

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA SURATHKAL
DEPARTMENT OF INFORMATION TECHNOLOGY
IT 301 Parallel Computing LAB 3
10th August 2021
Faculty: Dr. Geetha V

NAME: BHUVANESWAR DHARMASIVAM

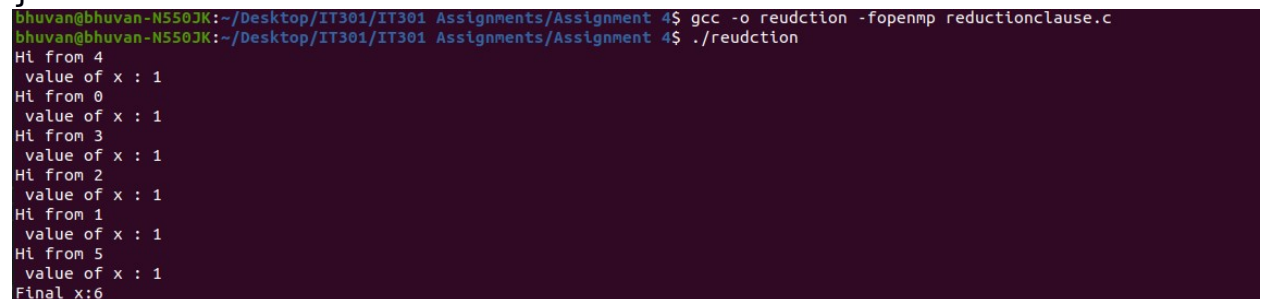
ROLL NO: 191IT107

Execute following programs and put screen shots of the output. Write analysis of the result before uploading in IRIS as a single pdf file. **For programming exercises, write the code and also attach screenshot of the results.**

Total Marks : 10

1. Demonstration of reduction clause in parallel directive. Write your observation. [2 marks]

```
#include<stdio.h>
#include<omp.h>
void main()
{
int x=0;
#pragma omp parallel num_threads(6) reduction(+:x)
{
int id=omp_get_thread_num();
int threads=omp_get_num_threads();
x=x+1;
printf("Hi from %d\n value of x : %d\n",id,x);
}
printf("Final x:%d\n",x);
}
```



```
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ gcc -o reudction -fopenmp reductionclause.c
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ ./reudction
Hi from 4
  value of x : 1
Hi from 0
  value of x : 1
Hi from 3
  value of x : 1
Hi from 2
  value of x : 1
Hi from 1
  value of x : 1
Hi from 5
  value of x : 1
Final x:6
```

Observation: The reduction operator is '+' so at each thread 1 is added and final sum would be 6.

2. Demonstration of lastprivate(). Write your observation [2 marks]

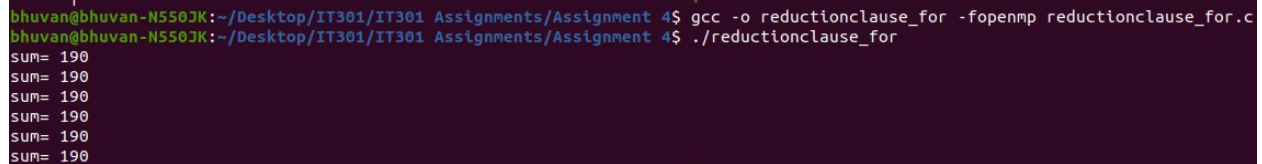
```
#include<stdio.h>
#include<omp.h>
void main()
{ int x=0,i,n;
printf("Enter the value of n");
scanf("%d",&n);
#pragma omp parallel
{
int id=omp_get_thread_num();
#pragma omp for lastprivate(i)
for(i=0;i<n;i++)
{
printf("Thread %d: value of i : %d\n",id,i);
x=x+i;
printf("Thread %d: x is %d\n",id,x);
}
}
printf("x is %d\n",x);
printf("i IS %d\n",i);
}
```

```
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ gcc -o lastpvt -fopenmp lastprivate.c
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ ./lastpvt
Enter the value of n6
Thread 0: value of i : 0
Thread 0: x is 0
Thread 3: value of i : 3
Thread 3: x is 3
Thread 5: value of i : 5
Thread 5: x is 8
Thread 4: value of i : 4
Thread 4: x is 12
Thread 1: value of i : 1
Thread 1: x is 13
Thread 2: value of i : 2
Thread 2: x is 15
x is 15
i IS 6
```

Explanation: The last value inside the parallel block will displayed.

3. Demonstration of reduction clause with 'for' [2 marks]

```
#include<stdio.h>
#include<omp.h>
void main(void)
{
int n=20,dsum=0,tid,a[20],sum=0;
for(i=0;i<n;i++)
{
a[i]=i;
dsum=dsum+i;
}
#pragma omp parallel num_threads(6)
{
int tid=omp_get_thread_num();
#pragma omp for private(i) schedule(static,5) reduction(+,sum)
for(i=0;i<n;i++)
sum=sum+a[i];
}
printf("sum= %d\n",sum);
}
return 0;
}
```



A terminal window showing the compilation and execution of the program. The user is at a prompt in the directory ~/Desktop/IT301/IT301 Assignments/Assignment 4. They compile the program with gcc -o reductionclause_for -fopenmp reductionclause_for.c. Then they run ./reductionclause_for. The output shows six lines of 'sum= 190', indicating that the program was executed six times, likely due to the parallel nature of the code and the way the terminal captures output.

```
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ gcc -o reductionclause_for -fopenmp reductionclause_for.c
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ ./reductionclause_for
sum= 190
sum= 190
sum= 190
sum= 190
sum= 190
sum= 190
```

Observation: The reduction operator is '+' so at each thread each array element is added and final sum would be 190.

3. Programming exercise [4Marks]

Write a parallel program to find the minimum number in a given array. Use 'for' directive for the same along with reduction clause. Write code, execution results and your observation.

CODE:

```
1  #include<stdio.h>
2  #include<omp.h>
3
4  void main(void)
5  {
6      int i,tid,minimum=__INT_MAX__;
7
8      int a[10] = { 10, 9, 71, 101, 3, 44, 12, 78, 34, 23};
9
10     #pragma omp parallel num_threads(6)
11     {
12         int tid=omp_get_thread_num();
13
14         #pragma omp for private(i) reduction(min:minimum)
15         for(i=0;i<10;i++)
16         {
17             if (a[i]<minimum)
18             {
19                 minimum=a[i];
20             }
21         }
22         printf("Minimum= %d\n",minimum);
23     }
24 }
```

RESULT:

```
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ gcc -o min -fopenmp min_array.c
bhuvan@bhuvan-N550JK:~/Desktop/IT301/IT301 Assignments/Assignment 4$ ./min
Minimum= 3
Minimum= 3
Minimum= 3
Minimum= 3
Minimum= 3
Minimum= 3
```

Observation: The reduction operator is 'MIN' so at each thread array elements are compared to find minimum element.