student name [ ]

American University of Central Asia
Software Engineering Department

Parallel Programming (COM 451)

Final Examination

You have one hour and fifteen minutes to finish writing the program.

1. Given a struct

```
typedef struct _body {
    float x, y;     // position
    float ax, ay;   // acceleration
    float vx, vy;   // velocity
    float mass;     // well...
} body;
```

and an array of bodies of size N

```
body bodies[N];
```

and a *DeltaTime*

```
static const DT = 0.1;
```

and a function to calculate a Newtonian acceleration

```
void calculate_newton_gravity_acceleration(body *a, body *b, float *ax, float *ay);
```

and a function to integrate the position from a velocity vector calculated from the given acceleration using a semi-implicit Euler method.

```
void integrate(body *body, float delta_time)
{
    body->vx += body->ax * delta_time;
    body->vy += body->ay * delta_time;
    body->x  += body->vx * delta_time;
    body->y  += body->vy * delta_time;
}
```

Parallelize the outer loop of the forward *N-Body* simulation with MPI.

```
for (size_t i = 0; i < N; ++i) {
    float total_ax = 0.0f, total_ay = 0.0f;
    for (size_t j = 0; j < N; ++j) {
        if (i == j) { continue; }

        float ax, ay;
        calculate_newton_gravity_acceleration(&bodies[i], &bodies[j], &ax, &ay);

        total_ax += ax;
        total_ay += ay;
    }

    bodies[i].ax = total_ax;
    bodies[i].ay = total_ay;
}

/*
   Final integration in the root process (move the bodies according to
   the calculated vectors).
 */
for (size_t i = 0; i < N; ++i) {
    integrate(&bodies[i], DT);
}
```

- You can use MPICH documentation (*mpich.org/static/docs/latest/www*).
- Minor *C*-language mistakes or typos are allowed.
- Common ceremonies (writing *includes* or the *main* function header) can be shortened or skipped.
- Considering the limited amount of time, try to reflect the core idea of parallelization of the algorithm in your code.

Your solution (you can continue on the back of the paper):