Exercice 1

```
/**
 *
 * @return
 */
public double[] textureDescriptor1st(){
    double[] hist = this.normalizedHistogram();
    double[] ret = new double[4];
    double val;
    for(int i=0; i<255;i++){
        ret[0] += i*hist[i];
    }
    for(int i=0; i<255;i++){
        ret[1] += Math.pow(i-ret[0], 2)*hist[i];
        ret[2] += Math.pow(hist[i], 2);
        val = hist[i];
        if(val == 0)val=0.000001;
        ret[3] = val;
    }
    ret[1] = Math.sqrt(ret[1]);
    ret[3] = -ret[3];
    return ret;
}
```

Exercice 2

```
/**
 *
 */
private Image imRescale(int p){
    Image ret = new Image(this.getWidth(), this.getHeight());
    for ( int y=0; y<this.getHeight(); y++){
        for (int x=0; x<this.getWidth(); x++){
            ret.setValue(x,y, (int)Math.round((this.getValue(x,y)*(p-1))/(double)255 ) );
        }
    }
    return ret;
}
```

Exercice 3

```
/**
 *
 */
public int[][] glcm(int p, int[] dxdy){
    Image img = this.imRescale(p);
    int[][] ret = new int[p][p];
    int dx = dxdy[0];
    int dy = dxdy[1];
    for (int x=0; x<this.getWidth(); x++){
        for (int y=0; y<this.getHeight(); y++){
            ret[img.getValue(x,y)][img.getValue(x+dxdy[0],y+dxdy[1])]++;
        }
    }
    return ret;
}
```

Exercice 4

```
/**
 *
 */
public double[][] normalizedGlcm(int p,int[] dxdy){
    Image img = this.imRescale(p);
    double[][] ret = new double[p][p];
    double divide = this.getWidth() * this.getHeight();
    for (int x=0; x<this.getWidth(); x++){
        for (int y=0; y<this.getHeight(); y++){
            ret[img.getValue(x,y)][img.getValue(x+dxdy[0],y+dxdy[1])]++;
```

```
            }
        }
        for (int x=0; x<this.getWidth(); x++){
            for (int y=0; y<this.getHeight(); y++){
                ret[img.getValue(x,y)][img.getValue(x+dxdy[0],y+dxdy[1])] /= divide;
            }
        }
        return ret;
    }
```