**DEPARTMENT OF INFORMATION TECHNOLOGY, NITK SURATHKAL**

**Parallel Programming**

**LAB 3 ------------14th January 2021**

**Note: Observe the results of each program, take the screenshot of the result and upload it in the Moodle.**

**parallel**

Forms a team of threads and starts parallel execution.

**#pragma omp parallel** *[clause[ [, ]clause] ...]*

*structured-block*

*clause*:

**if**(*scalar-expression*)

**num\_threads**(*integer-expression*)

**default**(**shared** | **none**)

**private**(*list*)

**firstprivate**(*list*)

**shared**(*list*)

**copyin**(*list*)

**reduction**(*reduction-identifier***:** *list*)

**---------------------------------------------------------------------------------------------------------------------**

**Note: Put screenshot of result for every output. Write few lines of observation wherever necessary.**

**Total marks = 10**

**1. Write a C/C++ simple program to display hello world by each thread.**

/\*hello\_omp.c\*/

#include<stdio.h>

#include<omp.h>

int main(){

**#pragma omp parallel num\_threads(4)**

{

int tid=omp\_get\_thread\_num();

printf("Hello world from thread=%d\n",tid);

}

}

**What is the output of this program? Change the number of threads to 8 observe the result. [1+1=2 marks]**

**Execute the program as follows:**

$gcc –o hello –fopenmp hello\_omp.c

$./hello

**2. Write a C/C++ simple program to display hello world by each thread and use *if()* clause**

#include<stdio.h>

#include<omp.h>

int main(){

int p=0;

**#pragma omp parallel if(p==1) num\_threads(4)**

{

int tid=omp\_get\_thread\_num();

printf("Hello world from thread=%d\n",tid);

}

}

**What is the output of this program? Change the value of p to 1 and observe the result.**

**Mention both result and write few lines about your observation. [1+1=2 marks]**

**3. Write a program to demonstrate private() firstprivate() and shared() clauses with parallel directive.**

#include<stdio.h>

#include<omp.h>

int main(){

int a=10;

//default(shared/none) private(list) firstprivate(list) shared(list)

#pragma omp parallel num\_threads(4) **private(a)**

{

a=a+20;

int tid=omp\_get\_thread\_num();

printf("Hello world from thread=%d value of a=%d\n",tid,a);

}

printf("After parallel loop a=%d\n", a);

}

**Execute the program. Observe and mention your observation for private(a), firstprivate(a), shared(a) [3 marks]**

**4. Write a C/C++ program to add elements of two arrays; a[i]=b[i]+c[i];**

#include<stdio.h>

#include<omp.h>

int main(){

int N=20, a[20], b[20], c[20];

int low, high, i;

// initialize array elements

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

{

b[i]=i;

c[i]=i\*2;

}

**#pragma omp parallel num\_threads(4) private(low, high)**

{

int tid=omp\_get\_thread\_num();

int numthreads=omp\_get\_num\_threads();

//distribute the work among threads

low = N\*tid/numthreads;

high=N\*(tid+1)/numthreads;

printf(" thread=%d low=%d high=%d \n", tid, low, high);

//perform operation

for (i=low; i<high;i++)

{

a[i]=b[i]+c[i];

printf("addition from thread=%d value of a[%d]= %d \t b[%d]=%d \t c[%d]=%d\n",tid,i,a[i],i,b[i],i,c[i]);

}

}

}

**Mention the result and write your observation. [1 mark]**

**5. Write a C/C++ program to calculate the sum of all the elements in a array. Assume array size =20 and number of threads = 04. [2 marks]**