گزارش کار سوال چهارم تمرین سری چهارم سیستم‌های عامل – محمدپویا تراشی – ۴۰۱۵۲۱۱۰۲

```
1. pthread_mutex_t* cellLocks;
2. …
3. cellLocks = calloc(MAP_SIZE * MAP_SIZE, sizeof(pthread_mutex_t));
4.
```

در این خطوط از برنامه، ابتدا آرایهٔ cellLocks تعریف شده است. این آرایه به تعداد خانه‌های grid بازی عضو دارد. هر عضو نشان‌دهندهٔ قفل mutex برای آن خانه است.

```
 1. void* agent(void* args) {
 2.     AgentArgs* agent_args = (AgentArgs*)args;
 3.     int current_x = agent_args->initialPosition_x;
 4.     int current_y = agent_args->initialPosition_y;
 5.     char* path = agent_args->path;
 6.
 7.     for (int i = 0; path[i] != '\0'; i++)
 8.     {
 9.         int next_x = current_x;
10.         int next_y = current_y;
11.
12.         if (path[i] == 'N') next_x--;
13.         if (path[i] == 'S') next_x++;
14.         if (path[i] == 'E') next_y++;
15.         if (path[i] == 'W') next_y--;
16.
17.         pthread_mutex_lock(&cellLocks[next_x * MAP_SIZE + next_y]);
18.         pthread_mutex_lock(&maplock);
19.
20.         if (next_x >= 0 && next_x < MAP_SIZE && next_y >= 0 && next_y < MAP_SIZE &&
map[next_x][next_y] == '.')
21.         {
22.             map[current_x][current_y] = '.';
23.             map[next_x][next_y] = 'O';
24.             current_x = next_x;
25.             current_y = next_y;
26.         }
27.
28.         pthread_mutex_unlock(&maplock);
29.         pthread_mutex_unlock(&cellLocks[next_x * MAP_SIZE + next_y]);
30.
31.         sleep(1);
32.     }
33.
34.     return NULL;
35. }
36.
```

در ادامه، هر مامور به شکل همزمان در لیست path خود iterate می‌نمایند. با مشخص شدن حرکت بعدی، روی خانهٔ مد نظر و grid lock می‌گیرند. Lock روی grid بدین منظور است که اگر ماموری در حال اعمال تغییر بر زمین بازی است، صفحه print نشود. سپس بررسی می‌گردد که خانهٔ مد نظر در محدودهٔ زمین بازی باشد و بتوان به آن حرکت نمود. سپس تغییرات مد نظر اعمال می‌گردد.