**NAME – GAUTAM CHANDRA SAHA**

**REG NO – 201900099**

**DATE – 09/02/2022**

**LAB 2**

**Q1)  Fill an array of size 100 with 1. Find sum of the array.**

/\*

AUTHOR : GAUTAM CHANDRA SAHA

DATE & TIME: Wed, February 09,2022 AT 10:50

DESCRIPTION:

\*/

#include <iostream>

#include <cmath>

#include <vector>

#include <omp.h>

int main(){

double sum = 0;

int size = 100;

std::vector<int> arr(size, 1); //array of size 100

#pragma omp parallel for shared(sum)

for (int i = 0; i < size; i++)

#pragma omp critical

sum += arr[i];

std::cout << "Sum: " << sum;

return 0;

}

**OUTPUT**

**Graphical user interface, text, application

Description automatically generated with medium confidence**

**Q2) Fill the array with different numbers and find sum.**

/\*

AUTHOR : GAUTAM CHANDRA SAHA

DATE & TIME: Wed, February 09,2022 AT 11:08

DESCRIPTION:

\*/

#include <iostream>

#include <cmath>

#include <vector>

#include <omp.h>

int main(){

double sum = 0;

int size = 100;

std::vector<int> arr(size, 0); //array of size 100

for (int i = 0; i < size / 4; i++)

arr[i] = 1;

for (int i = size / 4; i < size / 2; i++)

arr[i] = 2;

for (int i = size / 2; i < 3 \* size / 4; i++)

arr[i] = 3;

for (int i = 3 \* size / 4; i < size; i++)

arr[i] = 4;

#pragma omp parallel for shared(sum)

for (int i = 0; i < size; i++)

#pragma omp critical

sum += arr[i];

std::cout << "Sum: " << sum;

return 0;

}

**OUTPUT**

**Graphical user interface, text, application

Description automatically generated with medium confidence**

**Q3) Take an array of size= 100000 and fill with numbers. Then find sum. Also compare the same in serial environment using start time and end time.**

/\*

AUTHOR : GAUTAM CHANDRA SAHA

DATE & TIME: Wed, February 09,2022 AT 11:32

DESCRIPTION:

\*/

#include <iostream>

#include <cmath>

#include <vector>

#include <omp.h>

double parallel\_exec(std::vector<int> &arr, double sum)

{

double itime, ftime, exec\_time;

itime = omp\_get\_wtime();

#pragma omp parallel for shared(sum)

for (int i = 0; i < arr.size(); i++)

#pragma omp critical

sum += arr[i];

// Required code for which execution time needs to be computed

ftime = omp\_get\_wtime();

exec\_time = ftime - itime;

std::cout << "Execution time of parallel environment: " << exec\_time;

return exec\_time;

}

double serial\_exec(std::vector<int> &arr, double sum)

{

double itime, ftime, exec\_time;

itime = omp\_get\_wtime();

for (int i = 0; i < arr.size(); i++)

sum += arr[i];

// Required code for which execution time needs to be computed

ftime = omp\_get\_wtime();

exec\_time = ftime - itime;

std::cout << "Execution time of serial environment: " << exec\_time;

return exec\_time;

}

int main(){

double sum = 0;

int size = 100000;

std::vector<int> arr(size, 2);

double pe = parallel\_exec(arr, sum);

std::cout << std::endl;

double se = serial\_exec(arr, sum);

std::cout << "\nThe differences of execution times in both environment: " << pe - se << std::endl;

return 0;

}

**OUTPUT**

**Text

Description automatically generated**

**Q4) Use task 3 and find sum (core wise).**

/\*

AUTHOR : GAUTAM CHANDRA SAHA

DATE & TIME: Wed, February 09,2022 AT 11:32

DESCRIPTION:

\*/

#include <iostream>

#include <cmath>

#include <vector>

#include <omp.h>

void parallel\_exec(std::vector<int> &arr)

{

int count = 0;

#pragma omp parallel

{

double sum = 0;

for (int i = 0; i < arr.size(); i++)

#pragma omp critical

sum += arr[i];

std::cout << "sum for core " << ++count << ": " << sum << std::endl;

}

}

int main(){

int size = 100000;

std::vector<int> arr(size, 3);

parallel\_exec(arr);

return 0;

}

**OUTPUT**

**Text

Description automatically generated**