1. Which of the following statements is FALSE when applied to a map-redude computation?

 A. The same mapper may execute multiple map operations in sequence.

 B. The reduce operations can only start once all the map operations are concluded.

 C. It is not possible to have map and reduce operations executing simultaneously.

 D. The results from the map phase are passed directly to the reduce phase (without any processing by the infras-structure).

2. Which of the following statements is TRUE when applied to a map-redude computation?

 A. As soon as a map generates some output, the corresponding reducer starts working on that piece of data.

 B. A reducer produces as output a set of pairs sorted by their key.

 C. A reducer gets as input a pair key/set-of-values.

 D. A reducer always produces output(s) with the same key it received in its input.

3. Knowing that statement S1 occurs in the program before statement S2, which of the following conditions define an anti-dependency between S1 and S2?

 A. in(S1) ∩ out(S2) ≠ ∅.

 B. in(S1) ∩ in(S2) ≠ ∅.

 C. out(S1) ∩ out(S2) ≠ ∅.

 D. out(S1) ∩ in(S2) ≠ ∅.

4. OpenMP allows to define a parallel for with

 #pragma omp parallel for

 for (int i=0; i< n; i++) {

 do\_something ();

 }

 Select the option that presents a computation equivalent to the above parallel for!

 A.

 #pragma omp parallel

 {

 #pragma omp for

 for (int i=0; i< n; i++) {

 do\_something ();

 }

 }

 B.

 #pragma omp for

 for (int i=0; i< n; i++) {

 #pragma omp parallel

 do\_something ();

 }

 C.

 #pragma omp task

 {

 #pragma omp for

 for (int i=0; i< n; i++) {

 #pragma omp parallel

 do\_something ();

 }

 }

 D.

 #pragma omp parallel

 {

 #pragma omp for

 for (int i=0; i< n; i++) {

 #pragma omp task

 do\_something ();

 }

 }

5. We use parallel computing for…

 A. Splitting a very large data set into smaller pieces to be processed independently.

 B. Choice Solving a computational problem faster.

 C. Minimizing the latency on accessing data by creating and managing multiple replicas of the data.

 D. Merging different unrelated tasks into a single computation.

6. Select the ending that makes the following phrase FALSE: To be executed in parallel, a computation problem must…

 A. Be able to execute multiple program instructions at any moment in time.

 B. Be able to be broken apart into discrete pieces of work that can be solved independently.

 C. Be solved in less time with multiple computing resources than with a single computing resource.

 D. Be solved using a MapReduce framework.

7. Which of the following statements is TRUE?

 A. Reducers input key/value pairs are sorted by the key.

 B. Reducers output key/value pair must be of the same type as its input.

 C. Each reducer must generate the same number of key/value pairs as its input had.

 D. Is it possible to start reducing while some mappers still run.

8. The number of maps is usually driven by the total size of:

 A. Outputs.

 B. Processors.

 C. Inputs.

 D. Tasks.

9. The following loop

 for (i=0; i<20; i++)

 for (j=1; j<100; j++)

 a[i][j] = f(a[i][j-1]);

 A. Is dependent on both i and j.

 B. Is anti-dependent only on j.

 C. Is flow dependent only on j.

 D. Is output dependent only on j.

10. Suppose you have a microprocessor which executes a program consisting of 50% floating point multiply, 20% floating point divide, and the remaining 30% are from other instructions. Management wants the microprocessor at least twice as fast (execute the same program in half the time or less). You can make the divide operation run at most 3 times faster and the multiply operation run at most 8 times faster. Choose the correct ending for the following phrase:

The management's goal…

 A. can be met by making the improvement only on Multiply.

 B. can be met by making the improvement only on Divide.

 C. can only be met by making the improvement on both Multiply and Divide.

 D. can not be met.

11. Some algorithms use a divide and conquer strategy at its core. A parallel version of such algorithms may benefit from using a sequential cut-off mechanism. What is a sequential cut-off mechanism?

 A. Is a mechanism that enforces the outcome of a parallel execution to be always identical to the outcome of the corresponding sequential version.

 B. Is a mechanism to decide whether a task will spawn new tasks for parallel work, or it will execute the work sequentially.

 C. Is a mechanism that sequentially merges the results from two or more tasks that executed in parallel.

 D. Is a mechanism to decide where to split the work that shall be executed in parallel by two or more tasks.

12. Select the phrase that is TRUE!

 A. Superlinear speedup implies parallel slack.

 B. Parallel slack implies superlinear speedup.

 C. Linear speedup implies parallel slack.

 D. Parallel slack implies linear speedup.