**Name: Varun Menon**

**Reg No: 19BCE1438**

**Course Code: CSE4001**

**Faculty: Dr. Harini S**

**Lab Experiment 5**

1. Example Program -- Critical

Code:

#include <stdio.h>

#include <omp.h>

int main()

{

    int x = 0;

    #pragma omp parallel

    {

        #pragma omp critical

        {

            x = x + 1;

        }

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

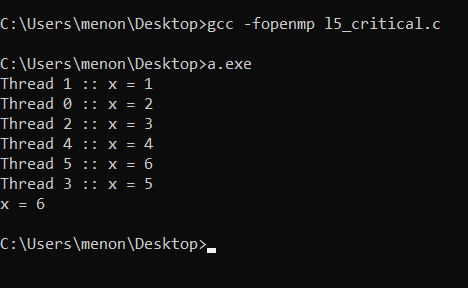
    }

    printf("x = %d\n", x);

    return 0;

}

Output:



1. Example Program -- Single

Code:

#include <stdio.h>

#include <omp.h>

int main()

{

    int x = 0;

    #pragma omp parallel

    {

        #pragma omp single

        {

            x = x + 1;

        }

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

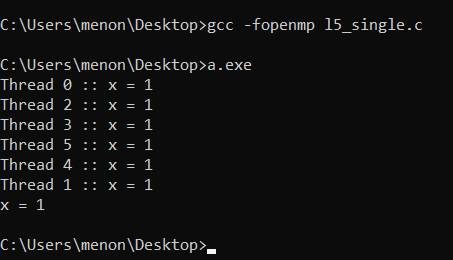
    }

    printf("x = %d\n", x);

    return 0;

}

Output:



1. Example Program -- Master

Code:

#include <stdio.h>

#include <omp.h>

int main()

{

    int x = 0;

    #pragma omp parallel

    {

        #pragma omp master

        {

            x = x + 1;

        }

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

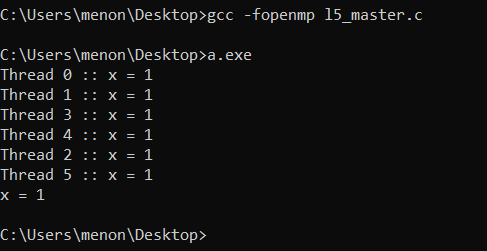
    }

    printf("x = %d\n", x);

    return 0;

}

Output:



1. Example program with x=x+thread\_id for critical, single and master. To prove concept, use one shared variable for each synchronization construct.

Code:

#include <stdio.h>

#include <omp.h>

int main()

{

    int x = 0, y = 0, z = 0;

    #pragma omp parallel shared(x, y, z)

    {

        #pragma omp critical

        {

            x = x + omp\_get\_thread\_num();

        }

        #pragma omp single

        {

            y = y + omp\_get\_thread\_num();

        }

        #pragma omp master

        {

            z = z + omp\_get\_thread\_num();

        }

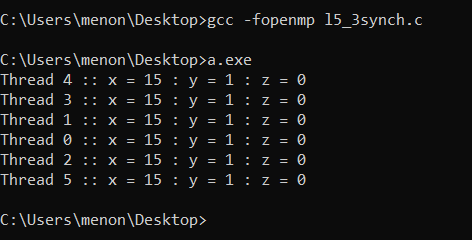
        printf("Thread %d :: x = %d : y = %d : z = %d\n", omp\_get\_thread\_num(), x, y, z);

    }

    return 0;

}

Output:



1. Consider you have to write a program for VIT placement cell where 100 students are placed in 4 companies namely, Amazon, Google, Shell, and Intel. Assume no student is offered more than one placement offer. The program has to do the following tasks in parallel and display the result with thread id.

* Get as input the name, register number, the pay package of students selected for jobs in the particular organization
* Display the total number of students selected in each company.
* Display the average pay package of the 100 students.

Code:

#include <stdio.h>

#include <omp.h>

#include<string.h>

#include<stdlib.h>

int main()

{

    int n = 100;

    char names[n][30];

    char reg\_nos[n][9];

    int packages[n];

    char companies[n][10];

    //char amazon[] = "Amazon", google[] = "Google", shell[] = "Shell", intel[] = "Intel";

    //printf("%s %s %s %s\n", google, amazon, shell, intel);

    printf("Enter name, reg no, pay package, company selected for the %d students (each input on a new line)\n", n);

    for(int i = 0; i < n; i++)

    {

        scanf("%s", names[i]);

        scanf("%s", reg\_nos[i]);

        scanf("%d", &packages[i]);

        scanf("%s", companies[i]);

    }

    int cnt\_amazon = 0, cnt\_google = 0, cnt\_shell = 0, cnt\_intel = 0;

    double average\_pay = 0.0f;

    printf("\n");

    #pragma omp parallel for shared(packages, companies, cnt\_amazon, cnt\_google, cnt\_shell, cnt\_intel, average\_pay)

    for(int i = 0; i < n; i++)

    {

        #pragma omp critical

        if(strcmp("Amazon", companies[i]) == 0)

        {

            printf("Incrementing Amazon: Thread num %d\n", omp\_get\_thread\_num());

            cnt\_amazon++;

        }

        #pragma omp critical

        if(strcmp("Google", companies[i]) == 0)

        {

            printf("Incrementing Google: Thread num %d\n", omp\_get\_thread\_num());

            cnt\_google++;

        }

        #pragma omp critical

        if(strcmp("Shell", companies[i]) == 0)

        {

            printf("Incrementing Shell: Thread num %d\n", omp\_get\_thread\_num());

            cnt\_shell++;

        }

        #pragma omp critical

        if(strcmp("Intel", companies[i]) == 0)

        {

            printf("Incrementing Intel: Thread num %d\n", omp\_get\_thread\_num());

            cnt\_intel++;

        }

        #pragma omp critical

        average\_pay += 1.0f\*packages[i]/n;

    }

    printf("\n");

    printf("Count of students selected - Amazon: %d\n", cnt\_amazon);

    printf("Count of students selected - Google: %d\n", cnt\_google);

    printf("Count of students selected - Shell: %d\n", cnt\_shell);

    printf("Count of students selected - Intel: %d\n", cnt\_intel);

    printf("Average pay package: %f\n", average\_pay);

    return 0;

}

Output:

For the sake of ease of demonstration, we take the number of students as 7 for this output image

