|  |  |  |
| --- | --- | --- |
| 1. | True/False | |
|  | Q: | T/F: What the text calls “embarrassingly parallel computations” are problems that are embarrassingly parallel because large amounts of communication are needed between processes, hence the need for parallelization. |
|  |  |  |
|  | A: | False, “embarrassingly parallel computations” refer to programs that should naturally lend themselves to being parallelized and ideally require relatively little or no inter-process communication. |
|  |  |  |
|  | Ref: | Page 79 |

|  |  |  |
| --- | --- | --- |
| 2. | Multiple Choice | |
|  | Q: | Which of the following is not a task allocation methods/algorithms is not discussed in Chapter 3: Embarrassingly Parallel Computations:   1. Static task allocation 2. Sequential algorithms 3. In-order work distribution 4. Dynamic task allocation |
|  |  |  |
|  | A: | C. In-order work distribution is a made-up term. The others are discussed in the text. |
|  |  |  |
|  | Ref: | Pages 89-90 |

|  |  |  |
| --- | --- | --- |
| 3. | Fill in the Blank | |
|  | Q: | When gathering (not necessarily with the MPI\_Gather function) data from slaves, the master should expect to receive the messages in \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ order as the slaves sent the messages themselves. |
|  |  |  |
|  | A: | a different order. Message passing does not guarantee that messages arrive in the same order they were sent in, only that messages sent from a particular node arrive in the same order that they were sent. |
|  |  |  |
|  | Ref: | Page 90 |

|  |  |  |
| --- | --- | --- |
| 4. | Short Answer/Code | |
|  | Q: | Explain why it is not necessary (or beneficial) for the master to send a prompt/coordinates/start value for every computation that should be performed by the slaves when the computations for each task/pixel/etc. are independent of one another. |
|  |  |  |
|  | A: | It is a bad idea to send numerous small messages for task assignment when computations are independent of one another because then time is wasted performing communication. It is often faster for a processor to do work to figure out what to do next than to have it wait to find out its next job. This is why it is beneficial to have a prearranged work load setup where messages passed indicate which segment of a problem a slave should solve. |
|  |  |  |
|  | Ref: | Pages 90-91 (or the whole chapter, really) |