

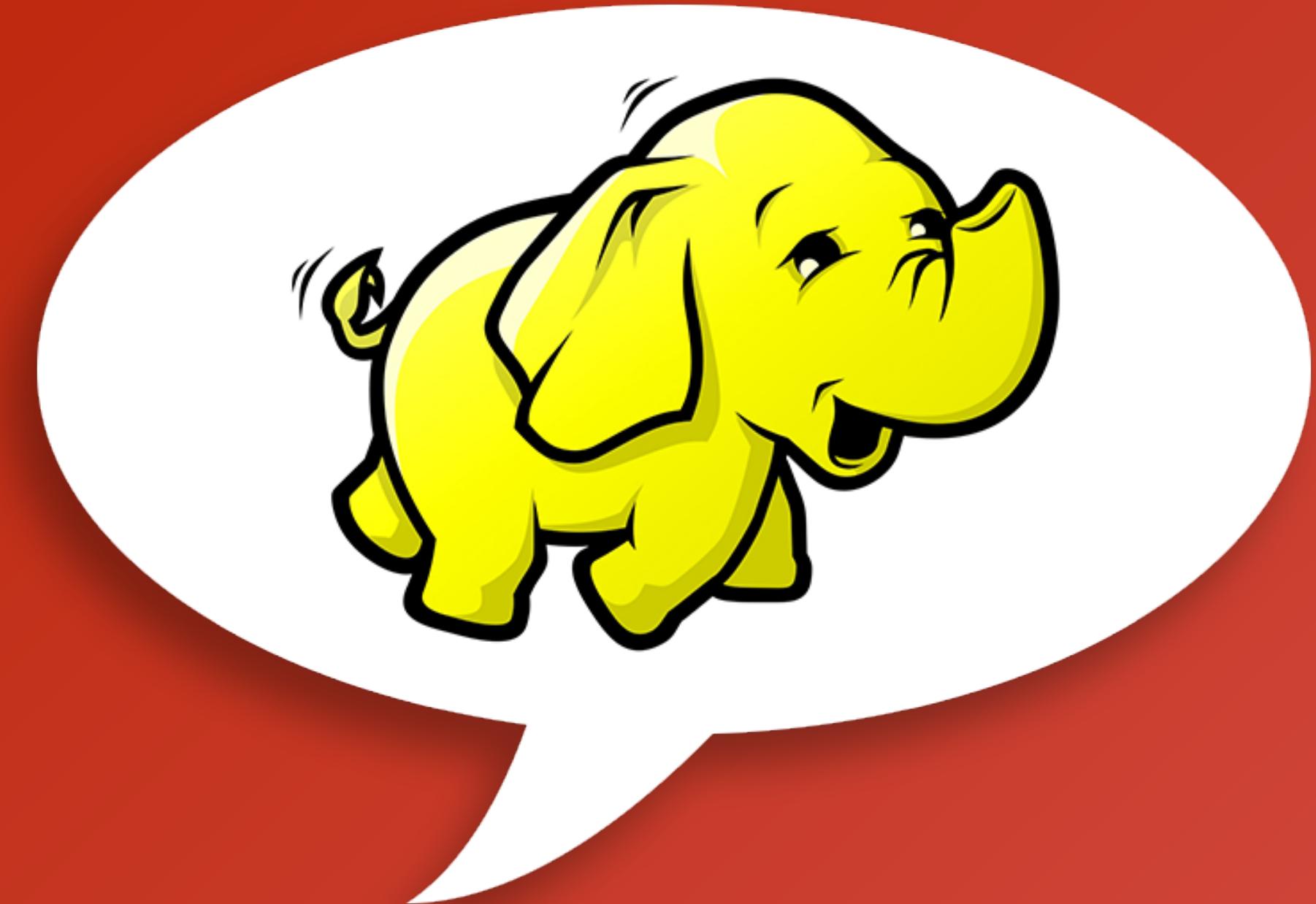
BRAND > code

Shane Curcuru, VP Brand Management, ASF



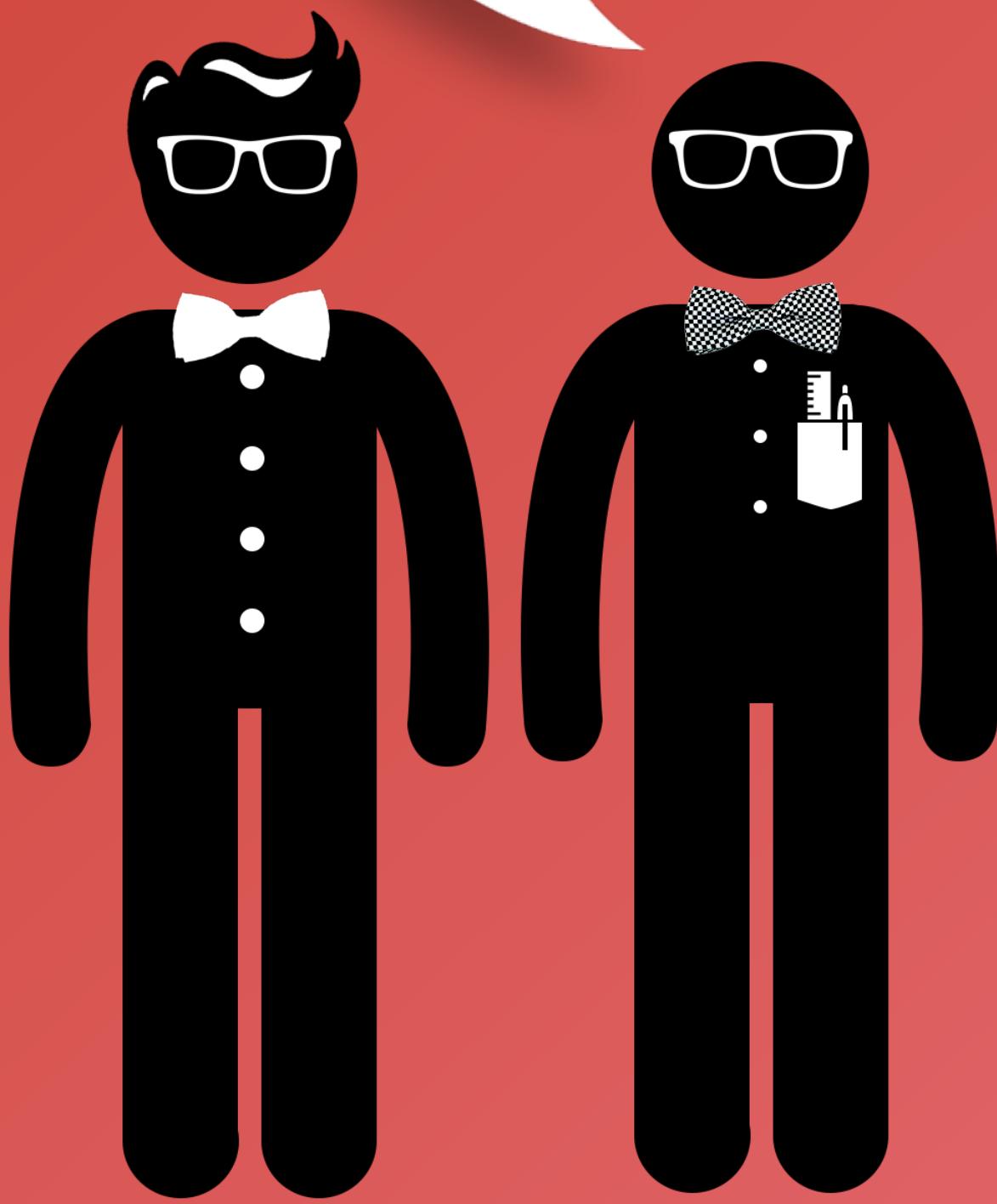
APACHE

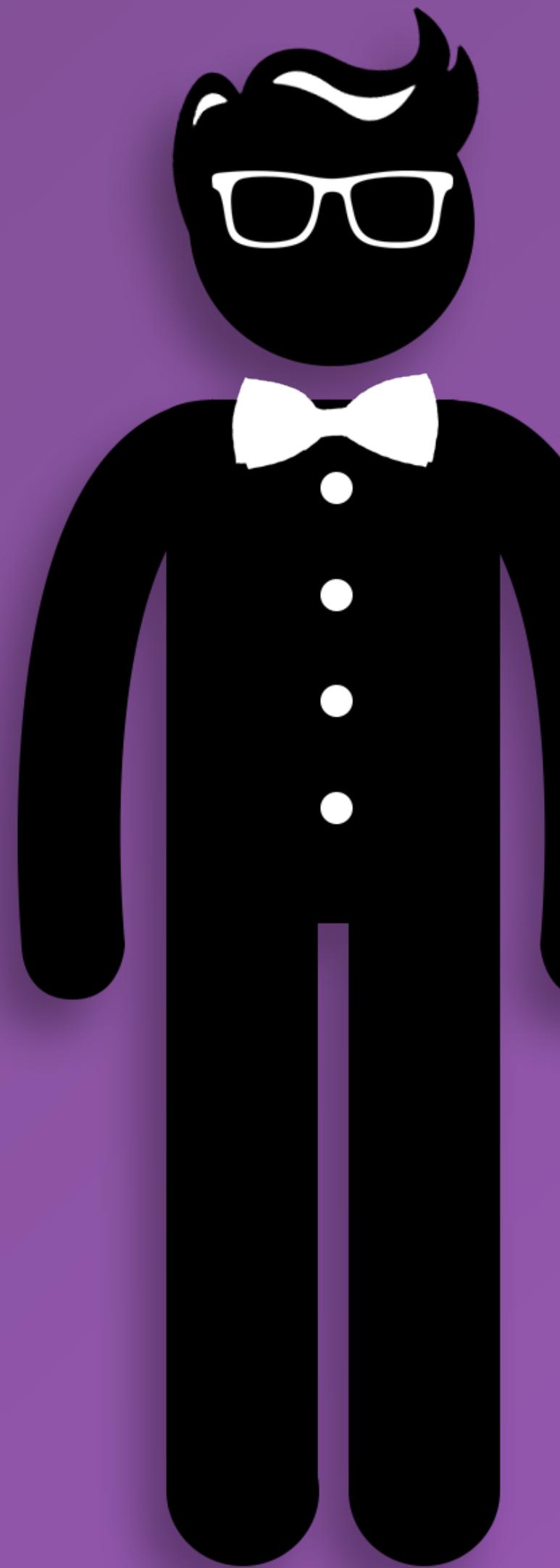




```
public static void main(String[] args)
{
    String[] argsTemp = { "project_test/input", "project_test/output" };

    Configuration conf = new Configuration();
    conf.set("fs.default.name", "hdfs://localhost:54310");
    conf.set("mapred.job.tracker", "localhost:54311");
    conf.set("mapred.jar", "JAR_Files/Hadoop_Example_04.jar");
    String[] otherArgs = new GenericOptionsParser(conf).getRemainingArgs();
    Job job = new Job(conf, "Example Hadoop 0.20.2 WordCount");
    job.setJarByClass(Hadoop_Example_04.class);
    job.setMapperClass(TokenCounterMapper.class);
    job.setReducerClass(TokenCounterReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    Path jobInputPath = new Path(jobs[0].getAbsolutePath());
    Path jobOutputPath = new Path(jobs[1].getAbsolutePath());
}
```



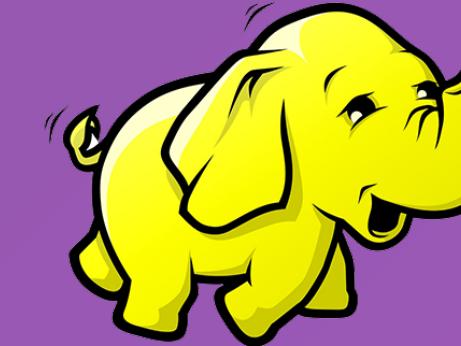
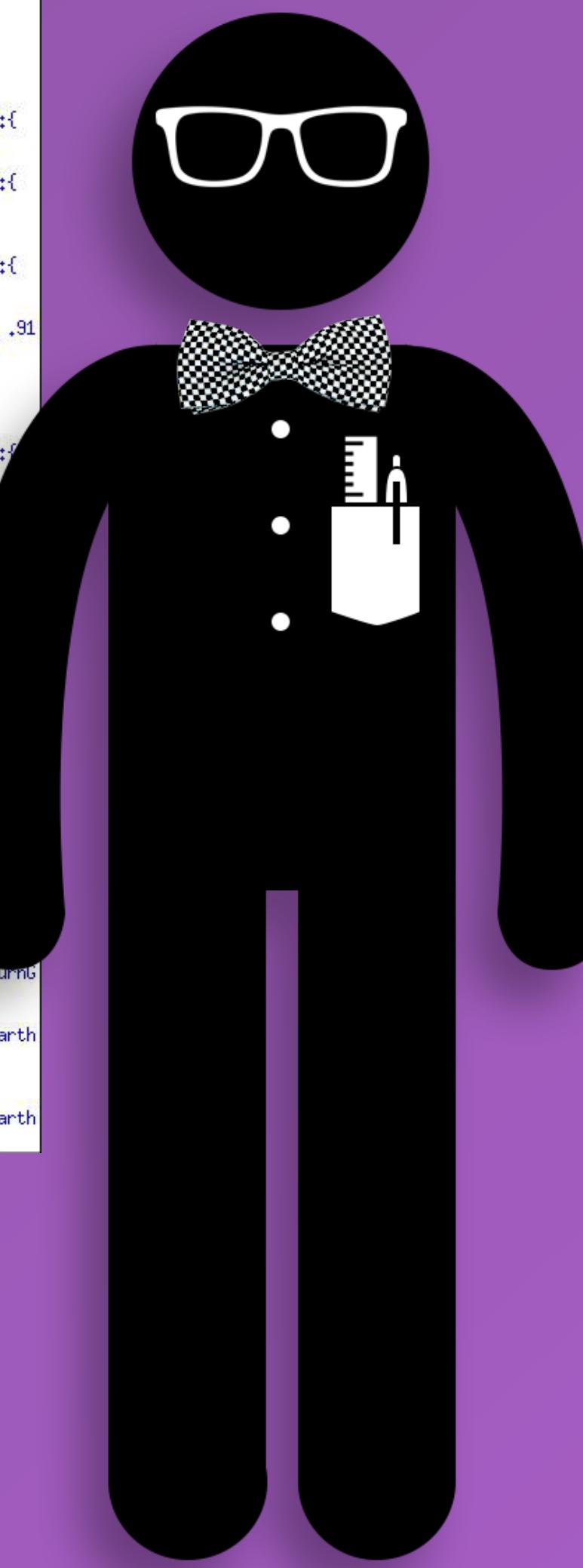


```
</>
<8> int mac_cathy (int a)
<9> {
<10>   int b;
<11>
<Union 223 envs> [mac_cathy:1:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .101.> }::{ <a,(Local),i_-10. . .111,>, <b,(Local), Bot > }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
<12>   if (a > 100)
<Union 223 envs> [mac_cathy:4:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .101.> }::{ <a,(Local),i_-10. . .111,>, <b,(Local), Bot > }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
<13>   {
<Union 223 envs> [mac_cathy:5:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .100.> }::{ <a,(Local),i_102. . .111,>, <b,(Local), Bot > }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
<14>   return (a-10);
<Union 223 envs> [mac_cathy:15:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .101.> }::{ <a,(Local),i_-10. . .111,>, <b,(Local),i_91. . .91.> }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
<15>   }
<16>   else
<17>   {
<Union 223 envs> [mac_cathy:9:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .100.> }::{ <a,(Local),i_-10. . .100,>, <b,(Local), Bot > }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
<18>   b = mac_cathy (mac_cathy (a+11));
<Union 223 envs> [mac_cathy:14:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .91.> }::{ <a,(Local),i_-10. . .100,>, <b,(Local),i_91. . .91.> }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
<Union 223 envs> [mac_cathy:14:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .91.> }::{ <a,(Local),i_-10. . .100,>, <b,(Local),i_91. . .91.> }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
<19>   return (b);
<Union 223 envs> [mac_cathy:15:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .101.> }::{ <a,(Local),i_-10. . .111,>, <b,(Local),i_91. . .101.> }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] []
of function
<Union 223 envs> [mac_cathy:15:mac_cathy.c]::{ <mac_cathy,(Global), Bot >, <mac_cathy_return,(Global),i_91. . .101.> }::{ <a,(Local),i_-10. . .111,>, <b,(Local),i_91. . .101.> }:{ <x1,(Local),i_0. . .1,>, <x2,(Local),i_100. . .100,>, <x3,(Local),i_-10. . .-10,>, <x4,(Local),i_10. . .10,>, <x5,(Local),i_91. . .101,>, <x6,(Local),i_1. . .11,> }:{ No signal }:{ To do }[mac_cathyG mac_cathy_returnG ] [aL bL ] [.x2L .x1L .x4L .x3L .x7L .x6L .x5L ]
}
}

int main (int argc, char * argv[])
{
  int x;

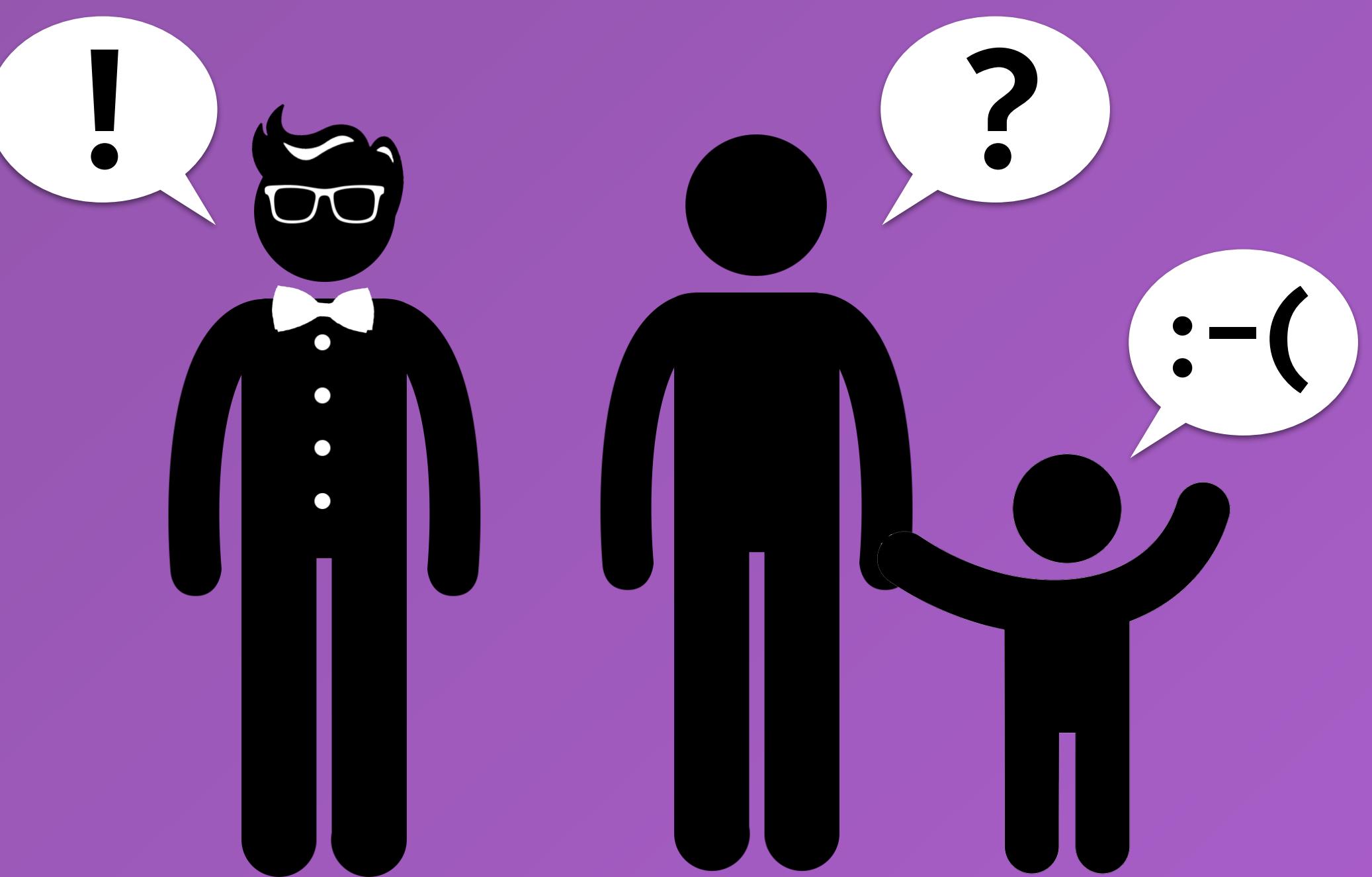
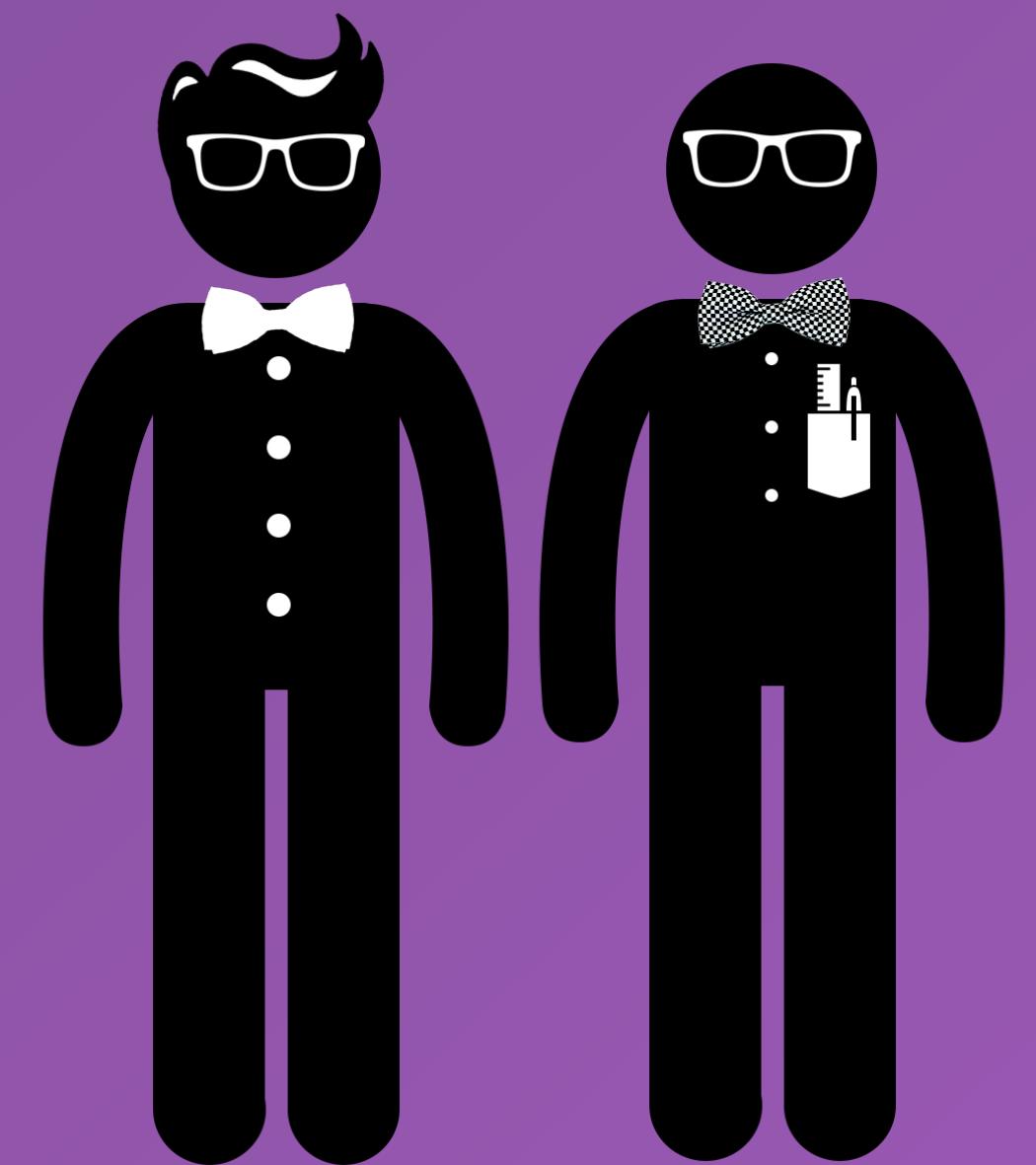
<Union 1 envs> [main:1:mac_cathy.c]::{ <mac_cathy_return,(Global), Bot >, <main,(Global), Bot > }::{ <x,(Local), Bot > }:{ No signal }:{ To do }[mainG mac_cathy_returnG ] [xL ] []
<28> x = -10;
<Union 1 envs> [main:3:mac_cathy.c]::{ <mac_cathy_return,(Global), Bot >, <main,(Global), Bot > }::{ <x,(Local),i_-10. . .-10,> }:{ No signal }:{ To do }[mainG mac_cathy_returnG ] [xL ] []
<29>

<Union 1 envs> [main:3:mac_cathy.c]::{ <mac_cathy_return,(Global), Bot >, <main,(Global), Bot > }::{ <x,(Local),i_-10. . .-10,> }:{ No signal }:{ To do }[mainG mac_cathy_returnG ] [xL ] []
```



```
public static void main(String[] args)
{
    argsTemp = { "project_test/input" };

    Configuration conf = new Configuration();
    conf.set("fs.default.name", "hdfs://localhost:54310");
    conf.set("mapred.job.tracker", "localhost:54311");
    conf.set("mapred.jar", "JAR_Files/Hadoop_Example_04.jar");
    String[] otherArgs = new GenericOptionsParser(conf).getRemainingArgs();
    Job job = new Job(conf, "Example Hadoop 0.20.2 WordCount");
    job.setJarByClass(Hadoop_Example_04.class);
    job.setMapperClass(TokenCounterMapper.class);
    job.setReducerClass(TokenCounterReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    Path jobOutputPath = new Path(job.getConfiguration().get("outputPath"));
    jobOutputPath.delete(true);
}
```



```

</>
<8> int mac_cathy (int a)
<9> {
<10>   int b;
<11>

223 envs> [mac_cathy:1:mac_cathy.c]:{: <mac_cathy,(Global), Bot>, <mac_cathy_return,(Global),i_91..101.>}:{<a,(Local),i_-10..111,>, <b,(Loca
gnal}:{ To do }[mac_cathyG mac_cathy_returnG ] [al bl ] []
(a > 100)
223 envs> [mac_cathy:4:mac_cathy.c]:{: <mac_cathy,(Global), Bot>, <mac_cathy_return,(Global),i_91..101.>}:{<a,(Local),i_-10..111,>, <b,(Loca
gnal}:{ To do }[mac_cathyG mac_cathy_returnG ] [al bl ] []
start_encoding_picture: // Begin encoding a video picture
input desired D; // Get the desired Distortion D value
float desired_D; // And the D value relates to the desired D
Qnorm = iD0; // Determine normal Q with no masking
lambda = f(Qnorm); // Determine the Lagrange multiplier lambda
start_coding_pixelblock: // Start encoding a pixelblock
Q = Qnorm; // Set to the norml Q with no masking
calculate visual mask M; // Determine the visual masking amount
while(lambda < Lambda_min(Qnorm) || Q > Lambda_max(Qnorm)) { // If lambda too small or too big
    Q = Q+deltaQ; // Raise the Quantizer Q size
    code pixelblock(M,lambda,Q); // Encode using M,lambda and Q
    if(encoder.buffer < TH1if){ // If buffer threatens to fill overflow
        lambda = lambda+deltaLambda; // Increase lambda
        if (lambda > Lambda_max(Qnorm)X // Test lambda
            Qnorm=Qnorm+deltaQ; // Increase Q if lambda too big
            lambda = f(Qnorm); // Calculate new lambda
        )
    }
    if(encoder.buffer < Tempif{ // If buffer threatens to fill underflow
        lambda = lambda-deltaLambda; // Decrease lambda
        if (lambda < Lambda_min(Qnorm)X // Test lambda
            Qnorm=Qnorm-deltaQ; // Decrease Q if lambda too small
            lambda = f(Qnorm); // Calculate new lambda
        )
    }
}
if (not last pixelblock) then goto start_encoding_pixelblock //Next
// Done with picture.

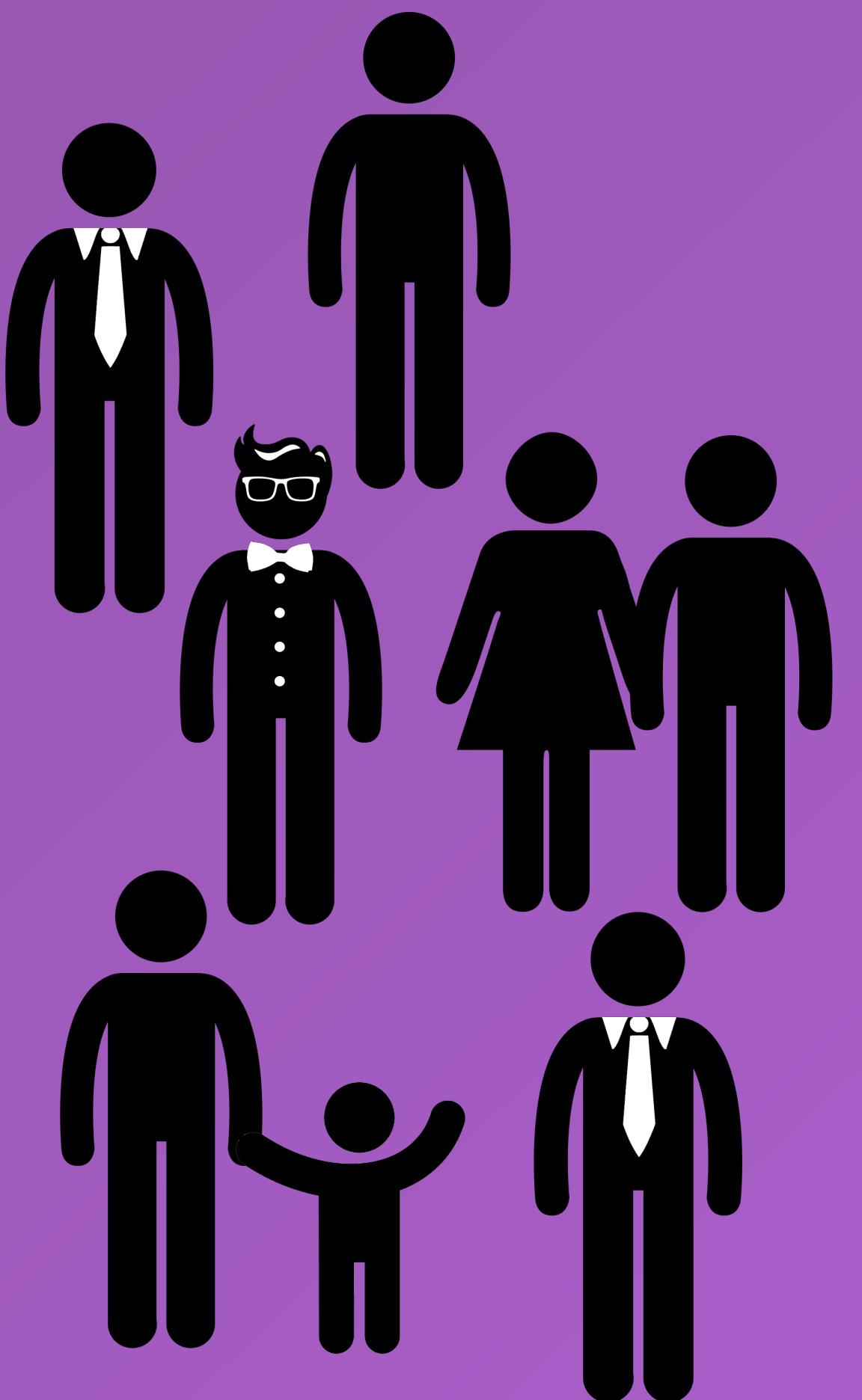
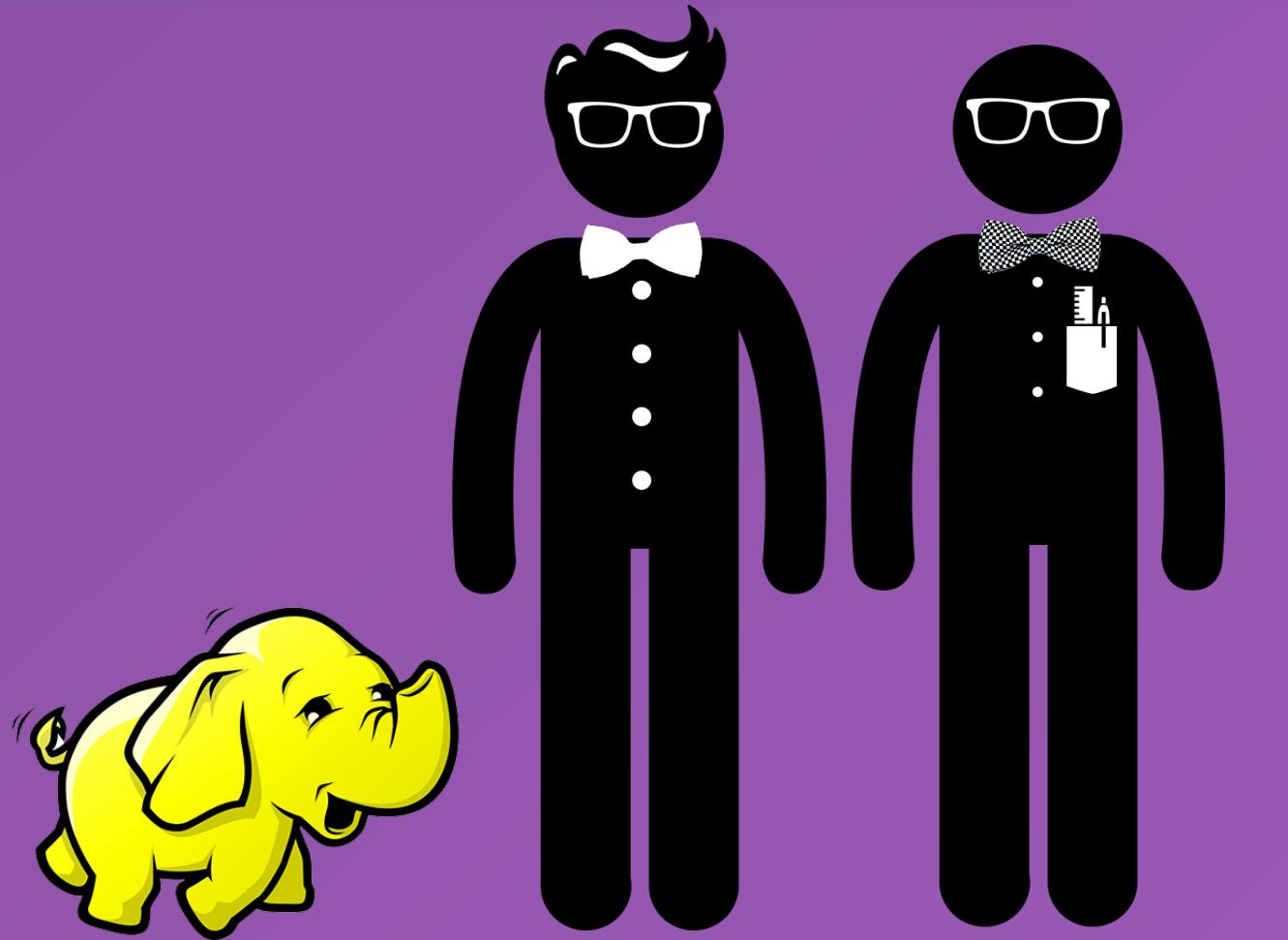
<Union 223 envs> [mac_cathy:9:mac_cathy.c]:{: <mac_cathy,(Global), Bot>, <mac_cathy_return,(Global),i_91..100.>}:{<a,(Local),i_-10..100,>, <b,(Local), Bot>}:{<
}:[ No signal}:{ To do }[mac_cathyG mac_cathy_returnG ] [al bl ] []
<18>   b = mac_cathy (a+11);
<Union 223 envs> [mac_cathy:14:mac_cathy.c]:{: <mac_cathy,(Global), Bot>, <mac_cathy_return,(Global),i_91..91.>}:{<a,(Local),i_-10..100,>, <b,(Local),i_91..91.
>}:{< }:[ No signal}:{ To do }[mac_cathyG mac_cathy_returnG ] [al bl ] []
<19>   return (b);
<Union 223 envs> [mac_cathy:15:mac_cathy.c]:{: <mac_cathy,(Global), Bot>, <mac_cathy_return,(Global),i_91..101.>}:{<a,(Local),i_-10..111,>, <b,(Local),i_91..91.
>}:{< }:[ No signal}:{ To do }[mac_cathyG mac_cathy_returnG ] [al bl ] []

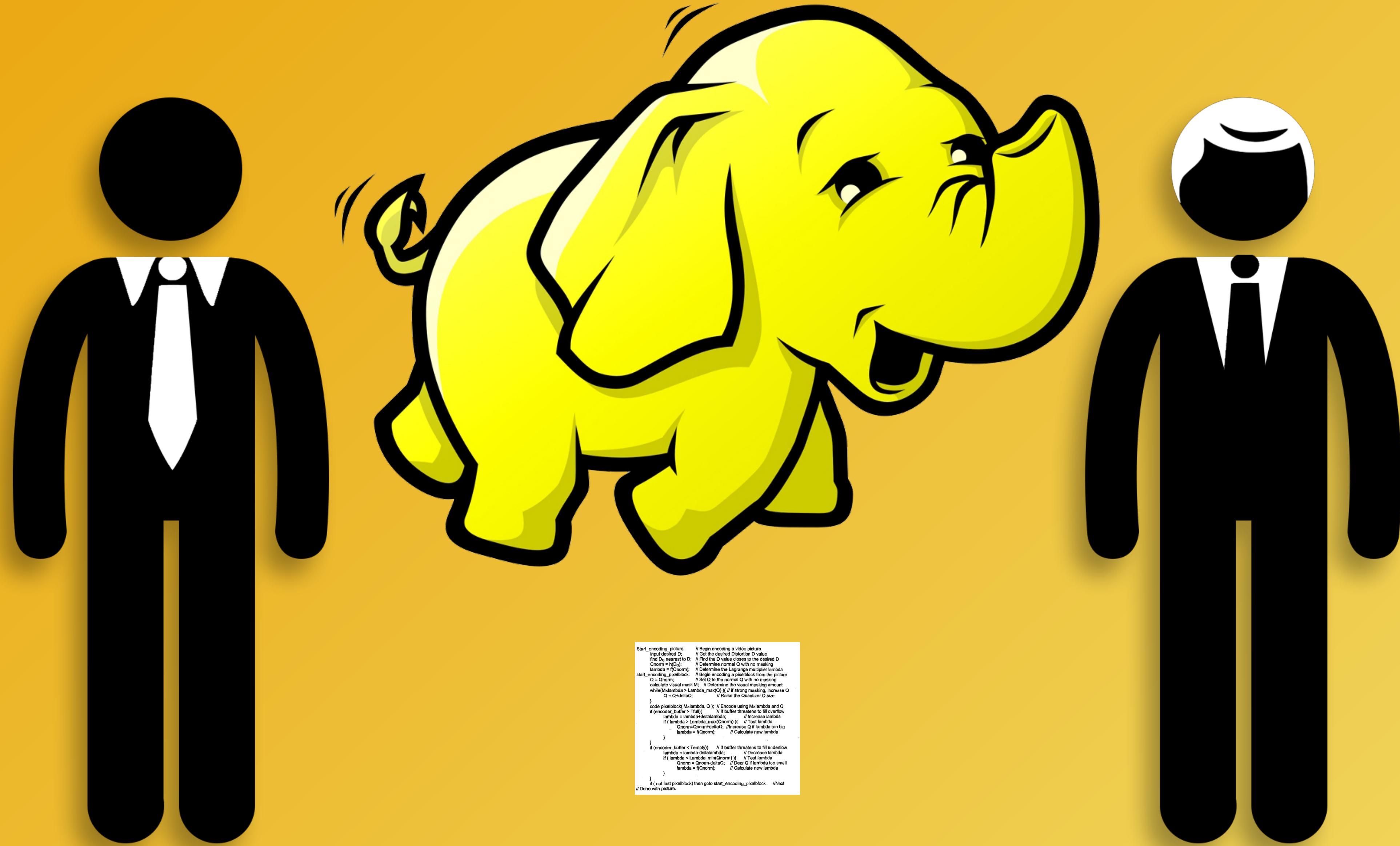
End of function
223 envs> [mac_cathy:15:mac_cathy.c]:{: <mac_cathy,(Global), Bot>, <mac_cathy_return,(Global),i_91..101.>}:{<a,(Local),i_-10..111,>, <b,(Loca
gnal,i_0..1,>, <x2,(Local),i_100..100,>, <x3,(Local),i_-10..-10,>, <x4,(Local),i_10..10,>, <x5,(Local),i_91..101,>, <x6,(Local),i_11..111,>}:{< }:[ No signal}:{ To do }[mac_cathyG mac_cathy_returnG ] [al bl ] [,x2L,x1L,x4L,x3L,x7L,x6L,x5L]

main (int argc, char * argv[])
:x;
envs> [main:1:mac_cathy.c]:{: <mac_cathy_return,(Global), Bot>, <main,(Global), Bot>}:{<x,(Local), Bot>}:{< }:[ No signal}:{ To do }[mainG mac
y_returnG ] [xl ] []
:-10;
envs> [main:3:mac_cathy.c]:{: <mac_cathy_return,(Global), Bot>, <main,(Global), Bot>}:{<x,(Local),i_-10..-10,>}:{< }:[ No signal}:{ To do }[mainG mac_c
y_returnG ] [xl ] []
223 envs> [main:3:mac_cathy.c]:{: <mac_cathy_return,(Global), Bot>, <main,(Global), Bot>}:{<x,(Local),i_-10..-10,>}:{< }:[ No signal}:{ To do }[mainG mac_c
y_returnG ] [xl ] []

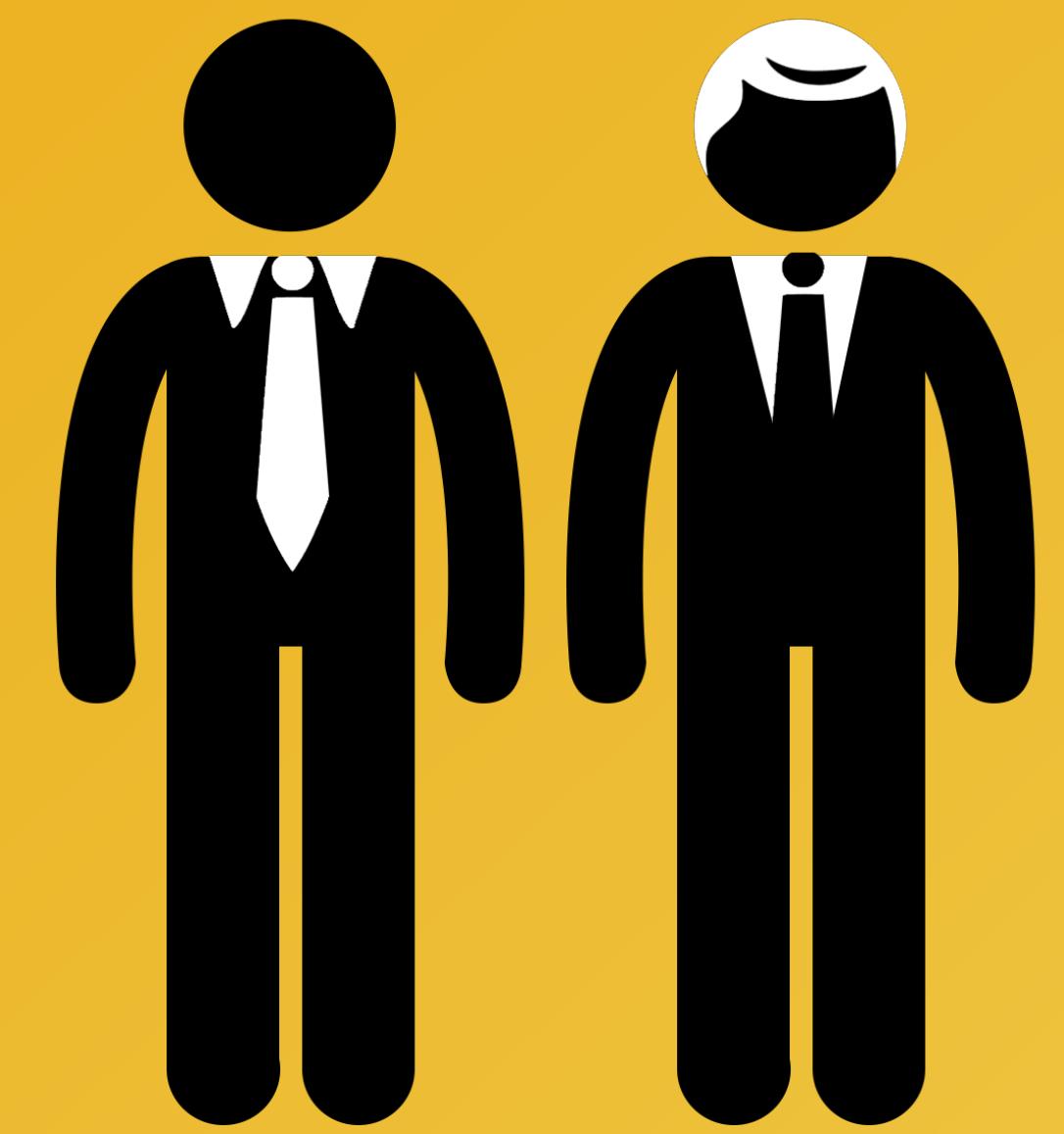
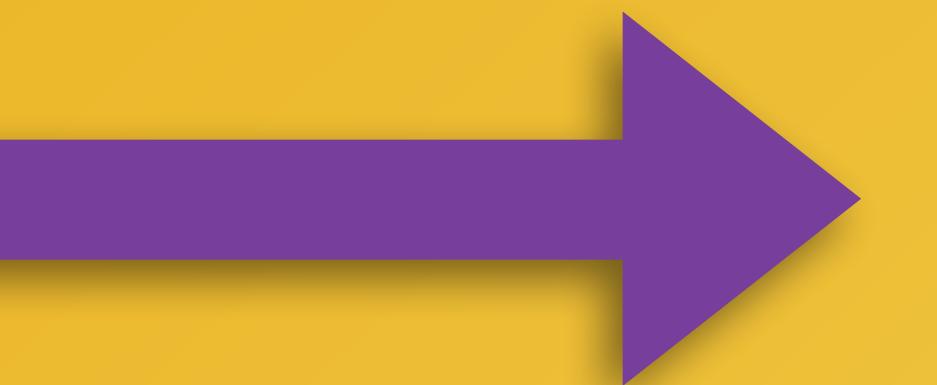
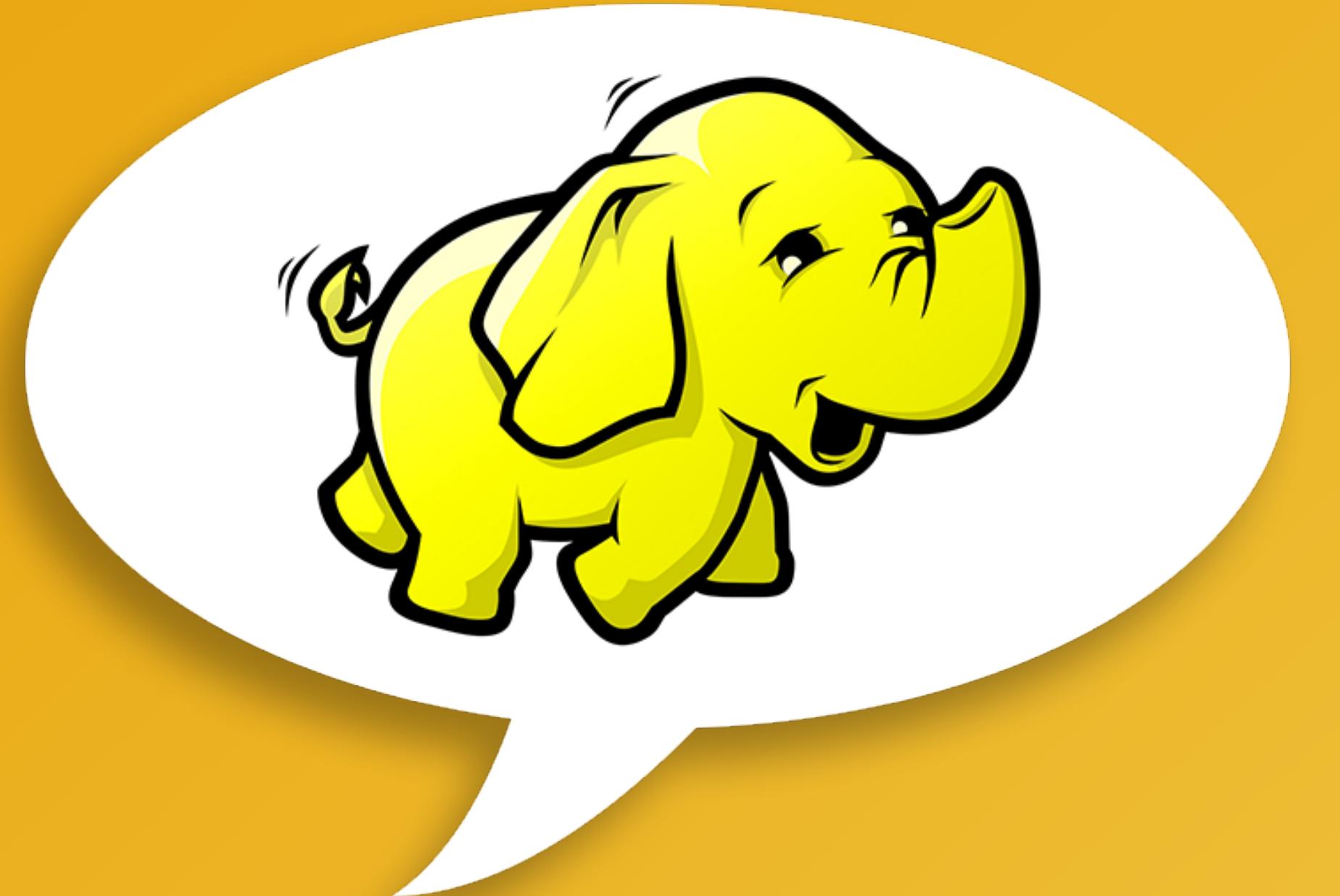
Start_encoding_picture: // Begin encoding a video picture
input desired D; // Get the desired Distortion D value
float desired_D; // And the D value relates to the desired D
Qnorm = iD0; // Determine normal Q with no masking
lambda = f(Qnorm); // Determine the Lagrange multiplier lambda
start_coding_pixelblock: // Start encoding a pixelblock
Q = Qnorm; // Set to the norml Q with no masking
calculate visual mask M; // Determine the visual masking amount
while(lambda < Lambda_min(Qnorm) || Q > Lambda_max(Qnorm)) { // If lambda too small or too big
    Q = Q+deltaQ; // Raise the Quantizer Q size
    code pixelblock(M,lambda,Q); // Encode using M,lambda and Q
    if(encoder.buffer < TH1if){ // If buffer threatens to fill overflow
        lambda = lambda+deltaLambda; // Increase lambda
        if (lambda > Lambda_max(Qnorm)X // Test lambda
            Qnorm=Qnorm+deltaQ; // Increase Q if lambda too big
            lambda = f(Qnorm); // Calculate new lambda
        )
    }
    if(encoder.buffer < Tempif{ // If buffer threatens to fill underflow
        lambda = lambda-deltaLambda; // Decrease lambda
        if (lambda < Lambda_min(Qnorm)X // Test lambda
            Qnorm=Qnorm-deltaQ; // Decrease Q if lambda too small
            lambda = f(Qnorm); // Calculate new lambda
        )
    }
}
if (not last pixelblock) then goto start_encoding_pixelblock //Next
// Done with picture.

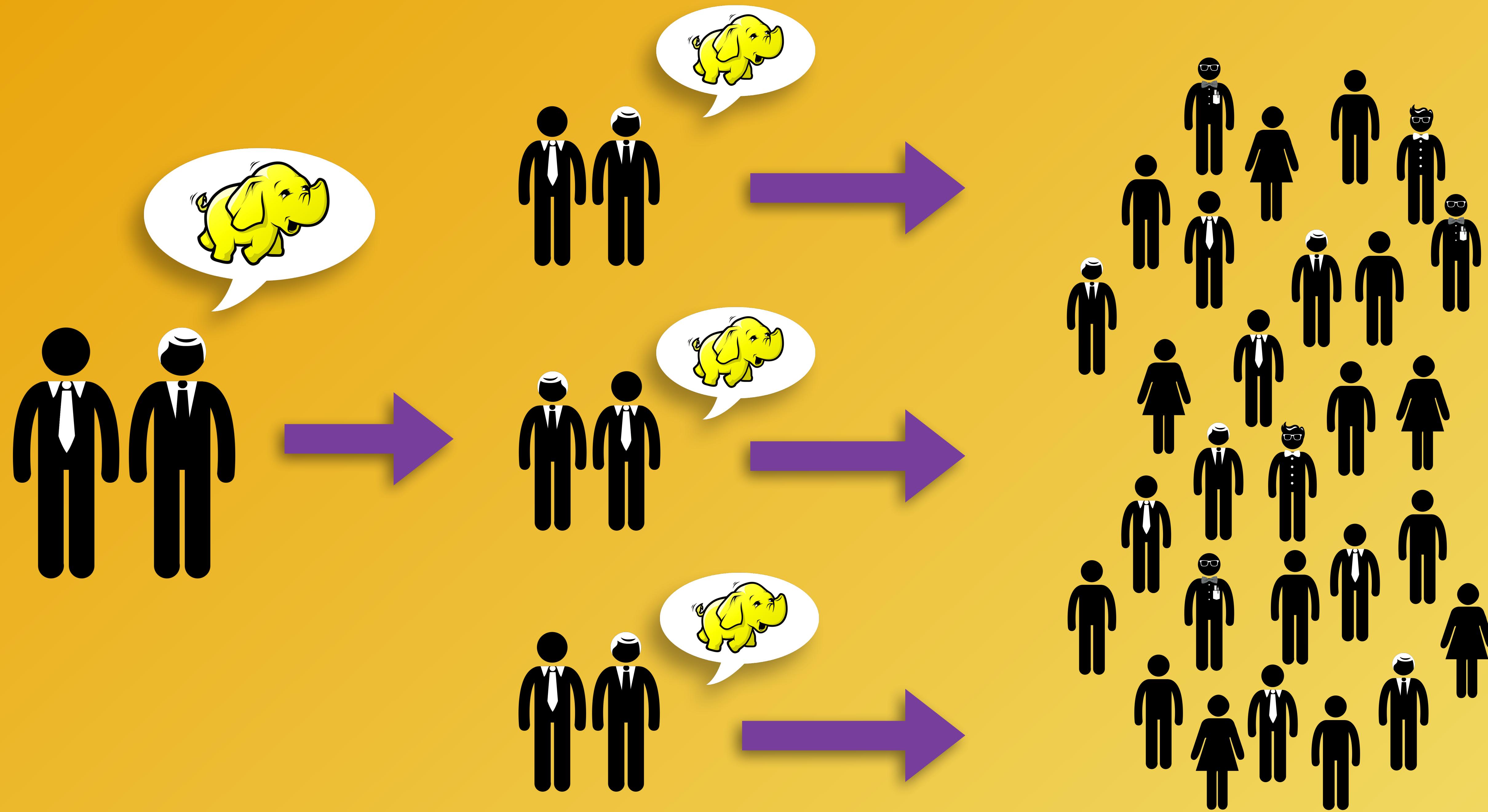
```

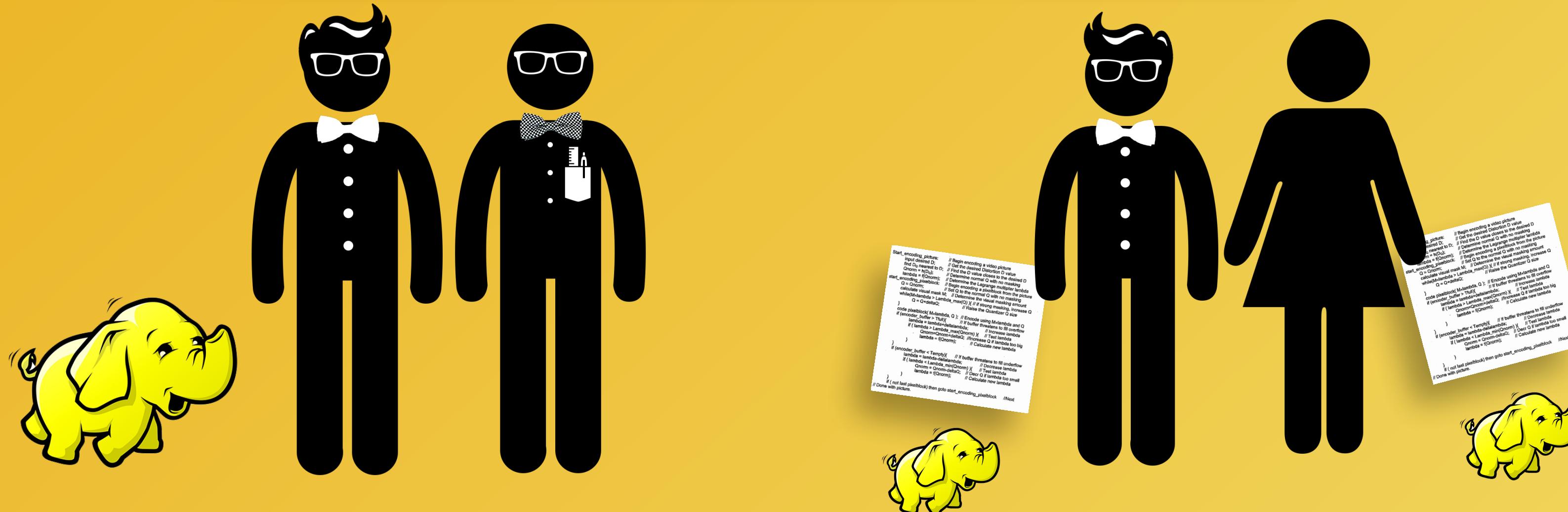


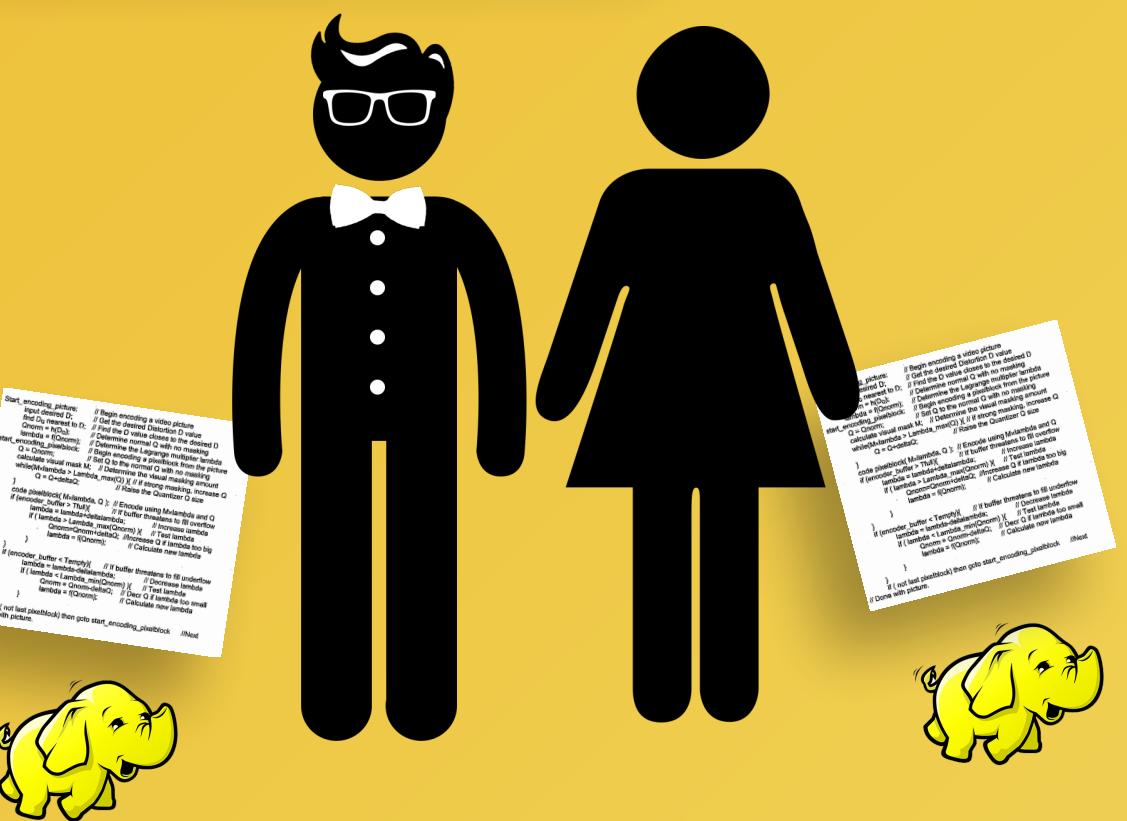
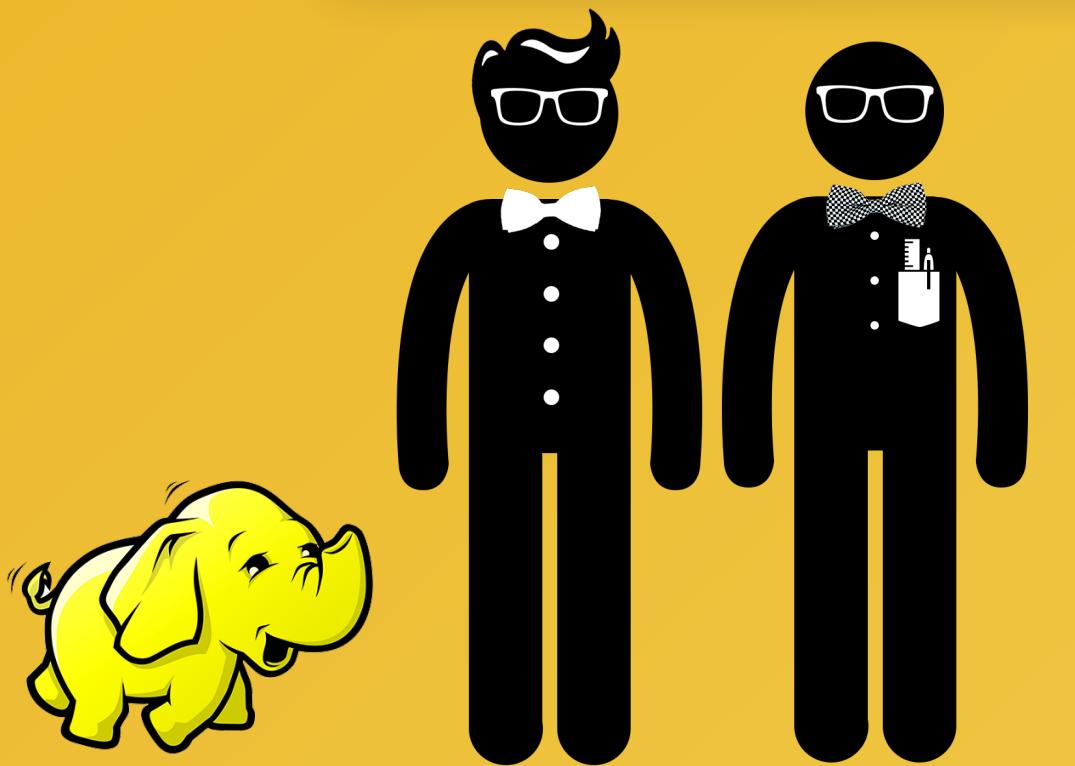


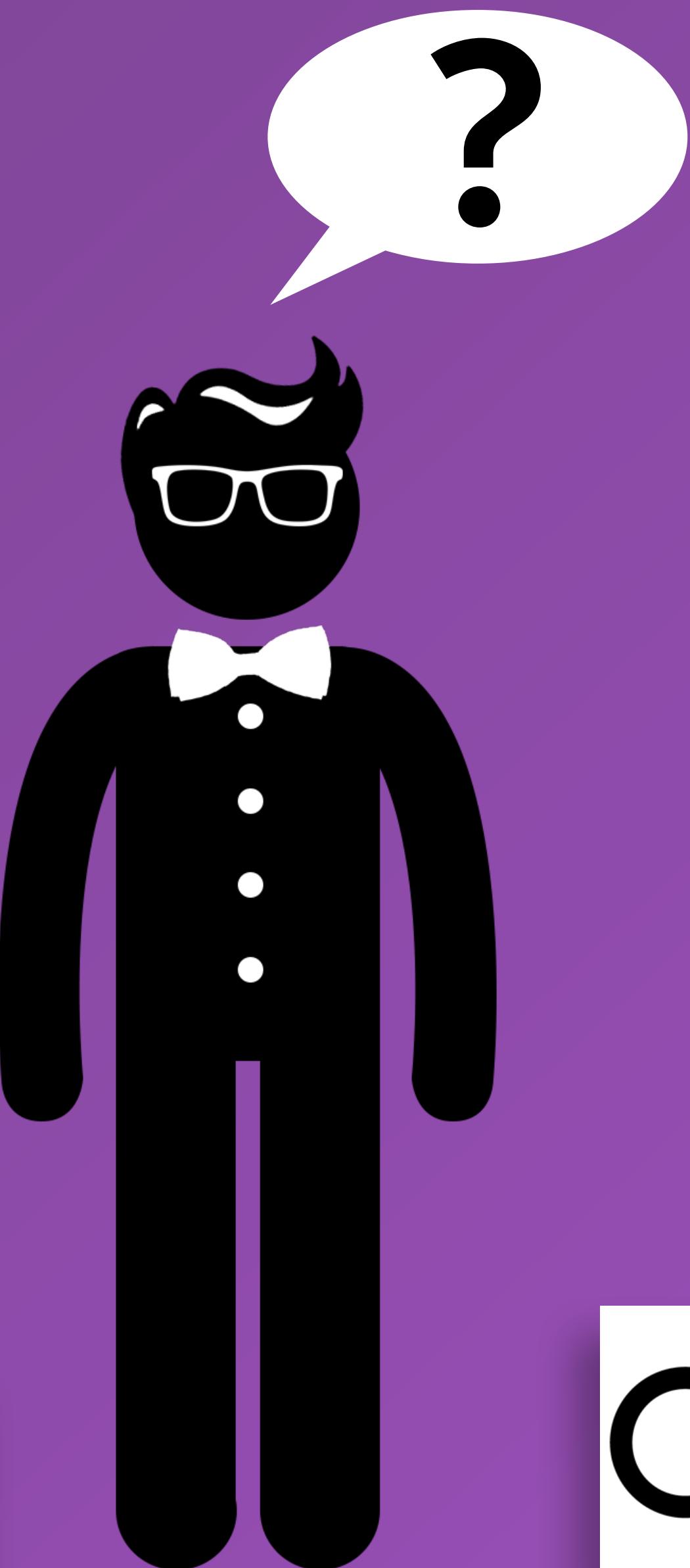
```
Start_encoding_picture: // Begin encoding a video picture
    Input_desired_D; // Get the desired Distortion D value
    find_D0_nearest_to_D; // Find the D value closes to the desired D
    Q=Q0;
    if (Q>Qmax) {
        Q=Qmax;
    }
    lambda = f(Q); // Determine the Lagrange multiplier lambda
start_encoding_pixelblock: // Begin encoding a pixelblock from the picture
    D = D0;
    calculate_visual_mask_M; // Determine the visual masking amount
    while(M*lambda > Lambda_maxQ) { // if strong masking, increase Q
        Q = Q+delta_Q;
    }
    code_pixelblock M*lambda, Q); // Encode using M*lambda and Q
    if (encoder.buffer < TempBuf) { // If buffer threatens to fill
        lambda = lambda+delta_lambda; // Increase lambda
        if (lambda > Lambda_maxQnorm) X; // Test lambda
        Qnorm=Qnorm+deltaQ; // Increase Q if lambda too big
        lambda = Qnorm;
    }
}
if (encoder.buffer < TempBuf) { // buffer threatens to fill underflow
    lambda = lambda-delta_lambda;
    if (lambda < Lambda_minQnorm) X; // Decrease lambda
    Qnorm=Qnorm-deltaQ; // Decrease Q if lambda too small
    lambda = Qnorm;
}
if (not last pixelblock) then goto start_encoding_pixelblock //Next
// Done with picture.
```

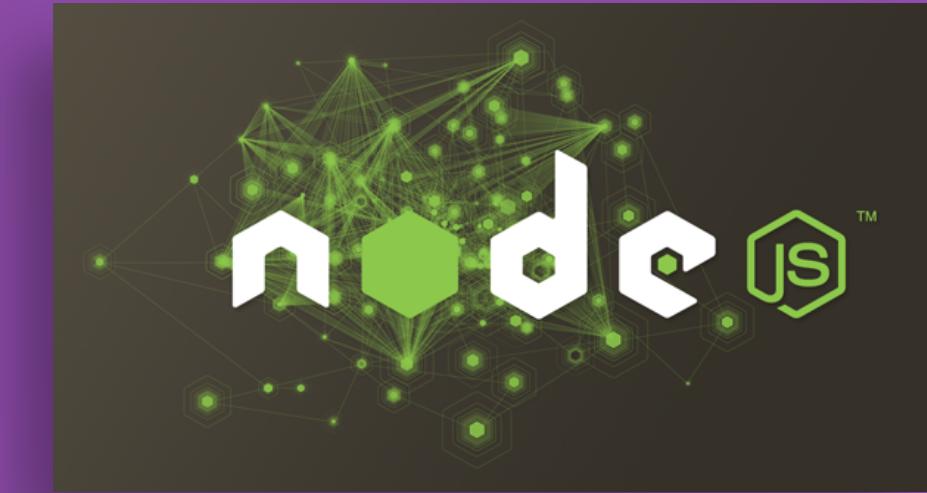




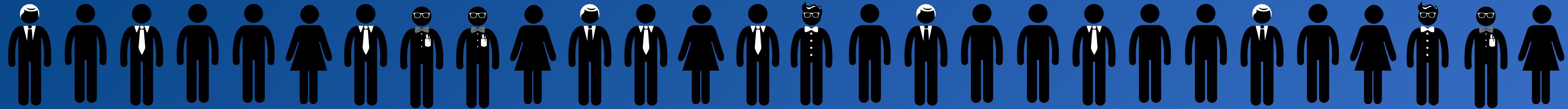






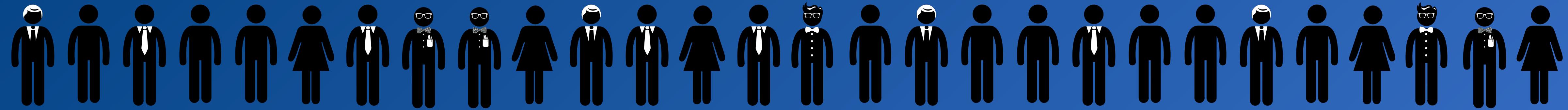






BRAND > code



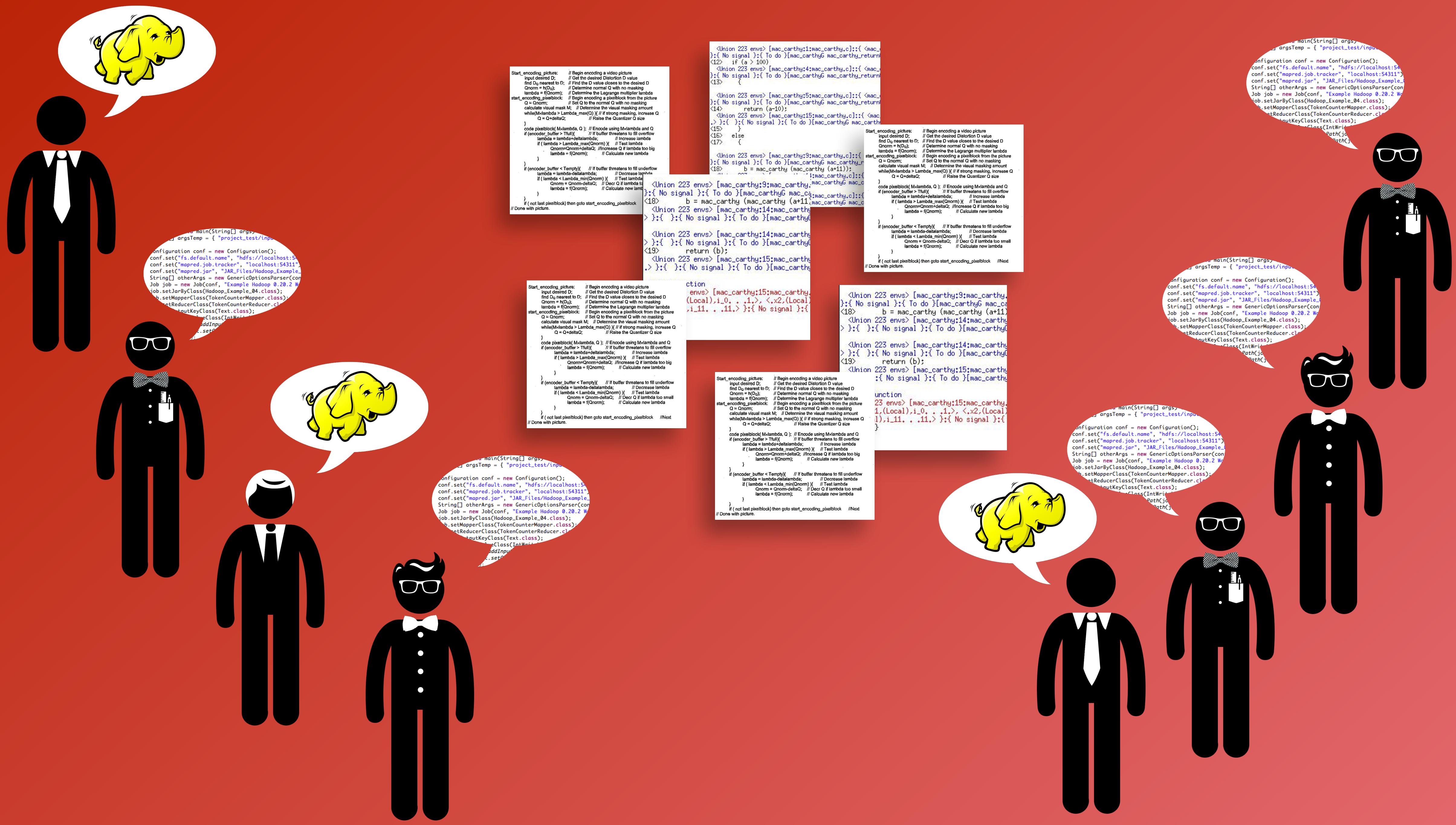


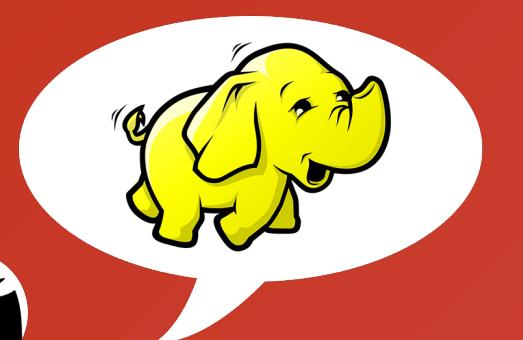
Community > code

BRAND > code



```
main(String[] args)
{
    argsTemp = {"project_test/input"};
    Configuration conf = new Configuration();
    conf.set("fs.default.name", "hdfs://localhost:54312");
    conf.set("mapred.job.tracker", "localhost:54311");
    conf.set("mapred.jar", "JAR_Files/Hadoop_Example.jar");
    String[] otherArgs = new GenericOptionsParser(conf).getRemainingArgs();
    Job job = new Job(conf, "Example Hadoop 0.20.2 WordCount");
    job.setJarByClass(Hadoop_Example_04.class);
    job.setMapperClass(TokenCounterMapper.class);
    job.setReducerClass(TokenCounterReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    Path jobOutput = new Path(jobsPath);
    job.setOutputPath(jobOutput);
}
```





```

start_encoding_picture: // Begin encoding a video picture
    input desired D; // Get the desired Distortion D value
    find D_Q nearest to D; // Find the D value closes to the desired D
    Qnorm = h(DQ); // Determine normal Q with no masking
    lambda = f(Qnorm); // Determine the Lagrange multiplier lambda
start_encoding_pixelblock: // Begin encoding a pixelblock from the picture
    Q = Qnorm; // Set Q to the normal Q with no masking
    calculate visual mask M; // Determine the visual masking amount
    while(Mxlambda > Lambda_max(Q)) { // if strong masking, increase Q
        Q = Q+deltaQ; // Raise the Quantizer Q size
    }
    code pixelblock( Mxlambda, Q ); // Encode using Mxlambda and Q
    if(encoder_buffer > Tfull){ // If buffer threatens to fill overflow
        lambda = lambda+deltaLambda; // Increase lambda
        if ( lambda > Lambda_max(Qnorm) ) { // Test lambda
            Qnorm=Qnorm+deltaQ; // Increase Q if lambda too big
            lambda = f(Qnorm); // Calculate new lambda
        }
    }
    if (encoder_buffer < Tempty){ // If buffer threatens to fill underflow
        lambda = lambda-deltaLambda; // Decrease lambda
        if ( lambda < Lambda_min(Qnorm) ) { // Test lambda
            Qnorm = Qnorm-deltaQ; // Decr Q if lambda to low
            lambda = f(Qnorm); // Calculate new lambda
        }
    }
    if ( not last pixelblock) then goto start_encoding_pixelblock
Done with picture.

```

```

Start_encoding_picture: // Begin encoding a video picture
    input desired D; // Get the desired Distortion D value
    find D0 nearest to D; // Find the D value closes to the desired D
    Qnorm = h(D0); // Determine normal Q with no masking
    lambda = f(Qnorm); // Determine the Lagrange multiplier lambda
start_encoding_pixelblock: // Begin encoding a pixelblock from the picture
    Q = Qnorm; // Set Q to the normal Q with no masking
    calculate visual mask M; // Determine the visual masking amount
    while(M*lambda > Lambda_max(Q) ) // if strong masking, increase Q
        Q = Q+deltaQ; // Raise the Quantizer Q size
    }
code pixelblock( M*lambda, Q ); // Encode using M*lambda and Q
if (encoder_buffer > Tfull){ // If buffer threatens to fill overflow
    lambda = lambda+deltalambda; // Increase lambda
    if ( lambda > Lambda_max(Qnorm) ) // Test lambda
        Qnorm=Qnorm+deltaQ; // Increase Q if lambda too big
        lambda = f(Qnorm); // Calculate new lambda
    }
}
if (encoder_buffer < Tempty){ // If buffer threatens to fill underflow
    lambda = lambda-deltalambda; // Decrease lambda
    if ( lambda < Lambda_min(Qnorm) ) // Test lambda
        Qnorm = Qnorm-deltaQ; // Decr Q if lambda too small
        lambda = f(Qnorm); // Calculate new lambda
    }
}
if ( (not last pixelblock) then goto start_encoding_pixelblock //Next
// Done with picture

```

```

<Union 223 envs> [mac_carthy:1;mac_carthy.c]:::{ <mac_
>:{ No signal }:{ To do }[mac_carthyG mac_carthy_return
<12>    if (a > 100)
<Union 223 envs> [mac_carthy:4;mac_carthy.c]:::{ <mac_
>:{ No signal }:{ To do }[mac_carthyG mac_carthy_return
<13>    {

<Union 223 envs> [mac_carthy:5;mac_carthy.c]:::{ <mac_
>:{ No signal }:{ To do }[mac_carthyG mac_carthy_return
<14>    return (a-10);
<Union 223 envs> [mac_carthy:15;mac_carthy.c]:::{ <mac_
>}:{ }: { No signal }:{ To do }[mac_carthyG mac_carthy_
<15>    }
<16>    else
<17>    {

<Union 223 envs> [mac_carthy:9;mac_carthy.c]:::{ 
}>:{ No signal }:{ To do }[mac_carthyG mac_carthy_r
<18>    b = mac_carthy (mac_carthy (a+11));
        ...; { mac_carthy.c }:{ mac_carthy.c }:{ 
vs> [mac_carthy:9;mac_carthy.
mac_carthyG mac_c
:::{ To do }[mac_carthyG mac_ca
mac_carthy (mac_carthy (a+11)mac_carthyG mac_c
vs> [mac_carthy:14;mac_carthy
signal }:{ To do }[mac_carthy
vs> [mac_carthy:14;mac_carthy
signal }:{ To do }[mac_carthy
ignal (b);
vs> [mac_carthy:15;mac_carthy
signal }:{ To do }[mac_carthy

```

Start_encoding_picture: // Begin encoding a video picture
 input desired D; // Get the desired Distortion D value
 find D0 nearest to D; // Find the D value closes to the desired D
 Qnorm = h(D0); // Determine normal Q with no masking
 lambda = f(Qnorm); // Determine the Lagrange multiplier lambda
 start_encoding_pixelblock: // Begin encoding a pixelblock from the pic
 Q = Qnorm; // Set Q to the normal Q with no masking
 calculate_visual_mask M; // Determine the visual masking amount
 while(Mlambda > Lambda_max(Q)) // If strong masking, increas
 Q = Q+deltaQ; // Raise the Quantizer Q size
 }
 code pixelblock(Mxlambda, Q); // Encode using Mxlambda and Q
 if (encoder_buffer > Tfull){ // If buffer threatens to fill over
 lambda = lambda+deltaLambda; // Increase lambda
 if (lambda > Lambda_max(Qnorm)) // Test lambda
 Qnorm=Qnorm+deltaQ; // Increase Q if lambda too s
 lambda = f(Qnorm); // Calculate new lambda
 }
 }
 if (encoder_buffer < Tempty){ // If buffer threatens to fill under
 lambda = lambda-deltaLambda; // Decrease lambda
 if (lambda < Lambda_min(Qnorm)) // Test lambda
 Qnorm = Qnorm-deltaQ; // Decr Q if lambda too s
 lambda = f(Qnorm); // Calculate new lambda
 }
 }
 if (not last pixelblock) then goto start_encoding_pixelblock //N
// Done with picture.

```

n
s> [mac_carthy:15:mac_carthy.
al),i_0..1.], <.x2,(Local)
1..11.}:{ No signal }:{

<Union 223 envs> [mac_carthy:9:mac_carthy
}:{ No signal }:{ To do }[mac_carthyG mac_c
<18>      b = mac_carthy (mac_carthy (a+1
<Union 223 envs> [mac_carthy:14:mac_carth
> }:{ }:{ No signal }:{ To do }[mac_carthy

<Union 223 envs> [mac_carthy:14:mac_carth
> }:{ }:{ No signal }:{ To do }[mac_carthy
<19>      return (b);
<Union 223 envs> [mac_carthy:15:mac_carth
:{ No signal }:{ To do }[mac_carth

Start_encoding_picture: // Begin encoding a video picture
  input desired D; // Get the desired Distortion D value
  find D0 nearest to D; // Find the D value closes to the desired D
  Qnorm = h(D0); // Determine normal Q with no masking
  lambda = f(Qnorm); // Determine the Lagrange multiplier lambda
start_encoding_pictureblock: // Begin encoding a pixelblock from the picture
  Q = Qnorm; // Set Q to the normal Q with no masking
  calculate visual mask M; // Determine the visual masking amount
  while(M*lambda > Lambda_max(Q)) // if strong masking, Increase Q
    Q = Q+deltaQ; // Raise the Quantizer Q size

unction
23 envs> [mac_carthy:15:mac_carthy
1,(Local),i_0..1.], <.x2,(Local)
1),i_11..11.}:{ No signal }:{
```

```

    }
    code pixelblock( Mxlambda, Q ); // Encode using Mxlambda and Q
    if( encoder_buffer > Tfull){ // If buffer threatens to fill overflow
        lambda = lambda+deltalambda; // Increase lambda
        if( ( lambda > Lambda_max)(Qnorm) ) // Test lambda
            Qnorm=Qnorm+deltaQ; //Increase Q if lambda too big
        lambda = f(Qnorm); // Calculate new lambda
    }
}
if( encoder_buffer < Tempty){ // If buffer threatens to fill underflow
    lambda = lambda-deltalambda; // Decrease lambda
    if( ( lambda < Lambda_min)(Qnorm) ) // Test lambda
        Qnorm = Qnorm-deltaQ; // Decr Q if lambda too small
    lambda = f(Qnorm); // Calculate new lambda
}
}
if( ( not last_pixelblock) then goto start_encoding_pixelblock //Next
// Done with picture.
}

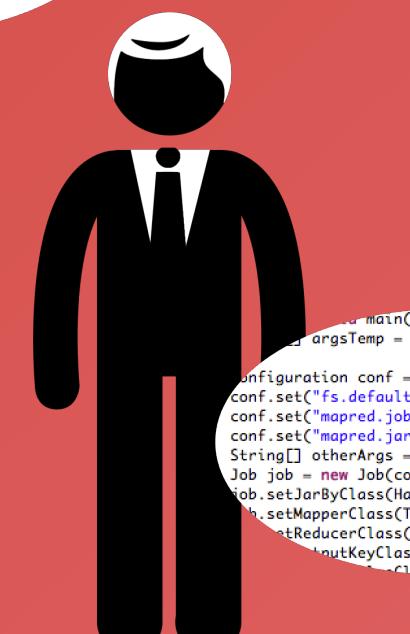
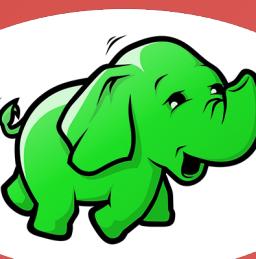
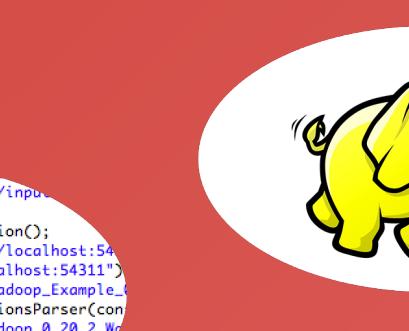
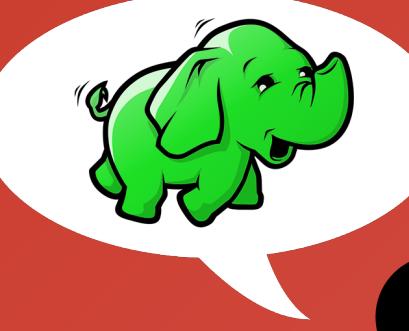
```

```

        if ( lambda < Lambda_min(Qnorm) ) { // Test lambda
            Qnorm = Qnorm-deltaQ; // Decr Q if lambda too small
            lambda = f(Qnorm); // Calculate new lambda
        }
    }
    if ( not last_pixelblock) goto start_encoding_pixelblock //Next
// Done with picture.

```

```
a main(String[] args)
Temp = { "project_test/input"
conf = new Configuration();
default.name", "hdfs://localhost:54310"
red.job.tracker", "localhost:54311")
red.jar", "JAR_Files/Hadoop_Example_4
rArgs = new GenericOptionsParser(Con
Job(conf, "Example Hadoop 0.20.2 Wa
lass(Hadoop_Example_04.class);
Class(TokenCounterMapper.class);
rClass(TokenCounterReducer.cl
KeyClass(Text.class);
i Class(IntWritable);
Path(jo
Path(jo
```



```
    J args);
    ects_test/input);
    nfiguration();
    "hdfs://localhost:54
    r", "localhost:54311");
    _files/Hadoop_Example/
    ericOptionsParser(con
    ample Hadoop 0.20.2 Wo
    ample_04.class);
    interMapper.class);
    interReducer.cl
    ass);
    Writ
    h(jo
    ath()
```

```
    public static void main(String[] args)
    {
        argsTemp = { "project_test/" };

        Configuration conf = new Configuration();
        conf.set("fs.default.name", "hdfs://127.0.0.1:9000");
        conf.set("mapred.job.tracker", "localhost:54311");
        conf.set("mapred.jar", "JAR_Files/Hadoop-Example.jar");
        String[] otherArgs = new GenericOptionsParser(conf).getRemainingArgs();
        Job job = new Job(conf, "Example Hadoop MapReduce Job");
        job.setJarByClass(Hadoop_Example.class);
        job.setMapperClass(TokenCounterMapper);
        job.setReducerClass(TokenCounterReducer);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        Path outputDir = new Path(jarPath);
        Path(jarPath).mkdirs();
    }
}
```

@shanecurcru
<http://CommunityOverCode.com>
<http://punderthings.com>
<http://www.apache.org>



</presentation>