Neslihan Çekiç 150160060

AoA2 –HW1

Compile: g++ -std=c++11 project1.cpp -o project1

Pikachu HP:200 Blastoise HP:200

Creating First Node :

```
BattleNode first_node(200, 100, 200, 100, 'P', 1, 0,1);
                    //(p_hp,p_pp,b_hp,b_pp,turn,prob,level,isleaf)
```

CLASSES

- ```
  class Attack {                    //To hold attacks
        string attack;
        int pp;
        int accu;
        int damage;
        int firstusage;
  }
  ```

- ```
  class BattleNode {                //Graph Node
        int p_hp;                   //Pikachu HP
        int p_pp;                   //Pikachu PP
        int b_hp;                   //Blastoise HP
        int b_pp;                   //Blastoise PP

        char turn;                  //Who's turn
        double prob;                //Probability of that node
        int level;                  //Level of that node
        bool isleaf;                //Leaf or not

        string att;                 //Which attack used
        bool effective;             //Is it effective

        BattleNode *mom;            //Parent of that node

        list<BattleNode*> *children;      //Children of that node
  }
  ```

FUNCTIONS

- `void readFiles() {…}`
  Reading txt files and set pikachu's and blastoise's attack lists.

- `list<BattleNode*> *createLevel(BattleNode *mother,int level, int count) {…}`
  Creating tree for max level and return last level's nodes.

- `void DFS(BattleNode* first_node) {…}`
  DFS recursive algorithm

- `void BFS(BattleNode* first_node) {…}`
  BFS recursive algorithm

- `BattleNode* PathPrint(BattleNode* child) {…}`
  Prints path that won the battle.

- `BattleNode* BFSFind(BattleNode* first_node, char pokemon) {…}`
  Finds node that won the battle using BFS algorithm

SAMPLE OUTPUTS (For HP=200 in both pokemon)

PART 1

```
[cekicl6@ssh algo2-1]$ g++ -std=c++11  project1.cpp -o project1
[cekicl6@ssh algo2-1]$ ./project1 part1 2
P_HP:170 P_PP:90 B_HP:160 B_PP:90 PROB:0.111111
P_HP:160 P_PP:90 B_HP:160 B_PP:80 PROB:0.111111
P_HP:140 P_PP:90 B_HP:160 B_PP:75 PROB:0.111111
P_HP:170 P_PP:85 B_HP:150 B_PP:90 PROB:0.0777778
P_HP:160 P_PP:85 B_HP:150 B_PP:80 PROB:0.0777778
P_HP:140 P_PP:85 B_HP:150 B_PP:75 PROB:0.0777778
P_HP:170 P_PP:85 B_HP:200 B_PP:90 PROB:0.0333333
P_HP:160 P_PP:85 B_HP:200 B_PP:80 PROB:0.0333333
P_HP:140 P_PP:85 B_HP:200 B_PP:75 PROB:0.0333333
P_HP:170 P_PP:80 B_HP:140 B_PP:90 PROB:0.0888889
P_HP:160 P_PP:80 B_HP:140 B_PP:80 PROB:0.0888889
P_HP:140 P_PP:80 B_HP:140 B_PP:75 PROB:0.0888889
P_HP:170 P_PP:80 B_HP:200 B_PP:90 PROB:0.0222222
P_HP:160 P_PP:80 B_HP:200 B_PP:80 PROB:0.0222222
P_HP:140 P_PP:80 B_HP:200 B_PP:75 PROB:0.0222222
[cekicl6@ssh algo2-1]$
```

PART 2

```
[cekicl6@ssh algo2-1]$ g++ -std=c++11  project1.cpp -o project1
[cekicl6@ssh algo2-1]$ ./project1 part2 2 dfs
NODE COUNT: 21
RUNNING TIME: 123 microseconds
[cekicl6@ssh algo2-1]$ ./project1 part2 2 bfs
NODE COUNT: 21
RUNNING TIME: 97 microseconds
[cekicl6@ssh algo2-1]$ ./project1 part2 6 dfs
NODE COUNT: 11271
RUNNING TIME: 11104 microseconds
[cekicl6@ssh algo2-1]$ ./project1 part2 6 bfs
NODE COUNT: 11271
RUNNING TIME: 1034495 microseconds
[cekicl6@ssh algo2-1]$
```

As we see, although BFS algoritm much quicker in small node count, DFS algorithm much more efficient in large node count.

PART 3

```
[cekicl6@ssh algo2-1]$ g++ -std=c++11  project1.cpp -o project1
[cekicl6@ssh algo2-1]$ ./project1 part3 pikachu
Pikachu used Thundershock. It's effective.
Blastoise used Tackle. It's effective.
Pikachu used Thundershock. It's effective.
Blastoise used Tackle. It's effective.
Pikachu used Slam. It's effective.
Blastoise used Tackle. It's effective.
Pikachu used Slam. It's effective.

Level count: 7
Probability: 6.94444e-05
[cekicl6@ssh algo2-1]$ ./project1 part3 blastoise
Pikachu used Thundershock. It's effective.
Blastoise used Tackle. It's effective.
Pikachu used Thundershock. It's effective.
Blastoise used Bite. It's effective.
Pikachu used Thundershock. It's effective.
Blastoise used Bite. It's effective.
Pikachu used Thundershock. It's effective.
Blastoise used Bite. It's effective.

Level count: 8
Probability: 2.71267e-05
```