UNIT III SHARED MEMORY PROGRAMMING WITH OPENMP

2MARKS

1. Initial task region:

An initial thread executes sequentially, as if enclosed in an implicit task region called an initial task region that is defined by the implicit parallel region surrounding the whole program.

2. Define odd even transposition sort:

Odd even transposition sort is a sorting algorithm similar to bubble sort.
The list ‘A’ stores ‘n’ ints and algorithm sorts them into increasing order. During even phase each odd subscripted element is compares with element to its left and if they are out of order, they are swapped.

3. What is lock? List types of locks on OpenMP:

A lock consist of a data structure and functions that allow programmer to explicitly enforce mutual exclusion in a critical section.
Two types : simple and nested locks. **Simple** lock can only be set once before it is unset, while **nested** lock can be set multiple times by same thread before unset.

4. Differentiate shared memory model and message passing model.

The message passing model all processes typically remain active throughout the execution of the program, whereas in the shared memory model the number of active threads is one at the program’s start and finish and may change dynamically throughout the execution of the program.

A special case of a shared memory parallel program: it is simply one with no fork/joins in it. The shared memory model supports incremented parallelization.
Ability of the shared memory model to support incremental parallelization is one of its greatest advantages over the message-passing model.

5. How to enable consistency between two threads temporary view and memory?

1. The value is written to the variable by the first thread.
2. The variable is flushed by the first thread.
3. The variable is flushed by the second thread
4. The value is read from the variable by the second thread.

6. List the criteria for sentinel of conditional complication.

1. The sentinel can appear in any column but must be precede only by white space.
2. The sentinel must appear as a single word with no intervening white space.
3. Initial lines must have a space after the sentinel.
4. Continued lines must have an ampersand as the last non-blank character on the line, prior to any comment appearing on the conditionally complied line.

7. What are the ICVs stored values affect loop region?

1. run-sched-var: Controls the schedule that the runtime schedule clause uses for loop
   regions. There is one copy of this ICV per data environment.
2. def-sched-var: Controls the implementation defined default scheduling of loop
   regions. There is one copy of this ICV per device.

8. What are the ICVs stored values affect program execution?

1. Stscksize-var: Controls the stack size for threads that the OpenMP implementation creates. There is one copy of this ICV per device.
2. wait-policy-var: Controls the desired behavior of waiting threads. There is one copy of this ICV per device.
3. cancel-var: Controls the desired behavior of the cancel construct and cancellation points. There is one copy of the ICV for the whole program.
4. default-device-var: Controls the default target device. There is one copy of this ICV per data environment.

9. List the restrictions to array.

1. An array section can appear only in clause where it is explicitly allowed.
2. An array section can only be specified for a base language identifier.
3. The type of the variable appearing in an array section must be array, pointer, reference to array, or reference to pointer.
4. an array section cannot be used in a C++ user defined []-operator.

10. List the restrictions to parallel construct.

1. A program that branches into or out of parallel regions is non-conforming.
2. A program must not depend on any ordering of the evaluations of the clauses of the parallel directive, or on any side effects of the evaluations of the clauses.
3. At most one if clause can appear on the directive.
4. At most one proc_bind clause can appear on the directive.
5. At most one num_threads clause can appear on the directive.
6. Expression must evaluate to a positive integer value.

11. List the restrictions to worksharing construct.

1. Each worksharing region must be encountered by all threads in a team or by none at all, unless cancellation has been requested for the innermost enclosing parallel region.
2. The sequence of work sharing regions and barrier regions encountered must be the same for every thread in a team.

12. List the restrictions to sections construct.

1. Orphaned section directives are prohibited. That is, the section directives must appear within the sections construct and must not be encountered elsewhere in the sections region.
2. The code enclosed in a sections construct must be a structured block.
3. Only a single nowait clause can appear on a section directive.

13. Brief about simple lock routines.

The type omp_lock_t is a data type capable of representing a simple lock. For the following routines, a simple lock variable must be of omp_lock_t type. All simple lock routines require an argument that is a pointer to a variable of type omp_lock_t. The simple lock routines are as follows:

1. The omp_init_lock routine initializes a simple lock.
2. The omp_destroy_lock routine uninitializes a simple lock.
3. The omp_set_lock routine waits until a simple lock is available, and then sets it.
4. The omp_unset_lock routine unsets a simple lock.

5. The omp_test_lock routine tests a simple lock and sets it if it is available.

14. Brief about Nestable lock requires.

The type omp_lock_t is a data type capable of representing a nestable lock. For the following routines, a nestable lock variable must be of omp_lock_t type. All nestable lock routines require an argument that is a pointer to a variable of type omp_lock_t. The nestable lock routines are as follows:

1. The omp_init_lock routine initializes a nestable lock.
2. The omp_destroy_lock routine uninitializes a nestable lock.
3. The omp_set_lock routine waits until a nestable lock is available, and then sets it.
4. The omp_unset_lock routine unsets a nestable lock.
5. The omp_test_lock routine tests a nestable lock and sets it if it is available.

15. Brief about pragma.

A compiler directive in C or C++ is called a pragma. The word pragma is short for "pragmatic information". A pragma is a way to communicate information to the compiler. The information is nonessential in the sense that the compiler may ignore the information and still prodce a correct object program. However, the information provided by the pragma can help the compiler optimize the program. Like other lines that provide information to the preprocessor, a pragma begins with the # character. A pragma in C or C++ has this syntax:

#pragma omp<reset of pragma>

.

16. How does the run time system know how many threads to create?

The value of an environment variable called OMP_NUM_THREADS provides a default number of threads for parallel sections of code. In Unix you can use the printenv command to inspect the value of this variable and the setenv command to modify its value.

17. Give general structure of OpenMP program.