Evelyn Moss

Michael Huang

**MultithreadedPokemon.c**

**MultithreadedPokemonLowCores.c**

Note: Two versions of this program are included, as titled above. The first is the intended version of the program. The second is specifically designed to aid TAs in testing, as they may (very unlikely, but possible) not have enough cores to test the second part of our program (which requires twice the number of cores as the number of types, meaning a minimum of 4 cores).

Variables

int found[] -

* Array of flags for the type being found. Init 0, shared, non-critical, check at each iteration of search to see if 1. When found, set to 1. Each search has its own index in the array

Int indices[] -

* Array of ints holding the indices of the found types. Init 0, shared, critical.

int multiplier -

* Total damage multiplier. Init 1, shared, critical, multiply by each type multiplier

const char \*type[18] -

* List of types to search. Init each to string of type (ex. “Bug”), search to find index, which is index in double-array type chart.

const float chart[18][18] -

* Type effectiveness chart. Init each to multiplier between types (ex. 2), plug in indices to get multiplier.

Main method

int main() -

Ask user for attack type. Asks user for number of defending types. Asks user for each defending type. If user has at least twice as many cores as the total number of types (attacking + defending), use OpenMP to create as many threads as the computer can and divide those into groups before searching. If the user has less cores than this, all threads do all searches. Each thread searches type array for given type (start and end points according to #threads allocated to that search), then consults double array type chart for damage multiplier. Each thread multiplies shared multiplier by specific type multiplier. Returns total multiplier at the end.

**Example Input and Output**

> ./mult

Enter attack type: Bug

Enter number of defending types: 2

Enter defending type 1: Ice

Enter defending type 2: Water

Total damage multiplier: 1.0000

**Issues**

* For some reason, the program runs very slowly. This occurred during a major fix, and we have no idea how to make the program fast again without breaking it. We apologize to whoever has to run this program.