```
{
 "cells": [
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<a href=\"https://www.bigdatauniversity.com\"><img
src=\"https://ibm.box.com/shared/static/
qo20b88v1hbjztubt06609ovs85q8fau.png\" width=\"400px\"
align=\"center\"></a>\n",
    "\n",
    "<h1 align=\"center\"><font size=\"5\">AUTOENCODERS</font></h1>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<div class=\"alert alert-block alert-info\" style=\"margin-top:
20px\">\n",
    "Welcome to this notebook about autoencoders.\n",
    "<font size=\"3\"><strong>In this notebook you will find an
explanation of what is an autoencoder, how it works, and see an
implementation of an autoencoder in TensorFlow.</strong></font>\n",
    "<br>\n",
    "<br>\n",
    "<h2>Table of Contents</h2>\n",
    "<ol>\n",
    " <li><a href=\"#ref1\">Introduction</a></li>\n",
    " <li><a href=\"#ref2\">Feature Extraction and Dimensionality
Reduction</a></li>\n",
    " <li><a href=\"#ref3\">Autoencoder Structure</a></li>\n",
    " <li><a href=\"#ref4\">Performance</a></li>\n",
    " <li><a href=\"#ref5\">Training: Loss Function</a></li>\n",
    " <li><a href=\"#ref6\">Code</a></li>\n",
    "</ol>\n",
    "</div>\n",
    "<br>\n",
    "By the end of this notebook, you should be able to create
simple autoencoders and how to apply them to problems that involves
unsupervised learning.\n",
    "<br>\n",
    "<p></p>\n",
    "<hr>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<a id=\"ref1\"></a>\n",
    "<h2>Introduction</h2>\n",
    "An autoencoder, also known as autoassociator or Diabolo
networks, is an artificial neural network employed to recreate the
```

```
given input.\n",
    "It takes a set of <b>unlabeled</b> inputs, encodes them and
then tries to extract the most valuable information from them.\n",
    "They are used for feature extraction, learning generative
models of data, dimensionality reduction and can be used for
compression. \n",
    "\n",
    "A 2006 paper named <b><a href=\"https://www.cs.toronto.edu/
~hinton/science.pdf\">Reducing the Dimensionality of Data with
Neural Networks</a>, done by G. E. Hinton and R. R. Salakhutdinov</
b>, showed better results than years of refining other types of
network, and was a breakthrough in the field of Neural Networks, a
field that was \"stagnant\" for 10 years.\n",
    "\n",
    "Now, autoencoders, based on Restricted Boltzmann Machines, are
employed in some of the largest deep learning applications. They are
the building blocks of Deep Belief Networks (DBN).\n",
    "\n",
    "<center><img src=\"https://ibm.box.com/shared/static/
xlkv9v7xzxhjww681dq3h1pydxcm4ktp.png\" style=\"width: 350px;\"></
center>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<hr>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<a id=\"ref2\"></a>\n",
    "<h2>Feature Extraction and Dimensionality Reduction</h2>\n",
    "\n",
    "An example given by Nikhil Buduma in KdNuggets (<a
href=\"http://www.kdnuggets.com/2015/03/deep-learning-curse-
dimensionality-autoencoders.html\">link</a>) which gave an excellent
explanation of the utility of this type of Neural Network.\n",
    "\n",
    "Say that you want to extract what emotion the person in a
photography is feeling. Using the following 256x256 pixel grayscale
picture as an example:\n",
    "\n",
    "<img src=\"https://ibm.box.com/shared/static/
r5knpow4bk2farlvxia71e9jp2f2u126.png\">\n",
    "\n",
    "But when use this picture we start running into a bottleneck!
Because this image being 256x256 pixels in size correspond with an
input vector of 65536 dimensions! If we used an image produced with
conventional cellphone cameras, that generates images of 4000 x 3000
pixels, we would have 12 million dimensions to analyze.\n",
```

```
    "\n",
    "\n",
    "This bottleneck is further problematized as the difficulty of a
machine learning problem is increased as more dimensions are
involved. According to a 1982 study by C.J. Stone (<a href=\"http://
www-personal.umich.edu/~jizhu/jizhu/wuke/Stone-AoS82.pdf\">link</
a>), the time to fit a model, is optimal if:\n",
    "\n",
    "<br><br>\n",
    "<div class=\"alert alert-block alert-info\" style=\"margin-top:
20px\">\n",
    "<h3><strong>$$m^{-p/(2p+d)}$$</strong></h3>\n",
    "<br>\n",
    "Where:\n",
    "<br>\n",
    "m: Number of data points\n",
    "<br>\n",
    "d: Dimensionality of the data\n",
    "<br>\n",
    "p: Parameter that depends on the model\n",
    "</div>\n",
    "\n",
    "As you can see, it increases exponentially!\n",
    "Returning to our example, we don't need to use all of the
65,536 dimensions to classify an emotion. A human identify emotions
according to some specific facial expression, some <b>key features</
b>, like the shape of the mouth and eyebrows.\n",
    "\n",
    "<center><img src=\"https://ibm.box.com/shared/static/
m8urvuqujkt2vt1ru1fnslzh24pv7hn4.png\" height=\"256\"
width=\"256\"></center>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<hr>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<a id=\"ref3\"></a>\n",
    "<h2>Autoencoder Structure</h2>\n",
    "\n",
    "<img src=\"https://ibm.box.com/shared/static/
no7omt2jhqvv7uuls7ihnzikyl9ysnfp.png\" style=\"width: 400px;\">\n",
    "\n",
    "An autoencoder can be divided in two parts, the <b>encoder</b>
and the <b>decoder</b>.\n",
    "\n",
    "The encoder needs to compress the representation of an input.
```

```
In this case we are going to reduce the dimension the face of our
actor, from 2000 dimensions to only 30 dimensions, by running the
data through layers of our encoder.\n",
    "\n",
    "The decoder works like encoder network in reverse. It works to
recreate the input, as closely as possible. This plays an important
role during training, because it forces the autoencoder to select
the most important features in the compressed representation.\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<hr>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<a id=\"ref4\"></a>\n",
    "<h2>Performance</h2>\n",
    "\n",
    "After the training has been done, you can use the encoded data
as a reliable dimensionally-reduced data, applying it to any
problems where dimensionality reduction seems appropriate.\n",
    "\n",
    "<img src=\"https://ibm.box.com/shared/static/
yt3xyon4g2jyw1w9qup1mvx7cgh28l64.png\">\n",
    "\n",
    "This image was extracted from the G. E. Hinton and R. R.
Salakhutdinovcomparing's <a href=\"https://www.cs.toronto.edu/
~hinton/science.pdf\">paper</a>, on the two-dimensional reduction
for 500 digits of the MNIST, with PCA on the left and autoencoder on
the right. We can see that the autoencoder provided us with a better
separation of data."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<hr>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<a id=\"ref5\"></a>\n",
    "<h2>Training: Loss function</h2>\n",
    "\n",
    "An autoencoder uses the Loss function to properly train the
```

```
network. The Loss function will calculate the differences between
our output and the expected results. After that, we can minimize
this error with gradient descent. There are more than one type of
Loss function, it depends on the type of data."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<h3>Binary Values:</h3>\n",
    "$$l(f(x)) = - \\sum_{k} (x_k log(\\hat{x}_k) + (1 - x_k) \\log
(1 - \\hat{x}_k) \\ )$$"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "For binary values, we can use an equation based on the sum of
Bernoulli's cross-entropy. \n",
    "\n",
    "$x_k$ is one of our inputs and $\\hat{x}_k$ is the respective
output.\n",
    "\n",
    "We use this function so that if $x_k$ equals to one, we want to
push $\\hat{x}_k$ as close as possible to one. The same if $x_k$
equals to zero.\n",
    "\n",
    "If the value is one, we just need to calculate the first part
of the formula, that is, $- x_k log(\\hat{x}_k)$. Which, turns out
to just calculate $- log(\\hat{x}_k)$.\n",
    "\n",
    "And if the value is zero, we need to calculate just the second
part, $(1 - x_k) \\log (1 - \\hat{x}_k) \\ )$ - which turns out to
be $log (1 - \\hat{x}_k) $.\n",
    "\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<h3>Real values:</h3>\n",
    "$$l(f(x)) = - \\frac{1}{2}\\sum_{k} (\\hat{x}_k- x_k \\ )^2$$"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "As the above function would behave badly with inputs that are
not 0 or 1, we can use the sum of squared differences for our Loss
function. If you use this loss function, it's necessary that you use
```

```
a linear activation function for the output layer.\n",
    "\n",
    "As it was with the above example, $x_k$ is one of our inputs
and $\\hat{x}_k$ is the respective output, and we want to make our
output as similar as possible to our input."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<h3>Loss Gradient:</h3>\n",
    "\n",
    "$$\\nabla_{\\hat{a}(x^{(t)})} \\ l( \\ f(x^{(t)}))  = \\hat{x}
^{(t)} - x^{(t)} $$"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "We use the gradient descent to reach the local minimum of our
function $l( \\ f(x^{(t)})$, taking steps towards the negative of
the gradient of the function in the current point.\n",
    "\n",
    "Our function about the gradient $(\\nabla_{\\hat{a}(x^{(t)})})$
of the loss of $l( \\ f(x^{(t)})$ in the preactivation of the output
layer.\n",
    "\n",
    "It's actually a simple formula, it is done by calculating the
difference between our output $\\hat{x}^{(t)}$ and our input
$x^{(t)}$.\n",
    "\n",
    "Then our network backpropagates our gradient $\\nabla_{\\hat{a}
(x^{(t)})} \\ l( \\ f(x^{(t)}))$ through the network using
<b>backpropagation</b>."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<hr>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<a id=\"ref6\"></a>\n",
    "<h2>Code</h2>\n",
    "\n",
    "For this part, we walk through a lot of Python 2.7.11 code. We
are going to use the MNIST dataset for our example.\n",
```

```
    "The following code was created by Aymeric Damien. You can find
some of his code in <a href=\"https://github.com/
aymericdamien\">here</a>. We made some modifications for us to
import the datasets to Jupyter Notebooks."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Let's call our imports and make the MNIST data available to
use."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 1,
   "metadata": {},
   "outputs": [
    {
     "name": "stderr",
     "output_type": "stream",
     "text": [
      "/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:519: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in
a future version of numpy, it will be understood as (type, (1,)) /
'(1,)type'.\n",
      "  _np_qint8 = np.dtype([(\"qint8\", np.int8, 1)])\n",
      "/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:520: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in
a future version of numpy, it will be understood as (type, (1,)) /
'(1,)type'.\n",
      "  _np_quint8 = np.dtype([(\"quint8\", np.uint8, 1)])\n",
      "/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:521: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in
a future version of numpy, it will be understood as (type, (1,)) /
'(1,)type'.\n",
      "  _np_qint16 = np.dtype([(\"qint16\", np.int16, 1)])\n",
      "/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:522: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in
a future version of numpy, it will be understood as (type, (1,)) /
'(1,)type'.\n",
      "  _np_quint16 = np.dtype([(\"quint16\", np.uint16, 1)])\n",
      "/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:523: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in
a future version of numpy, it will be understood as (type, (1,)) /
'(1,)type'.\n",
      "  _np_qint32 = np.dtype([(\"qint32\", np.int32, 1)])\n",
      "/home/jupyterlab/conda/envs/python/lib/python3.6/site-
```

```
packages/tensorflow/python/framework/dtypes.py:528: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in
a future version of numpy, it will be understood as (type, (1,)) /
'(1,)type'.\n",
      "  np_resource = np.dtype([(\"resource\", np.ubyte, 1)])\n"
     ]
    },
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "WARNING:tensorflow:From <ipython-input-1-aeda475fcce4>:10:
read_data_sets (from
tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated
and will be removed in a future version.\n",
      "Instructions for updating:\n",
      "Please use alternatives such as official/mnist/dataset.py
from tensorflow/models.\n",
      "WARNING:tensorflow:From /home/jupyterlab/conda/envs/python/
lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/
datasets/mnist.py:260: maybe_download (from
tensorflow.contrib.learn.python.learn.datasets.base) is deprecated
and will be removed in a future version.\n",
      "Instructions for updating:\n",
      "Please write your own downloading logic.\n",
      "WARNING:tensorflow:From /home/jupyterlab/conda/envs/python/
lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/
datasets/base.py:252:
_internal_retry.<locals>.wrap.<locals>.wrapped_fn (from
tensorflow.contrib.learn.python.learn.datasets.base) is deprecated
and will be removed in a future version.\n",
      "Instructions for updating:\n",
      "Please use urllib or similar directly.\n",
      "Successfully downloaded train-images-idx3-ubyte.gz 9912422
bytes.\n",
      "WARNING:tensorflow:From /home/jupyterlab/conda/envs/python/
lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/
datasets/mnist.py:262: extract_images (from
tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated
and will be removed in a future version.\n",
      "Instructions for updating:\n",
      "Please use tf.data to implement this functionality.\n",
      "Extracting /tmp/data/train-images-idx3-ubyte.gz\n",
      "Successfully downloaded train-labels-idx1-ubyte.gz 28881
bytes.\n",
      "WARNING:tensorflow:From /home/jupyterlab/conda/envs/python/
lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/
datasets/mnist.py:267: extract_labels (from
tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated
and will be removed in a future version.\n",
      "Instructions for updating:\n",
      "Please use tf.data to implement this functionality.\n",
      "Extracting /tmp/data/train-labels-idx1-ubyte.gz\n",
      "WARNING:tensorflow:From /home/jupyterlab/conda/envs/python/
```

lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/
datasets/mnist.py:110: dense_to_one_hot (from
tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated
and will be removed in a future version.\n",
      "Instructions for updating:\n",
      "Please use tf.one_hot on tensors.\n",
      "Successfully downloaded t10k-images-idx3-ubyte.gz 1648877
bytes.\n",
      "Extracting /tmp/data/t10k-images-idx3-ubyte.gz\n",
      "Successfully downloaded t10k-labels-idx1-ubyte.gz 4542 bytes.
\n",
      "Extracting /tmp/data/t10k-labels-idx1-ubyte.gz\n",
      "WARNING:tensorflow:From /home/jupyterlab/conda/envs/python/
lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/
datasets/mnist.py:290: DataSet.__init__ (from
tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated
and will be removed in a future version.\n",
      "Instructions for updating:\n",
      "Please use alternatives such as official/mnist/dataset.py
from tensorflow/models.\n"
     ]
    }
   ],
   "source": [
    "#from __future__ import division, print_function,
absolute_import\n",
    "\n",
    "import tensorflow as tf\n",
    "import numpy as np\n",
    "import matplotlib.pyplot as plt\n",
    "%matplotlib inline\n",
    "\n",
    "# Import MINST data\n",
    "from tensorflow.examples.tutorials.mnist import input_data\n",
    "mnist = input_data.read_data_sets(\"/tmp/data/\",
one_hot=True)"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Now, let's give the parameters that are going to be used by our
NN."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "metadata": {},
   "outputs": [],
   "source": [
    "learning_rate = 0.01\n",
    "training_epochs = 8\n",

```
    "batch_size = 256\n",
    "display_step = 1\n",
    "examples_to_show = 10\n",
    "\n",
    "# Network Parameters\n",
    "n_hidden_1 = 256 # 1st layer num features\n",
    "n_hidden_2 = 128 # 2nd layer num features\n",
    "n_input = 784 # MNIST data input (img shape: 28*28)\n",
    "\n",
    "# tf Graph input (only pictures)\n",
    "X = tf.placeholder(\"float\", [None, n_input])\n",
    "\n",
    "weights = {\n",
    "    'encoder_h1': tf.Variable(tf.random_normal([n_input,
n_hidden_1])),\n",
    "    'encoder_h2': tf.Variable(tf.random_normal([n_hidden_1,
n_hidden_2])),\n",
    "    'decoder_h1': tf.Variable(tf.random_normal([n_hidden_2,
n_hidden_1])),\n",
    "    'decoder_h2': tf.Variable(tf.random_normal([n_hidden_1,
n_input])),\n",
    "}\n",
    "biases = {\n",
    "    'encoder_b1': tf.Variable(tf.random_normal([n_hidden_1])),
\n",
    "    'encoder_b2': tf.Variable(tf.random_normal([n_hidden_2])),
\n",
    "    'decoder_b1': tf.Variable(tf.random_normal([n_hidden_1])),
\n",
    "    'decoder_b2': tf.Variable(tf.random_normal([n_input])),\n",
    "}"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Now we need to create our encoder. For this, we are going to
use sigmoidal functions. Sigmoidal functions delivers great results
with this type of network. This is due to having a good derivative
that is well-suited to backpropagation. We can create our encoder
using the sigmoidal function like this:"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {},
   "outputs": [],
   "source": [
    "# Building the encoder\n",
    "def encoder(x):\n",
    "    # Encoder first layer with sigmoid activation #1\n",
    "    layer_1 = tf.nn.sigmoid(tf.add(tf.matmul(x,
```

```
weights['encoder_h1']), biases['encoder_b1']))\n",
    "    # Encoder second layer with sigmoid activation #2\n",
    "    layer_2 = tf.nn.sigmoid(tf.add(tf.matmul(layer_1,
weights['encoder_h2']), biases['encoder_b2']))\n",
    "    return layer_2"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "And the decoder:\n",
    "\n",
    "You can see that the layer_1 in the encoder is the layer_2 in
the decoder and vice-versa."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "metadata": {},
   "outputs": [],
   "source": [
    "# Building the decoder\n",
    "def decoder(x):\n",
    "    # Decoder first layer with sigmoid activation #1\n",
    "    layer_1 = tf.nn.sigmoid(tf.add(tf.matmul(x,
weights['decoder_h1']),biases['decoder_b1']))\n",
    "    # Decoder second layer with sigmoid activation #2\n",
    "    layer_2 = tf.nn.sigmoid(tf.add(tf.matmul(layer_1,
weights['decoder_h2']), biases['decoder_b2']))\n",
    "    return layer_2"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Let's construct our model.\n",
    "In the variable <code>cost</code> we have the loss function and
in the <code>optimizer</code> variable we have our gradient used for
backpropagation."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 5,
   "metadata": {},
   "outputs": [],
   "source": [
    "# Construct model\n",
    "encoder_op = encoder(X)\n",
    "decoder_op = decoder(encoder_op)\n",
    "\n",
```

```
    "# Reconstructed Images\n",
    "y_pred = decoder_op\n",
    "# Targets (Labels) are the input data.\n",
    "y_true = X\n",
    "\n",
    "# Define loss and optimizer, minimize the squared error\n",
    "cost = tf.reduce_mean(tf.pow(y_true - y_pred, 2))\n",
    "optimizer =
tf.train.RMSPropOptimizer(learning_rate).minimize(cost)\n",
    "\n",
    "# Initializing the variables\n",
    "init = tf.global_variables_initializer()"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "For training we will run for 20 epochs."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 6,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Epoch: 0001 cost= 0.216099694\n",
      "Epoch: 0002 cost= 0.177453369\n",
      "Epoch: 0003 cost= 0.161362544\n",
      "Epoch: 0004 cost= 0.151179627\n",
      "Epoch: 0005 cost= 0.142130136\n",
      "Epoch: 0006 cost= 0.135626018\n",
      "Epoch: 0007 cost= 0.134643912\n",
      "Epoch: 0008 cost= 0.127500087\n",
      "Optimization Finished!\n"
     ]
    }
   ],
   "source": [
    "# Launch the graph\n",
    "# Using InteractiveSession (more convenient while using
Notebooks)\n",
    "sess = tf.InteractiveSession()\n",
    "sess.run(init)\n",
    "\n",
    "total_batch = int(mnist.train.num_examples / batch_size)\n",
    "# Training cycle\n",
    "for epoch in range(training_epochs):\n",
    "    # Loop over all batches\n",
    "    for i in range(total_batch):\n",
```

```
    "        batch_xs, batch_ys = mnist.train.next_batch(batch_size)
\n",
    "        # Run optimization op (backprop) and cost op (to get
loss value)\n",
    "        _, c = sess.run([optimizer, cost], feed_dict={X:
batch_xs})\n",
    "    # Display logs per epoch step\n",
    "    if epoch % display_step == 0:\n",
    "        print(\"Epoch:\", '%04d' % (epoch+1),\n",
    "              \"cost=\", \"{:.9f}\".format(c))\n",
    "\n",
    "print(\"Optimization Finished!\")"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Now, let's apply encoder and decoder for our tests."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 7,
   "metadata": {},
   "outputs": [],
   "source": [
    "# Applying encode and decode over test set\n",
    "encode_decode = sess.run(\n",
    "    y_pred, feed_dict={X:
mnist.test.images[:examples_to_show]})"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Let's simply visualize our graphs!"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 8,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAlIAAACNCAYAAAB8KJSgAAAAOXRFWHRTb2Z0d2FyZQB
NYXRwbG90bGliIHZlcnNpb24zLjMuMCwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy86wFp
kAAAACXBIWXMAAAsTAAALEwEAmpwYAADJS0lEQVR4nOy9d3Rdx3X2/
ZtT7rm9oHeQANhJkSJFdat327JjW65+p9iJkzjJm8SJ05w3iUvs2I67HSty3CVbs
rrVJaqygx1E7+X2csp8rl7z8yePXvvEVJKSiihBJKK
KGEEko4eSjz3YASSiihBJKKGEcxUlQ6qEEkoooYQSSijhFFEypEoooYQSSiihBJOE
```

SVDqoQSSiihhBJKKOEUUTKkSiihhBJKKKGEEk4RJUOqhBJKKKGEEkoo4RRxWoaUEOIWI
cRuIcQ+IcSfz1WjFhJKHM99nO/
8oMTxfMH5zvF85wcljq9JSClP6QdQgf1AC+ABtgArT/V5C/GnxPHc/znf+ZU4zn/
bShxL/Eoczy+OJ/tzOh6pi4F9UsoDUsoC8CPgTafxvIWIEsdzH+c7PyhxPF9wvnM83/
lBieNrEtpp3FsPdE/
5vQe45Hg3eIQhvQRO45VnF14C2JiERZnMkQZ4H69xjucyv+JHSeAHx7vnXOZY0tPDON8
5nsv8ih+V+iIljgsdOdIUZF4c75rTMaRmevBR580IIT4GfAzAi59LxPWn8cqzi0HZwyg
DrBQXsUk+ikkBXoMczxd+AI/In47wGpQhlDgudJT6oovzheNrWU/
h3OY4FZvkoye85nS29nqAxim/
NwB9R14kpfyGlPIiKeVFOsZpvO7sw8BHjuzUj16THM8zfh5egzKEEseFjlJfdHGecXxN
6imc2xxPFqfjkXoRWCKEWAz0Au8E3j0nrVogCBMjS4qsTCNdg/
uscBSaBkIBxXX6CSFA191/
cRMEME2k7YB0kJZ1yu+aL45nC1P5GfgAyoB757lZc4rzXXYZQ4ng+oNQXzw+8FjieLE7Z
kJJSWkKI3wcexI3i/
46UcsectWwBQBEKy+Q6XuUpsmQAfnymOQpNY+KdF5GtUHA8UIhIChUWG1Z2sCbsGv3Pj
rRw4MVG/AMC/6BD7OG92COjp/S++eB4NjGVX7HTj51P/
OD8lyGUOJ4PKPXF8wOvBY4ni9PxSCGlvB+4f47asiBRIWqpoJZN8lEScuwfz+S7tPo67
JoYoxcIrMo8qschEk6zpGyEP617gA2GB4BHg7v5q8KbGSgvI1+mEXspgognkWbhlN57N
jmeDITuQXgN7NUtoAj03jFkPIk9Pn5SznED+AR+dOBM9HWE0JR0eprkeEAhcoAQkqUr
IXWP44zPIKTz4M8Ksxg1lioMpxLlDie+1gQffEM45yQoaIiVBXhPWLLzXGQ+TyK3w9V5
Yi8iSwUIJ9HFkycTAY4RzieRZyWIVXC3KL3rYtQbxzhe6v+k9WePAAqAgUFXajY0gHgS
q/Jw2vuYtPSAE+nlvHIq68jbFpYnd2nNRkvNKg1VZhNFVz21Rdp8ozyb3e/
heqXTYz7Xpzvpp0UhO5BCfg4eGcT5toUd1/
ydSYcHz8evZhnf3IhjfcYiM4eZD4/300toYQSXgNQYxFEMIBZXwbicN6YsB20ziHyy+o
4+HoD35DAOyIJdxXwjGZhS/
t5NcfMFUqG1AKAtqiJ0SvqSF6c5cPNW2jWMviFD4BxJ0efJXgmu4gRM0SzMcIifYQrvC
YtWhwzsJ+fLL0GLVuNp6sXpD3PbOYOTixIptag3jNOSM2iFkAxz71OLLwGIhoh01rgbU
u30ayZVMsCr49t4eHmC0gvKycwPIZ9jhlSWk01dm0F/
VdHKGsv4HnwpVnfq5aXuSveiSQym8WOJ87pAVqrr8OuitF3bQSpQPVLOTy9E9j7OhYkL
2EYKH4/Ezctw/IJ/
MMW3sEsysF+nFP0bgvDQAmHkdXliMFR7NExcM6f8ehchhoOQ2Mt+ZoguTKNZKOKGYJcj
TUt/17YAm9fC/
kKhyVruxjNBEhlDUbiBt6uKM079FPe+TifUTKkFgCybZU47x7lr1uf4D2hfnADMXFw6L
Z0nsos5Tt7LyM57qeudpxLKw9yRc2LNGg+oso4mZU5Rk0vdY+qyPNo4DLL/
STrVcJKloxj4JmQaOlTD6yfLwi/
D7sizMblHfxd1SZAJ6QIbvUn+YfFY4yuqiT4qh9OcstyXqGo2I1VDF8Y5I9+96f8w31v
ofXBk7i/uoKJNWWEOn2oI0lIpM7pRYC5qIrRVT7+8v/
cTUjN8qm7PkTVKxX4O7pOKxnkTEEJBqCqnPjbkzSXjdO+q4HotjC1mQIiXzj5yVIIlFA
QWVfB2NooZVsVxET8vBqPzlkIgSiPMXphjLHVArE4zTuXv8wGfwc3+
+MoU5L3M7LAPalGomqGS73DBIWOLlQA/mpoA1u/
ECoZUjOgZEjNI4SmoSxrZWCNwd8vvY81niEcDJ7OeblnfD2P/
XQjngmJkZBUDprUZC3McDm/
2VjD+js7WW90U6cJPD4Ty+edbzpzjmylh3Sju525Ld1AzVNjMDDMOTM0KypqWZTx61sY
ujXPZys3z3eL5gxCVRldHSS+TFKmppAnWUgl1xBm6CLB0EY/
WjpIyxcnTjr2bSFA6B6UcJCO6/2suGEvG73dhBTB777tfv7TuJW2hz1IRy4Yz4zQNJRg
gK6PLid6zQB/
vfh+clLn38ZjyPYIsrsfJ5O5qWcqfj+irpr2vyinsibO8rIdbP7JahpS9TjdfQtjy1oI
lGCQwsalDFxikC93EJZgyRf2Yw+PLhj5zAXUcBhRHmPkyjpSjQL14nFqQ0kujb5Eq3eY
Rs8oy/
QhooqDUly0H4JXaFzjP4gOBIVn0ogq4fhYkIaU4vUiPB5EwA+AdNzJFEciMxl3hWfbC2
qAOiWoKoXKALlyyXW+MUAj6RR4KLGRhzuWs+j+MZSxJDKRxMlkkLaNr6qSYHUrfWaU5Z
5+wMZxBKoz32TmEEIgNJ1cTCBrckzYfjrS5ciDPTjp9Hy3blZQQiGUgB+7voKJJQrvWP
0ya4xejizd5lFtUjrIcAA1HEYWCkjLWpBejGlQBJkagazO4RXmSRtSlk/
BLjMJl6fJZj1HB72eIxAeHREJk2su8KmGB6lVPRhC493hHXyp/

MZp8SfzDiFQQiFkUw25NVm+uvx/
0HHYVqilkNcwchInmTzpxyqxKIX6KO+76Dk2+DtQhcOm2CrwuCVb5ntjU/
H7EYGA6y1bYcBFcS6q6Wc850cE/IixifPLc+bzYpcFGV8JvhXjfH/
t92jWbILK1D52dH8zpY2JjV78Pe5M9zxNmD5wFthEUwyaRxEITUP4vAhdB61o2lgWMu3
aDLJQQNr2GdlqX5CGVOGKVUy0esjcmEJRHDJxH5gCYSrUPCXwDxbQRzKooxNY/
edu4odQVXIVOmZIoqDwQt7Ls+klPP75y1i0dQLZfgDLtkE6ICVqNEL3+9rIXZjhXeEtR
BWNjHTQtgWp3Gy6SnIeQI1GcdoaiF+V47uXfZe/
3PMW+ndWscR6db6bNisI3cPI21aTXAyxjUO8vW4zv1f2Av4ZVndXVB/
g1+t8dCgVeEcqiBww8XfGsXfumYeWzxKKiuLzkl+T4dYl7XjEyeuddzhPYE+AyHWjtJa
NkAtHUMa9OLncGWjwmYNSWc7o5TW0NPdxsSHRhYf2QoY/
3P9Ogrt1N8tpIcRICYFaVUniysXIjw7zV4sepVF1+J32d9O9s4bl/
96DM3GAU5km+29vZvxCiy9GXuIXiQv54S+vpfG3eZw9B+Z9QSA0jeStaxhfpnLjW1/g/
cH9XO7rxisET2Vr+Wb5m1DGA9jj59F2VShAti7Autft4fNN91CpGiizmOqfzgXYna/
jweGV5O2jr9/XUc1ye/uZaPEpQ2usw4kEsGI+UvUGI+sE6qIULZWjOFLQM1GO/
kgLoV6LwP4ESlcfdiIx9+2Y8yfOAcaWGUyssXhTSzu6sOnNRik4KhnLwy4a0eMGnrgXY
zyGf6jppJ+v5iVqzsbYN4hMpbAn4meAxYkhbRvfYIHwfh9v3/
cmuieiJMYCLN2bRvQM4hy5F617SC22WFE7RFTRXLertFFNUAsLbKVwGhBeg1yFl3AowS
ItRd9glGCvAs4CmJBmAaEqjK+ShJeO88b67VzkP0BQ6DNee4G/
m5H6IC8oTUxM+MhWGZSHyoiMVuGMTSzIeAS1LArVFTRXjdHiG+a+
+FqM8VM7JKEmkKDZP8Z2oxx0Hc4lQ0oI7IowY6sEF0f7J7dBhh0/+w7UUDUkF4YRhbto
oyJGukblnfXbWGP0ANC5r4qydoHdP3DSRo8wDJRQkESbZMWSXnYVqnlquI3YLolnMIU9
30ZUsX2ja1ScFSneHH2FZi1BrepuZzXqo4ysCxIzmlFfzCBN69zZ4RACxTAQDbU4sSDK
eAqRzk46FqQCAa1AQCiTniVDKHSYCr12hO5COXHbT2eunImCj7G8n66xGLmUgTbgYaa1
UWRQuEWg5xOKihqLQEUZZk2IwSVe8jHh1lsst2lsGWZ9RTfrAl3YUmFPrIafXLqexIgH
78oyAn0xjISDMW6ij2Rwdu6dE5kvPENKCNKvS/
Oli37CDb4JDHFEE5dCXlr02wVeztfzdGLpST3ekQq7E1V0j0ap+p96Ah1JmC9DKp9Hee
pV6raEcR6poSGbgdw49vDIjIOQ8Hu5ZO0+7qh8CeMYE/
P5ABnwkarXqAsn8AtBYLuXqldy54zHTXg8vPOGZ/hM5WbMEwRQvy04wO8E+qHBdaU/
kW3kL559C77hBjzbJfbw8Flq9ezhLKplfEWIj9X/gkZ9lC8/
cz3V+05ugHUMFTMo2Rg9yFpvF1sCF6AZHvcY23MEimGQWBzgbbc8w+2RV3HrEsPBQgUV
z2pEd6Xmt4FTIDweUksiJFod/
rhsF6AyaBeof1QQeeYA1ikYPWpZDHNxNZdetovPN97HB/
bdwYGXG1ny4G6c1PxvwSvRCLK2gqtue5VPVj3KUt0L+HGKm411ah7/2wbofKWGJftjOP
HEZJ2khQ7FMFDKyxi8tobxVZLY9iDBXgtjcBjhSBRTMpoP0G0rDNsBdGFTrmT5zuiVPN
O/mPHOGJ5RlbJ2h0B/Hs/
+IRbl+qFgYieTx1wAzPeyQAn4sZY2MrwhQPLyDB9d8wg3B7ezwqOgMUM8V7iPf7r+FSx
sctLiwUwNWzNN/
GT3hWiby2jap+MUOG1jauEZUlLify7AJ+J3Ut00RsFSiSf8BIM5Yv4sq2P91BkTXB1sp
14b5y2xl+kyy8hJD62eQfQpprQjFWwEOakzYEYJqVmq1CRDkRDtFfXcvf5qyowIoS3zy
BdwsjmUwRGk6cbGzGQwOFeuY3CFjzfG7mW5ZxDwsLVg83x2OeGDDkbn2GSdqXMdZm2Ui
WtzXB0a5JV8lGCPg6d7/Jzgl3vDxQyt1/
i94HdQEMcM1szIAnHHJumomFJhse4QUjTWe3v46Ian+Wn5Osq+0Ij624VjSCmBAEp1JQ
dvDOO/
YgRdWPx6Yh21j6lEdidOalsoW+lBXZFkiTF4xtp7JiE0DbG4kVSDyu2RV2nRc0AAWzr0
FMop35ZE7RlmIUS6KYEASmU5I6s0vM3uovGvhjZwz941LDqYwkmcnPUqDAO1tpqha+sZ
vyHH/6l4haQjad9TT6wDNybFnH/mubVNDK/z8LZQBzUzdMOoovF/
Fj3J49HlPFK9guBOD7F9FsFnOhZ06QZhGNjrl7H3rT4qVgxza2U3T/
RuIDBQPEJsZIxAu6T7Jy3cUftHeMYFjg5mWOLvE/
hGHJrHbbRsAX0ohUhmcMYnXI+ks3C8qIegVlZCLMzoJVWk6wSFdWnaag7y3spdXB3YRb
Vq0mkJDppRnkwtpz8XIW17+GzDryhT4OlcNYu0UVZ5PNRocXLefq5vCfCQvZz0zRcQ3D
Hslik5DSw8QwqoeilNYMDHRGsVah5qehzS1T4Gy2J0tZRRUZGkuiXOEmOA1Z4kaelhwg
6wTI/jPSK4MyclcUcl7Ri06sOs82jAOF1GLz9ouYT0mI/Q/
NCchDQLbsedCUIgVJWxVT7G1tls9HXQUJTatnwDj44sJ9CTRfYNLrgOcEoQglylh3eue
pYGzxh7CzX4B01k/

9DC56eojFygseam3azUR1BFEGcGj1RemsQdm12FGGN2kLRjEFH2UaaqLNa8/
Gn5Tj5Vvp3XNf8hZZo273EmhyD8fvJNZdgXJvnmqv/
mhdxiXhhqpvzJgzjjEyf1rFxM8Lqm/
TRqY4zagTPT4DMIoWnka8NkqySXelXA5ZCVBQYKYUR7B9YCSYxQohGsqgjZ1gIbqwcwp
c2vD6zC93gItXc/1kl6YYTHg1kXY2yN5AeXfZsWLUOn5cN/UCfUbS2MODchiC/
Sya7LsMroJagYOEgcHGwp0YWKIXTeHhziDYEe+qof5kO172UwUkVwZxiRSCDzC9CQUlS
UaITxNj8fuPG3XBXcRaOa4jFjw+QldiIBiQR199hIrwenowvh8yFqq2B0HHtsfHIsXYA
MD6O4fSmry8k2hRi6wqZx8TBfW/Y/
VKoOEcWDLSEjBZtzdWxKtvLAwRVkxvyIvMLe6nIWaeM8NLGaS0P7WOHpJ6qYNOljlMc2
M9Hs49W1KzDGoij7Tq+pC9KQUrcdILJbJ/
qc3y1Zn8sR9XjAoyO9BtLn4Yc1byRTpZGuFUQ6HPS0zb8s13GO2PFSCqDmINRr0X2b5M
FbvkidqpKTCmLQwDu2sCdntW0xqZUVBN4ywGdbf0ObbmNKhx2mxT+8/
HoiT3ip6ejAymZP/
LCFDkVFa6on0ajyu2XPsduM8FKmBTVrLXiXu1pdRWFlA5m2AteV7SKgCExp4xzhpxm08
7x/93voGijD2OVDzYNiwT+sMqlrHOXXq36IX9GxpWT4Yhthb6Ts51sXRLaitbSe/
e9TePuSbRjC5t9/+iYqtjrYY93uMRKzQbEkRKoB/m/
Nw5QpGpsWWibQLCB8PvovMzCWT0x+Zkqbfx9dzwP7VtBqL5xkgZ47FpFcl+crV/
4QrzD5bqIV+UqE2of6sU/SAAa3BtXAugC+RXHWegr81eA1PHxwGU2/GkP0Ds775Kz4/
Sg1VYytt/
nCxp+yTM8z7kjuTy9m0IzQX4jwofKnWaG7k4VfeGjWbD6z5Nc8Wr2Sl565CF88iT04NM
9MpkNoGmp1Fe2fbqR5RR8fiL7EZ/pu4fF9S2h5KotnTz/
WFC+aPTAEinB3OVIpRGfB9RQu9AVpEWL9Sva9I0xo+RjX1r/MX0a2Ua8lWKypPJcPcv/
EWl4ebaJnJEr93R6M0TzN8RyYY6Br/N3iNxLwFMh+v5YHLl5L/
7WPcXVgF41agl1mBW+vepF3vncTf2F+gOY9VTijY6e8aF2QhtRkCu6xvDSKiq87gq+qn
GBdGO/+YWQ6i5ZuxNGme6RU00GYDmoqj5qIATDhWPTZEYwRBe/
YwljtHwtO2EeqVuXaik6u9I7jFx667CzPZNtQur2U7c658QjnSOc4HoSqYldGKESgVvW
zz5TELR+K5SAXMD9hGMiacobXeamrG2CJMYA+pVxwTlpkHJtOy8eOfBudO2sJ9CqUtVu
oeQdhS6Ri0GdVMLHCQREWOiqVzeOMJSooe7URdXgMe2h4fuQsBGpFBfF6L2taD9LqHWL
M8RI+AOH2OE6hMOt2KT4vdksdhQqbJi1Iv5XiYKESxbRhvgNZTwaaRq7WZnl0YvKjvDR
5uH85siuwoPQ1H4O6mnE2GqMM2Cq/
ylZjTIAcHJm9ASwECAWtvpbCokoSbQ5rK4bwCQ9P9LRhb48getsXRC0wYRjY5SGMsiwX
GQNMOLDTrOCr+68mlTXI53TUFQ5Xh3ex3hgipKgEhcFqzyh6ZCsPLbuMqmw96kR8YQSg
F797tb6WwqIKFq/s55aaHYQUla0jdXh3+ND7Bo7yCk9LVJFyYdTzOhGKpW/
U+hrGloaovmCQtzRs5tbgdmpUMJG8kA/
w49GLeXTvcug38A4rBF7YhzM2gV3krPj9JNKt5E2NqoM5MtV+ftW7hhWtvbToOUJKjqi
SY5Xu4U8i0i2ZIE4tYQYWqCF1Qji2uxU2No62W8E6VB7gybGZws3QqisZvmkxalOaxZq
X+zIRfj6ynsaHEygH++d9BXU85Ct8xJdKlvgG8Qv30OLnc838+0s3Ur/JQXlqK858d/
Q5gvAaDK8Nkmt2O8OAFeFApgKshTvBCk1Daaqn76oYf//
x77HcM0yDqqOLw3VaeizYlm/
kX3bdTOJAlOVfHYLhUeyEG4wsFEFDdzNjGyvZckMNKz2DNKgq/
7v6u2xbWsWnxPsp2x4j9uP5GdgVv5/
xG1sZvFTyreafs9us4jeJtZS1p3F27D4p407UVbP3vX4uuWAvprT5YWItv+xZS3Qk4Qa
5niMQhofmZQO8vnob4BpRw7ZF+tc1tLyUmr2BcpYgAVtKxmw/
7RM16Gl5Um0Umo4SDnLwziayK3J884pv0qLFsTDQfhlj0f+8vGCOOBLRMOMrgjSXd1Op
Gvzr6Bp+uGsjLR/vo6Jggkfn6Vsv4dfLLuFDb3qEywJ7ucJwaNCCVKo53vKhx/
nfvetZ3FmNMzp+SrW15pSPx4Pi99P7pkbia0zub/
sR5apkt+kh9WIFLd87gD08smBCAE4HwuNBrSjnwPsa0C6c4OnVd+EVGgoGe8wCj2eW8W
9P3Er5SyrL7juAk0whC+akAXX4QYJwIEfAU0BqPsp2muQGq3jij5bzen+KywwbBc+ctf
vcNKQOQcrpx0rMMMEofj92TTnDV5vctHgvCoLv9V3Bll1NrBwdWhDZJTNBraxk7KZWRi
8QLFnXxUpvLxlZ4H+TrXy/
81JizxgEO+LnjREFrlGSqxD4Im6Mxf1ja9i0rY0VqZEFbeyiqdgGLNFHKFNALcbpxZ0c
z+Yq+U7f69i6dRGhfSo1/
Q6MjLt6V5SddICxCYK9Ef5l3828pXEzvxdtp0xVWe4ZZsnFnexRm6h4ZH4yi4THQ7xVw
d8Qp1yVPDW6lPsPrGJxIod9EkaUGo1g1kRoWjbIpdEDODj8vHsdI1uqiKT3nTNZmVpNN

YVFlVxW8QobvAcBlZy0iDs6/iEHrXcMawF5pKaiTktyc/
VOvnx5LbaxHjUHnrRDuH1ixmKLTsAgV+UjW6GRKxN4Lx/
h2ppOWrQ4fbafB9ONeOPOwvJ2CIGtg6o4mNLmZx3rYFewOOkWEJpOrD2JWgjy6rWN1Or
jXGG423juAfE2QkjXQ7oAtp3VygoKrVXE1xW4ZuVuMlLj1/
EVfHXTtdRvt3ESyXOm75wIan0tuUXleNaPc0tzO0FhkHByDNjwsV130newgupnFML7Mz
gTcZx8fvpCTgh3azcWZbgnSroiQ/
71Bo7PgaDJxmAHqlBQgQ4zxf3pFRgjApnNntZ3eG4bUrOAEgmTWBzkK1d9nzWeEcDPlh
3NVD+j4AyNLIzAyBkg6yrgPSO8v2E7vxt7Gb/QGXFsvrn/
SlIvVbD4x+0LPm7opKFpZKscmiLuCvDZjhZqnlRgfH7KU8wWUlVxdGjWpnenMQd+PHwx
u55oYcV/deKMjuHkcjMahfbIKEaHn/4nG/n5Vev4vWg7fuFhsQa/
WHoPdyhvxKyKIUwTzrYh5TUwV2S4sXE/EcXDk/
1taC+GEIljbL3P+BABFWWkGg3+YvFDLPcMk5MKI1uqqH/
SQh4n5XqhwW6sIt7m4+3RF1lnuJ7HjGMzZIcJ9OawunvmuYXTISQ4UmADizUvfxDbS9l
VKR5bs5zBTJiOoXKs+2IoMzg0shUK8eU2VS3D3FB9gE9UPlmsw+Tj4UwD3z5wBeGJheU
JkaqC1EARkoy0yW6JUfmqMxkfJM0CvLSdWH8d7e+qZlmwGic0CNLBwSFlG5imCsUTNOY
bdm0Zw2t9vG/DE/xp+cs8kYvyg72XsOJfxmFoBHsBxE/OFfLNZYytMPiX1T/
kImMM8DHsSLbkG8jeU82SzRnEpu3g2EdnCRe3BZVYFKciQnCfTgo///Km/
6FJG6NOy1KheKDoidppVvD13a8j1OPgJFKn5ek/
rw0poWkM37SY0bWSNZ4RvELQa2cwhjWCfbkF6wpVK8rJ1AV5a9NjXO7fWzSiCjyfq8e6
v4K6XXmc5PmzCgFQl7SQaSvnist2sj7cxY9Slei7/
MReHMBJLpx6PLOBKW0eycb48dAN7Pnuchp3ZV0j6gSrdhlPUPNCns7ySn64aBE3+vfQo
PmOe8+ZhuL3I2NhNjZ3ck2kHYDx3WUs/
dUwzsjorJ8jNJ2ha2oYW+uwyjNE0tF5MF9FdA8EXunCWkgejRMg1eQnuUjgVywOHbXx5
dHL+cW+tSwezyw472n1iyajmWqebmlko7ebJs3Hdf4DrDR6yUmdsaYgjy9djiOPPs4mr
OVo8w5Sp41ToyUpUzQyssCWgo8v77qGwM/
CePf3LogyD4dgVocZvyLP28v3YQgFuSzNWDZI4NdHHOqua7yu/
gDXBtsP3yttnhhsQ3YGcBJJnII5DwymI1fpJbHUpskzSk7a3Dt2IemeELK/
G3k+JBlNgRQCqUBBqpjFhZUpFWwUUo3g6AHkpZdgBqAQcf8uJAgLbK9ElheoqEjSEBrm
UqOTBu84l3t7CQgFo3h2oCltuqws/z14Hf6fR4hsH8exTk/
O560hdehwzvgSCLdOUKZodFsOL+aaMcZAHzs9V94ZgxDI+ipSdRobfR00axl04aPP8rE
53Ux5ew5j39ApFdBbyLAqQ6TqNN5QvoWwkuPRxEq8oyC7+xZcvMnxoAqBKSX3ja1l04F
FLHtuDIbGsGfh+ZS5PEZfAmO0kr3Zaq7w7Z/
8m18rMBLzYYye3cOphUfH8emsDPXTpI0xaOfxDivY7XtP7kGKIF0v8NenKFMUBmyd9mw
9vlEba+DcqiWVjyjky230YnlCWzq8ONaMsyeIyCw876m/
I46jRfn16Fomon7WebsIKSqVSp46DTQS3OR7low0iR9xekBOqkw4Bs1ahupiRfARO8vO
XD3p3hCNr4zhjE3MA6tjwzFUIpE0MS2NimBRxRj7qv0Ir4FQFdB1hNeLXRlhdeBlGrUE
DgaHzsHMmZrrnVNOPfh4LmEGVXy1SSo1t1Zb0vQibIEwPO7Wo1CQlnnOeHSPB8V0UPOS
YStM0jNMRbGP6cLCrDFJBDREWZ5IJMPKmLuQc6QgbRpU+lJcGd3Lpb4DLNUF+ywHFUmt
6p98fsLJMeE4PJBewSvdDbRuGUcMnH4Sz3lrSClLFpNujfHm1z/Hx8qexif8/OH+NxH/
7wbqnxnA6ehakB4pxedj9x/
5uXHlZi400niLQcuf7Xwj7e0NrGzvxBoamedWzj3GVvgZW+ewzujjgFnGE/
1teMecBbv1OhNUIVBQSDo2uz63mmXPdrgptSdrsAt3W0JBFldQ8OaKV/mrjzVR+5MG/
D29Z4bATNA0HK/ORf4ORp0Af7LzDkJdJx83IoQgX2WzunwURQh25et4oG8F/
tQCXMycAKkGQUXLGAFFkHEK9NsFOl6tZ8lP4jgLsG86ew4Q6DQY7WjkJ5WtfGuxh3gb0
JTlrsu+yUq9wG5T4bsj13F/+6pp96r9BjWbHOTHhvntmp8AMGjrfO/
gZYT2qe55egtsQWr0xHEeq+KhspXcGTrIG2q28d85H86qxZhhD+kanZELJd6mJNcF9hB
SBAdMkxoVgorBf6z6EV8uu4Gx+1rQu0fnfas20aTw/9b+jPWeESKKwd83/
IqveuP8+hOXENsliezPoGzbvyBKpJwu9C37qRqv4yuvv4Y9TTV8svxp6lSFZv8gl97wR
UwJugAd0IuLVhvwCIECeIWKLlQ0VFbo040jC5vPjVzOQ13LifxXiLbOBM7u/ThzoL/
nrSFlVgRJNGms8vUQUQQvF2wO9FfQuicDYxML0ohSK8qhuoLWxiGuj+7EKzTG7Dydlo/
2ffVEd2g46cz8p+OeARQiAqWsgF9IhqwQwz1RmhMLn6fweEi3hslVHjYuHMA7VjjpOjT
Ca5BrjJCrcFjh6yOkHH7mgBWBXh+exFn2ztkOwrTpMsvxigIBvcC4R7glH2ZZ9kBoGiI
YwFud5uLYQXRU+s0og0MR2nILX8aHoAQCKNEIuXqTm2s6MISCic2E40FPKiiDY9gL0Ht
66LQE9WAf3ngENVuGnvGRHvLzAc8HiQSyJDJesr1BQvun5z17xyX+ngz9OXdB12NleSa

7gtGtldR02QtyHBXpLOEum55klLy0WGb0cXlNBw/
efDGWT2JFbVraBlgV7eeAWcbefA0/6VnPHQ2v8IbgDupUuCDUy89XLKNMgphnQ8rRYbk
+QkjRUFCoUFUuD+3j8XVtDFZEibcGqS5fiTGaRx3PgKbieDWUTAGRN5GJw/
GHwu9HGjqMTiDz+QUXZ+tkc6ijE+RfauHHfRfz9OIWNlR0szbQzSqjF13YJB3vZBFuFQ
dFOCzVhyhTTWLK4Uw8ZUoJmj1mjlfyjfysfR36bj+1ewdgeHTO9Pe8NaQSi7xMXGCxyD
NC3JH8R/+NGO0+lJdfXpCDHYC1tIGxVX4+Uvs0N/
sHUPDwYr6KHw1dQv0DCuFHd2Cnzq14odkiVylpqx3CIwRb0k2Uv6Dh6x4/
pdPozyaE30ffVQqNqwewpcTGxjxFL7GIhBnc4KFqxSBvD/agC99kUc/
HRpaz5K44onf4rMbgyGwWNZ7lgeFVrI30cl3Vbn5Y0YQai2KPzG4gUkIhqCrn/
cs2TZ7ztjNZi3ePF21idMHFFB0LoraK1IpKbrpwG/
9aswld+Biy03RblXjiTB4YuyAh5WTFa9HZTfhZCAN83v1z+Bi3qdEIsqkW23YNrIczS/
l+x6Us+c4QDI8tSNnZwyOEn7fZ+cYGBlbC5d4k1/
o28e8fe2HynD2AcSfHp3tv4fEty1n29Qyf//
9uInRVltsCHVwX3Mn3brwEqQaoeG4eyQCOAa16EFs6OEj8wsPtgXFuX383znqHpFPg5o
0fpLM7SmRXEDMAZkTiGxB44pLo3ihKsYRMut5HLqpQsdmPOpLA6couqC1BaRaw+gdo/
PsB99DpaIRn3nQRv9qwjrXLu4gZGbpSMYZTAVJxH4om0T0W1y7ay4bgQT4Q7pvxufcm1
/KNLVfSeJeG/8XdbvmkOeR9/
hlSQrhegjpB25J+wiLP5nwdLzy+guqd9oKs7Co0DcXvp39dgOSVGVZ6e/
EWD2vuLFSydbCO6gkLJ72wlH4uoHi9iGAAqz7PtZV72FkI8cxgC5UvTSD6Fs45c8eCMA
zCS8a5uvokY4aKULxeRMAP1RUklscov6afN9cfPvzRwaHHyjOR8xGwnLPujXQKJlouz8
GJMhYHRrm1bAu/
uG4te8tbqNnUjDF+dJCmoypkq3RMn6AQFhQiUIg6fNJ3cNp14lxTZY9OIajgU83JMxQP
mF5+0H8Zxvi5RmZ2kA21dL2hjA1128nIAv/v1ZvwbvXD6K4F5804BGlaOPEE5c/
p3Cb+gK+87i4uN8aIqX5U3Ji2z48v4dGh5fTf00zjQQulZ4jAgSV8q+lKNi7vpFrNcm3
LXh7fv46ahnqckdF5CzMI9Eje3XEtn6x7iAs90+O2FBT8QufjbU+yv6GKXaur8WsmUT3
LQC5EyjQYTIawi4kEUV+SCr3AvusrsSdqCe1vpGJbAf2hl+aD2nEhTQsZT1D1QpxQb4C
+Z1voVUEtSEIFiOYdpCIohLxsuqOJUEMOphhSCSfHlkKQP21/
GxPby6l7wSGwa+CMFLA+7wwpoekowQC5SocbqnahC4f2XB01m2yCeyewF+C2mPB4EGVR
kq0O71n1Ii1aCgUfeWlyMFdOeiCAlsxOr1R7nkD4fFAeo7oyzmWBvezM1TM0FCGy5eX5
btrsoGtcWtvJFYFTOxJEBAMQi5BaEmV8qcq/
tP6G5foI4MHBISctdpkVjGd8BM2Ce6jo2YRjQ77AxEQlw5VBlukWf9z2ME9WLueRwga8
I0cPIVKDVLODEzYpq0ywOBynNTjMEn28GNQLBUd1A3rPoYWB9GiYAYFfOdwPu6wytnY3
UJtY6L7Tk4fQNPJ1QdSLx7k8sp+8dDC2+al6pYA9Hl+4IQaOjZPJUPFSAj0V4vG1K2gs
ew5DuHIzsbmnZy0DO6pY9rNO7OER7HyeYHcrffsr6W0Ls86Y4PayV3moZhV2bRlKOgPz
ZEgF+yye37yUpyOdLNK2ERT6ZK06t+6VyvvCvRDuharZPbO9xWRnvpavtl7DCLXUPTH7
rfqzBsfGydmweSfG5kP5sdMhNI1wbQ27bqpkrGr6mZ0DNjydXkruiQoatxTQH3l12hE6
c4nzzpBSli6m643lXHDxXt4VeZVvj1/Cj/
esp2VTJ87EwsuoAWBJMwfeHGP9Jbv5aGwTlapBh5XjH/
pu5bknV7H87jjiYO+CdKOfLqwVTfS9LsBb656gUU3x9/3r0XvmruLsQkf8uiUMbhTccs
0rXBXZxUZjFKPo7dhekDybWc7X/uf1lO2ykZ1b5iUd25mI0/S/
TbyyZjUffr2HG8rb+UjFk1z7rnZyUj/qelsK4nYARTgElDwtniHq1RSV6uHhZmtPPW2/
GHAP2z5HkFocYuyqPBsCB7GlQ7+d4ZfD11J5n0Fo98Lc5jpVKH4/6RtX03+Zyk/
Wfo+QYtJn6XgSoCcKxSqyCxtidwexviC/
DV7Kvc2XUr1xANNRSOc9+H4WZcmr425F8GKoR/njXUR3l/FHNW/
nDS07+FTFM7xp3Wbu+dB6ln63AV6Yn+Nv/M/tY/
n+cn7y9E18e9EtfPTO+4vHUNks0cep0wwUTi7DsEWDOrWHlcvu5hPKO+gVG2i8b/
jks3HnGak3bWB0lcpnN/
4vG71dgJtZmpJ5bvvtHxDcYdD0k27kePyMOlHOH0NKCNSyGJnmMLk1WS6JHSQkFJ4YXI
LZGcAen1hY1XenwIz5kKuSXB7bP5liPGAHeHpPG2X7gP3d2OmF6UY/
XZghnWy1Q71nHF3ASCqAmj26ns35BjUaQS6qZ3yJgn/
JOO+reJrVusTNR3HRa0Vpz9RS3m4T3BOft60FaVn4940TM8p5uamF0SUBBqsi1Hom8Ii
jY6Ryjs5zEy3kbHd4uap8L17/
XipUa5KdmdOQvQM4uYXZJ6ehGC6QLVdY2dRPvebG7o3YOl3JGKGOLGI8Md+tnFMITSPZ
oGHX5Vjj0dllOmzON2LEHdTEzEVlFxqcTAZyecq212LEffR7ahAOKHlB7e4kdHRPmxPs

oRGUXI5c91JeijThr3ye9YGDvLS4iUKsDEPT5iW43p6YgHiCmCLwjkf51sYrqAimURWH
VdF+lvkH0IVNVE2z0eglpAgiyvHLpBhCxxA6EQUuKu/
iZ4trcIJnt7TK6UAJBFBiUcaXqdgrU1zq7aRBc31WKZmnz5L49xqU7zSxuvvOuPf0vDG
kFJ+P5NVL6LtS8NxVX6QgJXstg/Hf1LHo1ZwbG7VAkWw0eOjS/6BM0Tg0kT6dWsbS/
8ij9vZinUNnkJ0szKCKqM5TrqZwgHTGwHvuVDw4ZWQvWYL81Ai3lu9ifbCTRjUPR5z99
Hyqjad7F9O4qRurr39+GoprSNl79uPf28Hy+3VYuojf1l9Bql7DNo42erWspPK+/
cji8Uvf+d3buOvaEb656r9ZcYiiLRZsjM2REB4PSmMdE8slj7f9Ap/
wkJcWL+QW0zcQY8mmzWdsy2De4PMysdZkY0snAD+OX8TP9q+jcfOE67VYSFtAx4NjI57
fSuhFldA9hzMSZaGAcwQHaRZwEimqXoBOsx5zmcMVvoMEWgp8ru49+GIx7JGRs8+9eBS
avWsfnt0KjU8e3uTadeEqXli0AakK0nWC6972IjdEdnCrf/Zzxip/
L8+2DmOGY+eMQeCsaaX38iBXvvlVPlX9CE2abzJLb0vBx6PJVdQ+m0N/
YddZOUbthN+bEKIR+AFQg5vZ/Q0p5ZeEEGXA/wKLgIPA26WU8+L7FIaBUllO/
xWCxtUDRBQPv80GeTSxklCXjdE15h5sfAzkZIYdvEieHAJBPYtpEkswZYFtPE+WDAVyC
CFiZ4KjVCAklGnuWVU4bkE4w4MSCMx8n2VNP8RWURG6hjh0VImiILxeslGVzeO/
wcwlQQiqF19CXdvrMPMZ9j/
xfbLJIXz4kfOQI5ePKFzQ2EO9Nk7aUZDdfvwDJ9+O2cjQhx+Y8Vzr04YqnMm4BV1AssG
gbNUyAFJtEcaXatgGyOLbc00F/qThZRZ5hqlX4/gVdfL+Q/
jR1ouIvODFSfSAlPOrp8XB3MnZqP2j+HMmnokgUjt6S0ExbZyxicmYPk9CMjYRIC11KP
oyFK+NVl83eWzOIcx3X5wJimGQbyrDjlr4hAdVKGQck/
uH1qAOeE5qtTtbPZ2PvjgVQtepaxplQ6SLrCzwSP8y7M0RlGT3UQbIkZjvvngUpHQ9Sb
PwJknbJrI3jekP8o2Jtazw9rLEM0R8CQQ2LsL3WBInl5sfPT3UB6csQPSuEaKZCGgKnp
Sfx7qWEm7Jcat/Mw5uRt/diZVsT9fxQn8z66t7uCRygDcH9xIreq0UHFQhObKw/
ULsi0L3oNZVM7gmgHzdBNdHd1KpCBQEDpKMLPD1/
tt4bnsbK4biOGcpQ382BqgFfEpK+YoQIgS8LIR4GPgA8KiU8nNCiD8H/
hz49Jlr6rGhBANY1VHeet3z3B55BQ2VJ5PLuO/
AKha1j2MdOHjc+wWCJVxAWMSwpMkLPEqZrKafg5RRxSKxnCflfdhYZ42jLmzMkAeRD6A
cI8BYptI4mQyy4A66Qnez/0SwaHipCk40SGIxBNbfjLelHiebp+f//
ieFdy8m9cQr+FcuZ+Oua9iffIkuZ/fZoDYNuXLB79U/
Rpueo9PSieyF8MGTd0nNRoYH5S7GGKo5AzQAJg1hrxAkFitADIDBay2+de1/
scaTmDJ4CVShFFOaj+6GDg6VjxhE//
vZySl1oeipPTwMw8MI4FibsFM1Vk9L5LiHtGMA7iTg8xcwmyvRcrlpQbwLheNUCL+PeI
uHQFkCVbgyTjqSLfsaiXaf3Db0bPW0i3mOVfHo3F6/
jZuD25lwLPr3VdL62+ysqpgvhL54ynBs5EvbqSws5wd7LuZ3Wrdya2WSwKpxBswyWp7z
QS63YPTU6u6BYp2r8EgzQxfWsSmyCCo3Y0vJsC34xp4ryO2O0PRwgaevXcPWdXVcvKqD
2AlCURcKx6lQAj5ybVWMrrd5bP03qFY9GMIdU/
PSZMy2eW7bEhb9UkLf4Fnbij2hISWl7Af6i/
9PCiHagXrgTcA1xcu+DzzOfBhSikrXR5djb0jyT9EXCCkmT+VC3P3SJdQ+qsLQiQckQ/
gwikFqmtDxyxB5sgzTxwauBkDHQ4HsmzlLHN8U2sq2f6onbXnI2UcH9AJs72hB6/
PgSbiDuRmSmPUFVi8+XPk6oA6yzBfn8tDh7+HfVlncVHU/32vvJvQ3v8vAi400/
MZgf8f2M0tqCpRAAGdVC+lGm5V6HL84vQDz2ciwlmb2sT122o2fApnO8sBza9m/
qoIrl/
8CAL9Qef87HqY3H8WRCheFOoocNZxJs0gpHpIqp3zm4qepGr7XczmBwemB5QtRT2cN4X
rtDhmbmZSBtvuge1joFCxIjpqGGRL4PK48huw0L+brqXlEI7Irzsls9MxWT/
ezc65ZzBr2tesZWuXlEv+9VKgmI7aOd1BFb+
+aVazmfPXFOYOUKINj+O9r4Sc3X8inyjfxdyvv5b7atXQ8tAz9gAH9AwtOT2UyRcVWh4
6WClgOulCp02w+t+bnPNy0mntrL+DqZdt5R8ULNGs2h6b/
ESvMwHiI5vz0cWgh9kURCjFwsUF18xDVqmeyDEnCyfGf4+v5wa+vpelZG//
mLuzU2av0flJbokKIRcCFwCagumhkIaXsF0LMMvFyDqGoKF6DzIocn1r1OMt0h25L8Eh
yFf4DOtEtw5NxGrNFVqZJMkGEMgrkMYSrSMUJ4IxwVGzJsCOpVKxJxWjSfHyj8fHj3vf
F8qXc37+awXgIgNpwitfXbeOTsZlT8VNOnl1dBfrak7xpo4+vjOa5uC3Pjl6ojlQgT2p
KOD0In5d0kx8RK1CmGpjSZsIJoBRAKdin1ZJjydAQPpBzHAZgFgjtV9lXVklumYWOe0S
BW3jS9SzZUnIo/
sk+tC0iHEDBlDYmNnnpkHYkB6wIvxi6kP0761g6njnm9zAfejqXkAXFLYp3HCwYjqqCG
YSYx90mGLUFHfkqIntTKAf7Tjnw+nh6KucxBim+yCC+xKFRS6ACXVYMPQn2yCgI4Qbfq

6p7NMwJ2nlW+
+IcwkmmKNueYnxVgJ2ml8u9w5RXPMOfNK0lmi6DgcFJ7gtFT2W+QKA3hzPuZdDOUqZ48
AsPN/
nS1KjPkl2h89ayl7jWlwMMHBzG7Dz7slVYIz7U3AIfb4RABn1kmi0uLhvEKNZatLDpth
WeGm6j7mmLQPvQWT+/
c9aKLIQIAj8DPimlTAgxO5e2EOJjwMcAvPhPcPXJQS0vQ9ZVcMGiXt4f3otPePjv8XX8
5ttX0rgpibO346Rce5a02MpzLGMdmtCZ7Wx+uhy9ozafPHAHH65/mtsDs99q/
lh0O3eGt3DIb6EDfkUFPJMVsc0psWFf7F3DF9//
Kt7b38M7HltLyvwr9v56CdV7LJSRY5eGOCMyjEXov0Jw4aJuNFR+nSnnZ8MbqHh+GNnZ
c8qG1NmWoZPOUvf4OD2eGNsvN2jUMlSrM1U8mRn9doFeO8gD8Qt4sGc5wW9E8Q7nWDHQ
jzM0MmPz50tPTwsSbDn7FO2FxNGJBNA2jHNDjWscbyvU8nK8CWUi7Rb3OwUsJH5HvICR
S2zec+lzlCkKz+bL+MzO2wn0u+OIEgy6h+VGwxBPudu8x8CC5TgLOOk04tXdNFSs5X3K
x/n0LfdyW2AP1nvGOPhiBYv2+pH5PKaZWzAcnXQGrb2Lsi3Lee+y9/
CZll9xhdedHVbo8I+1j+AXOqBiSpt9lsPfdv0OO3+7hBXf68UZHF7Q440SDJJZFOW7N3
6LZXoCis8cs/
N8bOcHiL9QxaLHXsGahxIxszKkhBA6rhF1l5Ty58WPB4UQtUVvVC0w48FiUspvAN8ACI
uyuVlmCYESDGIub2Bwo4+3x14F4JvxRu7tWEPVrjzq0ATWSRhRjnTYynPU0ESVqAfAg0
FeZjGE79D2yxnh6B3IsO/5Zj5/
oZfOhs0oYrqLVUVSrU9Qrqa4zMhOeq38woNXdXg652XYCjNshbGLUSvbkg0M5kKMZv1I
KXAsmx2f/iXB1kupF+tQN8OAHiKwZQz/
qEYuOYw4RsTLnMtQUZF+A6MxxfKQu3J4Id3CloE6FqXGsU+xTMWJZJiXWXBj/
o7CqXKUto06EifUFeFvDryZdbEe1ga6uCXQORkPdTw8lW3hodFVPNfeiu+gh+pdA5BI4
cQTM9aMmk89PVVIASju1p5r3NvgHHshtqA4CoH0aDRExqjVJwC3vEPKNBCWfUoH9s5GT
89aX5wJhk21nkARgpzjIZvXUUMCdcUSksvLsLwCISG8PwjHMKTmoy/
ONaRZwNufIborzMtXNHOFbz/X1e/
hxwNhlLIY1ugoW83fLgw9BTdUIJMh2GPR80odP49tQC/
bRIuWwa+oRBQvKSfPoF3giWwLT00sZdvzbVTudHAGh2csRbIg+qKionh0kjetZHidwjI
9QUWxTFC/nWFLoYKxrZWU75m/
Q+5nk7UngG8D7VLKz0/5073A+4HPFf+954y0cAYohoGoq6brJi//
8s4fcIkxQJ8t+NeH3kj5ZoH+6PNYJ+Eal1Kyk5cIEKJZLJ38vJI6+ulkEcsxKcAZ4ihf
3cHiVyF1xyV8ff3Nk5ldk38XIOqzNFSO8z/
L7ppUIgBT2vzdvtvpHYoiBg1EcYKq3uQQPJAisn0vTiHPDl6kEg/L2j2Ae3hUjYwx/
sAviYjl9MhdaMwcizXXUHxeCuV+7lz6NFcH2wG4/+BK5KsRZLrnlNKLZyPDfjoBJuaGR
RGOjdXbR+xpSOYa+M2aBn6xZB2tr/
sWFxsn5vH1A1eRerKKlXd1YXX3HHebaL719FQhVYHUbVQktpQkHeuYhtRC4yg0HdurcU
lZJ0sM9yy9jGOQtXT8pnnS9Wlmq6dnqy+eCDYCx1ZItIAZrKTxzR3U+yd4tncx2UeiVL
149D3z1hfPAMS+LmrGojx1YyvXR3fymcoX6F4RY6ipid0jDywYPQXc7MR8Hv8zu1m6K8
avA+vpWxvhg7VPs0gbZ6kOByyNbfkGPvv8G/
HvMmj9wsvu4cUzPm5h9EXFo6NUlFPzyf38W/
2DVKmHvVvP5ur56dBFtH1nCOc0djJOF7PxSF0BvBfYJoTYXPzsL3ENqB8LIT4MdAF3nJ
EWzgAlFmXgmkpYmmatxy13MGhKgp0Kwb6TL3MfZ5QBuggS4Xn5MABtrKaZZWzjeXrlQW
x3A+1zc8/
mMCJbRvCNRJDK0ZNMIWKQ9ddyQ+OfTTO0hAOhTof6tIOWMifPLzO6J2A8gW2Zs+LnxYf
B2SnIJi0LxXQYKIT5bWolT6dt5KYotS/
kcbKntqKYLUeKiRNzDWciTnCHgTEaIbfD4L2J/w9/
fYq3tm7mssA+rvIm+UW6lldSzfxqzxosU0UIiX+rj/
J2a1axfAtFT08WyWbBdRe04xUmT+cCfGbvuwi3zzz0LDiO0kHNWTwxuIQqPcFV3m72Zq
sZmAjTYs+4ED8uFlpfPBHWGX18/IIn2NNWQ18mQtI0eOJgG/
7fBinfmZ3xnvnui3MJmc3ijEHw1w385cF3se6Of+fCSBdfXu6h7+ldC0dPp8DJ5lBGx2
n+dRUdW5fw+5c1Ey1Ls766h8d2Lse/
10P9LhvfQHqyqvtMWCh9UTTVk24t47LIU7ToOQ5VLwf45/
ZbyW4uo2V877zWipxN1t7THDvL+fq5bc4sIAQyHGR8jcOlDd00aa51mpMS/
4CDMZg+6QosUVHBDbxtxr8dyk7YJB8lIceOHx17mrD37Ec9xpFthwQVme2zpvx/
NvzA5XhWYNsopk1Xuox9yUqG00Eqtpp4Nu3COcVtvdlyfET+9IxUZ3PSadjXgbLPjVNr

zq5noiXCL5ULSDUaLNGf5L6RC3i5p5HoA370jANC4O/
LoA8mZlWccqHo6ckiV2Xz3spnyUmdZ9JLmXi+mqr9Mw96C42jdCRKzqJroIwdsXoy4f1
0Z2Pk4gbSPvlaTwuuL84ESyFu+8hLhzpV5YORdvLhHUw48K6tH8TqDFLz5BgMzXwsznz
3xbmEtCxkMknlI12Eumvo+50QK7291LxlPfXZfyN896ajFu7z3RelWcA2C3geeJGqUAj
FXE2m1uCx1iA1j2qUPdmJPTh8wrNbF0pfNKvDxFt0lnr7qVLd0j55aZKTFun2GPXPmch
kcl7PflywWRMzQgjEhlUMXRjiz667l0t9B2CBuMBLmD2kZaFs20/+E80AVFgOomcP9jl
S6fqEcGx8z+/Bv9mLfCLKDv8qfte/
FjVRoDWfg+H+yU4vCyZOoXDcleG5Du+gylf6rmPbE0sIHYSWR3uRE4lz4pgRHBvR1UfL
d9p44vL1vP7aag7urcbfrcF5eIg4UtL4a8H/
7r2OPb9TRc7WebWrEWOrn+hem8quDNXxIWRX77wclzJfsAeHMbwePtd5K9dW7uGfl/
+cj6/6GLGVS5H7O+ctNudEcFIpKu/fj/
B4kD4Dxiaw48kTGlELCX1X+lj1+t1sMHqBIADfmGjja+2vo+4JC9/
ze045rnaucI4ZUgq5ah+ZasHV/
r3Uaa6jLCsLjDpRFJtz5+iC1zicdBo2z1+tnDMNeyIOxN00aUDBTXY5J4yHOYa/X/
LyzsXUbpOEOjNYB7vOqX7qZHMY+waJVTTQXV5LqE/
BNyQX9LFTp4NARwLFCvHU0qVgKgQOalS9WsC3tRsnkXQN/
9eQEQWul4dkmt27FyGl4HfaNmOGHMwyP1qXPq2o7IKClNiDJ78FvZBQiEquK9tFRBHY0
iErC7yYaMbZGcLXO14ca+cX55QhJVSV8aU6ubY8dZogKNw081cKXp5MLkdP2YjsuWNpl
1DCawEV33+RyrvcA19nU3tooUGaBayeXoK/
HKTtVyo40s2QOk+NCWfbboztCssfdb390naQtn3+nSd4krCHR1n5WYXBNyxm25/
WIg2HVKNBrN0D5+9xqPMO2ytp8QzhFSp5abHbVHiqfSkrfjiE7B2Y7+YB55ghBe65dCg
SFcGQnaHT8vF7W95NfleE1q5xmChpdAklLCRIyzovPBizPa/
tnMeUcxVLmALHxoknKNuR4dP3vpvKbYLwgTTyFBNkSpgdAr0Kn++6iUWtPwZs/
vrgHfj3eWBsAjkPNaNmwjlnSCHdH1M6dFo+fpO8gMDPwjQ+vB97aPicW+2WUEIJJZRwb
sDJZBDPbqH12SmfzV9zXhOI7rU5EG3iQFMZOalz4MlFVG+13Er7CwTnlCElLZOG+4awn
g9w68N/jGKClnMo39yLPTZRMqJKKKGEEkoo4TxC+IVugh1R/
n7zB0FKFu8eQxmemLma6zzhnDKkkBJ79z4EEJmyIlhIX2gJJZRQQgkllDA3sHr7oLeP4
Gb3d4eF5wUUZ/
NwTCHEMJAGRs7aS08dFUxvZ7OUsvJEN53vHM8xfnD+cyzp6TFwvnM8x/
nB+c+xpKdFnO8cz6ohBSCEeElKedFZfekp4HTaeb5zPFf4wfnPsaSnZ+7es4mSnp6Ze8
8mShzP3L1nE6fSztkfx15CCSWUUEIJJZRQwjSUDKkSSiihBJKKKGEU8R8GFLfmId3ng
pOp53nO8dzhR+c/
xxLenrm7j2bKOnpmbn3bKLE8czdezZx0u086zFSJZRQQgkllFBCCecLSlt7JZRQQgkll
FBCCaeIkiFVQgkllFBCCSWUcIo4a4aUEOIWcRuIcQ+IcSfn633FCiE8FEO1Ci
B1CiE8UP/9bIUSvEGJz8ee2WTyrxHGeMFccFyo/
OP85lvS0xPGI5yxIfnD+cyzp6clxREp5xn8AFdgPtAAeYAuw8my8exZtqwXF/
8fAvYAK4G/
Bf6kxPG1w3Eh83stcCzpaYnjnjucDvtcCxpKez5yilPD2P1ElYmhcD+6SUB6SUBeBHwJtO
591zBSllr5TyRbtcAHYHBeHuPyeqB7yu89nGKDzyCgG2CiG+I4SEj+0+Ej+0/
gVtxrbh3CSFWHuPyeqB7yu89nGKDzySEEIuAC4FNxY9+H7gH2ApcTonjXAjxXGO/
F8hxDvOD85/ja1RPXwsczys9hf0f42tUT78GtALrgH7g30/4jOL+4Km8/G3ALVLKjxR/
fy9wiZTy/wiZTy94+47mPAHwF1KmrYT3jKH8GJBJCq+6uadyCVPfY7FYVcoxAb1cWq8yPKDMR
AxoimZl12/
P1AYThYHSbSNM65nUmBWxMvATIkcak8L7XIsciv48BARV1+bnKDyDJuAN87TyXYQ9wzy
lzFKDmFjzHUl8s9cUFzbHUF89dOU5FjjQFmZ/
JIJyENusWHI1ZWZpSym8IIb4D7PETDl8irp/8mxoOs/HJUf6ucgcAi3/zEZZ+
+KVjv1ECXVN+Hyv+HKs1x0LflP8feZ9SlKxjMyh7GGWAleIiNslHMSkcasX0Zp1rHKdg
NhyllN8AviGE0PyEzXOVH8Aj8qdZ5kOG48VGKsBsFy+nLsMezn+Oh1oxvVmv8b54yZMj
fKZyJ3AaMpTy/
O+LZ4HjguiL4iQcJWe5Ly5kXZ2KTfLRE15zOoZUD9A45fcGpjdzElJKSwjx+8B9zpXr2
P8RlYZfqPjv38yvvn4VP6m4GoCmzbOzEI8LIRCajrTMycFcXdZG+5/EKH9eo/

zbz7mfVVay629aCHSp1P3rs4x+9DJGN1qs+Pdx7N37ADDwkWOalbwwOM6Ak+L4b2PYe/
afEsewKOOM8BPi0EvONL/k8fidMRnOwrCYQ46/
LfI8nznOX188e7p60n3xnq9fzf9WXHPq/
M6uDEt98fzgeEp98azo6ooltP9x9KQ5nixOJ0bqRWCJEGKxEMIDvBO491gXSynvB0i0+
Hjyui8xulJDmgUqv/YcjX//LI1//yzeX71w+AYhUEIhFL9/
eoMDAZRA4NitEgpC1xCqOvlRoTbML274CqMb7cPXRYL8000/
xnv1CACjGy1+ceNXyC6KTj4/
oleTFWlyag7pGtwLg+MMOCmOi2OnzPGM8ROK+3MG+TnSAYgcj9+8yrAuzD03fnkuZPj6
s8rxkGEhTrzEO6EchZgTPT2jcjyRrtad433xBHhN9MW5G0/npy/
OkuNxxxshGL14gffF40EI8qcoR6F7UMNhhDY7X9NpnbVXLFT1RdxaEd+RUv7j8a4PizJ
5edU7KaxqxNg/
hNXdc8xrtcYGfP+T4+Vdi1n60RcB94scvLsB21Goe3cnTuYYe6NCTLNW1XAYc20r+mBi
0uIUhoGzYTlqIo+zfRfq0layi2PwJ8OMpf3UvbuTgQ+uY+WKH/
PLP9tHPj2GxPmreec4lZuiIhSBtKzDHAfi2HsPnDGOZ02GR0ANhzEvbEXvP3UZinSWLO
leKWXD8d51NjkKTUNoGk4+jxoJk7uoDW9vArt97ylxnC89lYUCwuPByeXBsadtkU/
FXMhxwfTFI1Hsm2eL41kZa+Colf9cjDULsS/
ONcd51dMj5gmkU9wGE+4iwLFRw2EKG9rw9CUmd2EWMsehH9XjOAo17+o6vhwVFaGqSMt
EDYVO3BeXtZFpiSH+eGgax9s/9gS//
Zsr2PrLfyYhx45roZ5WHSkp5f1SyqVSytYTfZGH3yiQigDl6HZpzY1Y121ArawEwHLUo
3ZepRSTHx15/ZSLpv1qJxIoT706zW0n83nEs1twtu9yr9mzH+/
T7UxkfDjy0AACzVfUcdH1f0aQCHPCcVET1vXH4agoM3NcsQStuRF1xRLUJS1oNdWowQC
oqqsY+Tzq8zuw93eekONY2n96HE9HhoqCEBKluHc/
eX11lfv3I1dUQriGRiaD8uTmWcnwWPyuELcCDJyQ3+lyPNb11VWgqKjhMEogMGlEoSgI
VcVJpTGe3oGzt+OUOc6Znp5kX1SbGsheswqtqgIAMcMzodgXn5hdX5yJ44YbzyJHRTnu
9e6kNOUHUKMRUFXUZ7Zh7zs4+fmC7IscMZ4eGpsm+
+LM08PkeFo0ME6F31nri8caTw9xPAZOl+NZ1VOOlqN5/
frpHI+UpRDYySTaU1ux93ackp6ebY6Oo2AX23DMuf8IzHa88T+z+6i533RUhDMrZmf/
rL3Rm1v54fe+xMF3Nx71t473NvLD732JsZtasbp7yN+WYvkfbpv8u5NOU3NnL/
Xv7cHJZKZdPxdw0mlq3tVFw3sO4mQyVH/
zJV69KoL3vpdP6jnH5KiodLy3gR9+d2aOQtOQhQK17+2bxvHu7/8Huz4dpuPORnb9WYj
dv1fF4OtbcJY1Izwe1LoalMY61Ioy1HAQoXuO6eJ10mnq3t15WhxPWYaKiiwUqPndFHX
/Z3yaDEdvai0aFrprXExZDSuhEEokjFDVouGhInSPy3PKdztX/
ABGbpklx57emfX0QyPUf3hwGsfh21pRYxHMta3IlS0o5WUIjwccB8XvR/
h8OAUT6cjjuujniuNp9cVMhpp3dVFfXAF3vLuOu7/+BUZvWAyAtKyjvFHTIMRhr9VJcP
Tfv/mscBSaBqZJzYfHqP/I0PTx5uZWd/vBMCZ/
EK4xnLuojdzGVpRIGMXndeWrqNO2P4XufjYTx81XR/H95pUzzg+KMnx3N/
XvcVf4HXe6Y9Poza3TvRiniPnW00MyrH1/
Pw3v650+3tzcOn2cmWOOvl+dxb6YTh8lx7u+446p0/
rY5JjquGOpqiIdiVCEq6cnOWecdY6znfsdG2kWpuvulMXONBQ9d3YicVpz/
+kEm58SHBVqtSByhjdXbLe4/KFPsuSAG8jmpNNH35/OuB380PUPH75+thh//
2VkqwSN39yBPRGf5gad6jKUZsEVyEnimBylQ8W26RxlPu9OnIDaWI9VGYZUHiElil6PG
ZIctDw014/
Qa0RZXTdIIu+luzzGxPIg3tE11GzK4RlKI5Pu9yV0jdE7N5CrEDR+qx17YsId6IsrgdP
leFoytG2cifikm7l8h83lj36CtoM5mIxrU1E0DaWuBrssCPEMIpuHVBpsG6EIRu/
cQLZK0PRfRRnKw0uHuZChnC1HKWfW02QKIQSK30+gT/
KuXXeSrRaM3bKUbKWCYoKxLICal2g5iTFWQE3lUbsGikaIw8jbL3BlOFVP55DjKcux2A
4nm51+/SOfZElH7qit9aMgBIrPhwgFGXpjK/lYUU/
jCXer2rbd73W+OArh9knLQk7EJz8r32G7401HDoSCtB03HtPnRa5vJF3nJRdR0PISo74
KZTyFMzrG6HvWF+Xo9kWXnwNCHMXRPpN9MZM5LJfiFo/M56fze/
QTtHXmELq7qDshhGD8fZceHk/jCXeCnm8ZTr0/m0MIgRoOEzng8Lqn/
oDFnblpHE5kMM7bnHE8jlPbMOWz8p02Vzz2CVo7sm5/
KhpLCB2hKqCq7hgDKJoAXUd4dIbevJ58uaDxO7tx4knXg2S6C6IFIcdU6pjXHxOHtjIB
pDNdjnOoq2fdkDoefL98gaW/
nPLBkXu7cHjCFALfPS+y9B6mDw4wc6c4pHRCEHpvL59d9CBf/

tmtMBEvftFHvucM1AyT8iiOwuMB20bmbfKLyhlfauAd8yMk5EMKVtjmN4m1XF+9G73Gp
s0YJKDk8bflcaTCgBXh34feSXmygFLsHMKjU/7eLj7Z9Aj/
8cvXI5JJdwuwaKgcUpzZDCAni6NkOElUgHSQljy8ClJVAr9+lWX3C4QQriHlOJMdO728
kvgincotGtpwcnIFhXSI3tnD3zc/xH/
87DaIJ+ZeXsd53DE5ToUjQVdRwiHKdqaJf7+G7EaH/AU5hJBIKUg4AienInIq0R0+/
EMGkWQWUTDBsqh+/0F+v+ExV0/jiWnG8KGB8EzhuHKc/H9xgBLgu/
dllv262DahgJzBGzXlXhEIIKvLaPnAHj5S8yRfvOd2RCaDOGTEzHT/
HOPYHJVJXT3UbqGqBH71Mkt/
abv9qGjwCY+O8PsZvCRAakMWrdOLMaZgNIfxqSpKMkn5nV38cdNDfPGXt7tydOzDA/
wZ5HkUP6G4Kd9OkcMhD6+UUCgc7ovFv8kid+DY/
Uso08fTRKrYj+dJhjO0V6iqK6eyGGXP9lH2hIUzPoHjzHLMmGnOOIuYUU8nDQRn2vwnN
J3AfZtZ9kDRQKC4zS4U9zvwGghdx0mmQEpX/
j4vwuel4X0H+Hj9Y3zxvrcg8gWEquCQQ+bnsS9OxRSex59npstVqEU7ApXI+3r4bPND/
McvbkOk0pPB5Kc7ni4oQ2pqIBwwOeC4Llh30BZeA3tNC+m/
SZJ4tIa6f3tu8j41GICqcjJLKvDEC2jxHHJ/J04uN6k0wjCwv1DN34U/
SHR4x+QE77rmD6+GzyTHw1tUirvqU1XU8jJEyiTYpyFsSaJJ4/
IPv8IDe1dwz9evJr5EotTk+MO1v2WZ0UeLliIjBQWpEui3UPZ146SzKNEIVERJfq2Mz3
g/
RPngDndisgug6dP3p88kz6N4K0V3suIq7dplJP8xS+KxGur+bZPbLqGgBHw4rQ30XRUh
uTZPc30vvBpF9g64gcxFWP9aw2ejHyTSv23G4PuzgkPvKw5YCAU1EkYEA+TbqphoMYi9
u4eOARV9j8KFG/bwhsqtpB0DXdgElDymVBk0I9z/wLUEN/fijI0j6mvINdeQ+abO3/
JBYsM7XI7SBkVzJ6ozradw9KRU1N1DkI5ErF1O8p+LcvzX56Z9N4rPi1JRhkykcDIZV9
eLzzKX1dP3Oj/
iV2V8Or2Munw3imFgp9LuOw55RM6mjh4m5v5b5K+sXUHic7lJjpMTlKoi6qoZurKa5IU
5rm7bx76frSS0awwxkUSaJtJ2yP1bHZ+JfJho/
45p252ut+AM8ijKa3JMc2y0+jpSFzYwfKFGtrlAeLsH2wsV1/
Ux9GQdi7+1H3QdpEQO2+BIhF7cAlMU1ytwqKxMOIwoj5H8Vi1/
FfgQ1ZkDKB4dZ4qX66zJb0pyg+L1osSi5FbUk6r3oBYkmWqFut85yP5nmmn93gCiEEDV
NKRlIUJBZG0FysAo9sjojOOH/H9V/
H30g4QHi9tN86GXcPg7lZKjjKh1K0n+c5aJJ2to/
lq7azh5POSWVJOPaaTqVLSMxJOWRB/djz06hlAK0NrI6Low9o/
gLwstVA3uQebySOkcHtvmi+uhxWNxcTM5b0zO/
Ye9TQgFxaMjAn7QNJzRMaRluUHnFRXI2nKS34nyV/
4PUZ3umEyOEcU449MZb866IaXlJA9ldLSjPXeAO7iotXVgeJAjY0gpXXdkMTBXSkkh5u
GPWx7hT/
e+C+HxoAQDCF0Hn5dCTYR4q44xrmLEDQKFOrRUBmnZiKAfOxrE15nEXzCRpjkpoElhcf
qKo2ePw7GYOi0a65CGB9HR7QYb6zoib+KZ0JCaADRuiO7gvuwaqp8aB2IkCz42t7j7y/
VaAl04BJQ8wpHugKcqUBEl3RYj0JtzsxLy+UmvnrRtBMeOS5ktTiRDAK2+Djw6dnev+9
5DQYbFAF476OEPFt/P/
617h7v6dxSECmga+QofiZUmVyzdz7WxXfww8kY0r4HI511DsTyGr2Mcn20fHrSnfr9zg
FlxbKxDeg2cA10uR1VFBPw4sSCpOg+JVnhg6Y/4fNll3J26hGvLd3Oz/
wDDjoZX2NSpKgoKg/
YB7vVfDx4dpbKcXGOMiSUeYrsLGANJ14CcMikJIY7nMJs1jqunAEJBq6spyrHP5Tg1Hd
iysMOuHP+y/
h2HFzuHVsA+L040iKK4A5wzEXe3Zg2DZL2X3Mos0ae8RA4UkLoGhgGJxKT3dC44nrSuz
jCJWkEPH1/8IH9Te8d0rzXgRPwkF0Nz7SjrQ110FJYjJpI4qTRKJIysr8LfmcBfMKdth
c4VZiXD+mrw6Mh4Aru2jPGlGmJdnPe0buUu6zLQHb7U+is+0vkh8HmRqoJwpOstB4Thc
eWuqgiv97AxWFNJuilMoD9PKGshpXSN/KnZYXPglZqdDGuRXg9OVy/C40EG/
SSbPMTbwBhXyFZJ/rz5fj7Q+WHw6Ai/
FzQVLBsiIbK1Afx5E5HOQDY7aXyqsRhUlqF2jOGzbOwjx5s5wqw4NtSDR8c62O3K4Iht
Witi8JdtP+P3u96L8PuQfi9OwEu6VidbpZBstdESCnpSIVoWQTUL4EgydX4mlkHVyw7B
zgwymyt6ZK3Dz58Dw/
HEuirQ6mqPnjcO7aYIQT7m5Y9bfnV47g+HEYa7q4OqIn0G0udBqiqq14CCW0/
SbqwiuTiAf9BEy1jgNRBWADKZORlvTqv8wckiLMrkZYE3otRUIUfGsBOJGVok6PnpSj6

+4km+/eU34B920NIOyUaNfJmg7qkMajyH49cnb+l/
XYhcpcQ3KMiVSYw1EziOMtmfC3kNu99Hzcoh/
nnpzwF4MrWcZ9+1FtnRPZkuKqbsHR+JTfLRE6ZAHpfjoUKhxfiI3p+t5CPLnuGnf3Mzo
X0J2N2BUl6GDPphdByhaVhNVahjKWTfIIVLV5Cp1rENQbZKkF6V43dWb+YdZZt4z4//
kPABCHVbdL1e4Ue3fQVTqvw2tZLnbl+KddAtCTs11X6mjjEbjtP4jY1Pj92Z0rF7frqS
t7e9yqa3LsPp7isG0puTKxzFayDqqiGZxh4aRjEMd+CuKmfo2hre/Ae/5QJfFzVanA+
+8gHMPWHafjDC/vdU8M13fw2A3yZX8sLtrVid3ZPvnebRnAGPyJ+
+LKW86JRkeOgdRcXq/dlK3tn2Ms+
+bRWydwDh8WAvaSDd6GdiiUq62eJrN34fv+IOvpWKuwd/b3ItFVqS6/
z7UItf3c5COV1mOU9PtLEu1MMNwZ1kHJ1HUqt49q0rcTq6XN080stwCjI8IccpXA9xfO
bOdYjeoWJqseLGEaUz7gRdV4mSzCJHx92JVFXcwPlYkExTmFxMxfZC5XPjCCkZWxdj+N
Y8T1z1ZcYcjWczrXz3c7cT3ZWCF7YhDMPV02xuRlnONcdputrZCxS3v51DuupFNNRCPI
U9PDypZ4rXIHP9aryf7OPWmu1c7t/
LVwau56XeJkL3hhi62uT7134LmEFXi+8+1gQ1674YvB2luvKEMvyjFY/y/
376O9g+SfXqIS4s72WZf4Bv7r2cVE+Y2ifAGLfwjOZQhyeQuRzEIkifByvsxQpoWH6V0
ZUqVkAibAiuG+Vrq+/
CkQpPpJfz+Ls2QEc3TjrtytDjcT0bU77LqZh1X5wlx3e3vcSTH9qIOpLACfrZ+8Eol16
2i60/W4l3RGL5wT/kEDyQwvHrOB4FM6BRCClkahSCPTb+/
jyejiFkMoWTzdHzqYv4yke/
DsATqeVHy7D4fuAoWZ6JvnhH66u88KY2d0yfEv8jdA21opyJK5oQtkTLOWTLNCwfmEFB
rkKiL0vgN0xUxSG+qQrPBKgFSerqNL+57KtMOB4eSa3iibevc+fFXG5aKYGzMd4c6osv
vHkJdneva+R4PAiPjmyswfFqSCFQLAcsh643RMg2miBAGDaRSIZkyoedVVH9lruT7Qg2
LOri0/W/wZQqO/
L1fOmbb6F8p4nngReLiUuau9A5RY5nP9g8k8E5cHDmPyoqikcnfyDMdzyXkSsXWH4VYa
mkGx3smElhm44vXUBNFoMFhcA/
EAAUEsssfBUZNtT0YEuBI10lS5kGXYEoGyu7aNFSPJurZ3+mEserofi8rlUKk0HfZ4zj
FD9+fm+Yb4vLiTogPRqKz4cM+HBCXpRBCyeVRjVNZC6Pk8tjDKYQToCJNgPbA5rHZsL0
szNfD41ZEtJHqAeMYZV/
6noDWUunayxGS354yusl2KeXiXNcflNQ2BfmbnsDrVbcXU0cCrAtfg9ONgsHuqZ4y4rt
CvoohARLvf1UqUm8wqYqnKIzFARH4u8X/M2+NwMwMBGiNT90WlxOmuOU7y6/
N8xd9kYWyzTCa4CmYYU85KIK6UUWFQ0T6MLigfgFPNC9AgBNdbi0+iCGYhJ39MkyECs9
ozRr43iERZmawpQKPxq/hCd62qizi4H4RSPfjSE6fV09JsdDsYlFjv/
jXESTKl0dzReQQT/Sa6BoGlgWYizpfl407kSx3wlbouZs9IxASEGhKoBU3IFdCOi0/
DyRWsGzYy0YCRslYzLZQ5S58SyelK7mx90PDnmoFQ15yOu5f/
ozhCIQhkEhpHBxrJdGfYyAsFgX6qE3GiVFCH1Q5zP734SU4ozpqpNO4xyYeYkvDAPF7y
d3IMS3fFdQqLTRI3k2VHTT6B1DEQ7JsQDeIRXfcA4tnkdJZpDZLLJgolh2MbREYAVUcl
GFfIWD43NQcgphbx4PDvck1/LowDKCtn14Ve/Iw33+NPcvj8fx0JyR2x/
mLnkR9bqKCPuxwl6EJehJRSfPrAsMOHhHCijJDMI0UHw6qXoP+bBCIQyZShWpelGsCtR
EEGV4HGNM8ncH3ghA/
3h4Zhmeyb44ZSsvvzfM3dYGWs3RSePtULiEEo0gI0Esr8CTlHjGCwgbzICCGXLT+DNxH
3mvjqo6mGUOtk+gxwVm3OBTB99K1tLpHo+yOJfEOeT9Pka5gjnlOMlVmTJvuM4E4TWgr
hqzPECu0oNUQE/
a6AkTLZ7B3xfG0TWqLhmgJpCg2T9GbzbKaC6Ag0BBYmgWV5ft4QKPSo+VpVfNkKuQ5GI
qnmO35qSwoGKkFK+BEg6x5K9fRUTC7Pq/MZTyAv5Ajqtrumn1D/PzZ69DzflwND+
+7gT2zj1EdkJ5yyKuv3crVwZ2A2BKFVNq7CnUkHd0QtWuW/
2RTAuffeENePd6afBmUCNhGB07HINyJnCoRscUb1fLp59D8XqxNyzH0RSUyjLyDVHyMY
1wpwcnkYLJgV1i79iNp6Kc+K1LoTHDVYv3k7QMfjl4Ie9Y+TLPVLag3VfGol+mKdwdQh
0ZY1F6BGuqK9qxkcdLST9VzDCQtPz58yAUZMStDmsnk5P73ZNBvFO+b2m6wY25mgD5cs
lyzyAB4X5fIU8eqUlEIkXN9/pxvuoavs2ANfXdh+IGzhJa/
vx5dyCrroJgABSFbKVOukHwvsue4dpgO5Vqmh89exlLfm8T4Bab2/29aqo8SbqsGACqc
FA1V9aL9BG25Rr5xegGdnxtNbXPDCCTaXd1f4a2FaZBCBSPXowtsGj59HPuim1FCzIcg
KEcdkWIbLUXX7+OMB2koaINTLjeGgBFRfP5EOks3l4Hb59A6irDF0WxAgIkOKMePttxO
z2PNBHbYxN5uRcZn7JKdU5/
Ap4VpKTl026cpV3cypKW5XqhvAZ2InU4OPyQrsliwkQkRK5M4a3Rl/

ArJgqSiJpBV2x8IxblP4zjfKYfads0S2e6rp6FYHMlHEbWV7L0H9oRgQATX8xyZdMB3l
/+DAN2mAP5asLbPEQ6LBCgpHPY+zom2yc8OgogQgaFoEKmWkBFDl1zsPI+BibC/
OfQdTz/
k7VUbs4jJnqRRQP4VDO7Tpqjz4sSCrLkr19FSomzfjm5uhDZchVjXNCzpRaPBwphQajL
cgtsFjmq0QiZ160iH5OYMQczqJCpE+RiAYwJP5E9OjUP9mB/
d3BmGZ4FTO5iOPY0PT10DJrw6IhgAKupinSjn6HLbCqfV/H/YgsewBeNkPzgKrS0QN/
jQQrXGLFXZpFCYnb6qHxOo/D5EOrQKE2ZYWxHTouNmqtQguMTdT3tLX/
xgrvQ9vkQAT8iHGLkonImlrlZfyjgeARVL2iEn9tC+Z79VLcs4pa3vsp1gV34hU1Gqkw
4Bj8fv4i8o3NxaD+LPCOMOznuSa1mS7IRq65AZtg4fIyyc3pjzYIypGShgJNM0fVH68m
0FdBGFIxtPoIdBluCUV72C6wKyFUYaFkYXVWB88YKbB8Uog6f8N5HeXEbpSAVTBRaPYP
YUiGg5Jlw/
IxaQVrrh9nvVMGjrqWvhEKuC9osnDhL5ZSIHfEsIej9s8vINNos+2YCZSSOzGTx5gsYh
geZyRSzs4pZdo50PWflMXzLJrigqp81IbcyrCMVnh5rpXuojMWAKFiIeNKtR3SsbMZjV
J2eS/T+2WVk6m1W/L8e7MHhSSPnWPOG0DSU8jK6r9eoWDVMSFh4BWQk7Oypxd/
pBh9Lyzq+bM505iVMbtP2fOoil+MXBpHjbvxPdIsH70iY+O0+ytQMXmFzweqD7Pnry6l
+0cRJWxzoh7Gsnyd8S4jnvORMjY013TT4xmk1Bnkx0czzBxcT1sGsj6LHU25HPzJz9Ux
ASpyCOfmO3k9fTrrZou1uE71/AplKo3UOERwLUKiLAOAZTnOoore0LDdWIRpC6qob/
2faCNNGy0scD9iGQMso7B+sQNchW64Q9nuhUEBkMm4WazHmTDryjOrpZF+st1n+jx3YQ
64x6BRMhF005o7cglNU1EiYoevqmVhtUa1m0YVrwqvCIWfpeAcziPHEMbdEDvUHJRRy3
5ec8Ti204JMJhH9goO/
t4psawGtQ+ep9gt4PrkWxQJhgxGX2B43JupQ35nM6PN6AVCTefS0Fz2lkjFV/
ME8K9cNUOeP0+wd5ZE2C8UyaNgBmOac1KCaNcd8HgfcOaPRouWnNsZgBl+XSbjDhxXUy
cU0FEviGUkjMoe3qwCqXs2RaDIYXSdwPBLHJxmvclDjGkbcjz9nHS5XMQ8B5tIyJ//f+
+nL3fHmX7qxB4dcgz6fB0VhYlmQTK1AmDaOXoz7UxSkoRPbYyIVgWJK8lGVQkiQq/
GAhLJ9EOouuHPGoYX+lPFF2naxrt0RGYJzTtQ13nr/
9BIyTTZLv5tGGY4jR8cpe8VHYCBI180a1ORZVD1Kd3WMkXWXIRWJHXT4tK8TBcmwY7Az
X09nvoLubAyPYmNKDRUHQyis8XbjSIUntTay1ZLkOy8lum0cu33f9Kz2Yptmi3kzpA7t
oTvpzORAKS1XaWNXD/CNpT/lw//9+0T32QR+9oK7ZWAYdP7FBnLVFmpaQdbkWdEwQL1/
ghojQas+jlEc0HJSYEqF8mJcSlrqqEi8ism6WA+mrWJ7y9G8HoTf5w7eZmFKFsDpD94z
cTyE0DWD/NniJ/
jRF1+H1dfvCm3c9UocimUSxUEW6SACfgqVQa6s38XFoQNUagm8wsSUGnfFL8IZMUDmwL
QOp7aqqhvwalpuvIk8nG3kPvb0OM6G3/9+4YrDK9MjJ6Pg9HOTZNBPzZpBbqvbQUARKI
AiJQwYBPokMpd3twsmGyBQgkG3NtXUIwPOhgwVQeSaAf5i8RPc9ZVr3fc7EtHZi28iSc
HRCAkLXcC7a5/
nwdeneSF7AZEOBYYEI0mdETWCmlBRc4JNjkJXJIZZprJrrBrZ7cfRIFvhwRMKuIPbkXW
AjvxO54rjIZ5CELl2gH9ofYgv3fsuPI7ELpg4g8OIkTGcpiiOoaCPCPB73WD5ZBppOzg
BA6kIhARh2mDZqAVQCmB7QM0JzAkPqiEpRAVWeQDNcWAiPlm7RugehHr6ejojxymYqqu
T3+ch723RsJh67pe0bUQwwPgqSePiYSpVDRtJXjoUpEbO0giMJHBS6WnyOkpXpZw0Vpi
SDTdX/JxcDvJ5IleG+OKSe/
nU1z9K+U4T32Pb3CQeTSN5y2osn8DyqmgBL2pZzO1n4GZOqgrCttGyDlpWAVvg1S1uq9
qGVxRQhSRckySdjiADPkQ6g7CsM+JoO96cEb5qkM+2PsxX7n0H6kQKq6MTAeiKin7RSh
yPirAc0DXUsqi7Te5IPNu7CctGxlYbOEEb4bdY1jhIfyJM4aUIXu+UKVJRUQL+o8ebM8
lxiv6ErhnkL1oe567/
vBaGinGKto2wbdL1gmyNAwpYfoHVWOHytSX+AxNg2YiCidZYjlJtoKUUhAWRgwWM/
qT7vkMJM34/0rIOZ2jK4pFPc+RBVbxe0PWj+6J0CF0zyD+0PcgXfvNuAomsm8W7ax/
eLj/
O7StoqJjgzbVbMOpN1HWuwedVTGrUNBlH46BZwSupZvYkqsiYHnyaSX8wSr0+jo5FnZo
kbQygeyzyMYuRtRr+wQDqDhtUDygCAZxstuK8GVIHPrOeW256iT2/
uwL50vbDf5AS64dVfKzx9wkNSPSU43buggmOQ9MDKYbXBfi/f3wXjfooUaXA3/
a8gR1jtVzg6ybteHgh2cr9m9YR2akSelM/yZxB2ZcCdLxF4/Hb/532bD2Lw6Ms/9IO/
uvlq1j60U53nzkUmnGgPRMco5/
W+WHw9WhDHYdrSRVjtKRt42xcxZqvbCOmFwOUu9eQyZv8bfmzVKpZbCn46+7beXlfM77
9BuW9En17h/s9AdK0UBvrCPwgyYu7lrD0o4crtE5d5ZxpfkqfW4b/
qFX9xatY+p87iWkZHAT3f+kqYu0ZsmaWjOMhKHTy0iIjFXwDCqGevFu8dMr2qNZQf9SZ

TMCcejBm5CglslAg+DcBvh39HXyJHsSKVoYujjB2scmGZQe5JtLOmONhtUdws3+AjUYv
N5etgQ6I7RQgVHDADAnMAOiqTcdABeJvyvFHdZxaEA4kG1Qu+PMBfrN1NUs/
Mja9cXO0OjyeHPUvlfOPte/FE3Kw11ShtVWgT+RR0gW8/
SniK6Nc+cMdRLQMjlT46RduoPLJAYRpo5g2YtzNwkNT8SRsPHGJf0c/
PXc0c+WtO3hs71IKaY2L3/0Kd716MUs/
dHhr6ZjenDnmeCJdlRetnKar9335KmK7soSXjHNllXv0hIpr2N7ds5HhXRXERrZNm2yP
pavO+OHt+zPFL/JXXv6h8YPkL5H0XK/C9etQswI1J/ANS7KVgrf+
+W9RkSRtLw9/9nWEfrMNFIVCQ4zhC7144m7h2GhVklTW4Ce/ezO9V/q44LZdmC/
HCI9I6u8a4pFtK9yx5gx4Lk7E8T9jd6CnC0i/1z2qx5FufE8qT/
KCMq77ysuE1Bw5R+fBf7qK6IPtkM8jbInUJcGaFKuqBrilfDs/
VTYgn0ngxBOuQWlaaI11+H6YPXq8OVsc/
9LL98K3o490utnL0RAy4MUKejBDErUmw1cuupvkTT66zTJ++IVbqX6s3/
WWA1JVUPNRFEtiGxLNFhhbDiLzBXeB6jiIxjqC343z4q5lLP3IS4ffP4dj6v6/
ufC4HP8zfAeBnR0Ir4G8ZDW5CoNMhcq6C/
bx5qpXuSXQiY7ARvIXfTexfawG21GIp3xY/
X7CexWCfTbxxSoJC558qZzvvOtqXn7zF9iUW0RfIcY3LvpvPrXz7VR9Lo9MJHF0z+F58
RR096wbUmp1FflVjZjlFnnHff2RhyeGDubQMwZKQaInTXdVKARSSrTeMUKVXp5KLqXWE
8evFOhLR0jkDJ5NttGbi/
JyZxORXSrl7Tn2X+CexRPTXOs17Sg8OdxGz2iUqqVJvME85lVrMLrGkQPDc7JtMpWjJd
XDHNe34el3OTpbdyGEwPF43A7P4bRxJRgmF3C/G0Mx8SsFLq/
pwJGCGjWDIWBMquwdrcS33yDUJfEPmu4KGIopyA44DgVbwxMqYF23Hu+
+ITfj5DQHuWPymyJDZ+su17I/
znMsR8Us3i9VcAyVCn+Gaj2BKgRpx2HYDqOnJFqyMOMqwXJUPOE81vUbDvObAxxTT6fI
UGk/iNfwgKZh+3XyZYL6hjHeV/MsAKN2gB5rnAHbz/
5CFVpGoOYlnpQE6WbMWAEV2ycpWKqbaRJPIUI6lk/
gHXPw4J755Ivk5lSGx+U4RY7+vSMY4yHGlwUwA6D6FYQtEaaDks6hFiQJy4sqHGypTAp
cFCxEruB6NzQNIXWE5eDoClZdGfmYpNk3ipPW0Mc0UraBPzyFY1fPnHM8bl/
kxLrqILClgqOB7VVZWj7Mcl8fCgp5aZF0JJ195QR7FXdBc0T7Z9LV0615Npu+qBzsJ5i
twLmyDKfcJFaeJJXxkk170DIeEDCQjxDU8q4MD0FKpCKwvKBlQLEEUX+WtGYjFddbXrB
VvCOSYL9N1tYxwvnpejoHmBXH/
T34AgGcyqj7vetF74IQSOHG5U3TU4E7TuoqtqHgGA4RX45G3zghJYdXNcmqPrfu3aGGO
HJexxu27UbTNDd13+uWNshX+cmVaVg+iYqbja4LG1OqqHkgX5iMrRQevbjAkegpgZYRr
hFlmsUzCSWKPWXOOMTxDPTF43FUhYKjCFQthlQFZkChEBF4VYuc9LCzEMIrTLzCYjAXY
iwRwOrzoycUIv2ScLeJdyhHqi7kxoN5NaQmsZHcO7SWfWMV5BdrZPI6uZYQ3gO4hWThl
HnO21l73l6drmsV5Ms7KKxv40vf/U/a/
6gMAH0wQfBAkuDLXWg7O3HiSexUGiebxZmIE3h6L3tuK+Pxd23g55+6ie7tNeRyOr94Z
iM7f76ctn/KUfPUOFoiT8NDAn+Xxr/+11f58i3f58fxi8h/
vZaWj3Sw7Y7FaM+Guft7/0HnHTXTis3NFceO6/
TDHL93mONkWmmhgLTMyRo9SjBA9pI2HI/CrjuaePT9l/
KTT93ClrF63hJ72TWiHI3nsy1kd0WpfyJL+W/24X1q5+QRK4c8N1ZnN/
k3ZAg9EuCH3/0SB9/TyDHPHJoLflNkeBSOCLQVL+1k/
3UeXrmmjFeur6TqyWEsv8qfLfoNH47sRUFhtxnhocRqAoMO2vDR6bKHzmQKPVzkN8MZT
nPBcVJPN0yXoVCVycrAUlVQc5DIGXQWKuksVPJSpoX/23M7d/
72Y3zp83dQ90wO30AGxZKopsSTsMhWSvSVCTLDAUROZdfHw3S+08G8Jk6wr0D5Xa/
Q+56qST2dlOGZ4nikHJNplFSBVIMg1SjIVCvYHgUlVwBNJdg+yr5bo7x4fQ2v3FBF5a/
2wEQChkaR4xOufudyyEwWISG+WKf6S53c8voXSdpeGh5UaP2Hrex+fxvihchhjnOEabp
6jTpzXzwSR1ZGfmknB643ePG6al65uZaax4aQmsLHax/jHaF+dKEy5jjsKFRR+ahB/
YNjh2OripjU1UfmVlePyW+KDK2ljcRXlyFVMAIF3tq8hZtb29m4tINcpcQ/IGm/
rcKV4Y01hB7a6bbdstGTBYI9brmDfFTQEhrl461P8PavPsBH3vUAy0ODBPtt/
L9+hbE36fgeD53Rvngsjs7iBvJLalBG4ojxhFtbKBrGqS7DivoIH0iz7+aQK8Mbqog91
YUIh3Caq0k1eAhUp1kUHqPWE2dfvpqM5WHk6gasFYvcUxekM3mu5lkbb47UU6G42aya5
tZa8uuMLfcweCmImhzWoJ9X7ljCC7ct4pUbqij75Xa3KKXpbn/
KgonIFtATJjWbLKpeKaBUlCEiYdcDbFpYHZ3T5oyOOxuZq9p8s+F4aFdGMQxkLo9nXz/
GhI1SgJe6G/

nanqv44KMf5j3PfIT3bv4gB0bLMbM6bXenaf1+H7X3dGCM5snU+5hYbeG7bZDPfOvbPH
bb5/EKle7vtdHwvl5eeP9aeKk43ryz/uiz+U4SZ9UjVagLMLYKLn/
kkzS9ak2epyMVQaXqgF70Bk0kUbJ5ZNqthjwt2M+2kY4D2SwC8DkOVAZY19DL/
seWEu4yUVIZBm6uZ3yNA0GLisoxqtUCpkxToScx/
YobFzU6QcXWMi556BM0bTtBEPPJcny4yDGZdJVULXL0HB5gJ8stFKt+Sykhn8cYzeF43
OJiyZYgI2sV3lrRQZ2WpNPyk3YMomqamgsH2BetRB9vw98nqP3OFve7mhLs6aRSxNqzX
P7wJ2ncYR0Oxj7FImsz8ptJhieAtG1kMjmZvtx353IS6/
I0aglU4WZPNWoJrgjt4UdvvojxJfU0fL7/qEwgJ52mrMivafvcVDQ/
iuMhPRXTOUrbcctJaCpmSCNTKxG7ovzHntu48dpX0YTNC68uIdCt4huz6b3aixkwqHpJ
oucdbI9CocbirYu3cbC6nOFckAN9FUQiGZaXD7FtwwoqPGvwb+
+jfEdhup6eZozUMTkeIUeZyaIk0xjxMhzdjXHSU5a7ZVeMLbQTiclq/
QPvXkWuUrD4B91uFp6iEH/
9KkbWCswKC39ZnDWhXnThehczlQqhpjrEeIKKrSEueegTrp7OAWbL8USQto2dTLoFAA2
D3ne2kVhboEZNQzGBOqIIWvVRxm/
NkGooo2mfdpS3yUmnKds5d7o6W37aSBKfV8Xxanh0i7jt49c71+Dd7SU4KvGOOZMZk8J
rMPzO1WRqBFWvmFg+hUyNILXYIlqf4MbYDpZ4BjGlypAdwq/
EsHXXk+4kElTsOEt98QiO6kgcJVOMNytuJYtMDpHOggy53tBi1XxhGPS9eRGZOknt0xa
FsGB1dT+XRDpY6+vElCqdwXLG7CY34QAm69Od1fHmyL5o26AIFF1HenQcQyXUbeMdE8R
b/
egpYCLpniJgWYzcuZ5cpaDpW8Wz84TD8BWVjFxsg8dBG9JZclBzvXbFornSdo6eM84iR
/
foMAUMw93aCwfwjOWIWQ7pq0DXbNSEihjXyOFl0UU9hCqGmIg2I0wHIgH6rgySviDHFW
0HWBvuZomWwl9MsEq0QujSpfj2jVCxLcglD39iTuR4Vg2pVRVDBC7sx7il+6g9V1vKSd
/
6oaJ3U89WOxwE6rhuSNtBTsQRqTRtdTH3vK4nYsi+QQgGSF6TYe9V30UVCqa0GbHBr5g
s8gyTrRA4dZWI9v3oj77C0kdmmIxO0dBYVTGEb90AvlsOTgkUVJCiyBGm86KoPB7dzUA
pFND6RlGCfgp1YUbWKdx+6/O8N/Y8DarOU5k6dGGxyDPC/6z4b2pXu0Gwfz20js3/
HUPm8+6RDlNirpRnt7H0mSMyLk4xC2OS380dR/
1tqgwPv2eGCX9KMPihyrU1t3fxyvJ7ycvDhVZbNR+tWoa913+Lv16zjle/
GjlsSE09t+2ZzSx9ZoZ3HvneueI4+YvtBq16dPJRFWVRmsqf+Ag9uJP2C6sp96ap+y3o
aRPFdLj8lh18tOoJ/ujV30NYEturUN84zD9VbwXg5XyB98c/
SEtslDdXvsL2K2voro6ybLeG96mdLH0oOycyPBmOTjIJjoNvuB7LK5AKaPEs1uAQWk21
u0J2JHhUhN9H3TsP8ocNj/
CFe9+GiCcQQjD4hjx7rv02qlCIO1meyJbjFJ3hyWbwrygj9MQo3kf+//bOO0ySstr/
n7eqOk9P9P9+S8M7Ozs5FlEzkqQcQfCqiooCAIuyACoghXvCiKmMAISBRQL+o1IFFyhg0s
m/Pu5Jx7ZrqnY3XV+/
ujetJmdmeWhdvf55lnd3q6q+rb7zlvP81Od81zP9BGqCg4HZec38O
z05xkyVXRp4BAKWYqLXFVh6ymP8v2Z81l7tw9isfHXZTe2uof7TiQ/
s7EVh55EuHPJcMbpjnvJftNB7p9WYhx3RIqfaemweTwUf6WBn5U/
wQXyBoQB0WKDC49fMWKnulTpMqK4FR27MEi6UtJOQ2GUtzcw/
c096CxOYhsmW1qtshvFhVZMk6bCYD/
GwACaKBmphSRSEiJlX6jnlinPcE3NdcT9cF7uWo52tlBlywAM6jKaWZ+YZw3OMF5fcdl
6pi+dhDFjX3ZqGkgdcFrVu5Mulcz1XRjNrTjPWmC9J55KyDElWRe1ckP5S9z1+P9DBgY
A6PtYnNrTrSKxv+ibxduPLUCI1ORFdSBMiRkxrTHjHWM0w3uiOI4dF3fHUViiysLtxPR
5iBVl4N7Rg7K5F2XJVLyOOKEhBUcAnAGTI89o44vZ73Jd/
rWYmhvDqZB1Zgcb5z4xcv24dIz0aVkLemhR85i+3cD92iamPxcfDec5iHHjkE6ktrXlc
/
QdOWA2jnvdsbGZ82/+zqiSsxBoJcU03eUnXpvJ1P9aAYDi8bD99zORpmDmN7ePZHEN3V
/CzdlXUNBr1ZDCbiP/cSdHbriGE85bz5lZmzjO2caK6FT+3bGAmZ/
djvY5k75YLvWryqz6HGOgzJ9taWu9WkjxncveP8c7s0GOcQrTwLmxxeLYGEtlP+hoxUU
03Z1FrMHL9J/
usDoyl4vtd+ShKCYVd5vY+wXv9lRweuYWCtV+FjmbaEzm8MzAAp7afiSi3o27XeDuNvC
xbUSHrfbnR1G5wKrS3LCmlGm3rBnJhgJGNdLeJ8cRfox3+pE2bIhaVcwNA624iMa7skj
UZo58x4rHw7a7Z4EJM6/bYsmfJHRaX5nCkQMX8+xRD1CsOQCYv/IrqK/

7sYUkroCBKzwaMF93x3FULrRKQDSuKrXqHQ3LCI3V8juYNtwTx8YYwuGwJvU5fgJ3Gnj
s7fT1ZNF+ThL5qRn8puJ/WTk0lcHeMrSBGCKms/nuI/i6/0h8/TpJt0KwXCW8NZ/
qwKUcXd6EQ03idiTIdw5RqA1y6+z/
0DPdy0vHzmbDyiOZdvOa0d1ZQDlyxsHb6R44lkYk5knzUaM6Cb+Twm/
U0RLMIvFqLtESLy5mUHOzg6SuMvOb0ZEjvOBdM/le1hXkx3qs6ubBEKV/
t3Hk5mvQju1nTl4nVxW9zltDM3mhfTYnnrYJ7ydi1F6bR917MyxfH9OJTYgv7q0d98dW
Jcy8doulxSUE29qr+WNhPud6GnEr1sT/
grqzWLd+Kv4tVqCrO7rGupkQu7VVxe22JHQGg4gjph8Qx33xm9YcR/
F4LNmhqXlcseAdvGqMF3tmEzk7xPYT57NgWhONA9lk3jePUJnGwCzJFzOX0WN4mH5mHU
5NZ5GviYWuRlqTQ+SpDhQU/IrGT7s+xotvzyfzc/
1wmQdpeievP91Lf2PdREUrKaLzXjf9jVnM+mkTgbNnEJgtyD2qi+5AJtU/tINNQ/
e72LLFzreiX6TnZJ3cgiB5WpAB087WRIR2w8umcAmRPBVXjwMbUPvzhVTObxvXhoeK47
TWOGpWFjIaRSnIo/tuB4EBB/
nPqQydVoTuLabwnGbaBn1MWaEhozGkYZC8o4Dbsi7D37PVUpJwuyh6xsbc7degRcHRL8
kPt1s3s9nZdksVM49sRpqZ1K8uY+p3VwKMaEJO+LjITu2YmrQp+bl03uUg0Oml+tEEXa
cXE5xazLVznsMmktyRXYThVIhnKwR0D+ti5XQfDUpSxcgwmKYl+UVfNZ/
xridPlXgVO3GZZMA0+f70Z2mpyOGF449g08p5VN00pr+R8oA5HtKJlNoXxvbK6l1eN3p
68P2lx9I1qp4KXb1Ip50Lqtbyt+SikfcJm8bXFi4lZtpYo+UiiSOlxP9OkxXwOxRGcTh
AVfHuGMTV5eL1udVEK2xUFPWyKVxCTVs+XzlyJadkbMOQCtf2Xjj+YYQg6XVY2lrFF0w
cx65ufH/pRs3JhmnlyNZOsNu4oGot/
2SBNRtWVaTbyZVHvo1NGPxH+Ti2IUlbj5/1xVPIU0PoUqVNz2JZdyW2DR4K343j3NJql
TwYoxnknjbIDyufIiZtXNt/
oRXLYxgjk28jw87VlS+8b477bMOcbKiuHOH3hWlr+Zsxvg0vXbQMXaqsVv2p3UaBp10S
aPASWaRiSElExolv81H9RDNGZ/f4Iz0hsFeGuLH8BQCu6/vSaIFDxdIyPFB+
+8UxNwdZXYno6AZV5fyy9RhS4X/6j2FaYQdz/
e0c6+ikPp6PSJiISBwRCpPzThzpsBMv9ZPwqSRd4OpSoNPN8lgVTm8cnyeKgkTF5GRXB
27RzQnuOr4SuMxaNZLKQDaNUQ24SeAoFs2h++hMMjrt6G6Fbxe/
zWue2bxozyWRqSLMDL4973m6dB8rHWUjx1jepQ1kappVvNPpQESjZGzuxtWRQZMji5XV
Tq4rMggZTnqDHs4tXc/
J7h2Es+x8o+eiXZ5nMjmqOdkwfSqypWPEVv+aHFUs2dlWpSkRhoExZKM2VgCeRgDiUmd
9UymFSwX+V3ZYwrdj7reLrULq+MkSYz/Q/maf/
AryMWdWkFQV9EyVac5OAEwpOLa0iXxHiOtz3uHNaBm/mHohwSqTWQuamOLow0BwQeEq/
GqY2bZedAQhU2FLwo1T0Vlo16kJ5uHfLqg8upevF78OwDf6Ltpl12JS2zArC6ZXQbc1Z
pxfvoG/JxcCMFSi4Dmyj/tm/
pU3wzN4svQMEJDI1LANKHR0ZlFW2sesrE68Sowew8uA4ead4HRW95YiNZCqpQ/
qrAzx3YrnPhCOWlEhyRllaI1dSLeTy6cuZdlgFRv9RxAtlMTzk9xb+TQvhebynlYy8nn
7y2txKAKZqhAuMjxkbh3E22hD6R9C6JbkFIoCiomvYoCbplh2enX/
RdYu3Jjdmkn3xepKK7bSbuNz5et5RjsCRXcTKRDYZgxysnsHQelA2iVJYZJ0CfriHrZH
ClFy4wghKcwK4VCTbAsX4lB0ymwBTnN10m4ItiWK8SpRZjo6qC7t5BtdF40rL4GUB8zx
sCrI2bJ4Fr+98gFuuWUx/
ifWsfK8aqpiPeOqyRpSQRWmtVUrTWQ8SbKrx5o1Gwa4nJgZTgbm+AkXKVT/
op+O3Cpu+i8/
LVsKKXnd5N27FvLe0GxEXGdapIskWMHfqWuoK7fw10+exIxQzQRIGI9H8xUz+d2SB/
j+9xaT+fgqVp47jXIZHIm3EdE4plTwayGClU4yOgyyHlF4+IwzeKDkZIQiUVudlL6aoL
ylG7p6MIbCVtFOu82SCsjKJO8uN7fYr8TV0E9lbBDpdjNckRlAW719Uji2XD6T315p8f
P+8z1WfqaKqng3I6fQO8vwpLZyB2ZA2bwOFCQNSYPVsSk4+gRyMLhrtp6UVF7Tw+/c/
w+Aqkg3SSlHg/ZVFW3Vjklrw5avzeDuK+/n5u8vwf/
MZl696kS6F7mZc8EO3FqCcNKBAWSoMSLFTrxRHdnSjpKbA+qoZENmo4ktItEiBuX/
6CFRkkXztW7qXTm87phNnb2XDj2Lvzx6Jv66JIrfZ5XniEaRUqC+t3XyOH7Cx1+W/
IbPP/5NfDWCG9d9nmjQSWYMwkUKwUo7xbZ+AskMZLYPxe1CqgrYNExNIZHnsTS/
crxIzZoATv1DE0PzS+B4OMZTT7JK4ZF/nMVTm88gc9sAVYOB8b6YTE4ux8tncveV9/
O97y0ZsdVp8a492qpQBNhsKK4kpXarFMWgmaBGd2Gvc5G1vAVjcKfCmruzVbASW8JWeQ
Rt9Xb+evbJzAhOsC9+dRq/v+pernr0anI2G9z8nwuxl4b5XPU6NgyWsK0/
n0uzljHX0Y7j/3VzVl4rX85ZNpIZPFUbJCQ1tus55KkhYlLjtv/6GlKBn/

3iAVyaTqRQ0PeTSn7VnA1Jg6pwn/X9jTmunNQ2HB4zvr8Y/
5MbWPa52ZQ6TIzSPEwNIjE7phQsdDXwm0tAxlREAvDFcbh1Cj1BCh1BnMLg3raPs3HrF
Kr/HCerO0hGYcTaTfZ4KP9OmF9p54FpUhXtw0gd+SIUZFKfVI5Nl0zl3ivv5ds/
+Tr5b3Zy3wPnEi2QyJMiSCmwKSZOkcQmDGvC5HRYMcTDQtMZHvBnEp6ahbMzgto9gNnV
A3Y7xsxy1MEo9AYo+p7kZ54vA1Ae0RF+H8Tj1m6zKdFWbZtUX/
ztlQ9w821LyH2uljevOAavTcW0g7tbEtyRSfe8DAC0oIppk5jeJBtrS9koS1FCGobHIJ
mpMjWjl2pXN//67llIFdp/8iL/
bF1I97uF5K82cLeGUSIJqoJ9o76emmMcaDt+YBMp86T5DFS7Rl8QEC0w+U3LJ3AMGkjD
JNnYjFpVQeCy48ld3Y+sbeaPK0/
E4YvhviCDnM0xlLfXWmfHMhXArSdRQlHCxdlEFkbpivqRiiA26MXRp+DqCKM0d2EODFp
BdsPBhNJEmgrD1WKTDU0o82YR+PR08t9oh/
qJ4RjPkdxefw7OfiuTItnchlpdSeenSsldH8HW0MUDq08hNy9I7zEm/k0qhTUDZG/
OJtrlxHCBq0vibAwQnplH5MQ88p9vwOjptVa5DjtmhhM1lkQbMpFNrUiwnEmI1DattDI
gD5LjLvyAWI7kR7WfwdmnW7smTS1oU3dqw/
dOwO5N4P2SB199AsemFjJaoMlbSFNlFhtjpdy3+lQcGdD9hTkUPFtPsrNr3H1Gfh/
usEdWiNYO1+7aMNnQ9P4I7oXjD+s+gzMwWgHXsMOxWda2tSkVftd7MttDBXQeL0g6M8l
udCDjlgyOFjVRYyZazKBvlpNooUpJIgfdoyFEFEVIHIpOTNrp1TPw1yVxN4fBZrNKZqS
qUE+anQKxfJOft5+NVCCaK5BS4PTGCc4D4grCFLwWnM2g7qLlnFwcfRJ3r4GnMYSIxDF
tXlTTQAnF6Dsmj1CFwN3uIVIoCEs7tfEC1gTK8LRJPG1Rkj4nmpTQroyUAUGxslonjWO
O5Ae15+63rXpbkrh39ODY4eI39tM59dga1sdL+dGGczCcko5PlVL0pE6yo3PcffZoq6l
Ch2YsjlnfOPG+mCu5te5cXF0SR7+Of6uDftXNkfOa2RYqYCjmYE28jDw1yGlFO/
BpUWoShdzZtYBAzM1XprzL9kghz+yYyyVz3uVzmWvon66iGLAtXkxTfxbOXnC2h5AtHQ
hVwdR3SoaASffF2+vPwRkwrNCFlnbkEdNo/
qSXWIGBmlRZHSvHr0Y4prqB+oEcevu8aG0OpO6gzeujM5zJEw1H4tAMsooHCZdm4ZGgB
WOIWBwUgdnZnTraTe0Km3Ikfg6hjGS8TYadxrMlP6g9D+egNVZ5ukwSWQqLKhoZ0h3ED
I2N8VJ69Qxazy8ja0chnrUtVnC9ECSripCKwNEfZ3Cml2iuj5zN+aixJLrXhl0RaHoSg
mG0gRBoKhgm0jRG1DGEIjDj8YMeM/bEMZYr+WnD/
8PVZ0A8jtbcTaK6mNbTXdgHIaNZ8qumsyj1DOCYFiQSdKL023B2K2hR0DNADigEu3JoP
CXIOf51DEzTUGOS57vn0NKQR34tuJvDqJ19GIF+y152woH2qR/
YRKruCpX6T9w37rXpb34V8won7t4dI4ppPacUsuLHv2fufddQdvs2pi9+j+Tpi3js0Ts
54eXrmf526o0p5zUjEcyGJobm5lLzsYdZfYLBm+GZ3PvO6eR0SrS2PusYzDBQCguQ8YQ
V3D7csY1B43lZbFhyNyfdfM0BGczuOE596XKc5/
UiEx3IlPZc90l5vHfL75n96Deo+lWcmd+qJ3xSNSvu/TXH+66h8PF+/
P9oxC9NlMopiGicZGsbLbdks/S0X/Pl1utxLA1ZMRdeD/
EcJ1IFRZc47XarAxiua6OqYIymeh4Mx93xq3z+CiugkMaRAaP71CJW3P575t5/DWU/
3sb0xavRT1/IXx65kzPeu5Kyn+VT+L/
bKPJn8vpJs3hi2zyqL11LzaMLeflrv+PSuutQU5IIO2O0SvvwJGq8TtRYfv4D6Lx3y/
GFK3B+qhVEO8LnJTDHzdDsBDdm19FrhKlP2rn0D99E2uC9y3/JsSVfJ2d5FmZnN2Z/
P06Py2rDhiYGLlvE2jPuYX7hN9EGVJw2g0xbjDJbgIRUCSadeDd2I3sDkJs9shM7FpNh
p3PfvYjAjWVoZwvis6KU+UIcm9fILwrW8edgLi/
0zeWpNQuwZSRYc92v+XHPcfxz6bFM/
5MGNU2YM3KxhUyM7bX03Oyj5syHaE5GCJh2ahIFPNM2l74VhZTUxVAiCWouziKjxUPhj
nprFZ0KgJWpTN2Gz2ax8Yq7OfF7E8fx/drquRsvw7wvn8o/
1CM9Ltb8p4zf1pzOlC9soubRhTx90e+4dNt1qJ1du7dVbTSRYlg/
bWzSzUT74tSXLsd5bg8uTwShqRRsMTHt0/jc5/p51REmGrXzl/Zjmetv5/
b81fwtVMCDjadg/2UW3s1t/PLXZyLq3VTesoJH/ngCN5+xhfsW30tNvJAX+
+YQ2eanasUgSv8Q0u2CbB9KJGZlZO0mO3hSfPHFy3Gc04nUUwtGu42eRV42XnUPS1pO4
fUd03ms9Vjm+Dv5n4qX+XOwhN8bp5L1R4FjcwtbcyvwNGuU/
GI5NY8uZNMZ93ESF9OzOZvq+wNWHSZTjmixKS6nlXEcTcUqilS2m5TIuDEpvlj54uU4/
187am4MmeHG2atj2O38qfw1OowIrUkX3284D5ems/w7v+XE1Zeg/
bYY+6YW0BN0nOTB3Snx/
3k5XVctYsMZv2fhO0uQjR58NeDIVHG6bTi6hhCRGKbHhYjFkf0D4xKmLJUBOeKLEzouv
vw17Of3Y4t1YhgGipT0zXGyafE9TP/71Ux/
ZID+UCkNR5Tx3ld+zVfrzqNxTRXFL3Qh2zoZ/PRcnIEktpdWsemPi/

j4tBhfvfwF3uibztblleRvgZxlnVb4TIYbegO79KNj8X7b8QObSJU+oTK77upxr+VvMa
C3ETMcHakymrt6gLn3XUPJG6NVgp213Zxx301W2uIYraxZP2siPqOY+ksFF8xbTXMywi
uhRbzUOYuc91TCZYKaX+ZRfm8ByjvrMPsHSBw3k4aLp1D6hIrrqZXjn/
GNKPOMaylfs1NF6YPgWLohadV5Sg320lTIWz1oTRSXxqziaaaJZ3sfH7v/
Roq3GshwmJbvHEW02GDm77qJzCmi6fZClF4bZ9x3ExW17RjJJERj9B2VhfKFHgZX5ONt
kjiL8gnPyqbt8zqFTzvIfHaDZUyGJT9Q+tqBc9wdvynrdi0jkbu6f3wbSomrtofTH7iJ
rBoTtbWBhqtnkZw7xJUZO3DPTvCnPx6H0mPjrPtuoqK2eeR4N3L+sbSdm2Tag0nEMqvc
g3nSfOquEJQ8acP95Mpx9x/
L70BKre6OY9mGlP4WBjISxVefwBa2MbvhatSFA3y8rIZEtoktqHDMX24gd71EBhpp/
uZ8oiWWvuJQdQGtt+YiYyoL//
ktMtoUTBskEhoeLcHRzhb6TAcdrizqi2cQXVhA7xciZP+7EN+T6zDj8RGeB9OGe+KY0S
xREiGK39GJbXXQcnYWTS25PNN4Ask5QxxV1kJWQRAhJDe0f5yXVs2l/
DmT+s9moF8xh+zVKoGZbsKLFzG7op1fBmZwVsYmbMIkaLo4s2gbrWd3sXXTEdi3Bpjyc
gaDlTZ2PHwkJU/
Z8Dy9erQ8iDQpey3KPP1ayldPHMcp64a1xUbtZW+26msw8Wzron7xVOQRIWbaO7h06nJ
+9ccz989WkzrmKfOpu1xQ8oQN9xPvjn/GCfbFkk2WZiHhMFIIhE0jf9UQc++7BlsI/
BFJU14FdZ5y/lFwFM5WO74ak86zBMYXy/Css5F0Q82jC/
nk7M2siMO9HaexqasIscyHLDRpu0WS81AR7uW10BMgctRUmn6UT+mTKq6n3js0vqgnRw
ZFmUyStyrI3PusgGpfDLozS2jOK8J/WoS/
bjiawufsNH9SkLxoCq4mjUSmpObRhRQUDHBJw6cIb8jG2wJGbibdR2cSPCVK+cMq9pU7
wKYRO346TReZlDxpw/
PMamTSHLHV0jeizDMnzhelgNJNBsKmWVmRQLCyAGevsCb7H2vm5orn+ETBFkyp8Hwkl4
G2TIrrO2i4sprYtDgyoTNUDT2nLcKfPcTntn8ep1NnqDhOOOqkr0CSOyWE+wE/
nhUBRCRKZFE5Lbf6KXrKjvfZ9eOSXEZ8cQI5lmw0QNfH3SdvjWWrZat1lIEQWZsUnH0e
jjG+TUYTFK4NUf+VfOKF2WRuURioshO7aBHlJb1c03YSL62Zi7NTI2+bycA0hcFzMsj9
uxvfmlSs4EnzqbtcmZCx/
wObSLmeWknZU7u+bowN4hMCuaWOigYnZjQ2EsCZbGqh9GctI+8Z1iB67A9nEpjpYOuZd
9GejNOUzGRZYCpNLbnM2DBEYL6bmlMe5YTnv4H/
bYkZDjNUYuet0+7krO03UbrT8yhvrqXsTQ7I6ffGcdw0Q4DcXEtFvXNUlBeQ9U1M+UWT
dQwnTTJP6eK/Kt/
k7785iXChzXrm+26i9OfLR87rpSkJToVt8/9FVeAyYkMu9HwvfbM0lp/
6G06tuZHM51OaScLagVPeXkvZ2wfGcU/82LkNt9ZT0di+SxtO+YWVMWKoKlknd/
Kv2X8mImGuvZuvn/YeH7/3Rkp/vnxcjFzvPJWaM+/jhNe+gX8ZICXBSidvfdxqQ/dT4/
WgDobf/
nCUiQSOpgDOOoOsZwdo+sYRrHGVYWQa2AYVpj9oSTQYwSEyT+ni+1Nf44+PfoqhIpWlp
/2ak//
xHab9LURwmpdwoULcUPBocapsGdiSQ+RqIeK5dgIzVF4/7j5O23QTPpsttcNoTBpHNS8
PWZSD690aXDY7PQum4e0QlP5hEw3fOoJEiUqhN0Qw7uTlbbPIXqfiem0tedcVc1v1U1z
XeCWR8iTvnX4XdweO4aWuWRzvqSFHWBJAp2Zs4+TcJMd5j8TsC2B7pQ/
14mN46+N3cda2m3AbxrhBeLJ80aq5NmovckstFY3u3duqqiLtdnJOEPxz9p8xgHMztvL
Zj2/ljPtuovRnyxh7WLBbWy138tbH7uSsrTfh3ilgeUJ9cYz4sBm1ymYoHg/
K5gYqtghEth8zw00yx4VhU5CagqulD2PLDsRXpvJf017kV899mb4jVNaecQ+r4hksi1S
zumkKSr2Lquf72H5FFuuOeYxjnv8GbsPAGAwSLtD23J9Oli+OgdSTiE21VNQ4oKQAPce
D1BTCxXaeqpxL5ion3n+8i/xqBbdWP8v3fvU1gtMkK0+/
iyUN57F601TyasAVMDAy7PQvSFJ76iMc/dY3KFxnAyEYrLSx9ON3csb2m3A/
JcftKk64nQphSYjZ7choFKlpxHIErm5J6R82s7VoJolylVM92xgw3Dw3OA97n4rsHyT7
JBv3zPgbl2+6mKqsPu4tf5ovbr+Q2vWlZE8P4MwJ0RuycfSRdfyt8mWOLfgG7nAYmUgQ
ya/inVN/w+nbb8L7LONKEE0KR1VFjg1uB9RtTVSkVJtMKaF/
AHe9yrQNGchwGGMwSNZtU/nZ9Ce4sudKkuUxtn/8D3yr/
QRe2DGb3JUqng4dV0uIzuP91Jz4JxYuvQbfKuseg1XOCRv7D6tgc+FwoLhSk6ZUqn740
ws47dZ3eOb+U8i7b/muH0ppZT1ceD59H7cTmm5wV/
9M2uJZNA7l0PG3CqbWJFBqW9FCM3f5eM6LdVzafB0Vdc3semI6SRhbr8I0CJ93FKfd+g
5PPnoqJX/cjDkU3iXAelgPTO2sIe/
FKJc2pZ45dZSluFyIkkJ0r2XsUwoCNE4vAOHCMSD5zHdvoGJjYKSS+qSpeMP4a0u5+zZ

MBb6LhbPoWeQl4x6Dz7tu4Es/
fIGnO48kflcR5Vs69yvgb6QNG1oOTRumMgSF3SpmZ7a0Ezp3AWf8dx1l8bV0RH206bmo
CSzRV58XLTODzFvdPOw6D1Xo5K6P8OWvfZOqYHhEsT3pgSNL25jvaQbg78F5vNA5h4RH
IX+dzpevuJ6pte0Y0eiEal/tDmYwiJJMWvIuqsL0Bzrp/
lghx77di97bwoa2YtQtGTj6objdIJEB7VctxHd/kttjX0MuAlebxmm/
uZH4sUN8pnojLXoOnSKJOUaGREhGVvN5L9ZzafN1lLd0IF2ukQnA5BId/z2GP7Not/
2N0DQCFywg9rkBbI/7+fzDN3DBD1/kybb5iF/
lUrGjjeR+1NjJeb521FZ3JwJ7gHV6dgeZqic0DDMSIfzZYzj9B+/
w5CNHUPLoJmytAttw3E8yibDb8X/
Xzt05XyRZILAPwoKnrgcThCHI3KGQ0WlAWydaxKpGrSasawtFWG3YtBd+kw3TIPz/
Uv3pH4+g5LEahN2Go8mGryYTEQ2gTCkh81Y3v/
R8mUxXElevwpl1N2LawauAFjcRSYmtbQB1qAAAJYlVbkdRKHylky903UD55q4JD7reBa
nFmtSTqJkZoGmU/62F7tNKOfbtXr7geBIThR7Dy/ZYMc/tmINz7gDxf/
uJPF7AN1uuJfBZk/7mLM783xsZKpPIwjh5niHcWgJ3tc7R/
kYAhIEVBgLkvdLMhd3foqKuG6lMsgCKtAS0hceNGbSShsxwhPD5R3H6D97h6QdPpeBBS
/
dPolv6iEKguN34b3Zwy7TFnPffy5njauVvoQI6Y15UzSCzKYGzrgcZGMA2lAWAbUhidv
UgDYPc5+u4tHFixv7DZyKV0tPbpRy9tLTGdieEpZWXEavKx1nThat/
iNgpZeDV2TZURM1gHp39Xkoadex9EWJHV4FiFa50DI43f6mIkYJtu7u+Y2MzdE8kWUZ5
Cmtrc5jr6EOlsusqphCdlodrUyvmpl5MwIzFUTvHZBcJBSXTS+9xBZgug7v7y9ENFcWj
I1UNKaxCighhdaxjOupJ5TjCZc9tGC1yE1hgoEVUbBGTDUOl1LXlMXNpHXJYOxBQ/
T4SC6ow7HKXNrQq81rlI4RNQyZGV09j+Rk9PQfPZSd5luEYHqnAFHsfK3orqWnLx92q4
eyRlvBrWQ5DZQ6yVnahdHQjKkpRBsKo67sQpUUk87xEcxRiuSZzfe0UaoMMmlEaonl0B
b1kYNmoVEBqKsJuHzma3ZnjhLWhYVjFXd0upJSYTW04B/
JRhUlLnx9R58Ffa+LoN3B2Rug5OpPQ/
Di5G8C5vQPHtAoUXeJr0Gmc6qK3PIMuuw+fGqHANsjGWBmvhDJxDI5Z8w23425kjA6Jn
cLubVUoCKeDSJHgqunvcP/
ST+MMwDuBaTQ15jHzzY3W0XoKwzpwu7NVlBRHZVjNwBzn67GqPBybWmB8vPpB8BlTsFV
KhGnxU3Rr8jO2MLBWWU5keh7u9S3Y61sJnzsHkQRPk2r1HxK8bUm0sEHsmGqkOtyfjh5
tDas4sJvBd8J9cY+cUxwTIEMhZCpJQ3T3IqdX0HdiMTnLOlA29aDMq8YWUnD1CCKFdmJ
ZAilAz1AIzs/H1OCXgRnYh6zg8pGQDAVMjxM1Jxuzv3/
kJGFS7HQ4ica0dr+SLe04gsU4FZ2lg9W0RXyWaG/
ciT7goDhnkG9XvMSticvw1PXjaM1Fiwj8NQkiRTYy/
BFK3IP4bRFK3QP06l5u7z0CR9AcqUM40o6qMvm+OHz9sVmypoEwrSx9YY6PeyVpokyrJ
DwjF1vY+t5PzNhBiTbAimgVQ7oDPa6hhaxY4LG26u4zMBM6SHN03JiAsX+fEykhRBnwZ
6AQa6frQSnl74QQ2cDfgQqgEfiClLJ/
37fcw31UFZlMWgXqFCu7DEXD89w61r7iIz+2epdxuOHiMl5dcgdn3HcTBasTJIp0nC6d
NV2lDG3LwtsIrpY+hqb7ue+3v+OcV69l3al+3JE1I9fqO6uKe6/+OR+7IBNkNwJBCZVM
EdXUfj6bondv5Y1oJklCCCGyDobjaNaOMsJZyfCQ+WYtq84qpTi4DmOMYjxAw1dKeXXJ
HXz6thvJ/1cMYzA4vvZF6jrRI8v4449/xZfWXc7j3/
kEbecKPPlhsrfYaKsKUVTzC95o9WPT4xY/
pqHLBMvEf8ivqyOgTEUe8GbtTvx2WlF7nlm92zYUqkr3Io2N5/
ya9rMNavQcrlt+IRnrnJihoVTsg/
WJxIIqfvfo7znnlWtZe4oPd3RNSoxU0nPWVO77+s859UuZKHofmOZu29Bj7YTsVKr3/
fFTXC5LuyqpW/Fsioqak41vW5AHf3Y+WRuDzNi6BWmYKC4nlBVRf76D5Rf8ik/
ceSMlz5iYdU3WQOp2EanKZmCaDePUQT5W3MR/
5ayl10zwbiyLDX3FhLs95AyadB+l8dezf85JX/
KiJvpBGpTIismxU7AmpZpmiZoCSmYGvqVNLPvsLKYOtI8kbIC1WxM+fwFrz7iH4+tuoD
SeT9GLnVZn7LLj25bFG8osQnMdHJvVwHWZ25j92hJmfLuVjMgmpE1D6kl6PlFpteMXMl
GGrIF2Qn1xpyNnYPzvUu7WVhW7DYryiRSbXOVr4oLr7mRd3M83/
nUFBZskMpkcN7FNLJq2R1vtOCkbT9utLAv6sSXjlMgxvsiz5NfWEWACfHEPu1rD/
Apiq3aRsGm4qJhXl9zBud+/kdz/7MCwW9+RLShRDEseyLuug8isQu65/
y7Oee1a1p2Wg9fYikydJIxrQ2PPbXjQvrgXjOVo6onRCvOKSvOnfby5+E7O/
vF3KHgigrajBeHNQC/OAglSCPQMCJULnrv4Ts5efjWvX3IM/

lgfIjcbAv10f7yQh674BSdekoESHUIxdUrklMkdMwAjtbAUqkrmW/W8ftmxqF0DyEA/
CpBdWsTglzTENMknXGGuOSYJ5FL1SBvS4yI4y0+yPMYV1UtZ5GwkW41Rrtk54s3FTL9l
AF+0CenNQA6F6T6zjAev/AUnf8GLEu4FKSdxXFSsyU1KXBmsPiXj+fWseSuP/
KE1yLE7x4pK45cKeWvJnayJ+wkYGfjVCDGpMdfZwsNDJ2BvcKL2DxCdUcDdD949Yque6
AZrUZqU9H5iKk/efmfqWH68+PTw3OL8730H/
mffFPZnRyoJ3CClXCOE8AKrhRAvA5cCr0opfy6E+C7wXeC/
9uN6u4U0JcKmoboddF8wh1iuYMoftllpisPq1TY7XUuOAgkFD60id1OSE16+nuyAJFxg
Y9H0GuyKwdbeAlwzBhgqt2GutKHokgLVBAFGKET/
V48jmi8oe3Azvroo5y9dzPSKIXJb2klKnZW8SrYsIPrUK2w48jROmnsiby/
7KQbJg+JokVAQNo2+ixYSLRBMebwTAgMY/QMIux21IJ/
2L05DJC2O2VsNTnrrWsrakpCXQ9tVc/C0S/
yPrSRy3lH0zVGp+Ec3pk2Qp0pcdh2pCfxFQU4treXFRccgowr1n1rMovUVuJ5dxYrYf8
gmnw6ayHNVYd7wNfJ+9CotNB8UNashrQ68/6vHW9/
xQ5sxBgYtXT3GtKGAor9tQyRhi67iFSaFapDTpu/g1X5LtiJ8/
lEEZqpUPtZCcliTSViyJf1fPZ5ogdWG/
lqrDadVhSkOhIiHBlhpvrRLGzYt+zsBugsPhtvYwNb+rx5PpFBQ/
o82RH+IrC0KQjcQU6fQflYuUoHC5WG8DQonLf06hU1JpMtB87cWYg9B3uowXcfYsC3oJ
88TQVMMbEKlRvfxeOAo5mR34LLpEMole4vCedmXUTUjTulQnESgm3eTL06anQohwOGg+
7PTiOUIyp/
ogf6UGHEiYenqLV4EEgofXUfuepP5L11LyRYDoRvUXVKAMyAofjVAuFQyd04z61tKaRr
M5rqsbUjD0mYb0QN7YCu+
+hjnr7ycadVRinr7SUSCrJSvTBzHnScXY201ZUu7s1XDCQWrokibJGjGMKSkUAtRcXQr
TWYpfiBy/jH0zVKp/
HMzSbFnW81s1qk9+0qOKpqC87mxvtg48b6Ywj59caf+tKopjsj00ndqHEI2Ct8W9C4Qm
GUxsjdkIhVS/anEHArT+9VFRPME5Q9tt9pw6WKmTQ9THBwiPtg/
Ob64vxwdDrqusAoCFz68htxNBse+/
E2m1iYQbicNF5ehJiBre5Le+QJ1WgjnG17UOOSqKkiBOjBE63klxLIl0x42yWxIcO7SK
yifE2WKz4+5aTsrwk9Pji+OZD8q9F9yjMXxT7XISBS1tcdasHkzaPtCFboHfLUm3XoJM
zuvIOc9DU+XQf2lpagx8DaZZGTEWORsJCztKKbJdGGzSgdFY7R/dirxbKh4sBZ/
XYxzl17BtKowRT2T4IvjOJqpUAk7fRdZ/
UHZw9sw+gchMQiA4nbT+bX5ABQ9thlHP9zefSone3dQbe8iW0kQkSptyUxOKaljhWZgP
mmVWMgbttVQiL6LjyaaL5jygDX270kzMXdTkhNeup7quuh+Udjn4aeUskNKuSb1/
xCwFSgBzgX+lHrbn4Dz9uuOe4JpWOekvkyKL2ngl4sfhizf+Id1OfnMkjc5e/
E7CKcD15MrmX7FarytScLFgh+VPcMlBcsQQvLNGa/
z5gn3kvQ6UKMmekrPR6gqvotbuXPJw4jsLMTSdcy6YQe5b1uBz5qw4cZLnCj9Nas58nE
VsXw9Nkuc9OA4SpnS1bOTc3Ez9yy+n2R+plUvJx5H8bgxS/O4ZMkLnL34HRSXE+/
zG5lxQxueTR3oxT4eWHIP7ovbLaHf8xO8seROwtXZmJrABDz2BLEslTPKtvO9/
DeZckozanYm834dw/
v8RlQ03GQSJ0oP7UzZqjL9ilUUN9vR2Y3O2fvkB1gJABe3ceeV1nc8FsNteN7iNxDeDN
QYPB+cR30yGxXJPaVvUD6rEyEEHeclePXKO4hX5oEco1Woadb1d2rDgtfaMSMRNKnstg
2LKAcY/0Dvl6JuJQQIVcV3SSu/XvwQpteF2dOHXLUJNIWBI7P51pX/
4srLnyGRZafg3RDTfhDGu7IZ0+PgvsX3ctplKxiY4ab81CbeO/
rPlGX0kzA1TEzWRCt4vW46Z2dt5MbyF9D6o/
he3s6cHwcpeaMXo6cH1RCTZ6dg1axyOii5qIHfX34/
iUIvqCpGMIiUEiXTy+eWvMbZV7yDsNvIeGo1M67cgPf5jSgxnQcueoDK8+oQhsRWHeRP
U/
+NWuMmuDaHiKlb44OqknNRC7+84mFElg91xWaO+EEPxWsiCIcDTXFMqi8CoKgjtkRu9r
i3KB4Xn1nyJldf/hSRAgdSM2k3BF2GVcbg2Zn/
pvioDgDaztUtW63KB8boFY65vsjOwvleHUfc1oH7qVXj2rCHdqZsUSbOF0dIqPv0xZ37
U9vqGowsD4+c9EeOW7ADd1cC75w+lp1yD7FCNwis/
hSr+nvOhS385vKHID8HdeUWZn2nhuLVEYTHg021T5ovjmBP/
U0qjua8JW9Ydup04Hl6NTOWrMP21nqk28m9l93PJ7+0nHCByvTjG3n7uPsxbaCFQZcmp
qlAQqf8vHruvehBjBwvzpU1zPpOHeX/6UbZUIMS1SfXF1Pjhu/
iVn615CHI8SNjcZKdXQhVwSjL55IlL3Dhha+R/V4PFY/

UMePbreQ9thZPTYB7Ln6AMy5YiWEXFGUGOdoh6DMyaNRz0aVhBSrabZR9oZ7ffvUh8Hl
Rl21m9k31FC7tBdOcmHFR7HqENsJPVVEcDnIubrZ8MdtvTbBMS49VeDP43JLXrHEjy09
Gm8Ezbx3F1lgxeWqCKZqbbMVgwHRzbe4bPHHEH0n6HQhDprT8rMmo/yut/HLx6Lgx/
fJVOJ9ZucsjWb6wCrF8/
X5Re18xUkKICmAB8C5QIKXssL4H2SGEyH8/19odzGgMenoZ+P18/
jvraxT0bhm+8cjfX7/1RADcifVopSUkS3OIZmtIBZZGq6iJFtDf7qO5PAc8tcCono82V
2C8WETb62X8eMdlZHZt3OUZojJMiAF8ZJMgjkNYs1rFmnMeNEdpGBCN0vfYEVxdfBXF9
hjxI6cQzasiXKQQ98Mjj30SWxDyZw8h4gYYBuHyTGJZKpf/
zzW4OyWuMhW7M4khJVrUwBaIcebqK6jMCrDw+hV0xb18dvPFJP8nn/Km6Mi9I/
F+QvSn+MVwYCmmO4RrpPjaQUNKzDvzuc1/
Gb6dvmMzGuOVn5xM0inwlUSRGizrncqf1h2HCNjJrFPIbExiJlqR5qjjOTa3cN4tN6Id
AfoLxXS8WcptD+10fWkds+ypDR3CBXJi4gJlMknyzkJuzb6crK56RKYXpk4hWO1lqFjh
tnfPQdEkyskaatSLGvOSvS2JMOHHDZ8mxxnG+5U2vlb6DiEzwYa/
HoGiS+67toUyW4A7jnqcb7/0ZbLWKxS0bbfildTRdc9k26kZjoCuM3DPPG7IvpLC2hZk
OGwlhDgcSNPkhR+dimJI3OF1I+UghKogugJ877+XYB80cLfXEo+XoyPJ2WTgrR3i9LYb
UGZKzGeyaXljCrftsNrRCjIdfYaoEZxUjhZRY9RWO1K2lNoBkAmdZ+89hUiBIHG8SXll
D2Gp8cPGc9neWkDmchf+eh1pjK+J5Njcwnn/
bdlq8qWS3duqaRAxI3u00wnzxWF+WZfh69k87njTjESs/
lRiaQMOJ1EU5mE4NK7/7VUIA1yFJn6XtROnxk2c2zv4zA9vRJ1jtWHtxlJufHYxJX21o
7cdCiOiUSLJ3bfhRPjisE4iUiLvyOfH/
svI7Nww+gZp7Zi98sOTAfCE147EcgmbBr0DfPvOK5GKwB6V6IaKCfgak2TUDHJW+DuYs
yXqX022v13JzbVLyGurx0wkQJpWELhQiJpDk26n0jDQf1nID7IvJ7t9q8XB4bAWPIkk/
/vLs1B0SXZ/HWYsBrqO8HoRepJbvr8YRZf426N0hbwEzRi3PfxlvM0mtxcpGNVJ1D/
r1Lw6lR9ur8DXsdlSiLBZsmKmKSeG415sWiZ1zIhJ9NczuTX7a+RGmlB9mZaYdiq295+
PnkYiE5xnw8B8ncXHvUWf7uH3fSfxBf9K3gwfye/
XnspVC97iK5nrQYJjUwvnfy819r9QMDr2j7WTCcB+G7IQIgN4HLheShkUe5pd7vq5JcA
SACfuvb/
ZNDDjJv63GsBhx4jGRoOypYlM6rifXYNwuRBTp5DI8RAtdBD3CwyXpCZawNZgIfY+lfc
C5Ux3dqIkjBE9n/
CNJ3BD+Uv8eMdlZPxjxS5RCEmZZAPLmcF8NGHbbXD0QXNMDfa5qwfwNbgwVUHCrxEuVI
jmSZIZJvlrJdqQwcB0D2pCokVNQmUaSTf46iS2iEmi1I+ixGlKutC9KmpEY6jRxYA7yp
klm7h56/kM1GQz470eCAyAKUkaCTbIZZPHb0wnbX9pNXZSaatj6Sd1Mp/
fjJKXQ98JhegZks6QF0e9k4xmSf7LTZj9A5imAYM2Xo5UWG3Y3UPWXwOEbziGGyte5La
HLsP7j3d3uf4hacMUHK+sxaGqlpaV34fhdWDaLL9QOxwYGSa2qiF0XSUW0dCidrSwJBD
0kuMMc/
WUN5jr6CAiJRntBrawyRu9Mzgqq4mj3A1kr1MoeL4ZIzgEikAYJlLKQ8PRNDBjBr63Gv
Db7ZgDg2AYjPh9MknG02sRNg1RUQoDoZEAYhkawvuP96zLKAJj0M6KWB7u7gRiax2F7Z
lEr5zGt8pf5rYd49txuH5c0tQPWTuOs9XhuMOUXFTe6hCD1RlEvjjIDH8XIdPJjo58nF
tdFD1ei9kXsCY9Y221qxv/Yz1Ebjx+j7ZqteEk+uLO/
ISCuVMBQplM4npyJYrTiaicggiFLZ1AjxPTrlL0ZgA9x01gpgOp23g7VpLyxV7y/
h0mUjiHb5W/
zH89cznFrwUgmopFkhKpJ9ATk8xxTBkX20ursLFrurrUE7ifWmWJ9laUQnAIo6fP+lss
RtHzrRh5PoJVGXQPZfBePAf7YBLZ2Eph3wCRgiquLnmdH9RUkvv0NsxEwmpvoYA0SRrx
Q2OnUuJ8ZT0uVcFI6NYujWq3dnMSSfJebrIKUhfloARCJNva0fw+0JP4/
rkGxeOConyCwQzWJLzkrddxrW0iy+el7pJ8ri55ndu2V4zaqTJaUueQcEyNi+7XNuGx2
5A2eypjWI4E2eduiDNUbKfnaJM501v5in819/
edQGMkh23uIlYOVmCvcfFS8WyKbQMjvuh7rHufY//
BQuzPykcIYQOeBV6UUv469dp24GOp3agi4A0p5Yy9XSdTZMtjxel7vo/
NblWOldKqJKso1r9GKtLeNFDz8hg6oZJv3vm/
PNx2ErXLy1Grh8jLHCKcsDFQm83UJ+OoYR0lqiMbWjBTAdyq34fITlWY3imo25Qm61hK
rlpMuToLqSdYJl9gEafiEC6Wy5cJM7jjYDkCVoByhgfhcRNeUAYS7EEdrTuIiMap/
foUbLOC/GH+n2lLZrEpWsr2oQJihsZcXzvN0WxWtk0hFrFjJlQKi/
sxTIXexmyy1isUvtIB0RjSNCEzAxFLoLe3szb5JjkUUC6mA4zjF5dR3uF5TGnsdYa8T3
7DmZdjs4V2gur3ETlhOt+764/8vfdYXls/

i4waG54Ok+yX6jCDQWQigZqfB14PsqPbKmKaTO7ahmMm9KZpsI6le+X4Nv+JSymdB8Ux
xVOoqZpcirCOpd1uSCYRTgdNv8/
hgqq1XJa1ki2JHN4Zms6qwBRiSRuLy9+mzNZHhTZEwLDRZ7r5R98xrO8tYXBFPlk7TLJ
WdlrHhdGoVUfGNC2Jhn1wnDA7VSwBbBQrm9bySTky0Rn5zk+az+KH/
83Nz13ItOtXjHw3Y2M75HFHEKpwkVkbRh2IQGAAYbNZuphj2zG1aDKlyTrzbXJEAeVUT
x7H4fo1O2s5gjVA2ez0XbyIvnmSl8/7JTV6Dq+HZvHEi8eTtRlyntqMEQqBlKh5eQivx
2qzRMKyX1/mbm3VlCbr5DuT74tjeA5zGvl9zP/NU+az+KF/
890XvsT0G9dhHDebWI4dLWwQmmIjcHKczPec5G6MYe8IIoZVElLZlbI3gBmNjVSil/
H4SH96SHxxP2CevIDFD/+b7z5/
IdXXvztSY0vY7Sg5WUSn5xPN0dAzBJ4OA3tIx9Y+iEhJiJldPZjRmDUh0zRwODCCg6yJ
vTr5vigEQrONK1Y5+reUpJKqop94BFff/
09ufP4iqr+5YmQyBFilE4ryCSzKIVhp+ZmjH4qfbWVY8WLEToc/
J01MYL18m2yZP6kchaZZP3a7lTmYiptCVZBDYaSUBC5cSN+Rkic+
+1vylCROoTBgmrQbbm7a8Xk6tucz5QUDR38ctT+CbGkfKaGyx7F/P8qNvCtfJSgDe/
XF/
cnaE8DDwNbhSVQKTwNfBX6e+ncfpdL2A0dW03l0JoVv9SEbW60VsMtpnZfmZKBn2Ig5V
QamaVTZejghp57W2T4WFbaS7wixcaCYflsWWn+UoSof0RyFgr6BkS/
OGBiEgcFhYiNfoJSSLazCg5dyZcaIseZRTAdNVDATnQQTwVGZP5vehT5yVw8guvtx9Ma
swUo3SOZ6MdxZ6KVx5uZ1Mc0Wo0Btp0Trp9zRS8hwUe3oxK1MYZ1WgtMfRhHQ2ZaFiKk
oUQXHoLR0rISVXq1qGmY8webku3jIHHGGnfl10ISGbS9Pvp/85s2id6GP/
Dc6SNY37voGIcBmB2BlpIqwYceWmSCZYUP3iNGBWyjoM0sYrHCS91KUZJe12h3XhqSyP
U2JNI3RNtwLR2BgQjgu8pP3dhdmyk4RAvQE5OWg52fiskewCQMFyFNDzHK1485NoEuVC
lsPNmHQaThQkXhEgndaphJt9ZLVJfF0xDGaWxkWdMY0kYaJNM19cpwoOxXzZtK7KJP8N
7sw6pqszhrGT5IVFXUowS9rPoGnZUy45RhZCcVuY3CKi8AsgbfBWhjJwjzEQIhkY8v4O
k7SRCLYYq60OKYmUZPFUZk3i55FPvLf7MSoaxzPX9MQdhtJt0A6k9gEOIWOW0mABCU5y
hFM9JmlDE51kvdKAiMlZ7SzrVpfjWSLfG+fdjpxvugn/
82OUW27sYOyaVj9hG6yLFQNAjhiGqFSBwmvwNkvMOwCGVOJFkp6NCfFgQgioaOXZKPEk
yihGMKXaS2AQ0OgJ8f3p4fCFxf696nfpwVju9qpKUHXIRbH3hdloCqTUAVktErUSBLpc
UI4BsEhhNOBatNGUvRlIs7mxPJD4ovKvFn0HOWj4LWd+lQhLJ1ZK68B22CMX9ScRUZzi
uOwbymqFZQe14llK0Sn6GRutuHqNZE2DfoHMfoCu0wopGmyRazCIzIpZ3I5MncG3Udnk
vdeEKW1GxmJIOx2SyM2ywcuO4NV4JgyRLkmcQoHujTwKBIMaG/
Mxduk4OoIMlTpJTbbQ/
5zIWsixa7jxkRjfyptnQhcDJwmhFiX+vkU1gTqTCFEDXBm6veDQtM5Pt79/
j10n5DDsG6e8GbQf3Q+tV90E7o+RPtFCeRJA7wWnsUx7jpeWfQHflv6EjfmLuXSkqXY8
yMog2G6vhjlyR/
cSXxmiXXx4do0QoyutlMYpI9Omumnh+X68yzXn6dXdlDODProYql8AcMK/
jxojo3n+Vn243toOz0L6XEh1mxF3d4CQtB+ioemr5l8cvYWTs2uoSlpwyZgkQMu9LZxm
W871bY+bMJgcMDNFyrW8vSRj1D1mMmsX7Th6hLYwqa1ynI6EDaNZHsHgZ5tdNJEP92sk
C+zQr68C78+ukbipSaCX8/JReODCxXV+t41G8LrQYsY/
Om5j7OmpZRTp9YSm5IgUiSsnQrVap+6ywX//
vGdxGcU774ApWKtKIVNY1D2jrTh3jgCHQfN8fysFMcCqy5QMomZ0JF6ksAx+dSf78ShJ
amL5PFurJiwtHO0s5klWWu4Pmc1xWqEmLTxSsjKTizXohTd7WDmD7eT2aijDcZR3G4Un
xfFm2Gl1yf1cXa6J44TZadNn/Gx/
Ef30Hd8gdVZp1L8hU0bLd9h02BLLVmfrqfo17sWyxU2DeH10vOZGK9+9U5MhwqDQ/
TP95OozEdxOsasfq36ZoNm9z5tdSJ9cflt99BzcqHVH6SeQagqwuVCyfSSdAGaZFsii5
i0Mc3ZhalhjVzDZVqAuitStlpduEtZgdEvRGFQBPbLTifGF7NYdvs99J5UNDJIClVFcT
mttlOsumu2zkFe++sxuNpVms/20XMUDMwxCRcpmHZwNdk44fRN/
Odbd6BnuyEWp2eem84TMmn7ZD4Dx5UQn1M2Ij21P3Y6Yb44zPHk4r2+z1y/
laxzain+1Xg7lVJaAvZbGxg8IcbrF96JFjVQapqJ57uJVWYTP6IMo7oUWV6McFuxQYGB
ejrMhkPii43n+Vn+o1Q77gHSMJBrtpB1Ti1Fv1q26xt0HRmOED4uwrKzfkPeuhi+ZzYi
PU5EhsfaCRqp52RNsgfpo1M2ETC7Dkl/s+LWe+g82Ydw2K14aUWQLPTTf3Q+rWdlM/
djNVw3+3Xi0kSXVoZzQkpa9BwK3lYofjuE3FpH2/k6T/
wwNfbv68RtgmIR9+tob6Kwr+098+QFtJ3qovzZAdhWj5KXi16aQ+88N/
1H6cyZ1kY0aSPPNcRVhW9QoA5RqEKPKYlJFacwuGTzV8n5WoiBUyoJzFSIlejYAhrVdz

VgBkOpLfaxq+c9H0GNxf5s7+0Xx1MX0Hayi7JXhlC3pxS67TaUTC9Di6YwWKkRmmoisx
McVdXEkZltHO+pwavE8IgkxarkuUgZP17/KfR2D/
aAxVEdUql8Oo6tbQCjvtkaxISwAg/3E/vDcX/
5lT8ZwNy8fdzfhN1uOWx5CbGSDNpPtqGXxTmiop2NW6egDaoYhXFc251MuXsjoU/
Mpm+OSqxEx96rMfUn63epdi00LSUqun81hl+R/
1otpTxqQjg+04+5cYf1HDYNxeUkOaeS0BQnwQqFWIHJsUdvp8Q1QImjn9mONgq1EGWqy
V9DM/
jlW2ejDahoEUGsREcb0Kh+pAcGgpiBAYTdBopiBX7vJ7+JttPyJwOYm7aNxg6ldgCR1o
R9OGZKGubIsZ+an8e2WypxdahUPFpP/
ykV9M+w7FSJqJS9YuBqCiFrGlIFYs1xdZgOOcenA5gbt48cVwnNhuLzQpaPbdfnUVzVw
3VTX2XA8NAcz+HvL52Eu00QnJvAU2un7LdrGPrUPHqPGGOrP91gDQZj2+19VC2fEF8c7
k+fDmBuqrEmrmMgNA19QRXhQgeBOQLDCYbTRNolwmkws7yDmo58fK+5SPgEuoeRvqbim
QRK3EAYJrECF1IFz8ubrf51PzlOqC8O2+neMGZhpzgcKIX5bLmpEGenxtTH2gkcV8jAN
MtOtUGNaX8ZQETi1lEmgGFi9g8gEzpST+yT36T54lg+e/uuFRWtII+t/12Os8MqzdF/
Yin901Mc+zVLwioYGqkL93760gnlmLLVKS+GULY3WYsUv4/
Y1DwCsxyEKiRXnv0Sp3m2Uq7pdBkKjcks7qz/
JO19PvyZEQbW51J129pxvmjr05h2x5ZR/
d7UUegeFzsHyPHwqWzOeC0mYbNjZnvRfXZMu0BxGhQ4Q7hUnULHINNsQZxCoAiFej2LP
iODU11NaIqJ1HW8T60l83k71a/
HsAmDrX+uQiQSEImM1q1QBNI8tDIGwxo+CIGRqiklEzrJrh48293YB324+hxE8pyskhW
0FfoIFTgpsg+QrQ7hdjWiSxUhoOgdiffFjVS/HiMpVRp/
ZB+J2cBMVSM+xNhFo2hM5yVSRR5Nh4pUBWpckIhqdA55UaIKUpX89eSHuKHgAsR9GhnP
rsP7kn2U350OiI6v6/
F+HGKisDNHoWkjVfltrX1k9btwd3kJldlZV1pCU0YWBe5cBn1uKh3d+N2NbBgqJWeVSt
57A1DfOsKx4XYbZqoGjnUzMelyMPvDcRjj4olMCWpKCyyRQCaxFimZGfz0E//
g3saPwZ9VfM9uxP+8yvTXI/
i0KMsePxplKIJhylRV4dFqypMuCbM3jmPitCw7taNmxZni7acpkUt3IpOmSDYiCUk3PP
rxR7i55HzEvXY8T68m42UX1a/HMKVC/
W8yEPH4+JCWQ8gNxvSnqVgw4XBY8abRmLUr5XHTP81JNF+gZ5iYbhPhTkJcRWgmv6r8F
3e4zqLm8dnkv9WL7OhmxhuW/9XcVYTZG8CMRFDPXETcr30wHA9E9y11ZG76M/jNmX/
lD20no7+QS/
ZrDWQ9G2XW60MAbPtdIWZfACMWQ3G7rc+8j4niRGGPHPfxHEIRyMwMfv6J/+W+po/
B3zT8z23B/
xyjHH+bh4xEhz+AUMxdwrAOBYZtVQJGagIo3U5MmyCRCUaOzjxnM1O1JDEpaUxm8UZwF
u2ri3AEBHdc9T98V3wWAPeTq6h42U3169YmQt3v8xCxODLJuJOoicRhNZEaC5nUUdq6c
XcFcG/
S2D6ljKIjBjklYxt+NULAHH307zx0OUUrovxmmhN3nwFGH5gSMxxhx1WzrFVmQ42VSjk
cXGoaViGyQ+wUowStbBMMMSrT0NOHPRIju9YkW9MwX83A8HpZmXkUWsRa/
f3ZriKSJuVhHaW1HiMcYdt11hGREtlsXVuIXYKCPzCkYmkUuxXzIRMJlLpWPO0uSoKF6
Jk24r4cqreFUIai3Lj0alz9OjLablUQ15NsvX4uAEpw4wfXXnuDUKwdmUgERZoQi+PoG
8DZ6MbdnYfpcNNlz+et3iksjRs8nG1HJCXZwShK7yDJcITNN81HSIkWHU3LPWzacGcML
0KGK5vDqOSJYSBb2nn4ivNwGibSGRxRrl95x1FIFbJb2pCDQWtVv4dq+B8IpESoAuF0W
LWyhiJU3u2g0zOVJ3JmYNgFhgOmrg6i9g7y45WX4RuMW9qghoEc9kUJysDmD2SSv1vIl
MTGcHvZNGRFMZFSL6EK0DNNUIezJhXK/iNwt+qc0/
ttlLjAlwVelx0zHGHDzQtIeFXUuRJ3swc21eBcWYNTVXdRZTgsMEZNQqSCmKUpMaMx1P
pWfv7Dr2A4BGI25ER05PZ+3njwWKSAwviOkcsMx9ocFnY6jL0tPIS16yKbWnnwys+hJS
XSMWRNBofCvP6HY5FCUJTcbsUtJhKHDTehCCvsJRrHtaoe44SZHD+jjoh08HYsl//
0z+PVmpl4VrmoXBpC6xrgBzuuwBvQLZ+T5ujYD9C9Y0RfdrL61EM+kVIL8q3zdECYEtv
qmvEr8BQUh4PY/ApswQSs2oKnpZznmmdTUBWk2tFJmTZAj+GhWc/G2Sext/
STN+hGRBOjmj2mYRVJJJWxOZxpNYxJMpz95uh0kjx2FmowgVy3BRmLWwNyLI6SmUF8Vi
H2viiu1a2YgyGknkAdXjXbNIyUsLOy2truHbdbMIlOsb/
8hMOBcfQs1LAOW+tGdaricVSXi1C5E+eAgW9HCKWhFSM4RGYobMUcpRwCKdHWWB2aeQh
3Zt4Px+Txc9BCcVi/
HZnQwTCtOi42G7EcFWfAwFM7gGhuxwiF8GRnIZxOZIbbWg1KE8caqwaPcYjacHcc7Wtq

MYLB3XI0jpmNFopjrrdq2AzX41F8XiLHTMXVNgSbawAwYzHU5RtR3G5Elm8kW9q/
oQ+UVBbO8HHJIea4T1uN6rCtERI6EtB2tKBl+QgXF2IfMnE2xFFaOkn29mHv6bWy1Ibb
zDRQ3kt9P4doEvW+
+tPjpqOFdNS12xGGRElK7EGBaQN1aph40IHWbcPdEkSpayVr42ykBmochotCOtc24fRl
EK3KQRjWAtCYxCDe98NxuA3H2unwca3qyyR2dBXOthDG1lprsI3GyNo4gJ7jZqjEjplh
R8vPJWt73OKW0EfreR1mdqoNJUY5Dl8jpfFo7wxhbLP6EzMex/
buNitpK8tnHZnZNHK2xCzZoqSVHHAoJlHvh+PgwgIcAR3ba604e2FDVzEV7qlk2cKoQm
IMaXibDbSOfoyOTrzxBFLXrV1uGDf2j8Mk8Tzkhz99Z1Xx2B9/x9/+dBe/
fPQ+kvOn7fZ9orSIq+79F/23xkCaFN27isLLevjti2fz8/
qzqdRUNsdLuHPrJ+ifI2m6oBilK4Bs7Rg1/p1rXaVqVUy20ewXRyEQU0pY/NC/
CdxunbebsRhmyJowJY6s4O4H72b7ErdV22X4TF5aZ9gyHk9l3QirNEA8PhIse1jwA5Sy
YhY/8gSB2xOYqWeU8TjSlMRml/
DoT35N07kg126zOmPTINnZZfEd005mJLJLuYpdsJ91zSaD45IHHydwWwJpGNazhkKYsR
jx6gIevf3XNH1WQm3jSHye0T9IsqMLY0cdZiiEUFWMwaD1HRzCVeHOHPV5VXvkuPjhf9
P3k9TkJxWMKlSV2MJKfn/vXWz7RsboB4S1y2pGIhid3Ug9iZLhwaxpwNiyA6MvsO/
2nCC8b1u9LWE9d38/Rlc3ZnCI2NQc/
njbr2j9pIm6YpOV4YTlryN+OJwBrCf2HT8zgbb6vvrTe/
7JwC0RzHgcY8sObK+speyRbRStMHjpuPtw+WNM/fcQatcAAHn/
2ETBk3XkrAmg9A+heL2Y/f0YdY3YX16LsWXHIbHX99uGfT/
daRIrJYn5ldz94N1svS4TSIUZqApKKIq9pZ+cd9qJ5TrpOrscx/Z2lKUbrCr+Y/
TfJhPvm+NPdt1ZSSy0NB63Xu+3FtsjweMmMhpDtneBpqHk5aC9uxX1rfWWnM5hxlEU5f
Pt2/9K5NuDIE0K71/FlMUd/OO1E3im7Uh+VvQ6/sIQjoGkNTF0ODD6+i1JmeHF9gSPB/
vCId+RGta3AUBXmNXZy273GYQgXw3htllbcorfB1mZ5K2GQGsxc5q+ga1Pw9MsyB8wcQ
xaWQkyoVuZVPsZvKrMm0XjeVmUvhFFeXPtoeOYerYRjjvB3h7knBe+Se67+zjT3V+O52
dR+vrEcDygNhwLaeJoD/KpF663+O1rp2kfxz975HcQMTcHaqdjYR/
muFIdFbYdi+HdDKHs8zlH7PS1KMrbh9BOYY/
tKA0DZ2vKTlfuVI9JmlbooZkqqDccyLoXTLSdwsRxHLbVfe40HaitHiAOyk6lgTkUxlM
f5NTnvo1/g4bW3oQcGrJ2x1P1/BQhkLGYFWQ93K/
uIa70cGxDGNOfrhw+3lOsneNozPLDhI672Y095MIcCu+1TzpsfHHn/qZjPEfr/
VbYAYZpvRaJjuxCWTHChyfHHHUIp2YtppUMDyLTS95qGGgrYn7zdfg22nC0dCGDQ9aYP
5bIfvb5E8nxkE+kxNJ1TF86+vsezVVKDASmTGUFFeUQLfGS/
Z+t+AYG2V0iqDGcSp0ck4m3jwKRvQv9bFhyN/
OMa60g8AnAfnM0zVGOOz2bsb2W6Vft7Sbji+qNYE8cF9/
NvOTEcDygNhz7XFJibK3Zb35Cs1mTYz2523bcI79UjZ8DmUwdqJ2OHUjHtuEuTzBcimN
4UNqfNhy207ffN53d4qDb0TQwtuzYtR3HFuSUJlI3kLqVQALqSLzCzphoO4VJ5DjuJjv
ZKjvFt4253mHhi2OeW8bjyA3bRvglx/xNqCqmnhwfYJ0qH7PX/vRwaMOdsEt/
Kg2kNEey1cx4HPr7UbCC8/
c2IT5sfHEn7MJxOKN22N+EGBFzVjweGCOxszMOK475OSQKvfif2UxmKDQy9o/
9vNA0a2Io3+e4OEEcP4C8rp0w1mjHQHZ084ObF6PdaQmJisZ2PGubrdXCnmAaJD6xkOT
LZQxcfDyK282OhxdhvlyM+Uop7TeesMtH8t9o56Sbr6H8ycCEUdpf7Mxxf5H45NG7cny
l9LDjOFH8tj84F/lcHubLxbR/5/hd3r9Hfub+7UoeDHbhuJ9Hq4mzjvpQtCEcQDum+O/
McfuDc5Ev5H+4Oe7UV+3WVoc5fuf4XWzhg/TF739vJ357s9PUbumHrT/
dheNeP2AFnUdPm/vR9MVUGAhS7mqnv5+JfMb/
oeAoWzuwbW5G7pSxPe4zhkH8jAUfWDt+8Fl7w7tFO8EMh8n4x4qR341gEHYTCLszhoo0
3pv9NLMrriZH07h00TJuzbPEjysbr9jl/cmGJvwNTROuvbM/2Jnj/uLDwnG/
+e10BLcLv4XLD0t+8NFvQ5hAjh/GdtzHUd2HhaMZDuP9+/tsQyk/
dHb6vjmaBuGCDxfHifDFry44PO0UduVohsMQ3ssGCnzgtnpIC3IKIXqAMNB7yG564Mhl
/HOWSynz9vWhjzrHDxk/+OhzTNvpHvBR5/gh5wcffY5pO03ho87xkE6kAIQQq/
ZVzfZwwME850ed44eFH3z0OabtdPI+eyiRttPJ+eyhRJrj5H32UOJAnvODj5FKI4000k
gjjTTS+JAiPZFKI4000kgjjTTSOEB8EBOpBz+Aex4IDuY5P+ocPyz84KPPMW2nk/
fZQ4m0nU7OZw8l0hwn77OHEu/
7OQ95jFQaaaSRRhpppJHGRwXpo7000kgjjTTSSCONA8Qhm0gJIT4phNguhKgVQnz3UN1
3XxBClAkhXhdCbBVCbBZCfDP1+g+g+FEG1CiHWpn0/
tx7XSHD8gTBTHw5UffPQ5pu00zXGn6xyW/OCjzzFtp++Po6VePsk/

gArUAVMBO7AemH0o7r0fz1YELEz93wvsAGYDPwS+k+b4f4fj4czv/
wLHtJ2mOX4Y+P1f4Ji20/3nKKU8ZDtSxwC1Usp6KWUC+F/
g3EN0771CStkhpVyT+n8I2AqUHMCl0hw/
QEwQx8OWH3z0Oabt9H3ho87xsOUHH32OaTt9fzhUE6kSoGXM760c4ANPJoQQFcAC4N3U
S9cIITYIIR4RQmTt4+NpjocJDoLjh4IffPQ5pu30/zzHDwU/
+OhzTNvpPjkesonUrqrEcFilCwohMoDHgeullEHgPqAKmA90AL/
a1yV281qa4yHGQXI87PnBR59j2k7THPkQ8IOPPse0ne4Xx0M2kWoFysb8Xgq0H6J77xN
CCBvWF/kXKeW/AaSUXVJKQ0ppAg9hbVHuDWmOHzAmgONhzQ8++hzTdprmmMJhzQ8+
+hzTdrrfHA/ZROo9oFoIUSmEsANfAp4+RPfeK4QQAngY2Cql/
PWY14vGvO18YNM+LpXm+AFigjgetvzgo88xbacjSHM8jPnBR59j2k5HsD8cD03WnrSi4
j+FFRVfB/z3obrvfjzXSVhbjRuAdamfTwH/A2xMvf40UJTm+NHneLjy+7/
AMW2naY4fBn7/
Fzim7fT9cUxXNk8jjTTSSCONNNI4QKQrm6eRhppJFGGmkcINITqTTSSCONNNJII40D
RHoilUYaaaSRRhpppHGASE+k0kgjjTTSSCONNA4Q6YlUGmmkkUYaaaSRxgEiPZFKI400
0kgjjTTSOECkJ1JppJFGGmmkkUYaB4j0RCqNNNNJII4000kjjAPH/
Aex2zyI9uOXHAAAAAElFTkSuQmCC\n",
"text/plain": [
"<Figure size 720x144 with 20 Axes>"
]
},
"metadata": {
"needs_background": "light"
},
"output_type": "display_data"
}
],
"source": [
"# Compare original images with their reconstructions\n",
"f, a = plt.subplots(2, 10, figsize=(10, 2))\n",
"for i in range(examples_to_show):\n",
"   a[0][i].imshow(np.reshape(mnist.test.images[i], (28, 28)))
\n",
"   a[1][i].imshow(np.reshape(encode_decode[i], (28, 28)))"
]
},
{
"cell_type": "markdown",
"metadata": {},
"source": [
"As you can see, the reconstructions were successful. It can be
seen that some noise were added to the image."
]
},
{
"cell_type": "markdown",
"metadata": {},
"source": [
"<hr>"
]
},
{
"cell_type": "markdown",
"metadata": {},
"source": [

```
    "## Want to learn more?\n",
    "\n",
    "Running deep learning programs usually needs a high performance
platform. __PowerAI__ speeds up deep learning and AI. Built on IBM’s
Power Systems, __PowerAI__ is a scalable software platform that
accelerates deep learning and AI with blazing performance for
individual users or enterprises. The __PowerAI__ platform supports
popular machine learning libraries and dependencies including
TensorFlow, Caffe, Torch, and Theano. You can use [PowerAI on IMB
Cloud](https://cocl.us/ML0120EN_PAI).\n",
    "\n",
    "Also, you can use __Watson Studio__ to run these notebooks
faster with bigger datasets.__Watson Studio__ is IBM’s leading cloud
solution for data scientists, built by data scientists. With Jupyter
notebooks, RStudio, Apache Spark and popular libraries pre-packaged
in the cloud, __Watson Studio__ enables data scientists to
collaborate on their projects without having to install anything.
Join the fast-growing community of __Watson Studio__ users today
with a free account at [Watson Studio](https://cocl.us/
ML0120EN_DSX).This is the end of this lesson. Thank you for reading
this notebook, and good luck on your studies."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "### Thanks for completing this lesson!"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Created by <a href=\"https://www.linkedin.com/in/
franciscomagioli\">Francisco Magioli</a>, <a href=\"https://
ca.linkedin.com/in/erich-natsubori-sato\">Erich Natsubori Sato</a>,
<a href=\"https://ca.linkedin.com/in/saeedaghabozorgi\">Saeed
Aghabozorgi</a>"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "### References:\n",
    "- https://en.wikipedia.org/wiki/Autoencoder\n",
    "- http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/
\n",
    "- http://www.slideshare.net/billlangjun/simple-introduction-to-
autoencoder\n",
    "- http://www.slideshare.net/danieljohnlewis/piotr-mirowski-
review-autoencoders-deep-learning-ciuuk14\n",
    "- https://cs.stanford.edu/~quocle/tutorial2.pdf\n",
```

```
    "- https://gist.github.com/hussius/1534135a419bb0b957b9\n",
    "- http://www.deeplearningbook.org/contents/
autoencoders.html\n",
    "- http://www.kdnuggets.com/2015/03/deep-learning-curse-
dimensionality-autoencoders.html/\n",
    "- https://www.youtube.com/watch?v=xTU79Zs4XKY\n",
    "- http://www-personal.umich.edu/~jizhu/jizhu/wuke/Stone-
AoS82.pdf"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<hr>\n",
    "\n",
    "Copyright &copy; 2018 [Cognitive Class](https://cocl.us/
DX0108EN_CC). This notebook and its source code are released under
the terms of the [MIT License](https://bigdatauniversity.com/mit-
license/)."
   ]
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python",
   "language": "python",
   "name": "conda-env-python-py"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.6.11"
  },
  "widgets": {
   "state": {},
   "version": "1.1.2"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 4
}
```