a given entity; for example, each customer in a database has their information in a table row.
- run-time polymorphism [Polymorphism]
  - A process that allows programmers to access methods using the same interface on different types of objects that may implement those methods in different ways
- runtime stack [Error tracing with stacks]
  - A stack that keeps track of the state of a program, including values and function calls; related to a call stack

S

- search, binary [Ordered list search; Search arrays]
  - An algorithm that determines whether an item is in a sorted list; it works by starting in the middle of the list and, depending on whether the value sought is less or greater than the item, discards the "wrong half" of the list. It repeats this process until the item is found or there are no elements left to search. Its time complexity is logarithmic, O(log n).
- search, ordered list [Ordered list search; Search arrays]
  - An algorithm that determines whether an item is in a sorted list; it works by starting in the middle of the list and, depending on whether the value sought is less or greater than the item, discards the "wrong half" of the list. It repeats this process until the item is found or there are no elements left to search. Its time complexity is logarithmic, O(log n).
- search, unordered list [Unordered list search]
  - An algorithm that determines whether an item is in a list whose elements are in no particular order by examining each item in turn; its time complexity is linear, O(n).
- searching algorithms [Common algorithms in programming]
  - Algorithms that find a specific piece of data inside a larger data structure
- second normal form [Second normal form]
  - A database requirement that, for every non-key table column, each of the values must rely on only the whole key; the values must describe something about the row that can't be determined from just the part of a key.
- secondary actor (use case) [Diagramming use cases]
  - An actor that takes more of a reactive role than the primary actor
- security requirements [Defining requirements]
  - Requirements that determine what security an application will need
- SELECT statement [Narrowing query results]
  - An SQL statement that asks for specific rows in a database table
- sequential access [Pros and cons of lists]
  - Access to a data structure in which you need to go through each item in turn to get to a specific index, as in a linked list
- sequential algorithm [What are algorithms?]
  - An algorithm in which each operation is done in sequence
- set [What are sets?]
  - A collection of different objects where the order doesn't matter; no two objects in the set are identical.
- SET statement [Modifying data]
  - An SQL statement that lets you update data in a database
- setter method [Creating class diagrams: Behaviors]
  - A method that allows a program to set the value of an object's attribute, also called a mutator
- short int [Numerical data types]
  - An integer data type that is usually 16 bits long
- sibling nodes [Introduction to tree data structures]

    - Two child nodes in a tree with the same parent node
- signed data value [**Numerical data types**]
    - A numerical value that contains negative, zero, or positive values
- single inheritance [**Inheritance**]
    - The condition where a derived class can only inherit from one class; Java is one such language.
- singly linked list [**Singly vs. doubly linked lists**]
    - A linked list that only has a pointer to the next node in the list
- snake case format [**Converting class diagrams into code**]
    - Naming convention where words in a variable name are separated by underscores, such as price_after_discount; Python tends to use this convention.
- SOLID [**General development principles**]
    - A set of five separate but interrelated principles that apply to object-oriented design
- sort, bubble [**The bubble sort**]
    - A sorting algorithm that works by repeatedly comparing each item to its neighbor and swapping them if needed; its time performance is quadratic, O(n2).
- sort, merge [**The merge sort**]
    - A sorting algorithm that recursively divides a set of data into smaller parts, sorts them, and then recombines the parts; in general, its time performance is log linear, O(n log n). This sort requires additional memory.
- sort, quicksort [**The quicksort**]
    - A sorting algorithm that recursively divides a set of data into parts that are greater and less than a pivot point, sorts the subparts, and recombines them; its time performance is log linear, O(n log n). This sort does not require additional memory.
- sorting algorithms [**Common algorithms in programming**; **Sort arrays**]
    - Algorithms that take a set of data and place it into a particular order
- space complexity [**What are algorithms?**]
    - The amount of memory and storage space an algorithm needs to do its work
- SQL [**Basic SQL**]
    - Structured Query Language—a set of commands for interacting with the data in a database
- SQL clause [**Basic SQL**]
    - A part of an SQL statement that has a keyword specifying some action to take and something to act on or use
- SQL expression [**Basic SQL**]
    - Part of an SQL statement that sets parameters within which to operate
- SQL injection [**Access control, compliance, and injection**]
    - A technique in which a user enters a value that is part of an SQL command to alter the query that should be running and change how it works
- SQL predicate [**Basic SQL**]
    - Part of an SQL statement that sets parameters within which to operate
- stack [**Stacks and queues**; **What are stacks?**]
    - A data structure where the first entry stored is the last entry removed; you can think of it as a stack of mail, where you handle the top items (most recently put on the stack) first and the bottom items (least recently put on the stack) last.
- static analysis tool [**General development principles**]
    - A tool that will analyze your code and highlight things such as duplicated or unnecessary code
- static variable/method [**Static attributes and methods**]
    - A variable or method that is shared across all objects in the same class, also referred to as a class-level or shared variable/method
- statically typed language [**OOP support in different languages**]
    - A language in which the type of all variables is known at compile time; Java and C++ are statically typed.
- stored procedure [**Indexes, transactions, and stored procedures**]
    - In databases, a kind of program that's stored in the database server; it contains a series of commands that you can then reference and use when interacting with the database.
- string [**Primitive types in memory**]
    - A data type composed of a sequence of characters
- string data type [**Primitive types in memory**]
    - A data type composed of a sequence of characters
- structural patterns [**Design patterns**]
    - Design patterns describing how classes are designed, using inheritance, composition, and aggregation
- subclass [**Inheritance**; **Identifying inheritance situations**]

    - A class that inherits from another class, also called a child class; for example, a Car is a subclass of a Vehicle.
- superclass [Inheritance; Identifying inheritance situations]
    - The class from which another class inherits attributes and methods, also called a ; for example, a Vehicle is a superclass of a Car.parent class
- support requirements [Defining requirements; FURPS+ requirements]
    - Requirements that determine how an application will be supported
- supportability [FURPS+ requirements]
    - How well an application can be tested, extended, serviced, and installed and configured
- surrogate key [Keys and unique values]
    - A field that is added to a table and has no natural key so that each row can have a unique value
- synthetic key [Keys and unique values]
    - A field that is added to a table and has no natural key so that each row can have a unique value

T

- table, associative (database) [Relationships]
    - A table that contains columns for foreign keys from the tables being associated
- table (database) [Relational databases]
    - A collection of rows (representing individual entities) and columns (the attributes of those entities)
- table, linking (database) [Relationships]
    - A table that contains columns for foreign keys from the tables being associated
- third normal form [Third normal form]
    - A database requirement that it should not be possible to determine any value in a column from a field that isn't a key
- time complexity [What are algorithms?]
    - How efficient an algorithm is relative to the size of the input it is given
- TIMESTAMP [Columns and data types]
    - A database type that captures and stores the date and time when a row is updated
- transaction, atomic [ACID and transactions]
    - A transaction must be indivisible; pieces of it can't be separated out.
- transaction, consistent [ACID and transactions]
    - Whatever a transaction does, it must leave the database in a valid or consistent state.
- transaction, database [Indexes, transactions, and stored procedures]
    - A group of queries or statements treated as a block of activities; if one of the components fails for any reason, the whole group of statements is not executed, and anything that is partially completed is rolled back.
- transaction, durable [ACID and transactions]
    - The information changed in a transaction must actually get written to the database.
- transaction, isolated [ACID and transactions]
    - While activities in a transaction are being completed, nothing else can make changes to the data involved.
- tree [Introduction to tree data structures]
    - A collection of nodes in which a node might be linked to one, two, or more nodes
- tree, binary search [Understand binary search trees]
    - A binary tree in which the left child must be less than its parent and the right child must be greater than its parent
- tree, unbalanced [Understand binary search trees]
    - A tree in which there are more levels of nodes on one side than the other

U

- UML [Unified modeling language (UML)]
    - Unified Modeling Language— a graphical notation for drawing diagrams to visualize object-oriented systems
- unbalanced tree [Understand binary search trees]
    - A tree in which there are more levels of nodes on one side than the other
- union (set) [Sets in Python 3]
    - Given two sets, the union is a new set that contains all the items in both sets; for example, the union of A, B, C and B, D, E is A, B, C, D, E.
- unit test [Software testing]

    - Code that tests a specific part of a program
- unordered list search [Unordered list search]
    - An algorithm that determines whether an item is in a list whose elements are in no particular order by examining each item in turn; its time complexity is linear, O(n).
- unsigned data value [Numerical data types]
    - A numerical value that can only contain zero or positive values
- usability [FURPS+ requirements]
    - How intuitive an app is; accuracy and completeness of documentation
- use case diagram [Diagramming use cases]
    - A UML diagram showing the relationship between actors and the different use cases in which they're involved
- use case scenario [Identifying the scenarios]
    - A description of a goal that an actor can accomplish in a single encounter; this should focus on the user's intention—what they really want to accomplish.

V

- VARCHAR [Columns and data types]
    - A database data type that contains a variable-length character string
- variable, static [Static attributes and methods]
    - A variable or method that is shared across all objects in the same class, also referred to as a class-level or shared variable/method
- visibility [Creating class diagrams: Behaviors]
    - Whether a method or attribute is exposed to other classes; in general, this can be public (accessible by other objects) or private (not accessible by other objects).

W

- WHERE clause [Narrowing query results]
    - An SQL clause that lets you choose only specific records or records that match given criteria

Y

- YAGNI [General development principles]
    - "You ain't going to need it"—a principle that says you should avoid trying to account for every possible variation of everything you could ever possibly see

Previous Next