Final work: Computer Networks (due 13.08.2020):
Version 1.1

The exam is divided into two parts: (1) programming questions with theoretical questions; and (2) the oral exam (see below).

Instructions:

1. The work should be done in singles without advice from each other. All clarifying questions can be asked in our whatsapp chat or in Moodle.
2. If a programming question is not compiling, you will get zero for the corresponding part.

Questions:

1. Packet classifier (50 points)

a) In this part you will implement an IP lookup table based on a binary trie that we studied. Your implementation should support three operations: ADD/REMOVE inserting/deleting a given prefix with the action to/from the trie; and FIND a new prefix, The input is read from a file in the following format: action, prefix, result.

Input file:
ADD 255.255.255.0/24 A
FIND 255.255.255.255
REMOVE 255.255.255.0/24 A

Expected output:
Added 255.255.255/24 A at the depth <depth in the binary trie>, total nodes <notal nodes in binary trie>
Found 255.255.255.255 <found action> at the depth <depth in the binary trie>
Removed 255.255.255/24 A at the depth <depth in the binary trie> total nodes <notal nodes in binary trie>

b) Consider the following optimization: if two prefixes differ only in the last bit and point to the same action than both prefixes can be replaced by a single prefix with the same action.

Example: Two prefixes 1111* -> A 1110*-> A can be replaced by 111**->A.

See the binary trie example here both for the regular and the optimized cases.

Add this optimization to the previous implementation of prefix tables. The FIND action should return the optimized prefix with the right action.

c) For implementations in a) and b) in the worst case the lookup time is equal to the maximal prefix length. Assume that you have an explicit constraint on a number of lookups. For instance, 8 lookups in the table containing 32-bit prefixes. Suggest your ways how to address a fundamental tradeoff between lookup time and the required memory to represent this table.

d) Think about alternative ways to represent a table of prefixes with the desired balance between lookup time and memory requirements. Compare cons and pros of various proposals. More creative proposals are better valued. Not necessary binary trie should be considered.

What to do:

1. Implement two prefix table representations as in a) and b). In the makefile the target for a) should be **prefix_table** and for b) **prefix_table_opt.** Have a clear README file explaining how to run both implementations. Also add your sample input file (**sample_input**) that was used with your programs.
2. Answer c) and d) in a separate document entitled 1_<your_id>.pdf
3. All code and <your_id>.pdf have in a separate folder entitled 1_<your_id>;

For the oral exam: in addition to what you did in 1, you will need to know all routing protocols that we covered: RIP, OSPF, IS-IS, BGP . You should have a clear understanding about their purpose, the whole taxonomy of routing protocols that we studied and their cons and pros.

2. The nature of congestion (50 points).

One of the central questions in computer networks is how to deal with congestion. As we studied congestion in packet networks is unavoidable if we want to exploit network resources efficiently. During the lecturers, we considered congestion control whose logic is implemented on end hosts. This type of congestion control was based on implicit network feedback - loss. This is the so-called end-2-end design principle that does not assume any knowledge about how switches are implemented. While the end-2-end design principle is nice, additional performance requirements as for low-latency applications require support in buffer management. In this case the congestion control is a composition of control logic implemented on end hosts and buffer management policies implemented inside switches. In this question we explore the performance of two different buffer management policies: Early-Deadline-First (EDF) and Bounded-delay we will define them shortly. As we studied each buffer management policy, managing a single queue is defined by admission policy defining which packets can be admitted and processing policy that defines in which order the buffered packets are processed.

Inputs. we assume that each packet p is prepended with a **packet slack** s(p) defining a maximal offset in time when a packet should be transmitted and a **packet value** v(p). You can assume that packets with expired slack are automatically removed from the queue.

Model. We consider a single queue buffering architecture. Time is slotted. Each time slot consists of two phases:

1) **arrival phase** where the admission policy decides which packets are admitted and
2) **processing phase** where a single packet p (chosen by the processing policy) is transmitted out and its value v(p) is added to the total transmitted value; The slack values of all remaining in the queue packets are decreased by one. All packets with the zero remaining slack are removed without added value to the final objective.

Our objective is to maximize the total value of transmitted packets whose slack is unexpired.

EDF policy:
admission: during arrival phase, for each incoming packet p if a buffer is not full accept p. Otherwise (full buffer), if in the queue there is a packet q whose remaining slack s(q)<s(p), q is removed from the queue and p is accepted. Otherwise, p is dropped.
processing: from all buffered packets transmit a packet p with the minimal s(p).

Bounded-delay policy:

admission: during arrival phase, for each incoming packet p if a buffer is not full accept p. Otherwise (full buffer), if in the queue there is a packet q whose value v(q)<v(p), q is removed from the queue and p is accepted. Otherwise, p is dropped.
processing: from all buffered packets transmit a packet p with the maximal v(p).

What to do:

1. Implement both EDF and Bounded-delay. Each scheduling algorithm accepts as input a queue size and a file name with arrivals.

Format of the input file:
Each line in the input file contains arrivals in tuples (amount, slack, value).
(5,3,6) (2,4,1) (6,2,4) // during this time slot arriving 5 packets whose slack is 3 and the value is 6, etc.

Input_file example:

```
(2,3,6) (3,4,1)
(2,4,5)
```

At each time slot the packets are admitted from left to right. For example, (2,3,6) and only later (3,4,1). The number of lines in the file defines a number of time slots in the run. The given file contains arrivals during two time slots.

For an input file each run returns:
Expected output:
Total arrived packets %d, total dropped packets %d, total transmitted packets %d, total transmitted value %d.

In your make the target for EDF is named **edf** and for the Bounded-delay is named **bd**.

Input_file example:

```
(2,3,6) (3,4,1)
(3,4,5)
```

Running example: edf 4 input_file

```
Total arrived packets 8, total dropped packets 4, total transmitted packets 4, total transmitted value 17.
```

The detailed run of EDF on the sample_input can be found here.

During the Zoom meeting I will give several dynamic examples of the requested policies. After this meeting more examples can be found here.

In this part you will have the implementation of the both policies, README explaining how to compile and run, sample input (sample_input) that you used for testing and the results of the both policies on this sample_input (edf_sample and bd_sample).

2. In the theoretical part you will need the following questions:

a. If all input packets have the same value but heterogeneous slack values, what is preferable EDF or Bounded-delay for the defined objective. Please explain.
b. If all input packets have the same slack but heterogeneous packet values, what is preferable EDF or Bounded-delay for the defined objective. Please explain.
c. In the general case with heterogeneous packet values and slack values when should EDF outperform Bounded delay and vice versa? Explain why.
d. Suggest other policies that can be useful in the defined model for the defined objective. Add cons and pros of all three policies.

All your questions add to 2_<your id>.pdf

For the oral exam: you should clearly understand the problem of congestion control and explain your answers for a-d.

The answers for the both questions are submitted as a single zip file <your_id>.zip containing two folders, one per question.

Zoom recording with explanations:
https://zoom.us/rec/play/u5B8cuv7rDs3SNOVsQSDA_N7W420L6is0SgY_6YIyEvgB3YBNQCkM-BGYes-GLxfAhWCCgqILDIm03Ev?continueMode=true&_x_zm_rtaid=ViSvpBVbSQyp9-luCiV3KA.1595253575090.d4a68ef823cc93f87abdc1461e33c720&_x_zm_rhtaid=617

...