NAME – GAUTAM CHANDRA SAHA
REG NO – 201900099
DATE – 02/02/2022

Q1. Write a menu driven Open-MP program in C to implement the following clauses:

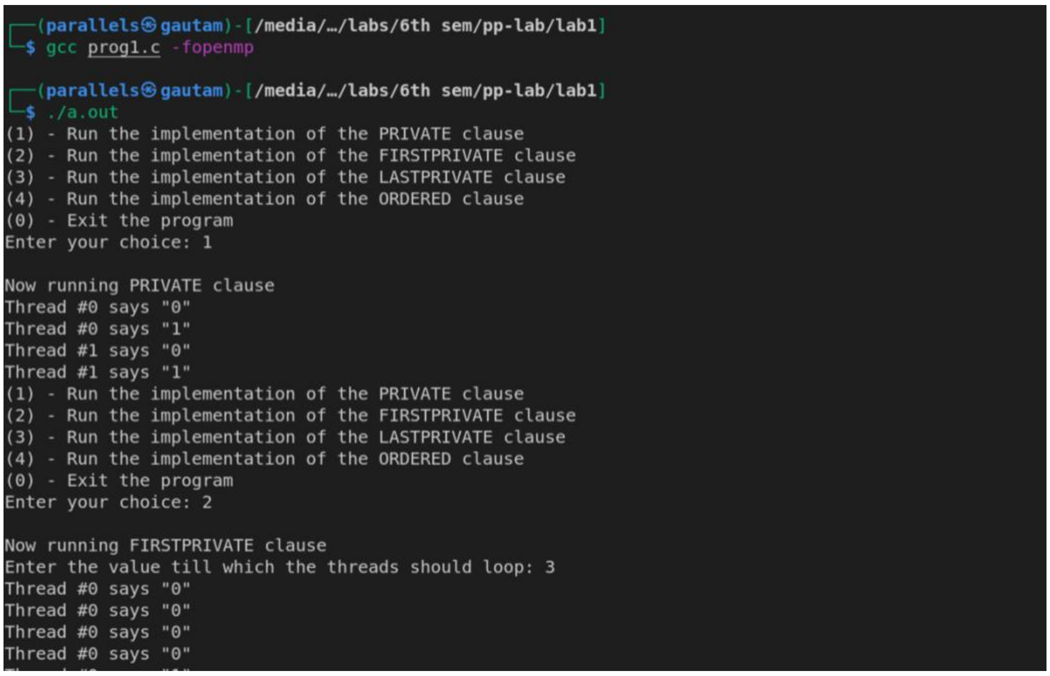
a) PRIVATE b) FIRSTPRIVATE c) LASTPRIVATE d) ORDERED

```c
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
//each thread writes all numbers from 0 to MAX_THREAD
void private_clause(){
    printf("\nNow running PRIVATE clause\n");
    int a, i, thread_id;
#pragma omp parallel for private(a, thread_id)
    for (i = 0; i < omp_get_max_threads(); i++){
        for (a = 0; a < omp_get_max_threads(); a++){
            thread_id = omp_get_thread_num();
            printf("Thread #%d says \"%d\"\n", thread_id, a);
        }
    }
    return;
}
//each thread writes a number between 0 to MAX_THREAD //all threads
collectively write all such numbers unquely
void first_private(){
    printf("\nNow running FIRSTPRIVATE clause\n");
    int i, j, r, thread_id;
    printf("Enter the value till which the threads should loop: ");
    scanf("%d", &r);
    // #pragma omp parallel for firstprivate(r) private(j, thread_id)
    for (i = 0; i < omp_get_max_threads(); i++){
        for (j = 0; j <= r; j++){
            thread_id = omp_get_thread_num();
            printf("Thread #%d says \"%d\"\n", thread_id, i);
        }
    }
    return;
}
//each thread rolls a die
//the number rolled by the last thread is retained
void last_private(){
    printf("\nNow running LASTPRIVATE clause\n");
    int a = 10, r, thread_id, i;
#pragma omp parallel for firstprivate(a) lastprivate(r) private(thread_id)
    for (i = 0; i < omp_get_max_threads(); i++){
        thread_id = omp_get_thread_num();
        srand(thread_id);
```

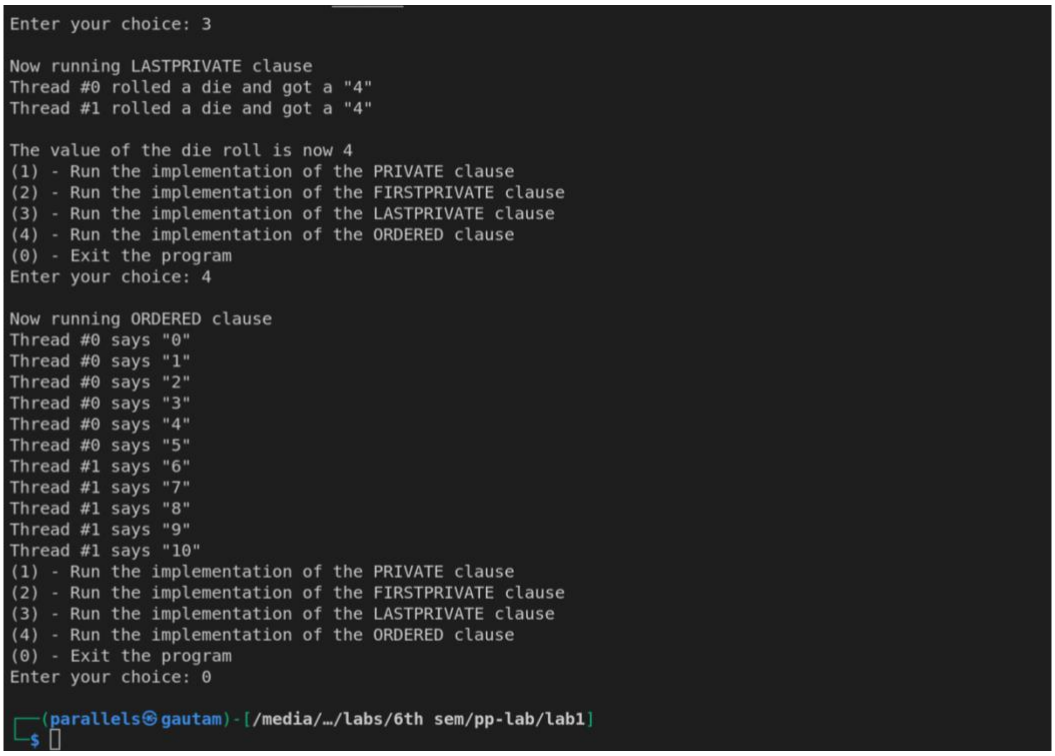
```c
            r = rand() % a + 1;

            printf("Thread #%d rolled a die and got a \"%d\"\n", thread_id, r);
        }
        printf("\nThe value of the die roll is now %d\n", r);
        return;
    }
    //threads iterate over a range, treating it as a serial construct
    void ordered(){
        printf("\nNow running ORDERED clause\n");
        int i, thread_id;
    #pragma omp parallel for ordered private(thread_id)
        for (i = 0; i <= 10; i++){
            thread_id = omp_get_thread_num();
    #pragma omp ordered
            printf("Thread #%d says \"%d\"\n", thread_id, i);
        }
        return;
    }
    //the menu
    int menu(void){
        printf("(1) - Run the implementation of the PRIVATE clause\n");
        printf("(2) - Run the implementation of the FIRSTPRIVATE clause\n");
        printf("(3) - Run the implementation of the LASTPRIVATE clause\n");
        printf("(4) - Run the implementation of the ORDERED clause\n");
        printf("(0) - Exit the program\n");

        printf("Enter your choice: ");
        int choice = 0;
        scanf("%d", &choice);
        switch (choice){
        case 0:
            break;
        case 1:
            private_clause();
            break;
        case 2:
            first_private();
            break;
        case 3:
            last_private();
            break;
        case 4:
            ordered();
            break;
        default:
            printf("\nInvalid choice");
            choice = 5;
            break;
        }
        return choice;
    }
```

```c
int main(void){
    while (menu()){
    }
    return 0;
}
```

OUTPUT



```
┌──(parallels㉿gautam)-[/media/…/labs/6th sem/pp-lab/lab1]
└─$ gcc prog1.c -fopenmp

┌──(parallels㉿gautam)-[/media/…/labs/6th sem/pp-lab/lab1]
└─$ ./a.out
(1) - Run the implementation of the PRIVATE clause
(2) - Run the implementation of the FIRSTPRIVATE clause
(3) - Run the implementation of the LASTPRIVATE clause
(4) - Run the implementation of the ORDERED clause
(0) - Exit the program
Enter your choice: 1

Now running PRIVATE clause
Thread #0 says "0"
Thread #0 says "1"
Thread #1 says "0"
Thread #1 says "1"
(1) - Run the implementation of the PRIVATE clause
(2) - Run the implementation of the FIRSTPRIVATE clause
(3) - Run the implementation of the LASTPRIVATE clause
(4) - Run the implementation of the ORDERED clause
(0) - Exit the program
Enter your choice: 2

Now running FIRSTPRIVATE clause
Enter the value till which the threads should loop: 3
Thread #0 says "0"
Thread #0 says "0"
Thread #0 says "0"
Thread #0 says "0"
```

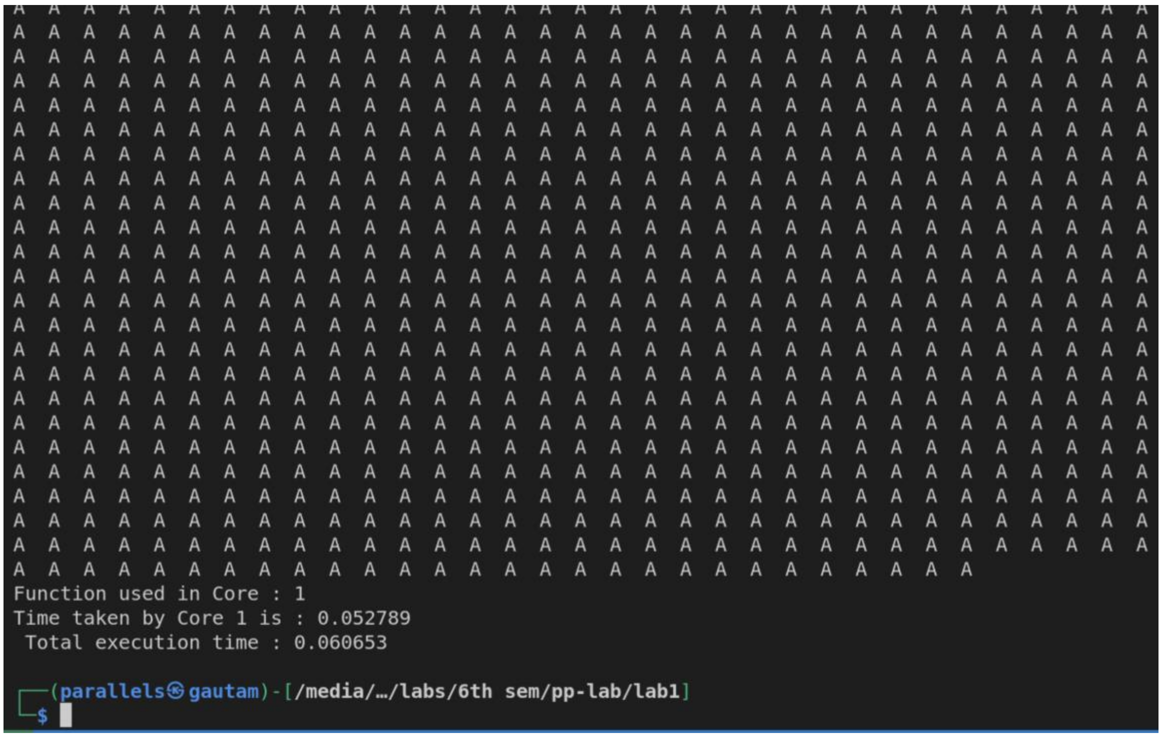


```
Enter your choice: 3

Now running LASTPRIVATE clause
Thread #0 rolled a die and got a "4"
Thread #1 rolled a die and got a "4"

The value of the die roll is now 4
(1) - Run the implementation of the PRIVATE clause
(2) - Run the implementation of the FIRSTPRIVATE clause
(3) - Run the implementation of the LASTPRIVATE clause
(4) - Run the implementation of the ORDERED clause
(0) - Exit the program
Enter your choice: 4

Now running ORDERED clause
Thread #0 says "0"
Thread #0 says "1"
Thread #0 says "2"
Thread #0 says "3"
Thread #0 says "4"
Thread #0 says "5"
Thread #1 says "6"
Thread #1 says "7"
Thread #1 says "8"
Thread #1 says "9"
Thread #1 says "10"
(1) - Run the implementation of the PRIVATE clause
(2) - Run the implementation of the FIRSTPRIVATE clause
(3) - Run the implementation of the LASTPRIVATE clause
(4) - Run the implementation of the ORDERED clause
(0) - Exit the program
Enter your choice: 0

┌──(parallels㉿gautam)-[/media/…/labs/6th sem/pp-lab/lab1]
└─$ 
```

Q2. Run a parallel prog to display/print A for 1 Lakh times using different cores.

```c
#include <stdio.h>
#include <omp.h>

int main(){
  double sum = 0;
#pragma omp parallel
    {

        int i = 0;
while (i < 4){
      double start, end;
      start = omp_get_wtime();

    if (omp_get_thread_num() == i){
        for (i = 0; i < 25000; i++)
                printf(" A ");
    }
 end = omp_get_wtime();

 printf("\nFunction used in Core : %d ", omp_get_thread_num());
 printf("\nTime taken by Core %d is : %f ", omp_get_thread_num(), end - start);
            sum += (end - start);
            i++;
    }
 }
 printf("\n Total execution time : %f ", sum);
return 0;
}
```

OUTPUT

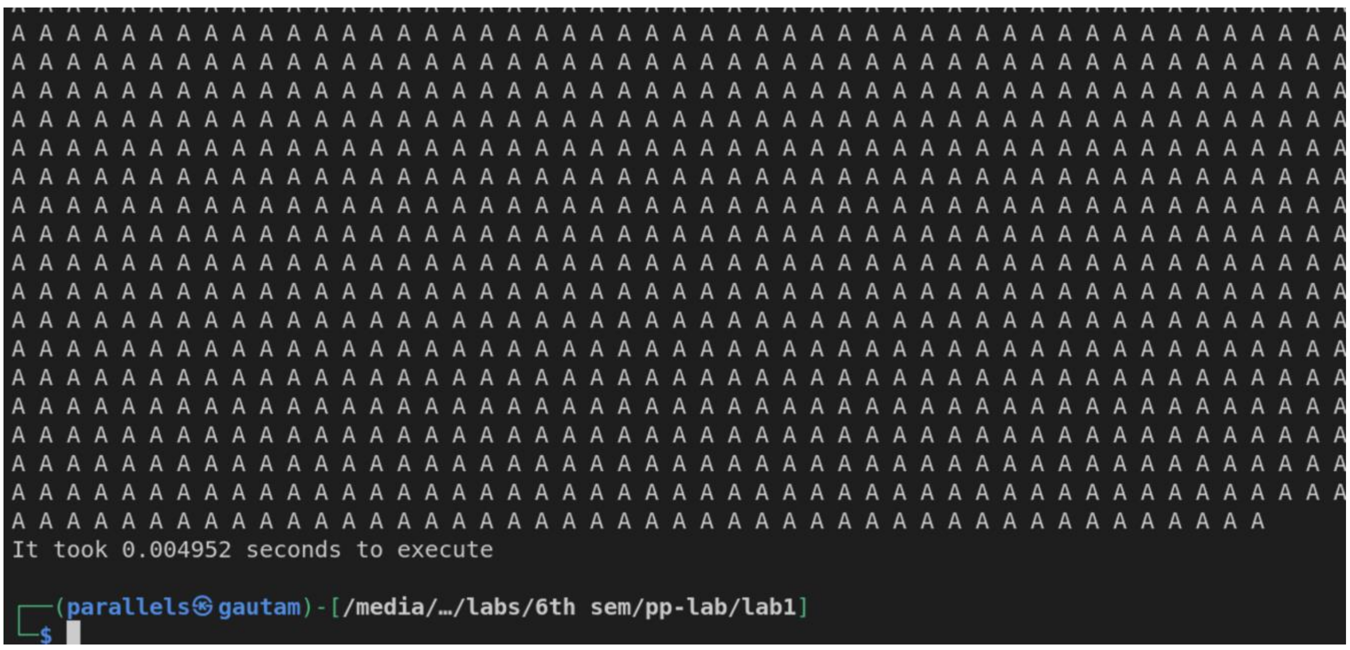


```
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
Function used in Core : 1
Time taken by Core 1 is : 0.052789
 Total execution time : 0.060653

┌──(parallels㉿gautam)-[/media/…/labs/6th sem/pp-lab/lab1]
└─$ 
```

Q3. How to calculate execution time of a C program without omp.h.

```c
#include <stdio.h>
#include <time.h>

int main(){
    clock_t start, end;
    start = clock();
    for (int i = 0; i < 100000; i++)
        printf("A ");

    end = clock();
    double time_taken = ((double)end - start) / CLOCKS_PER_SEC; // in seconds

    printf("\nIt took %f seconds to execute \n", time_taken);
    return 0;
}
```

OUTPUT



```
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
It took 0.004952 seconds to execute

┌──(parallels㉿gautam)-[/media/…/labs/6th sem/pp-lab/lab1]
└─$ 
```