public class LogAnalyzer

{

private ArrayList<LogEntry> records;

public LogAnalyzer() {

records = new ArrayList<LogEntry>();

}

// read a new file

public void readFile(String filename) {

FileResource fr = new FileResource(filename);

for (String line: fr.lines()) {

LogEntry le = WebLogParser.parseEntry(line);

records.add(le);

}

}

// print all information in records

public void printAll() {

for (LogEntry le : records) {

System.out.println(le);

}

}

// count the number of unique IPs.

public int countUniqueIPs() {

ArrayList<String> uniqueip = new ArrayList<String>();

for (LogEntry le: records) {

String ip = le.getIpAddress();

if (!uniqueip.contains(ip)) uniqueip.add(ip);

}

return uniqueip.size();

}

// return unique Ips that visited that day.

public ArrayList<String> uniqueIPVisitsOnDay(String someday) {

ArrayList<String> uniqueip = new ArrayList<String>();

for (LogEntry le:records) {

String whole = le.getAccessTime().toString();

String date = whole.substring(8, 10);

String month = whole.substring(4, 7);

if (someday.substring(0,3).equals(month) && someday.substring(4,6).equals(date)) {

String ip = le.getIpAddress();

System.out.println(ip);

if (!uniqueip.contains(ip)) uniqueip.add(ip);

}

}

return uniqueip;

}

// count the number of unique IPs having status code in range

public int countUniqueIPsInRange(int low, int high) {

ArrayList<String> uniqueip = new ArrayList<String>();

for (LogEntry le:records) {

int whole = le.getStatusCode();

if (whole>=low && whole<=high) {

String ip = le.getIpAddress();

System.out.println(ip);

if (!uniqueip.contains(ip)) uniqueip.add(ip);

}

}

return uniqueip.size();

}

// print status code higher than a number

public void printAllHigherThanNum(int num) {

ArrayList<Integer> uniquestatus = new ArrayList<Integer>();

for (LogEntry le:records) {

int whole = le.getStatusCode();

if (whole> num) {

if (!uniquestatus.contains(whole)) uniquestatus.add(whole);

}

}

for (int status: uniquestatus) {

System.out.println(status+"\t");

}

}

// return number of visits per IP.

public HashMap<String, Integer> countVisitsPerIP() {

HashMap<String, Integer> uniqip = new HashMap<String, Integer>();

for (LogEntry le:records) {

String ip = le.getIpAddress();

if (!uniqip.containsKey(ip)) {

uniqip.put(ip, 1);

}

else uniqip.put(ip, uniqip.get(ip)+1);

}

return uniqip;

}

// return the number of most visits by IP.

public int mostNumberVisitsByIP(HashMap<String, Integer> uniqip) {

int max=0;

int current;

for (String ip: uniqip.keySet()) {

current = uniqip.get(ip);

if (max < current) max = current;

}

return max;

}

// return the IP list with most visit numer above.

public ArrayList<String> iPsMostVisits(HashMap<String, Integer> uniqip) {

int max = mostNumberVisitsByIP(uniqip);

ArrayList<String> show = new ArrayList<String>();

int current;

for (String ip: uniqip.keySet()) {

current = uniqip.get(ip);

if (max == current) show.add(ip);

}

return show;

}

// return each day and IPs for that day.

public HashMap<String, ArrayList<String>> iPsForDays() {

HashMap<String, ArrayList<String>> eachday = new HashMap<String, ArrayList<String>>();

for (LogEntry le:records) {

String whole = le.getAccessTime().toString();

String date = whole.substring(8, 10);

String month = whole.substring(4, 7);

String time = month + " " + date;

String ip = le.getIpAddress();

if (!eachday.containsKey(time)) {

eachday.put(time, new ArrayList<String>());

}

eachday.get(time).add(ip);

}

return eachday;

}

// return the day with most IP visits.

public String dayWithMostIPVisits(HashMap<String, ArrayList<String>> eachday) {

String mosttime=null;

int most = 0;

int current;

for (String time: eachday.keySet()) {

current = eachday.get(time).size();

if (most < current) {

most = current;

mosttime = time;

}

}

return mosttime;

}

// return the IP list with most visits on particular day.

public ArrayList<String> iPsWithMostVisitsOnDay(HashMap<String, ArrayList<String>> eachday, String day) {

ArrayList<String> ip = new ArrayList<String>();

HashMap<String, Integer> eachip = new HashMap<String, Integer>();

ArrayList<String> mostthatday = new ArrayList<String>();

int mostnumber = 0;

int currentnumber;

for (String time: eachday.keySet()) {

if (time.equals(day)) {

ip = eachday.get(time);

}

}

for (int k = 0; k < ip.size(); k++) {

if (!eachip.containsKey(ip.get(k))) {

eachip.put(ip.get(k), 1);

}

else eachip.put(ip.get(k), eachip.get(ip.get(k))+1);

}

mostthatday = iPsMostVisits(eachip);

return mostthatday;

}

}

# test class

import java.util.\*;

public class Tester

{

public void testLogEntry() {

LogEntry le = new LogEntry("1.2.3.4", new Date(), "example request", 200, 500);

System.out.println(le);

LogEntry le2 = new LogEntry("1.2.100.4", new Date(), "example request 2", 300, 400);

System.out.println(le2);

}

public void testLogAnalyzer() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

la.printAll();

}

public void testUniqueIP() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

int number = la.countUniqueIPs();

System.out.println("The number of unique IPs is "+number+"\t");

}

public void testuniqueIPVisitsOnDay() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

ArrayList<String> uniqueip = la.uniqueIPVisitsOnDay("Sep 27");

System.out.println("The number of unique IPs on Sep 27 is "+uniqueip.size()+"\t");

}

public void testcountUniqueIPsInRange() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

int uniqueip = la.countUniqueIPsInRange(400, 499);

System.out.println("The number of unique IPs is "+uniqueip+"\t");

}

public void printAllHigherThanNum() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

la.printAllHigherThanNum(400);

}

public void testmostNumberVisitsByIP() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

HashMap<String, Integer> uniqip = la.countVisitsPerIP();

int most = la.mostNumberVisitsByIP(uniqip);

System.out.println(most);

}

public void testiPsMostVisits() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

HashMap<String, Integer> uniqip = la.countVisitsPerIP();

ArrayList<String> result = la.iPsMostVisits(uniqip);

System.out.println(result.get(0));

}

public void testiPsForDays() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

HashMap<String, ArrayList<String>> eachday = la.iPsForDays();

for (String time: eachday.keySet()) {

if (time.equals("Sep 30")) System.out.println(eachday.get(time).size());

}

}

public void testdayWithMostIPVisits() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

HashMap<String, ArrayList<String>> eachday = la.iPsForDays();

String mosttime = la.dayWithMostIPVisits(eachday);

System.out.println(mosttime);

}

public void testiPsWithMostVisitsOnDay() {

LogAnalyzer la = new LogAnalyzer();

la.readFile("weblog2\_log");

HashMap<String, ArrayList<String>> eachday = la.iPsForDays();

ArrayList<String> result = la.iPsWithMostVisitsOnDay(eachday, "Sep 30");

for (int k = 0; k < result.size(); k++) {

System.out.println(result.get(k));

}

}

}