Dr Punit Gupta

Roll No:.....

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

TEST -2 EXAMINATION- March-April 2018

B-Tech (8rd SEM)

COURSE CODE: 14M1WCI433 **MAX. MARKS: 25**

COURSE NAME: Parallel Programming Techniques

COURSE CREDITS: 3 **MAX. TIME: 1.5 Hrs**

Note: All questions are compulsory. Carrying of mobile phone during examinations will be treated as case of unfair means. **All questions are compulsory**

Q1. [1x 5 Marks]

a. Explain the difference between **private** and **threadprivate** directive in OPENMPI?

b. Explain the importance of **omp Single** over **Barrier**?

c. Explain the difference between **MPI_Ssend** and **MPI_Send?**

d. Suggest a suitable programming technique for a task with 2% functional dependency and 20% data dependency and why?

e. Define granularity and agglomeration and its importance in parallel programming?

Q2.

a. Explain Foster's design methodology for parallel execution and its advantages over pipelining?

b. Explain various classifications based on memory arrangement?

[2.5 x 2 Marks]

Q3. Explain various ways to parallelize a program with data dependency using OPENMPI with examples? [5 Marks]

Q4. Design a parallel MPI program to find the frequency of set of numbers and print them at master where we have 50000 elements using 5 processors including master? Find the speedup over sequential process? [5 Marks]

Roll No:.....

Q5. Design a parallel in-order parallel selection sort using OPENMP for 2000 elements and 8 threads? With condition to restrict parallel threading if the problem size is less than 200 elements? [5 Marks]

```
#include <stdio.h>
 int main()
{  int array[100], n, c, d, position, swap;
   printf("Enter number of elements\n");
  scanf("%d", &n);
   printf("Enter %d integers\n", n);
   for ( c = 0 ; c < n ; c++ )
    scanf("%d", &array[c]);
   for ( c = 0 ; c < ( n - 1 ) ; c++ )
   {     position = c;
     for ( d = c + 1 ; d < n ; d++ )
     {       if ( array[position] > array[d] )
          position = d;
     }
     if ( position != c )
     {  swap = array[c];
       array[c] = array[position];
       array[position] = swap;
     } }
  printf("Sorted list in ascending order:\n");
   for ( c = 0 ; c < n ; c++ )
    printf("%d\n", array[c]);

  return 0;
}
```

a. Using sections and critical section.

b. Using openmp schedule.

Find the speedup over sequential?