his Data Scientist course, in collaboration with IBM, accelerates your career in Data Science and provides you with world-class training and skills required to become successful in this field. The Data Scientist course offers extensive training on the most in-demand Data Science and Machine Learning skills with hands-on exposure to key tools and technologies including Python, R, Tableau, and concepts of Machine Learning.

Become an expert in Data Science by diving deep into the nuances of data interpretation, mastering technologies like Machine Learning, and mastering powerful programming skills to take your career in Data Science to the next level.

This joint partnership between Simplilearn and IBM introduces students to an integrated blended learning approach, making them experts in data science. This Data Science course, in collaboration with IBM, will help students become industry-ready for top data scientist job roles.

**What can I expect from this Data Science course developed in collaboration with IBM?**

Upon completion of this Certificate course in Data Science, you will receive IBM certificates for the IBM courses and Simplilearn certificates for all the courses in the learning path. These certificates will testify for your skills and assert your Data Science expertise. You can also avail the following benefits as part of this Data Science online course:

* Masterclasses by IBM experts
* Ask Me Anything sessions with IBM leadership
* Exclusive Hackathons conducted by IBM
* Industry-recognized Data Scientist Master's certificate from Simplilearn

Data structures serve as the basis for [abstract data types](https://en.wikipedia.org/wiki/Abstract_data_type) (ADT). The ADT defines the logical form of the data type. The data structure implements the physical form of the data type.[[5]](https://en.wikipedia.org/wiki/Data_structure#cite_note-5)

Different types of data structures are suited to different kinds of applications, and some are highly specialized to specific tasks. For example, relational databases commonly use [B-tree](https://en.wikipedia.org/wiki/B-tree) indexes for data retrieval,[[6]](https://en.wikipedia.org/wiki/Data_structure#cite_note-6) while [compiler](https://en.wikipedia.org/wiki/Compiler) implementations usually use [hash tables](https://en.wikipedia.org/wiki/Hash_table) to look up identifiers.[[7]](https://en.wikipedia.org/wiki/Data_structure#cite_note-7)

Data structures provide a means to manage large amounts of data efficiently for uses such as large [databases](https://en.wikipedia.org/wiki/Database) and internet indexing services. Usually, efficient data structures are key to designing efficient [algorithms](https://en.wikipedia.org/wiki/Algorithm). Some formal design methods and [programming languages](https://en.wikipedia.org/wiki/Programming_language) emphasize data structures, rather than algorithms, as the key organizing factor in software design. Data structures can be used to organize the storage and retrieval of information stored in both [main memory](https://en.wikipedia.org/wiki/Main_memory) and secondary memory.[[8]](https://en.wikipedia.org/wiki/Data_structure#cite_note-8)

There are numerous types of data structures, generally built upon simpler [primitive data types](https://en.wikipedia.org/wiki/Primitive_data_type). Well known examples are:[[10]](https://en.wikipedia.org/wiki/Data_structure" \l "cite_note-10)

* A [*byte*](https://en.wikipedia.org/wiki/Byte) is the smallest amount of data that a Computer CPU can copy from memory to a register or back in a **single CPU instruction**, therefore a [bytestream](https://en.wikipedia.org/wiki/Bytestream" \o "Bytestream) is the most efficient way to run big data through a computer, hence [Stream processing](https://en.wikipedia.org/wiki/Stream_processing). [Ref](https://en.wikipedia.org/wiki/Turing_machine). No CPU can copy a single bit from memory to a register or back, because every bit in memory is part of a byte. [[11]](https://en.wikipedia.org/wiki/Data_structure#cite_note-Cline_FAQ-11)
* An [*array*](https://en.wikipedia.org/wiki/Array_data_structure) is a number of elements in a specific order, typically all of the same type (depending on the language, individual elements may either all be forced to be the same type, or may be of almost any type). Elements are accessed using an integer index to specify which element is required. Typical implementations allocate contiguous memory words for the elements of arrays (but this is not always a necessity). Arrays may be fixed-length or resizable.
* A [*linked list*](https://en.wikipedia.org/wiki/Linked_list) (also just called *list*) is a linear collection of data elements of any type, called nodes, where each node has itself a value, and points to the next node in the linked list. The principal advantage of a linked list over an array is that values can always be efficiently inserted and removed without relocating the rest of the list. Certain other operations, such as [random access](https://en.wikipedia.org/wiki/Random_access) to a certain element, are however slower on lists than on arrays.
* A [*record*](https://en.wikipedia.org/wiki/Record_(computer_science)) (also called *tuple* or *struct*) is an [aggregate data](https://en.wikipedia.org/wiki/Aggregate_data) structure. A record is a value that contains other values, typically in fixed number and sequence and typically indexed by names. The elements of records are usually called *fields* or *members*.
* A [*union*](https://en.wikipedia.org/wiki/Union_(computer_science)) is a data structure that specifies which of a number of permitted primitive types may be stored in its instances, e.g. *float* or *long integer*. Contrast with a [record](https://en.wikipedia.org/wiki/Record_(computer_science)), which could be defined to contain a float *and* an integer; whereas in a union, there is only one value at a time. Enough space is allocated to contain the widest member data-type.
* A [*tagged union*](https://en.wikipedia.org/wiki/Tagged_union) (also called [*variant*](https://en.wikipedia.org/wiki/Variant_type), *variant record*, *discriminated union*, or *disjoint union*) contains an additional field indicating its current type, for enhanced type safety.
* An [*object*](https://en.wikipedia.org/wiki/Object_(computer_science)) is a data structure that contains data fields, like a record does, as well as various [methods](https://en.wikipedia.org/wiki/Method_(computer_programming)) which operate on the data contents. An object is an in-memory instance of a class from a [taxonomy](https://en.wikipedia.org/wiki/Taxonomy). In the context of [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming), records are known as [plain old data structures](https://en.wikipedia.org/wiki/Plain_old_data_structure) to distinguish them from objects.[[12]](https://en.wikipedia.org/wiki/Data_structure#cite_note-12)

In addition, [*hashes*](https://en.wikipedia.org/wiki/Hash_table), [*graphs*](https://en.wikipedia.org/wiki/Graph_(computer_science)) and [*binary trees*](https://en.wikipedia.org/wiki/Binary_trees) are other commonly used data structures.