Operating System Lab – Final Exam

Problem description

This following code is for question number 1 and 2:

```
#include <stdio.h>
#include <omp.h>
using namespace std;

#define NUM_OF_COLUMNS 6
#define NUM_OF_ROWS (3*(NUM_OF_COLUMNS - 1))

int whichThread[NUM_OF_ROWS][NUM_OF_COLUMNS];

void fillColumn(int j)
{
    int i;
    #pragma omp for
    for (i = 0; i < NUM_OF_ROWS; i++)
    whichThread[i][j] = omp_get_thread_num();
}

int main()
{
    int i, j;
    for (i = 0; i < NUM_OF_ROWS; i++)
    for (j= 0; j < NUM_OF_COLUMNS; j++)
    whichThread[i][j] = -1;

    #pragma omp parallel num_threads(NUM_OF_COLUMNS - 1)
    fillColumn(0);

    #pragma omp parallel for num_threads(NUM_OF_COLUMNS - 1)
    for (j = 1; j < NUM_OF_COLUMNS; j++)fillColumn(j);
    for (i = 0; i < NUM_OF_ROWS; i++)
    {
        for (j= 0; j < NUM_OF_COLUMNS; j++)
        printf(" %2d ", whichThread[i][j]);
        printf("\n");
}
return 0;
}
```

1. (20pts) Explain the output of this program if it is compiled without `-fopenmp` flag!
2. (20pts) Explain carefully the output of this program with `-fopenmp` flag, also why the output is difference than the serial version of this program!

3. (60 points – 10 per function explanation) Suppose that we have a computer with 4 cores. Suppose we use OpenMP to parallelize a for-loop that initializes to zero the upper triangle of a 100 × 100 matrix.

```
#pragma openmp parallel for private(j) schedule( ... )
for (i = 0; i < 99; i++)
    for (j = i+1; j < 100; j++)
    { a[i][j] = 0.0; }
```

Notice that the `schedule()` clause has been left undefined. Below are six example schedule clauses that could be used. Rank these clauses from slowest to fastest. In particular, since each iterate of the inner loop above does just one assignment, we can estimate the execution time by counting how many assignments each thread does (notice that all together, there are exactly 4,950 assignments). So for each schedule clause, estimate how long the parallelized loop will run. Explain how you arrived at your estimates.

- `schedule(static)`
- `schedule(static, 10)`
- `schedule(static, 1)`
- `schedule(dynamic, 1)`
- `schedule(dynamic, 10)`
- `schedule(dynamic, 20)`