实验 3：实现网络命令行工具

一、实验目的

在这个项目中，将使用三个命令行工具来分析互联网行为。你们将使用 ping 命令来测量 RTT（往返时间）和丢包率，traceroute 来检查网络路由，dig 命令来了解域名如何映射成 IP 地址。该实验的重点是研究 100 个你最常访问的网站（按 Alexa 衡量）的连接。在某些情况下，我们会要求你对较小的一组网站进行更详细的测量，以深入了解某些特定的测量。

二、实验要求

注意事项

- 截止日期：6 月 10 日

最后需要提交三个命令的源代码，测试程序的输出文件（下文中会给出格式要求），可执行程序，使用说明以及实验报告。

三、提供的文件

1、alexa_top_100：Alexa 排名前 100 的最受欢迎的网站。（请在 http://netlab.csu.edu.cn/index8 下载）

四、实验内容

Part 1. Ping 命令

在这部分作业中，主要使用 ping 来测量不同网站的往返时间。
首先，你的程序中应该具有一个函数：run_ping

1) run_ping 函数。应该运行 ping 命令并生成 json 输出。它应该有 4 个参数：

- hostnames：要 ping 的主机名列表；
- num_packets：发送给每个主机的 ping 数据包数；
- raw_ping_output_filename：将原始 ping 结果输出到（作为 json）的文件名；
- aggregated_ping_output_filename：将汇总的 ping 结果输出到（作为 json）的文件名。

例如，假设 hostnames = [“www.baidu.com”]和 num_packets = 100，那么你应该 ping baidu.com 100 次。你可以通过使用子进程模块调用 ping shell 命令来执行此操作：

```
ping -n 100 baidu.com
```

注意：Windows 上的-n 参数可能不同。 如果不行，请尝试使用-c。
run_ping 函数应该生成两个 json 输出。

① 原始 ping 结果。该文件包括每个数包级别的详细结果。数据格式为：

```
{
      hostname1:  [rtt_1, rtt_2...],
      hostname2:  [rtt_1, rtt_2...],
      ...
}
```

每个主机名映射到 RTT 列表。 按照上一个例子，如果你 ping baidu.com 100 次，那么应该有一个主机名“baidu.com”映射到 100 个 RTT 的列表。 主机名应为字符串，RTT 列表应为浮点数，以毫秒为单位。 请注意，并不是所有的网站都会响应 ping，有些可能丢弃数据包。如果 ICMP 数据包超时，则请使用-1.0 作为 RTT。

② 统计汇总 ping 结果。对于每个网站，你应该汇总原始 ping 结果以产生中间值 RTT，最大 RTT 以及丢包率。格式应该是：

```
{
      hostname1: {“drop_rate”: drop_rate1, “max_rtt”: max_rtt1, “median_rtt”: median_rtt1},
      hostname2: {“drop_rate”: drop_rate2, “max_rtt”: max_rtt2, “median_rtt”: median_rtt2},
      ...
}
```

主机名是字符串，并且每个映射到三个汇总数字：丢弃率，最大 RTT 和中值 RTT。 应以 0.0---100.0 之间的百分比指定丢包率（例如：如果你观察到在 ping 运行中 500 个包中丢弃的数据包为 5，则丢包率为 1.0%）。 最大 RTT 和中位数 RTT 应为毫秒。所有这三个数字应该是浮点数。如果一个网站有几个丢弃的 ping 数据包，请不要将这些数据包包含在中值/最大值计算中。如果一个网站根本没有响应，那么最大和中值 RTT 应该是-1.0，丢弃率应该是 100.0%。

样例输出，请查看 sample_ping.txt 和 sample_ping.json。文本文件显示 ping google.com 的原始文本(sample_ping.txt)输出 10 次，json 文件(sample_ping.json)显示 json 格式的原始 ping 结果。

sample_ping.txt

```
PING google.com
(216.58.194.206):
56 data bytes
                    64 bytes from 216.58.194.206: icmp_seq=0 ttl=54 time=2.561 ms
                    64 bytes from 216.58.194.206: icmp_seq=1 ttl=54 time=12.351 ms
                    64 bytes from 216.58.194.206: icmp_seq=2 ttl=54 time=2.963 ms
                    64 bytes from 216.58.194.206: icmp_seq=3 ttl=54 time=5.848 ms
                    64 bytes from 216.58.194.206: icmp_seq=4 ttl=54 time=2.590 ms
                    64 bytes from 216.58.194.206: icmp_seq=5 ttl=54 time=5.180 ms
                    64 bytes from 216.58.194.206: icmp_seq=6 ttl=54 time=3.610 ms
                    64 bytes from 216.58.194.206: icmp_seq=7 ttl=54 time=3.741 ms
                    64 bytes from 216.58.194.206: icmp_seq=8 ttl=54 time=2.752 ms
```

```
64 bytes from 216.58.194.206: icmp_seq=9 ttl=54 time=12.958 ms


--- google.com ping statistics ---
10 packets transmitted, 10 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 2.561/5.455/12.958/3.749 ms
```

sample_ping.json

```
{"google.com": [2.561, 12.351, 2.963, 5.848, 2.59, 5.18, 3.61, 3.741, 2.752, 12.958]}
```

2)将文本结果进行可视化，y 轴的值应为 RTT
注：Java 语言图表绘制，推荐使用 jfreechart 开源库
(http://www.jfree.org/jfreechart/)

***3）使用编写的代码，运行以下实验。**

- ping 每个 Alexatop100 网站 10 次。你应该将代码的输出存储在两个文件中。
- 接下来，我们要更详细地检查几个网站的 ping 行为。网站是：baidu.com，microsoft.com，sina.com.cn，taobao.com。ping 每个网站 500 次。将代码的输出存储在两个文件中。

Part 2. Traceroute 命令

虽然 ping 对于识别端到端行为很有用，但是 traceroute 是一个可以为你提供有关 Internet 路由的更详细信息的工具。特别是，traceroute 允许你跟踪从机器到远程机器的整个路由。

本部分要求：你应该编写一个程序，与 Part1 类似该程序应该运行 traceroute 列表的网站并生成 json 输出。该文件还应该能够直接解析 shell traceroute 命令输出。该程序应包含两个函数：

（1）run_traceroute（hostnames，num_packets，output_filename）：用于在主机名列表上运行 traceroute 命令，输出 traceroute 的结果，它包括三个参数：

- hostnames：traceroute 的主机名列表；
- num_packets：发送到每一跳的数据包数量；
- output_filename：在哪里保存 traceroute 命令的结果。

（2）parse_traceroute（raw_traceroute_filename，output_filename）：此函数应该能够从 run_traceroute 函数中获取输出，并写出 json 数据。

- raw_traceroute_filename：从 shell 命令存储 traceroute 输出的文件的名称；
- output_filename：存储输出 json 数据。

你可以通过使用子进程模块调用 traceroute shell 命令来跟踪网站：
Windows 系统: `tracert baidu.com`
Linux 系统: `traceroute baidu.com`

你将查看以下网站的路由行为：baidu.com，taobao.com，www.csu.edu.cn，以及一些你感兴趣的网站。将输出结果保存为 json 格式。

Part 3. Naming

在这部分中，你将使用 dig 命令理解 DSN 延迟。写一个程序解析 IP 地址并生成汇总结果的 json 输出。 你的脚本应该有几个函数运行 dig，并且处理 dig 输出。
你的程序应该包含函数名为 run_dig（hostname_filename，output_filename，dns_query_server =None）解析排名前 100 的网站对应的 IP 地址。这个函数应该解析每个地址 5 次。

- hostname_filename: 该文件包含 hostnames 列表， 例如：alexa_top_100；
- output_filename: json 输出文件的名字；
- dns_query_server: 指定要查询的 DNS 服务器的可选参数；

你的脚本应从根开始解析每个站点（即首先查询根域服务器，然后查询 TLD 域服务器等）。 你应该通过调用以下 shell 命令来执行此操作：

```
dig +trace +tries=1 +nofail www.google.com
```

windows 下安装 dig 命令请参考：
http://jingyan.baidu.com/article/f25ef25444b89a482c1b82b5.html

（安装完成后，如无法在命令行直接运行，请进入安装目录下的 bin 路径下尝试运行）

请注意，此命令还包括一些额外的标志。 当 DNS 查找失败时，+ attempts = 1 和+ nofail 标志信号 dig 不进行故障转移，以便你可以计算出多少次查找失败。

如果指定了 dns_query_server，你的脚本应该向指定的服务器发送 DNS 请求，不应使用+ trace 参数; 例如。：

```
dig www.google.com @8.8.8.8
```

8.8.8.8 是 DNS 服务器的地址。

run_dig 应该生成一个 json 输出列表，每个 json 字典表示一个单独的“dig”调用，并将输出保存到 output_filename。 每个调用“dig”的表示形式应该如下所示：

- “Name”：正在解析的名称
- “Success”：dig 调用是否成功（如果是 false，则 json 输出中唯一的其他字段应该是“Name”;不应该有任何查询）
- “Queries”：为单次 dig 调用所做的所有查询的列表。 每个查询的格式为：

- “Time”：整数表示完成查询所需的时间
- “Answers”：查询的结果列表。 每个结果的格式是：
  - “Queried name”：查询的名称（例如“。”或“.com”）。 这是挖掘输出中的第一个字段。
  - “data”：结果（例如，对于 NS 记录，DNS 服务器的名称，或对于 A 记录，IP 地址）
  - “Type”：结果类型（例如“CNAME”或“A”）
  - “TTL”：表示结果的 TTL 的整数

程序还应包括以下处理功能：

- get_average_ttls（filename）：该函数应该接受具有上面输入的 json 文件的名称作为输入。它应该按以下顺序返回包含以下平均值的 4 个项目列表：
  - 根服务器的平均 TTL 是多少？
  - 顶级域名（tld）服务器的平均 TTL 是多少？
  - 任何其他名称服务器的平均 TTL 是多少？（例如，对于 google.com，这包括 google.com 名称服务器）。
  - 终止 CNAME 或 A 条目的平均 TTL 是多少？

  换句话说，它应该返回[average_root_ttl，average_TLD_ttl，average_other_ttl，average_terminating_ttl]。所有时间都应该在几秒钟内，这些平均值应该超过给定文件名中的所有 DNS 查询（不仅仅是特定主机的条目）。

  这里有一件棘手的问题是如何处理返回多个答案的查询。例如，假设你的 json 输出只对两个站点进行查询。例如，我们来看看这些站点的终止条目：

  www.google.com:

  ```
  www.google.com. 300 IN A 172.217.5.100
  ```

  weibo.com:

  ```
  weibo.com. 60 IN A 180.149.134.141 weibo.com. 60 IN A 180.149.134.142
  ```

  这里，对 weibo.com 的 DNS 请求返回了两个不同的终止记录，这通常用于帮助负载平衡和处理故障。要计算平均 TTL，你应该首先计算每个查询的平均 TTL。在这里，这将给你两个平均值：300（谷歌）和 60（对于 weibo）。然后计算平均值，在这种情况下结果为 180。这样做的原因是为了避免从返回多个答案的查询中增加 TTL 的重要性。

- get_average_times（filename）：此函数应该接受一个 json 文件的名称，其输出如上所述，作为输入。它应该按以下顺序返回包含以下平均值的 2 个项目列表：

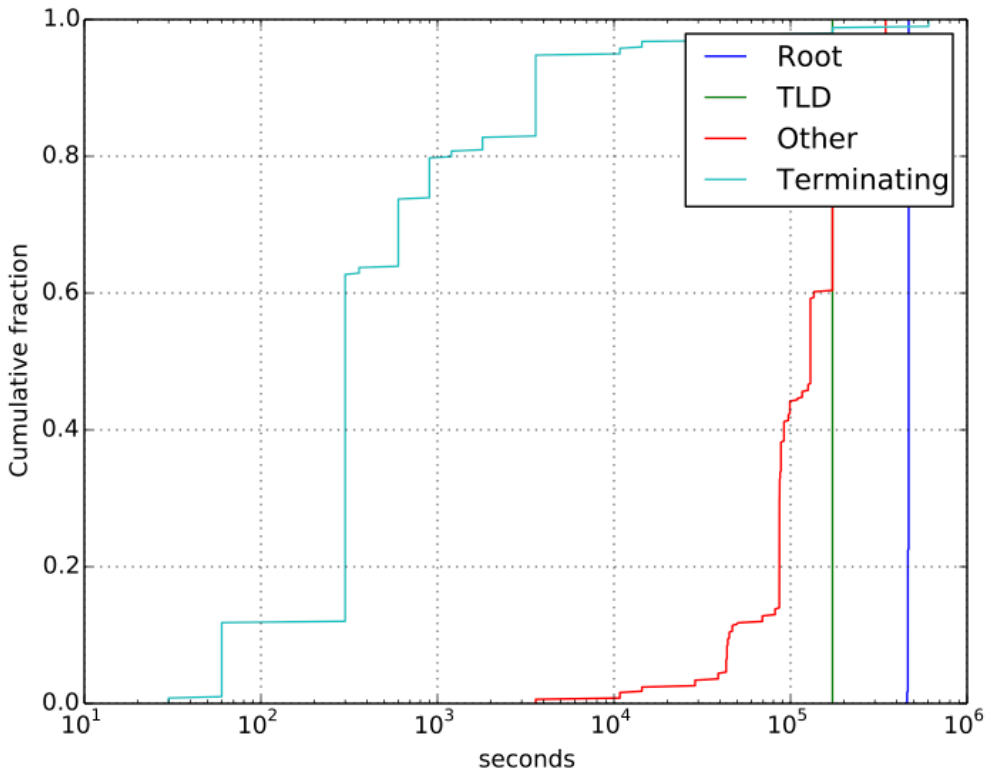
  - 解析网站的总时间的平均值。这应该包括解析层次结构中所有步骤的时间。例如，对于 google.com，它应该包括联系根服务器以确定顶级域名服务器（com）位置的时间，以及联系 com TLD 服务器以解析 google 的时间，以及 google 域名服务器解析 google.com 的时间。
  - 导致 A（或 CNAME）记录的最终请求的平均时间。

  与 get_average_ttls 一样，这些平均值应该超过给定文件名中的所有 DNS 查询（不仅仅是特定主机名的 DNS 查询）。

- generate_time_cdfs(json_filename)：该函数应该接受 json_filename，一个 json 文件的名称，其内容如上所述，作为输入。它应该生成两个图：一个显示解析站点的总时间的 CDF 图，一个显示解析最终请求时间的 CDF 图（这是相同的两个分布，都是 get_average_times 返回的平均值）。

附录 1：什么是 CDF

CDF（累积分布函数）对于 x 轴上的所有值 x 显示变量取小于或等于 x 的概率（更多参见维基百科：https://en.wikipedia.org/wiki/Cumulative_distribution_function）。例如，这是一个 CDF，我们为 Alexa 前 100 个网站的 DNS 条目产生的 TTL：



从 CDF 可以直观的观察到中值：中值是线的 y 值为 0.5 的 x 值。在上面的示例 CDF 中，终止 DNS 记录的中值 TTL（即 A 或 CNAME 记录）为 200 秒。根服务器的 DNS 条目的中值 TTL 大约为 5*10^5。CDF 也可用于了解分布。例如，从上面的 CDF，我们可以看到根服务器和顶级域服务器基本上是恒定的 TTL，因为 CDF 是一条直的垂直线（因为所有查询都使用相同的根服务器）。另一方面，终止记录和其他记录的 TTL 更广泛地变化。