AUBURN UNIVERSITY

M2-Homework

Questions and Exercises to work out and turn in:

Grading Guidelines:

Exceptionally for this homework, you will not have to justify your answers. Just be neat and provide complete answers.
========== The following rubric does not apply to this homework.
A right answer will get full credit when:

1. It is right (worth 25%)
2. It is right **AND** neatly presented making it easy and pleasant to read. (worth an **extra** 15%)
3. There is an **obvious and clear link** between 1) the information provided in the exercise and in class and 2) the final answer. A clear link is built by properly writing, justifying, and documenting an answer (worth an **extra** 60%).
4. Calculation mistakes will be minimally penalized (2 to 5% of full credit) while errors on units will be more heavily penalized.

Late Submission : as specified in the syllabus. Days counting starts one minute after the deadline.
Check Your Submission: after submitting, download your submission to check whether it is the right version and it is complete.

You are welcome/encouraged to discuss exercises with other students or the instructor. But, ultimately, **personal** writing is expected.

- USE THIS FILE AS THE STARTING DOCUMENT YOU WILL TURN IN. **KEEP IN THE QUESTIONS** AND INSERT YOUR ANSWERS.
- IF USING HAND WRITING (STRONGLY DISCOURAGED), REWRITE THE QUESTIONS.
- FAILING TO FOLLOW TURN IN DIRECTIONS /GUIDELINES WILL COST A 30% PENALTY.

Objectives of this assignment:

- to implement programs that communicate over the Internet
- to implement such applications using socket programming.
- Understand the relashionship *IP addresses - Host Names*

What you need to do:

Answer the questions and/or solve the exercises described below.

Exercise 1 (100 points)

The objective of this exercise is to get you familiar with the types used in Java to handle IP addresses. Consider the program ***InetAddressExample.java***[1] provided with this homework:

```java
import java.net.*;  // for InetAddress

public class InetAddressExample {

  public static void main(String[] args) {

    // Get name and IP address of the local host
    try {
      InetAddress address = InetAddress.getLocalHost();
      System.out.println("Local Host:");
      System.out.println("\t" + address.getHostName());
      System.out.println("\t" + address.getHostAddress());
    } catch (UnknownHostException e) {
      System.out.println("Unable to determine this host's address");
    }

    for (int i = 0; i < args.length; i++) {
      // Get name(s)/address(es) of hosts given on command-line
      try {
        InetAddress[] addressList = InetAddress.getAllByName(args[i]);
        System.out.println(args[i] + ":");
        // Print the first name.  Assume array contains at least one entry.
        System.out.println("\t" + addressList[0].getHostName());
        for (int j = 0; j < addressList.length; j++)
          System.out.println("\t" + addressList[j].getHostAddress());
      } catch (UnknownHostException e) {
        System.out.println("Unable to find address for " + args[i]);
      }
    }
  }
}
```

a) (20 points) Download, read, examine, compile, and run this program to understand this program. Execute the following commands and **provide the screenshots** of the responses:

1) (5 points) `java InetAddressExample`

```
[➜  Reference_Book_Examples java InetAddressExample
Local Host:
        Kings-Macbook.local
        127.0.0.1
```

[1] This file is in Instructional Resources in the folder Simple examples from reference book.

2) (15 points) `java InetAddressExample www.harvard.edu www.yale.edu`

```
[➜  Reference_Book_Examples java InetAddressExample www.harvard.edu www.yale.edu
Local Host:
        Kings-Macbook.local
        127.0.0.1
www.harvard.edu:
        www.harvard.edu
        199.232.34.133
        2a04:4e42:45:0:0:0:0:645
www.yale.edu:
        www.yale.edu
        199.232.34.133
        151.101.206.133
        2a04:4e42:45:0:0:0:0:645
```

b) (80 points) Modify this program to create a program named `MyInetAddressExample.java` to perform the following tasks:

1) Prompt the user to enter a hostname (e.g., www.auburn.edu)

2) Display the IP addresses in binary, binary dotted-quad, and decimal dotted-quad formats. We are interested only in IPv4 addresses (32 bit IP address). For example, if the user enters the hostname`www.auburn.edu`, then your program must display:

- (25 points) `binary  format             : 10000011110011001000101010101010`
- (25 points) `binary  dotted-quad format : 10000011.11001100.10001010.10101010`
- (25 points) `decimal dotted-quad format : 131.204.138.170`

3) (5 points) Provide a screenshot of an execution of your program.

```
[➜  hw2 java MyInetAddressExample

[Enter a hostname (e.g., www.auburn.edu): www.auburn.edu

 Hostname: www.auburn.edu
     binary  format             : 10000011110011001000101010101010
     binary  dotted-quad format : 10000011.11001100.10001010.10101010
     decimal dotted-quad format : 131.204.138.170
```

What you need to turn in:

- Electronic copy of this file (including your answers) (standalone) and the program source `MyInetAddressExample.java` (standalone) Submit this file as a Microsoft Word or PDF file.

- Recall that answers must be well written, documented, justified, and presented to get full credit.

- How this assignment will be graded: (**No need to justify answers for this homework assignment**)
- A right answer will get full credit when:
- It is right (worth 25%)
- It is right AND neatly presented making it easy and pleasant to read. (worth 15%)
- There is an obvious and clear link between 1) the information provided in the exercise and in class and 2) the final answer. A clear link is built by properly writing, justifying, and documenting an answer (worth 60%).
- Calculation mistakes will be minimally penalized (2 to 5% of full credit) while errors on units will be more heavily penalized.

- You are welcome/encouraged to discuss exercises with other students or the instructor. But, ultimately, personal writing is expected.