Evelyn Moss

Michael Huang

**MultithreadedPokemon.c**

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 multiplier -

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

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 int 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. Use OpenMP to divide up threads between different searches (in the future, will determine how many cores the running computer has and decide whether to give the different searches to different groups of threads which will then further divide the search into sections, or to do the multithreaded searches in series). 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**

> run MultithreadedPokemon

Enter attack type: Bug

Enter number of defending types: 2

Enter defending type 1: Ice

Enter defending type 2: Water

Total damage multiplier: 1