Exercise 1: Rate-Distortion Optimized Quantization

The goal of the exercise is to implement a function

```c
void Quantizer::quantizeRDOQ( const Block& blk, // input block of residual samples
                        int* qIdx, // output quantization indexes
                        int QP, // quantization parameter
                        double lambda, // Lagrange multiplier
                        ... // addition required parameters
                    );
```

This function should determine the quantization indexes $q_k$ in a way that a Lagrangian cost function

$$J(q_0, q_1, \cdots) = D(q_0, q_1, \cdots) + \lambda \cdot R(q_0, q_1, \cdots)$$

of the resulting distortion $D(q_0, q_1, \cdots)$ and the number of bits $R(q_0, q_1, \cdots)$ required for transmitting the quantization indexes $q_k$ is minimized.

- Think about how such a function can be implemented for our codec.
- How can we calculate (or estimate) the distortion $D$ in the transform domain? – What properties has the implemented transform?
- How can we estimate the number of bits $R$ required for transmitting the quantization indexes? – What factors determine the (average) number of bits needed for a certain quantization index?
Exercise 1a: Distortion Estimation

Implement a private function

```cpp
double Quantizer::xDistortion(int absCoeff, // absolute value of the original transform coefficient
                              int absQIdx, // absolute value of quantization index
                              int QP // quantization parameter
);
```

This function should estimate the contribution of the quantization of a single transform coefficient to the total distortion for a transform block. In order to simplify the implementation it is suggested to restrict the consideration to absolute values.

For implementing the function, think about the following:

- How do we obtain a reconstructed transform coefficient for a given the quantization index and a given quantization parameter?
- How do we calculate the distortion (sum of squared differences)?
- How do we have to scale the distortion, so that the distortion in the transform domain corresponds to the distortion in the original signal space?
Exercise 1b: Rate Estimation

Implement a private function

```cpp
double Quantizer::xRate(int absQIdx, // absolute value of quantization index
                        bool isLast, // whether last non-zero index in scanning order
                        ... // additional required parameters
);```

This function should estimate the number of bits that are required for coding a quantization index. Since the number of bits is independent of the sign, it is suggested to restrict the function to absolute values of quantization indexes.

For implementing the function, consider the following:

- Note that the number of bits depends on whether a quantization index is the last non-zero index in scanning order (i.e., the first in coding order) or not.
- For estimating the number of bits that the arithmetic coder produces for a bin (binary decision), you can use the class RateEstFixed and in particular the function RateEstFixed::encodeBin(\ldots).
  - The function returns the number of bits multiplied by $2^{15}$.
  - It has to be used with the correct probability model.
Exercise 1c: Putting Things Together

Implement the function `Quantizer::quantizeRDQO(..)` with help of the functions:

```cpp
double Quantizer::xDistortion( int absCoeff, int absQIdx, int QP ); // distortion estimation
double Quantizer::xRate( int absQIdx, bool isLast, ... ); // rate estimation
```

- For simplifying the implementation, use the absolute values of the transform coefficients and quantization indexes. The signs can be added as a last step.

- For each absolute transform coefficient, it is sufficient to consider two potential quantization indexes (see lecture):
  - the quantization index obtained by mathematically correct rounding;
  - the quantization index obtained by rounding towards zero.

- The dependencies between quantization indexes (that are introduced due to the coding of the last x/y location) can be elegantly considered by a simple trellis structure with two states:
  
  1. All quantization indexes that precede the current scan position in coding order are zero (and, thus, no rate is spend for coding these indexes).
  2. At least one of the preceding quantization indexes is coding order is not equal to zero.
Exercise 1d: Selection of Lagrange Multiplier & Testing

Selection of the Lagrangian multiplier

- As shown in the lecture, the Lagrange multiplier has a strong relationship to the quantization step size, which can be characterized by

\[ \lambda = \alpha \cdot \Delta^2 = \alpha \cdot 2^{QP/2} \]

- According to the theory of high-rate quantization, the factor \( \alpha \) should be roughly

\[ \alpha = \frac{2 \ln 2}{12} \approx 0.05 \]

- Experiment with factors \( \alpha \) in the range of 0.01 to 0.25 and choose the one that provides the best coding efficiency on average

Testing

- Run simulation for all test images with the simple quantization (rounding) and the implemented rate-distortion optimized quantization (RDOQ).
- Compare the obtained rate-distortion curves.
- Does the implemented optimized quantization improve coding efficiency?
- What impact has the improved quantization on the encoding run time?