========================
BUILD OUTPUT DESCRIPTION
========================
Input:
For Low Rank Approximation, there are 4 inputs. First input is "**3"** as it is the third part of the project. The second input is "**header.txt**" which contains the width, height and grayscale of the image. The third input is the "**SVD.txt**" which contains the matrices **U**, Σ and $\mathbf{V}^T$ after the Singular Value Decomposition of the image. And the fourth input is the value of **k** rank up to which the image will be compressed.

For decompressing the image, there are 2 inputs. First input is "**4"** as it is the third part of the project. The second input is "**image_b.pgm.SVD**" which is generated while storing the magic number, width, height, grayscale, value of k, and k columns of **U**, k Singular values of Σ and k rows of $\mathbf{V}^T$ matrix in part 3. So part 4 cannot be run before part 3 as the **image_b.pgm.SVD** which is input of part 4 will be generated as output of part 3.

Purpose of the jar file:

When the jar file is run, it reads the input files, it creates a file image_b.pgm.SVD that contains the magic number, width, height, grayscale, k, k columns of **U**, k Singular values of Σ and k rows of $\mathbf{V}^T$ in part 3. We also generate a file called image_b(value of k).compression.txt without the additional spaces in between the elements to measure the compression rate after low rank approximation. Moreover, the data of the matrices are stored as short not the float values of the SVD after bitwise operation on them to get 10 bits mantissa, 5 bits exponent and 1 bit for sign.

For part 4, the jar file performs the matrix multiplication $P = (\Sigma)*(\mathbf{V}^T)$ and then $A = (U)*P$ and puts in the magic number, width, height, grayscale and the A matrix to get the decompressed image.

Required Library:

None

Required command:

To run for any image in part 3, the header.txt, and SVD.txt should be in the same directory as svdargs.jar
To run the part 3 from the command line, go to the folder containing the jar and type the following:

```
java -jar svdargs.jar 3 header.txt SVD.txt 20
```

Here 20 is the rank up to which the matrix values are going to be stored.

To run for any image in part 4, the image_b.pgm.SVD will be generated in the same directory as the svdargs.jar

The command will be

```
java -jar svdargs.jar 4 image_b.pgm.SVD
```

Output:

The output **"image_b.pgm.SVD"** is generated while storing the magic number, width, height, grayscale, value of k, and k columns of **U**, k Singular values of $\sum$ and k rows of $\mathbf{V}^T$ matrix in part 3. It is truncating all the other values in **U**, $\sum$, $\mathbf{V}^T$ for k rank approximation where k_{max} = min(width, height).

The output **"image_b(value of k).compression.txt"** is without the additional spaces in between the elements in **"image_b.pgm.SVD"** to measure the compression rate after low rank approximation.

The output **"image_k.pgm"** provides us with the image before applying SVD after low rank approximation.