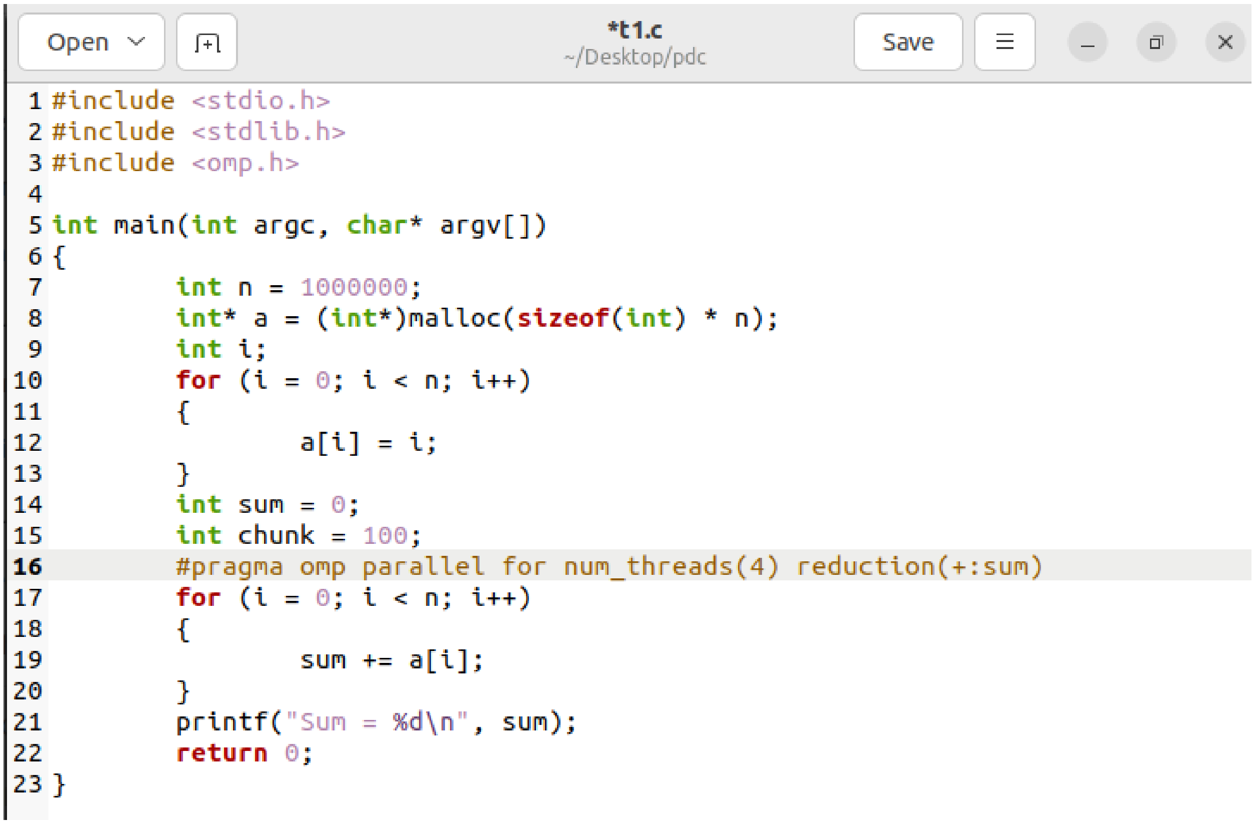
PDC A1

21K4653 Ali Zain

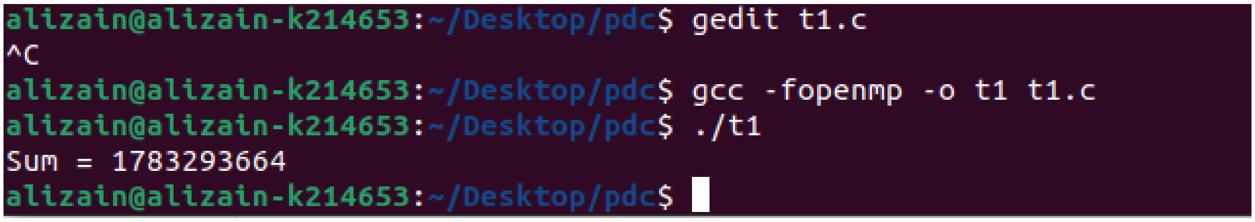
Task #1:

Code:



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

int main(int argc, char* argv[])
{
        int n = 1000000;
        int* a = (int*)malloc(sizeof(int) * n);
        int i;
        for (i = 0; i < n; i++)
        {
                a[i] = i;
        }
        int sum = 0;
        int chunk = 100;
        #pragma omp parallel for num_threads(4) reduction(+:sum)
        for (i = 0; i < n; i++)
        {
                sum += a[i];
        }
        printf("Sum = %d\n", sum);
        return 0;
}
```

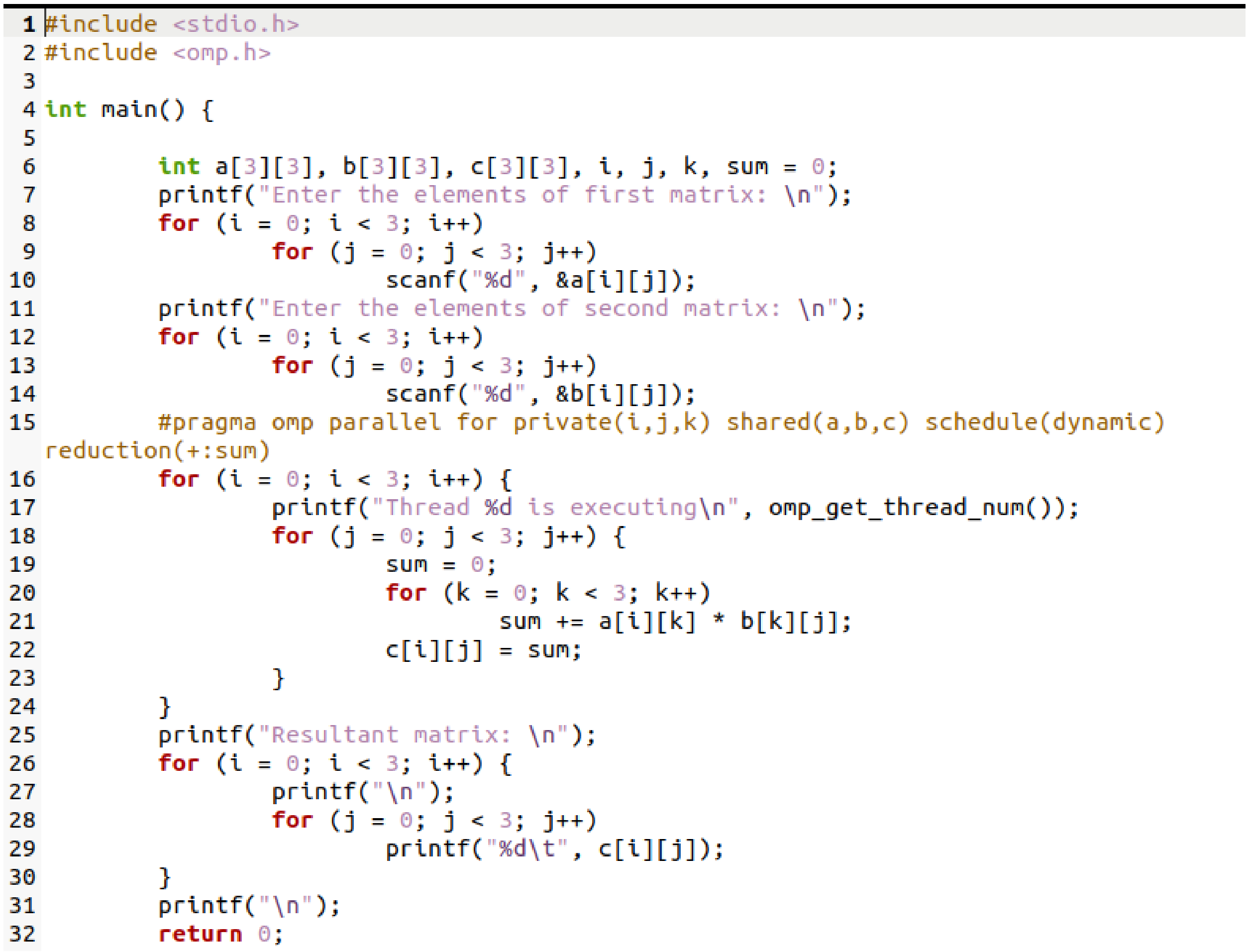
Output:



```
alizain@alizain-k214653:~/Desktop/pdc$ gedit t1.c
^C
alizain@alizain-k214653:~/Desktop/pdc$ gcc -fopenmp -o t1 t1.c
alizain@alizain-k214653:~/Desktop/pdc$ ./t1
Sum = 1783293664
alizain@alizain-k214653:~/Desktop/pdc$ 
```

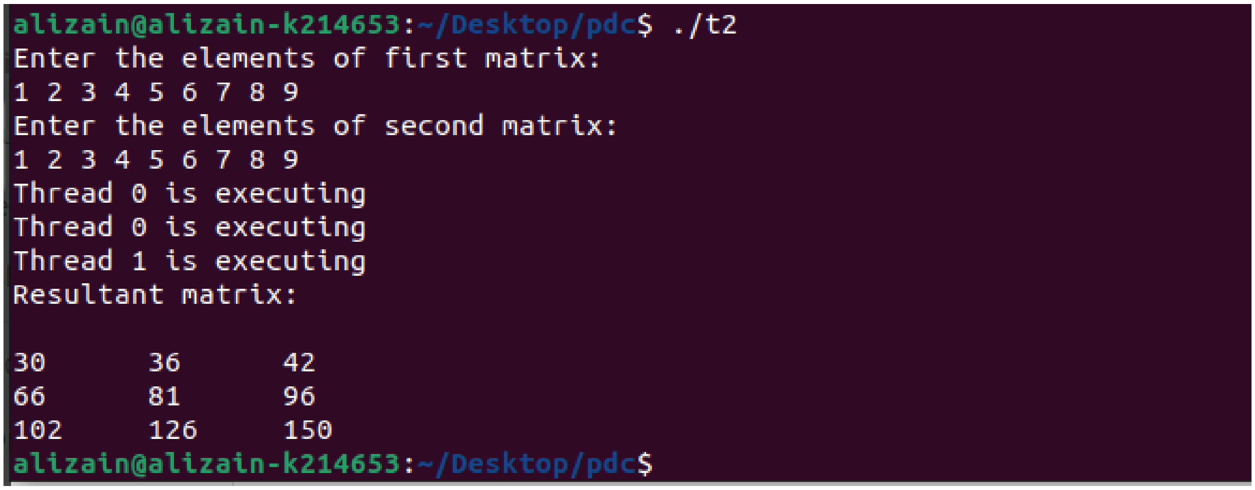
Task #2:

Code:



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

int main() {

        int a[3][3], b[3][3], c[3][3], i, j, k, sum = 0;
        printf("Enter the elements of first matrix: \n");
        for (i = 0; i < 3; i++)
                for (j = 0; j < 3; j++)
                        scanf("%d", &a[i][j]);
        printf("Enter the elements of second matrix: \n");
        for (i = 0; i < 3; i++)
                for (j = 0; j < 3; j++)
                        scanf("%d", &b[i][j]);
        #pragma omp parallel for private(i,j,k) shared(a,b,c) schedule(dynamic) reduction(+:sum)
        for (i = 0; i < 3; i++) {
                printf("Thread %d is executing\n", omp_get_thread_num());
                for (j = 0; j < 3; j++) {
                        sum = 0;
                        for (k = 0; k < 3; k++)
                                sum += a[i][k] * b[k][j];
                        c[i][j] = sum;
                }
        }
        printf("Resultant matrix: \n");
        for (i = 0; i < 3; i++) {
                printf("\n");
                for (j = 0; j < 3; j++)
                        printf("%d\t", c[i][j]);
        }
        printf("\n");
        return 0;
```

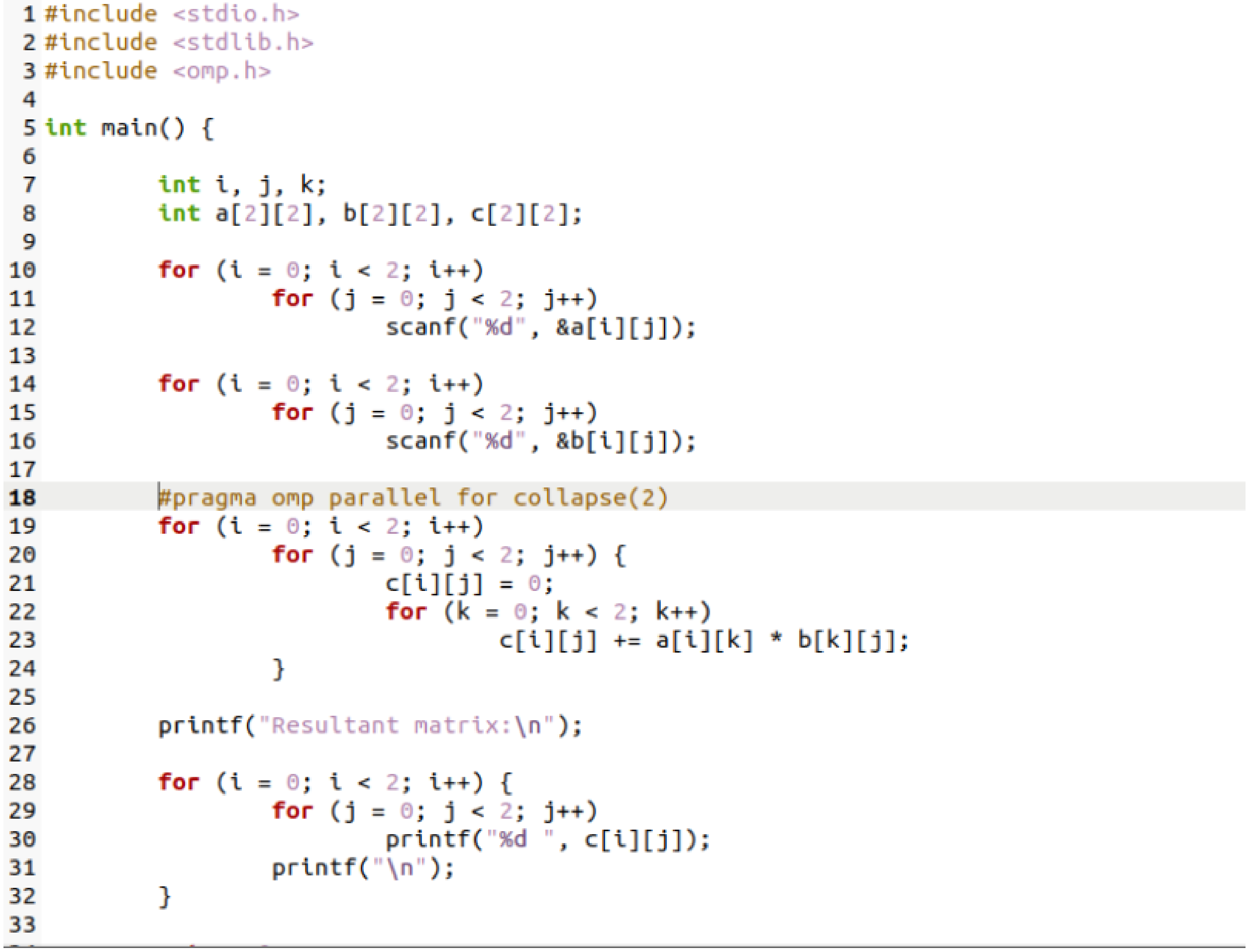
Output:



```
alizain@alizain-k214653:~/Desktop/pdc$ ./t2
Enter the elements of first matrix:
1 2 3 4 5 6 7 8 9
Enter the elements of second matrix:
1 2 3 4 5 6 7 8 9
Thread 0 is executing
Thread 0 is executing
Thread 1 is executing
Resultant matrix:

30      36      42
66      81      96
102     126     150
alizain@alizain-k214653:~/Desktop/pdc$
```

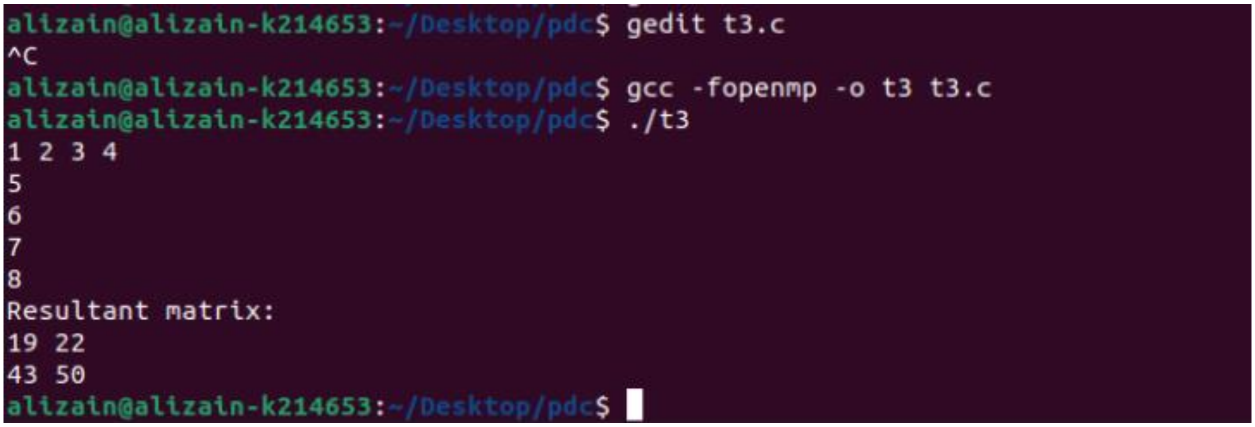
Task #3:

Code:



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

int main() {

        int i, j, k;
        int a[2][2], b[2][2], c[2][2];

        for (i = 0; i < 2; i++)
                for (j = 0; j < 2; j++)
                        scanf("%d", &a[i][j]);

        for (i = 0; i < 2; i++)
                for (j = 0; j < 2; j++)
                        scanf("%d", &b[i][j]);

        #pragma omp parallel for collapse(2)
        for (i = 0; i < 2; i++)
                for (j = 0; j < 2; j++) {
                        c[i][j] = 0;
                        for (k = 0; k < 2; k++)
                                c[i][j] += a[i][k] * b[k][j];
                }

        printf("Resultant matrix:\n");

        for (i = 0; i < 2; i++) {
                for (j = 0; j < 2; j++)
                        printf("%d ", c[i][j]);
                printf("\n");
        }

```

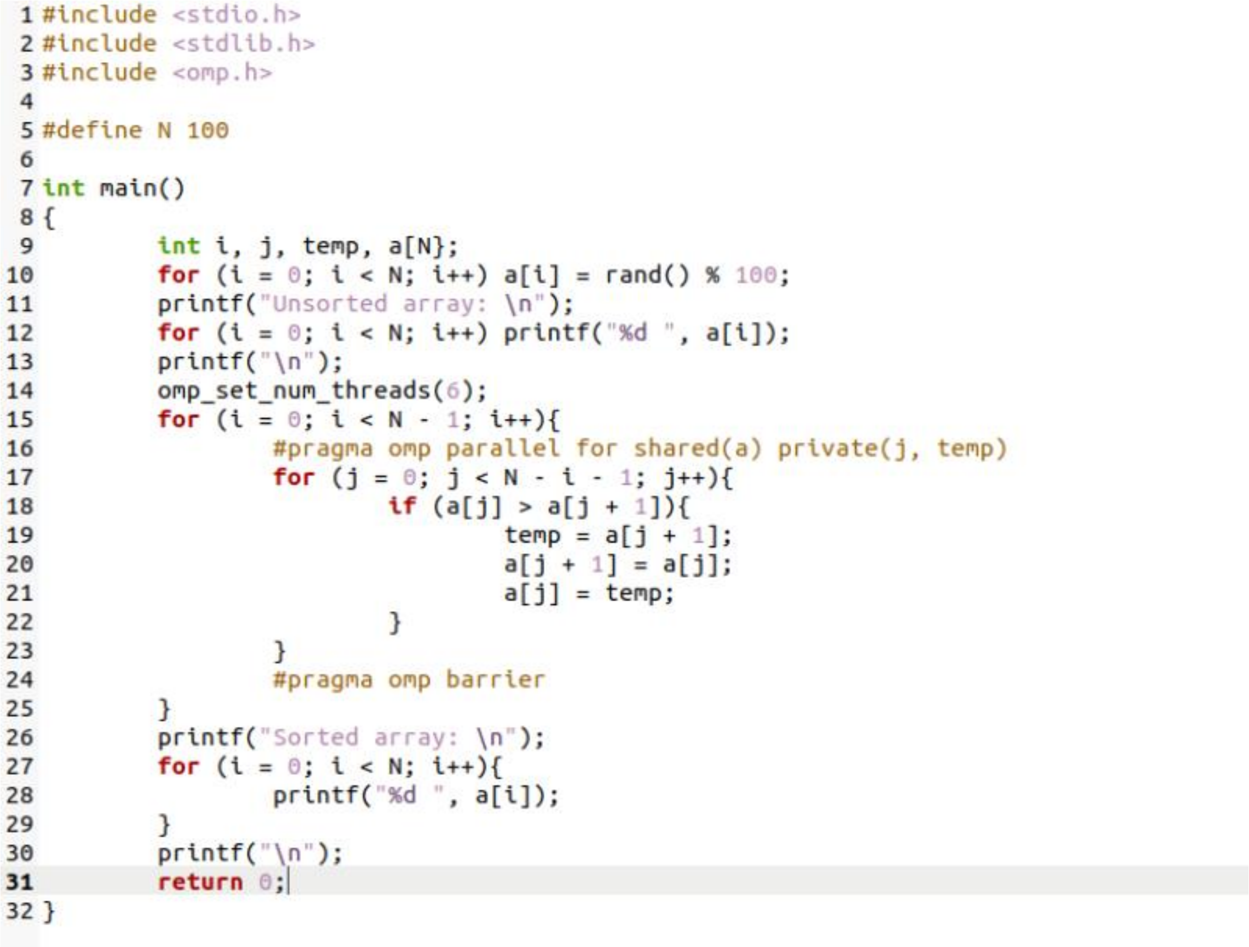
Output:



```
alizain@alizain-k214653:~/Desktop/pdc$ gedit t3.c
^C
alizain@alizain-k214653:~/Desktop/pdc$ gcc -fopenmp -o t3 t3.c
alizain@alizain-k214653:~/Desktop/pdc$ ./t3
1 2 3 4
5
6
7
8
Resultant matrix:
19 22
43 50
alizain@alizain-k214653:~/Desktop/pdc$ 
```

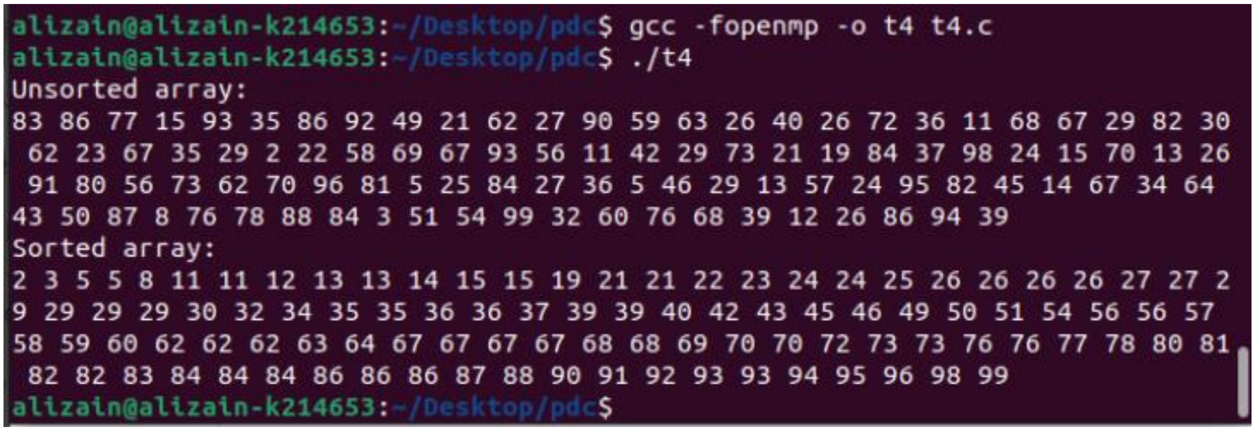
Task #4:

Code:



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

#define N 100

int main()
{
        int i, j, temp, a[N];
        for (i = 0; i < N; i++) a[i] = rand() % 100;
        printf("Unsorted array: \n");
        for (i = 0; i < N; i++) printf("%d ", a[i]);
        printf("\n");
        omp_set_num_threads(6);
        for (i = 0; i < N - 1; i++){
                #pragma omp parallel for shared(a) private(j, temp)
                for (j = 0; j < N - i - 1; j++){
                        if (a[j] > a[j + 1]){
                                temp = a[j + 1];
                                a[j + 1] = a[j];
                                a[j] = temp;
                        }
                }
                #pragma omp barrier
        }
        printf("Sorted array: \n");
        for (i = 0; i < N; i++){
                printf("%d ", a[i]);
        }
        printf("\n");
        return 0;
}
```

Output:



```
alizain@alizain-k214653:~/Desktop/pdc$ gcc -fopenmp -o t4 t4.c
alizain@alizain-k214653:~/Desktop/pdc$ ./t4
Unsorted array:
83 86 77 15 93 35 86 92 49 21 62 27 90 59 63 26 40 26 72 36 11 68 67 29 82 30
 62 23 67 35 29 2 22 58 69 67 93 56 11 42 29 73 21 19 84 37 98 24 15 70 13 26
 91 80 56 73 62 70 96 81 5 25 84 27 36 5 46 29 13 57 24 95 82 45 14 67 34 64
43 50 87 8 76 78 88 84 3 51 54 99 32 60 76 68 39 12 26 86 94 39
Sorted array:
2 3 5 5 8 11 11 12 13 13 14 15 15 19 21 21 22 23 24 24 25 26 26 26 26 27 27 2
9 29 29 29 30 32 34 35 35 36 36 37 39 39 40 42 43 45 46 49 50 51 54 56 56 57
58 59 60 62 62 62 63 64 67 67 67 67 68 68 69 70 70 72 73 73 76 76 77 78 80 81
 82 82 83 84 84 84 86 86 86 87 88 90 91 92 93 93 94 95 96 98 99
alizain@alizain-k214653:~/Desktop/pdc$
```