Exercise 1 (4 points)
Assume you have a randomized adversary and an instances set \( \mathcal{I} = \{I_i | 1 \leq i \leq k + 1\} \) with \( k > 8 \). We have two deterministic algorithms \( A_1 \) and \( A_2 \) that try to maximize their gain. We define \( h(j, i) = \frac{\text{gain}(\text{OPT}(I_i))}{\text{gain}(A_j(I_i))} \). The table below is filled with the corresponding \( h(j, i) \). Thus, the table gives \( \forall j, i : \text{gain}(A_j(I_i)) \) and \( \text{gain}(\text{OPT}(I_i)) \).

The adversary chooses the following probability distribution. Instance \( I_i \) with \( i < k \) is chosen with a probability of \( \frac{1}{2i+1} \). Instance \( I_{k+1} \) is chosen with a probability of \( \frac{1}{2} \) and for \( I_k \) remains a probability of \( \frac{1}{2k} \).

<table>
<thead>
<tr>
<th>( I_1 )</th>
<th>( I_2 )</th>
<th>( I_3 )</th>
<th>\ldots</th>
<th>( I_k )</th>
<th>\ldots</th>
<th>( I_k )</th>
<th>( I_{k+1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A_1 )</td>
<td>( \frac{1}{3} )</td>
<td>( \frac{1}{3} )</td>
<td>( \frac{1}{3} )</td>
<td>\ldots</td>
<td>( \frac{1}{3} )</td>
<td>\ldots</td>
<td>( \frac{2k}{2k} )</td>
</tr>
<tr>
<td>( A_2 )</td>
<td>( \frac{1}{5} )</td>
<td>( \frac{1}{5} )</td>
<td>( \frac{1}{5} )</td>
<td>\ldots</td>
<td>( \frac{1}{5} )</td>
<td>\ldots</td>
<td>( \frac{2k}{2k} )</td>
</tr>
</tbody>
</table>

Use Yao’s principle to derive a lower bound for an arbitrary randomized algorithm \( A_R \).

Exercise 2 (4 points)
The cow wants to help the cats on the farm. They are fighting every day over the sleeping places \( S \). The cow knows that the number of cats \( |C| \) is equal to the number of sleeping places \( (|C| = |S|) \). Moreover the cow knows that it is possible that every cat has a sleeping place and is happy (perfect Matching). As a cow, she does the obvious. She creates a list of sleeping places \( S \) and asks every cat \( c_i \in C \) independent of each other which sleeping places \( S_i \subseteq S \) are acceptable.

The cow chooses now arbitrarily for each cat \( c_i \) one of the acceptable sleeping places \( s_i \in S_i \). Then the sleeping place is removed from the list of sleeping places \( S \). If a cat \( c_j \) cannot be assigned to a sleeping place because it was assigned to a different cat \( c_i \) with \( i < j \), the cat is allowed to sleep on the back of a cow. The problem is that there are only half as many cows as cats.

(a) The cats are worried that this approach will not work. Can you give a proof that the cow will always find a sleeping place for at least \( \frac{|C|}{2} \) cats?

(b) The cats realized that it is much more comfortable on the back of a cow than in any sleeping place. They try to trick the cow. They change their order in which they get assigned by the cow and lie about their favored sleeping places. What will the cats do? Make sure that your approach works for arbitrary many cats (more than one). Prove that it is possible that at most \( \frac{|C|}{2} \) cats have a sleeping place.

(c) Do you see a connection to the one-sided online matching problem for graphs with a perfect matching? Write one sentence with a short explanation.

Exercise 3 (3 points)
(a) Use self-delimiting encoding to encode the numbers 4, 13 and 17.

(b) Assume a different way to encode the advice string: Three consecutive bits are interpreted as one sign. The sign 111 denotes the end of the advice string. Argue why this approach is not superior to the method from the lecture.

(c) Derive a method, different from the first encoding presented in the lecture, that uses \( 2 \lceil \log(m) \rceil \) bits to encode the number \( m \).
Exercise 4  

Think of sorting as an online problem. The deterministic online algorithm knows the length of the Instance \( n \). A number \( a_i \in \mathbb{N} \) arrives in time step \( i \) and needs an index \( k_i \in \{1, \ldots, n\} \) such that \( a_{k_i} < a_{k_j} \) for \( k_i < k_j \) holds. If a number \( a_i \) gets a wrong index, the algorithm has to pay a penalty of 1.

(a) Analyze the strict competitive ratio for this problem.

(b) How many bits of advice are necessary to solve the problem optimal? Try to minimize the number of required bits.

Bonus presentation exercises: Write your tutor (fischer@algo.rwth-aachen.de or tarik.viehmann@rwth-aachen.de) a mail and announce that you would like to present a presentation exercise. For every exercise group, only one student is allowed to present an exercise. So, write in your mail which exercise you would like to present and your group number. You are allowed to use the whiteboard and the slides from the lecture.

Bonus Exercise 5  
Lower Bound for Randomized Paging.
Slides: 3:1 to 3:14 (Handout)

Bonus Exercise 6  
Ski Rental.
Slides: 3:18 to 3:32 (Handout)

<table>
<thead>
<tr>
<th>Exercise:</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>Bonus:</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Deadline:** The solutions are to be handed in until June 05, 17:45, in the lecture or at the drop boxes at the Chair i1.