CSE4001 - Parallel and Distributed Computing

Lab 21+22

Lab Task2

Submitted by: Alokam Nikhitha

Reg No:19BCE2555

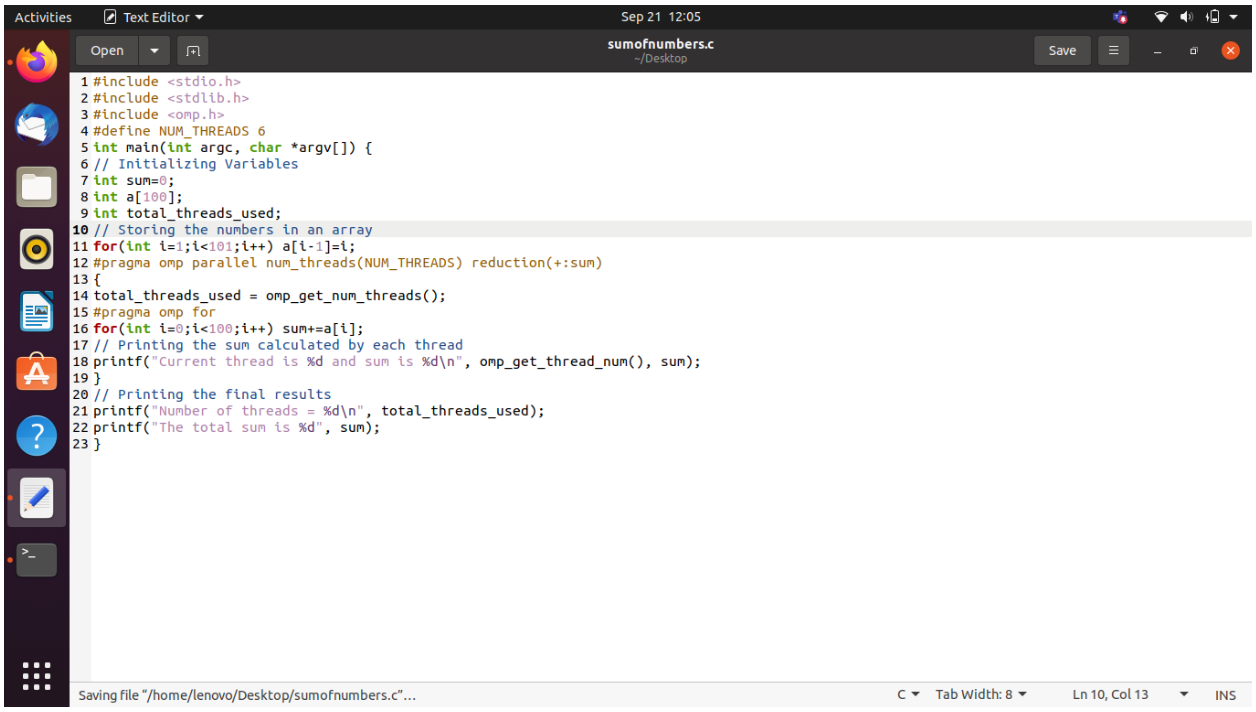
Ques:

Create an OpenMP program that uses the work sharing directive to add all the numbers between 1 and 100. WorkSharing directives simplify and effectively split normally serial tasks into fast parallel sections of code. To declare the loop as work sharing, use the #pragma omp.

CODE:

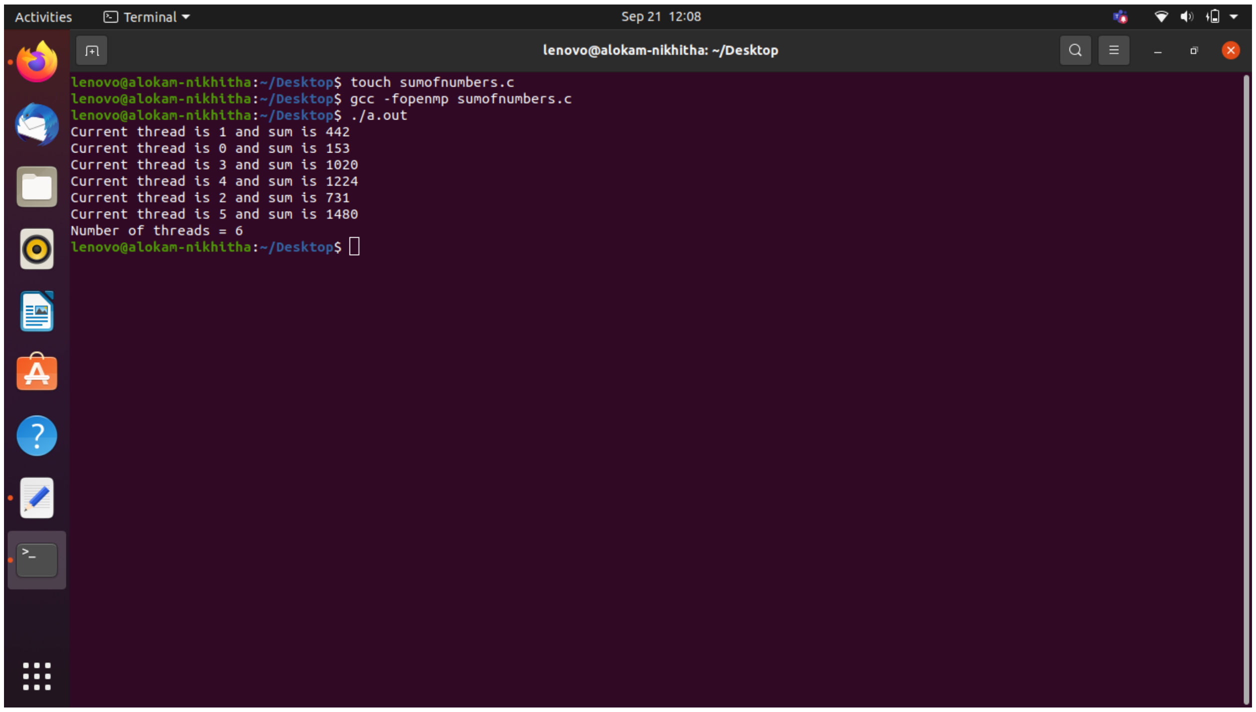
```
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#define NUM_THREADS 6
int main(int argc, char *argv[]) {
// Initializing Variables
int sum=0;
int a[100];
int total_threads_used;
// Storing the numbers in an array
for(int i=1;i<101;i++) a[i-1]=i;
#pragma omp parallel num_threads(NUM_THREADS)
reduction(+:sum)
{
total_threads_used = omp_get_num_threads();
#pragma omp for
for(int i=0;i<100;i++) sum+=a[i];
// Printing the sum calculated by each thread
printf("Current thread is %d and sum is %d\n",
omp_get_thread_num(), sum);
}
// Printing the final results
printf("Number of threads = %d\n", total_threads_used);
printf("The total sum is %d", sum);
}
```

CODE :



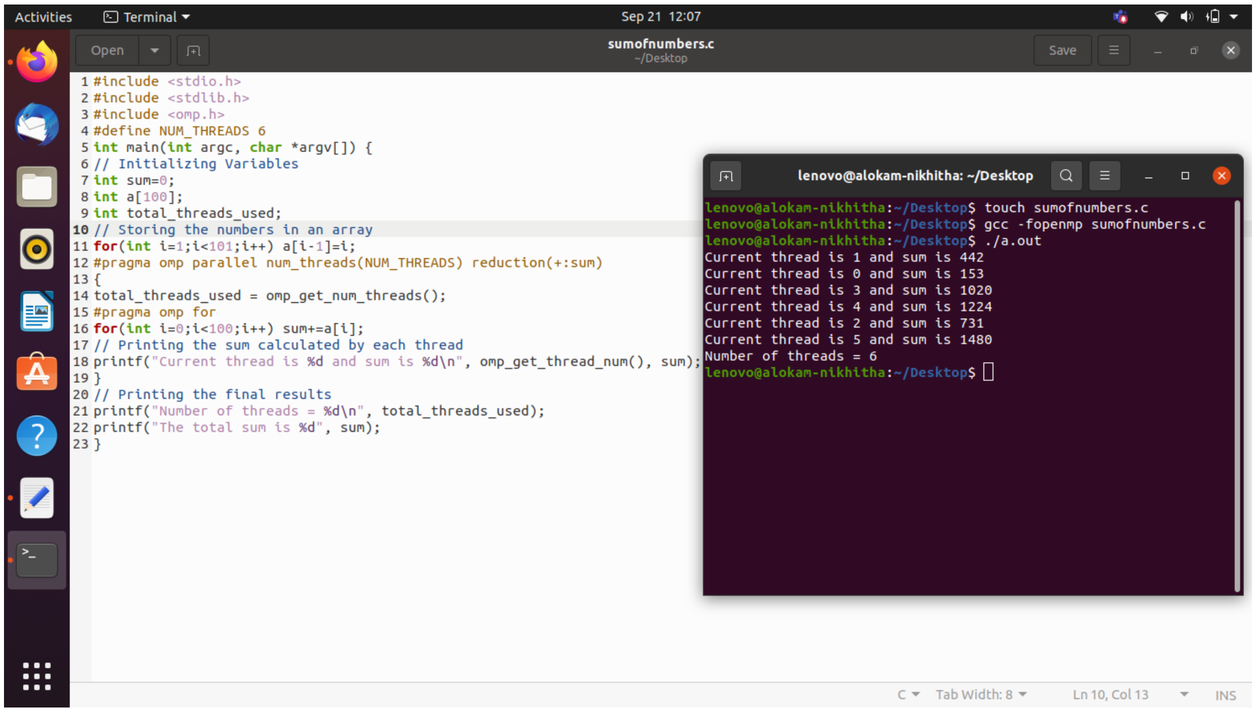
```
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#define NUM_THREADS 6
int main(int argc, char *argv[]) {
// Initializing Variables
int sum=0;
int a[100];
int total_threads_used;
// Storing the numbers in an array
for(int i=1;i<101;i++) a[i-1]=i;
#pragma omp parallel num_threads(NUM_THREADS) reduction(+:sum)
{
total_threads_used = omp_get_num_threads();
#pragma omp for
for(int i=0;i<100;i++) sum+=a[i];
// Printing the sum calculated by each thread
printf("Current thread is %d and sum is %d\n", omp_get_thread_num(), sum);
}
// Printing the final results
printf("Number of threads = %d\n", total_threads_used);
printf("The total sum is %d", sum);
}
```

OUTPUT:



```
lenovo@alokam-nikhitha:~/Desktop$ touch sumofnumbers.c
lenovo@alokam-nikhitha:~/Desktop$ gcc -fopenmp sumofnumbers.c
lenovo@alokam-nikhitha:~/Desktop$ ./a.out
Current thread is 1 and sum is 442
Current thread is 0 and sum is 153
Current thread is 3 and sum is 1020
Current thread is 4 and sum is 1224
Current thread is 2 and sum is 731
Current thread is 5 and sum is 1480
Number of threads = 6
lenovo@alokam-nikhitha:~/Desktop$
```

CODE WITH OUTPUT:



```c
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#define NUM_THREADS 6
int main(int argc, char *argv[]) {
// Initializing Variables
int sum=0;
int a[100];
int total_threads_used;
// Storing the numbers in an array
for(int i=1;i<101;i++) a[i-1]=i;
#pragma omp parallel num_threads(NUM_THREADS) reduction(+:sum)
{
total_threads_used = omp_get_num_threads();
#pragma omp for
for(int i=0;i<100;i++) sum+=a[i];
// Printing the sum calculated by each thread
printf("Current thread is %d and sum is %d\n", omp_get_thread_num(), sum);
}
// Printing the final results
printf("Number of threads = %d\n", total_threads_used);
printf("The total sum is %d", sum);
}
```

```
lenovo@alokam-nikhitha:~/Desktop$ touch sumofnumbers.c
lenovo@alokam-nikhitha:~/Desktop$ gcc -fopenmp sumofnumbers.c
lenovo@alokam-nikhitha:~/Desktop$ ./a.out
Current thread is 1 and sum is 442
Current thread is 0 and sum is 153
Current thread is 3 and sum is 1020
Current thread is 4 and sum is 1224
Current thread is 2 and sum is 731
Current thread is 5 and sum is 1480
Number of threads = 6
lenovo@alokam-nikhitha:~/Desktop$
```