**NAME:ARYAMAN MISHRA**

**REGISTRATION NUMBER:19BCE1027**

**Static**

***Q1 Find the sum of elements of 2 arrays using parallel for.***

**Code**

#include<stdio.h>

#include<omp.h>

void main()

{

int a[10],b[10],I,sum[10];

for(int i=0;i<10;i++){

scanf("%d",&a[i]);

scanf("%d",&b[i]);

}

#pragma omp parallel for schedule(static,1)

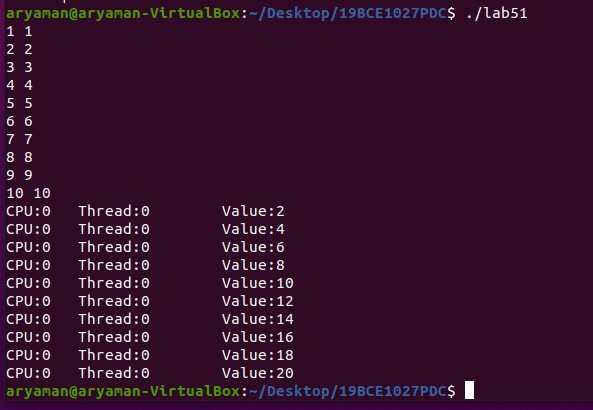
for(int i=0;i<10;i++){

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

printf("CPU:%d\tThread:%d\tValue:%d\n",sched\_getcpu(),omp\_get\_thread\_num(),sum[i]);

}

}



**Dynamic**

#include<stdio.h>

#include<omp.h>

void main()

{

int a[10],b[10],I,sum[10];

for(int i=0;i<10;i++){

scanf("%d",&a[i]);

scanf("%d",&b[i]);

}

#pragma omp parallel for schedule(dynamic,1)

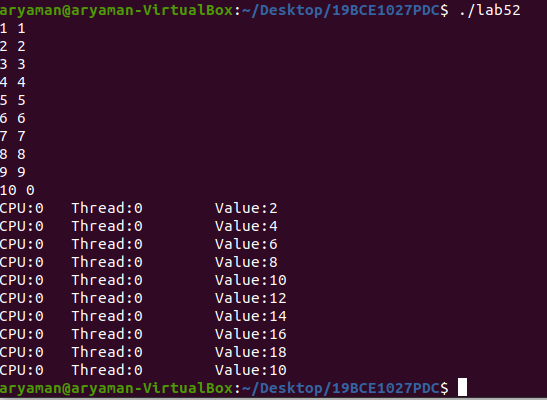
for(int i=0;i<10;i++){

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

printf("CPU:%d\tThread:%d\tValue:%d\n",sched\_getcpu(),omp\_get\_thread\_num(),sum[i]);

}

}



**Exercise 2 Sample Problem on firstprivate**

**Code-static scheduling**

#include <stdio.h>

#include <omp.h>

int main (void)

{

int i = 10;

int j;

omp\_set\_num\_threads(3);

#pragma omp parallel for firstprivate(i) schedule(static,2)

for(j=0;j<6;j++){

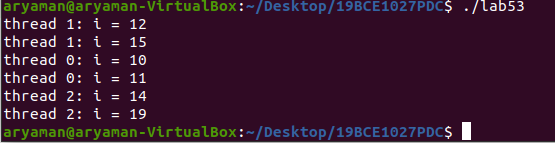
i = j+i;

printf("thread %d: i = %d\n", omp\_get\_thread\_num(), i);

}

}

Output



**Dynamic**

CODE

#include <stdio.h>

#include <omp.h>

int main (void)

{

int i = 10;

int j;

omp\_set\_num\_threads(3);

#pragma omp parallel for firstprivate(i) schedule(dynamic,2)

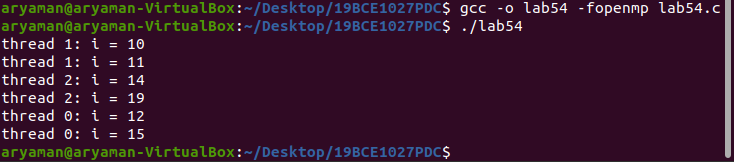
for(j=0;j<6;j++){

i = j+i;

printf("thread %d: i = %d\n", omp\_get\_thread\_num(), i);

}

}

OUTPUT

**Exercise 3 (using Parallel For)**

***Addition of factors of a number < n***

**Static**

Code

#include<stdio.h>

#include<omp.h>

int main(void){

int a[]={1,2,3,4,5,6,7,8,9,10};

int n;

printf("Enter value of n: ");

scanf("%d",&n);

int b[n];

for(int i = 0; i < n; i++){

b[i] = 0;

}

int j,k;

int c = 2;

omp\_set\_num\_threads(6);

#pragma omp parallel for schedule(static,2)

for(k = 0; k < n; k++){

for(j = 0; j < 10; j++)

{

if(a[j] % c == 0){

b[k] += a[j];

}

}

printf("Sum of Factors of %d -> b[%d] = %d\n",c,k,b[k]);

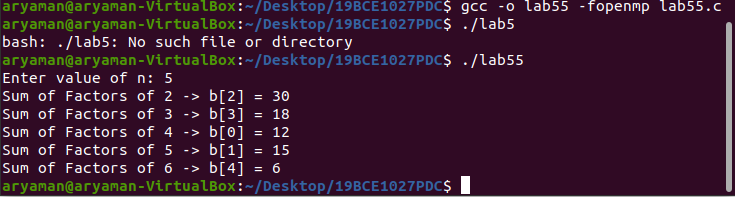
c++;

}

return 0;

}

**output**



**Dynamic**

Code

#include<stdio.h>

#include<omp.h>

int main(void){

int a[]={1,2,3,4,5,6,7,8,9,10};

int n;

printf("Enter value of n: ");

scanf("%d",&n);

int b[n];

for(int i = 0; i < n; i++){

b[i] = 0;

}

int j,k;

int c = 2;

omp\_set\_num\_threads(6);

#pragma omp parallel for schedule(dynamic,2)

for(k = 0; k < n; k++){

for(j = 0; j < 10; j++)

{

if(a[j] % c == 0){

b[k] += a[j];

}

}

printf("Sum of Factors of %d -> b[%d] = %d\n",c,k,b[k]);

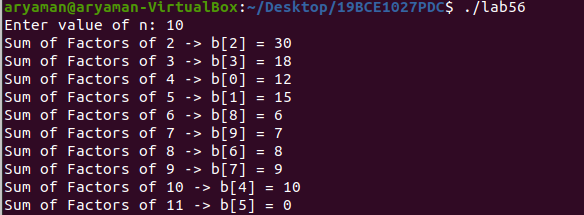
c++;

}

return 0;

}

**OUTPUT**



**Guided**

Code

#include<stdio.h>

#include<omp.h>

int main(void){

int a[]={1,2,3,4,5,6,7,8,9,10};

int n;

printf("Enter value of n: ");

scanf("%d",&n);

int b[n];

for(int i = 0; i < n; i++){

b[i] = 0;

}

int j,k;

int c = 2;

omp\_set\_num\_threads(6);

#pragma omp parallel for schedule(guided,2)

for(k = 0; k < n; k++){

for(j = 0; j < 10; j++)

{

if(a[j] % c == 0){

b[k] += a[j];

}

}

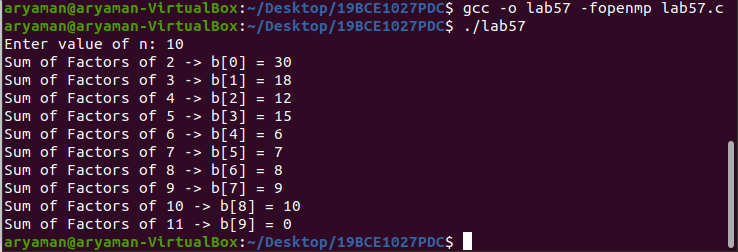
printf("Sum of Factors of %d -> b[%d] = %d\n",c,k,b[k]);

c++;

}

return 0;

}



**Exercise 4**

**addition of odd and even nos**

code

#include<stdio.h>

#include<omp.h>

void main()

{

int a[]={1,2,3,4,5,6,7,8,9,10};

int i,k;

int oddsum=0;

int evensum=0;

omp\_set\_num\_threads(2);

#pragma omp parallel for schedule(static,2)

for(i=0;i<10;i++){

if(a[i]%2==0){

evensum+=a[i];

printf("Thread:%d\t a[i]:%d \t evensum:%d\n",omp\_get\_thread\_num(),a[i],evensum);

}

else{

oddsum+=a[i];

printf("Thread:%d\t a[i]:%d \t oddsum:%d\n",omp\_get\_thread\_num(),a[i],oddsum);

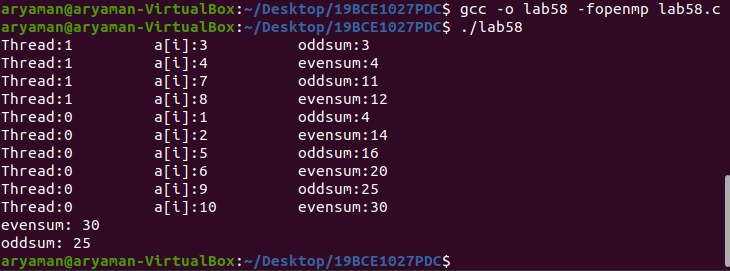
}

}

printf("evensum: %d\n",evensum);

printf("oddsum: %d\n",oddsum);

}



**Dynamic**

**Code**

#include<stdio.h>

#include<omp.h>

void main()

{

int a[]={1,2,3,4,5,6,7,8,9,10};

int i,k;

int oddsum=0;

int evensum=0;

omp\_set\_num\_threads(2);

#pragma omp parallel for schedule(dynamic,2)

for(i=0;i<10;i++){

if(a[i]%2==0){

evensum+=a[i];

printf("Thread:%d\t a[i]:%d \t evensum:%d\n",omp\_get\_thread\_num(),a[i],evensum);

}

else{

oddsum+=a[i];

printf("Thread:%d\t a[i]:%d \t oddsum:%d\n",omp\_get\_thread\_num(),a[i],oddsum);

}

}

printf("evensum: %d\n",evensum);

printf("oddsum: %d\n",oddsum);

}

**OUTPUT**

