Big Data L3: MapReduce True/False Questions

#	Question	Answer
101	MapReduce programs consist only of a Map phase.	False
102	In MapReduce, the shuffle phase happens before the map phase.	False
103	The motivation behind MapReduce was to process massive datasets like the entire web efficiently.	True
104	In MapReduce, each key-value pair emitted by a mapper must be processed independently.	False
105	The mapper in MapReduce emits outputs of the form (key, value).	True
106	A reducer receives a key and a list of values associated with that key.	True
107	Associativity means that the order in which operations are applied affects the result.	False
108	Determinism ensures that the same input always produces the same output in MapReduce.	True
109	In a classic MapReduce word count, the shuffle phase groups identical words together.	True
110	Floating point numbers are ideal for use as keys in MapReduce.	False
111	In functional programming, ‘map‘ applies a function element-wise over a list.	True
112	‘reduce‘ in functional programming recursively combines a list of elements into a single result.	True
113	Google’s Ngram Viewer inspired the need for systems like MapReduce.	True
114	Hadoop is an open-source implementation of MapReduce.	True
115	In MapReduce, the shuffle step is free and does not involve any data movement.	False
116	MapReduce was introduced after relational databases became standard technology.	True
117	Key-skew happens when some keys have far more values than others during the reduce phase.	True
118	In MapReduce, all values for a particular key can be spread across multiple reducers.	False
119	Using combiners can help minimize the amount of data shuffled between mappers and reducers.	True
120	Combiners are optional in MapReduce and depend on whether the reduce operation is associative.	True
121	Reducers can process multiple keys at the same time in the same MapReduce job.	False

#	Question	Answer
122	In MapReduce, a combiner executes after the shuffle phase.	False
123	MapReduce is particularly efficient for highly iterative algorithms like gradient descent.	False
124	Word counting is considered an embarrassingly parallel task.	True
125	In MapReduce, shuffling and sorting are handled automatically between the map and reduce phases.	True
126	One of MapReduce's strengths is supporting interactive queries with low latency.	False
127	MapReduce jobs scale linearly with the number of processors, assuming no key-skew.	True
128	MapReduce removes the need to manage communication manually in distributed computing.	True
129	The philosophy behind MapReduce emphasizes embracing locality to minimize communication.	True
130	You should generally use floating point values as keys because they hash consistently.	False
131	In MapReduce, intermediate key-value pairs are grouped by key before reduction.	True
132	The associativity property in MapReduce ensures that reducers can work in any order.	True
133	A major drawback of MapReduce is that it lacks support for transactions and indexing.	True
134	MapReduce was criticized for being a revolutionary, completely novel computing model.	False
135	Iterative machine learning tasks fit well with the MapReduce paradigm.	False
136	MapReduce shines when solving "one-shot" batch processing problems.	True
137	In MapReduce, mappers should avoid using complicated loops to keep jobs efficient.	True
138	All mappers in a MapReduce job need to communicate with each other during the map phase.	False
139	Shuffling data between mappers and reducers is a major bottleneck if not optimized.	True
140	Using sort operations smartly during mapping can reduce overall communication in MapReduce.	True
141	Modern frameworks like Spark were developed because MapReduce struggles with iterative tasks.	True
142	Combiners in MapReduce must perform non-associative operations to be effective.	False
143	MapReduce imposes severe constraints but enables high scalability.	True
144	In MapReduce, each reducer must see **only** part of the values associated with a key.	False
145	Combiners perform a local pre-reduction on a mapper's output before shuffling.	True

#	Question	Answer
146	Key-skew in MapReduce can cause some reducers to take significantly longer than others.	True
147	Divide Et Impera ("divide and conquer") is a guiding principle behind MapReduce design.	True
148	Data exploration and visualization tasks are ideal for MapReduce.	False
149	Hadoop's ecosystem today is limited to basic MapReduce tasks only.	False
150	Learning MapReduce helps in understanding fundamental distributed computing concepts even today.	True