IBM-Project-47385-1660798812

PROJECT DOCUMENTATION REPORT

CRUDE-OIL PRICE PREDICTION

TECHNOLOGY: ARTIFICIAL INTELLIGENCE
Team ID: PNT2022TMID46457

TEAM MEMBERS

Team Size: 4

Team Leader: SIVASAKTHI A

Team member: PRIYADHARSHINI B

Team member: SWETHA S

Team member: AFREEN YUSUFA A

INDEX

1. **INTRODUCTION**
    1. Project Overview
    2. Purpose
2. **LITERATURE SURVEY**
    1. Existing solutions
    2. References
    3. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
    1. Empathy Map Canvas
    2. Ideation & Brainstorming
    3. Proposed Solution
    4. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
    1. Functional requirement
    2. Non-Functional requirements
5. **PROJECT DESIGN**
    1. Data Flow Diagrams
    2. Solution & Technical Architecture
    3. User Stories
6. **PROJECT PLANNING & SCHEDULING**
    1. Sprint Planning & Estimation
    2. Sprint Delivery Schedule
    3. Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
    1. Feature 1
    2. Feature 2
    3. Database Schema
8. **TESTING**
    1. Test Cases
    2. User Acceptance Testing
9. **RESULTS**
    1. Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

1 INTRODUCTION:

1.1 PROJECT OVERVIEW:

This document is provided as a report for the project **Crude Oil Price Prediction.**

Crude oil is amongst the most important resources in today's world, it is the chief fuel and its cost has a direct effect on the global habitat, our economy and oil exploration, exploitation and other activities. Prediction of oil prices has become the need of the hour, it is a boon to many large and small industries, individuals, the government. The evaporative nature of crude oil, its price prediction becomes extremely difficult and it is hard to be precise with the same. Several different factors that affect crude oil prices.

1.2 PURPOSE:

The purpose of this document is to provide a clear-cut view of the project undertaken and produce a neat and greater understanding of the project.

2 LITERATURE SURVEY:

2.1 EXISTING PROBLEM:

One of the most significant commodities in the world, crude oil is responsible for one-third of the world's energy use. It serves as the foundation for the majority of the items we use on a daily basis, ranging from plastics to transportation fuels. Since changes in the price of crude oil have a significant impact on national economies around the world, price forecasting can help reduce the risks brought on by oil price volatility. For a variety of stakeholders, including governments, public and private organisations, policymakers, and investors, price projections are crucial.

2.2 REFERENCES:

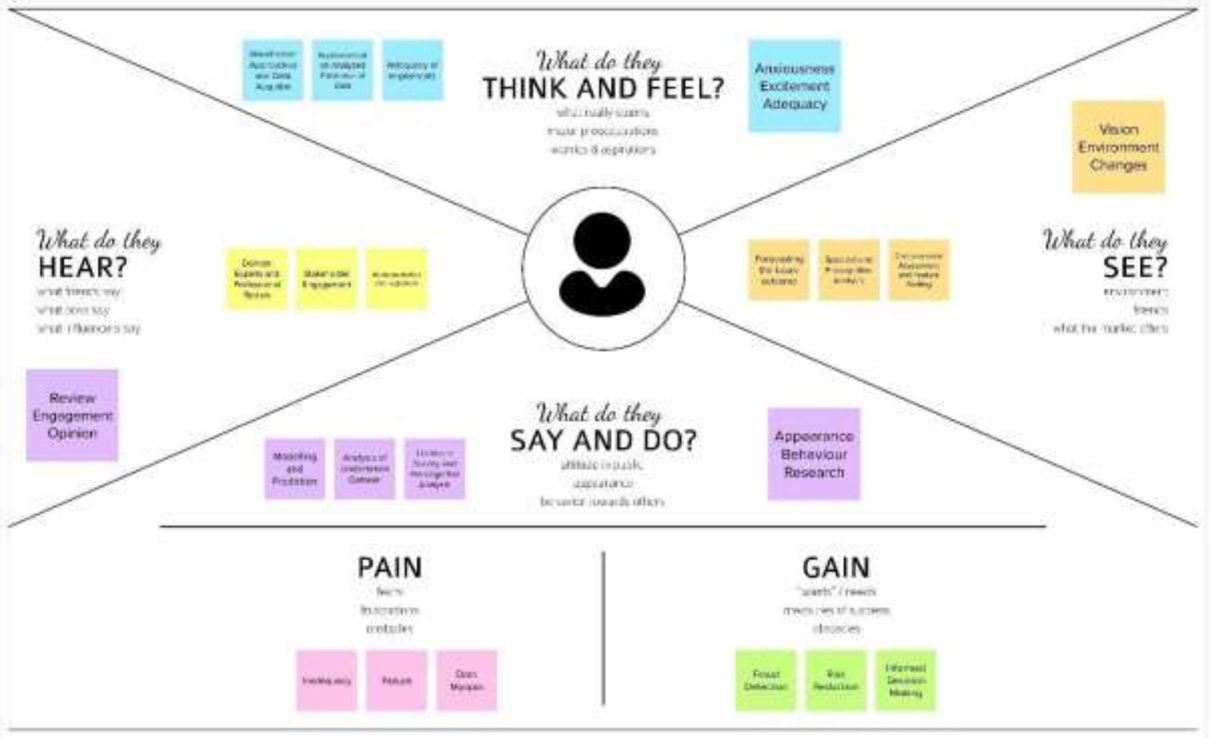
https://drive.google.com/drive/folders/1yq9UqoGpyAQFKR6ARNFwpVMofYtOHdCm?usp=sharing

2.3 PROBELM STATEMENT DEFINITION:

It is required to forecast CRUDE OIL PRICE in international market. The input and output should also be shown as charts and/or dashboards in various formats (like day, week, work-week, month, quarter, year, etc.). The models should be built with comprehensive explanation of data (using EDA), trend analysis, assumptions, data cleaning and validation, data augmentation (if required). Performance of various models need to be clearly evaluated and best model needs to be recommended based on some robust evaluation criteria e.g., AIC (Akaike information criterion), Accuracy, RMSE, MSE etc.

3 IDEATION & PROPOSED SOLUTION:

3.1 EMPATHY MAPS:



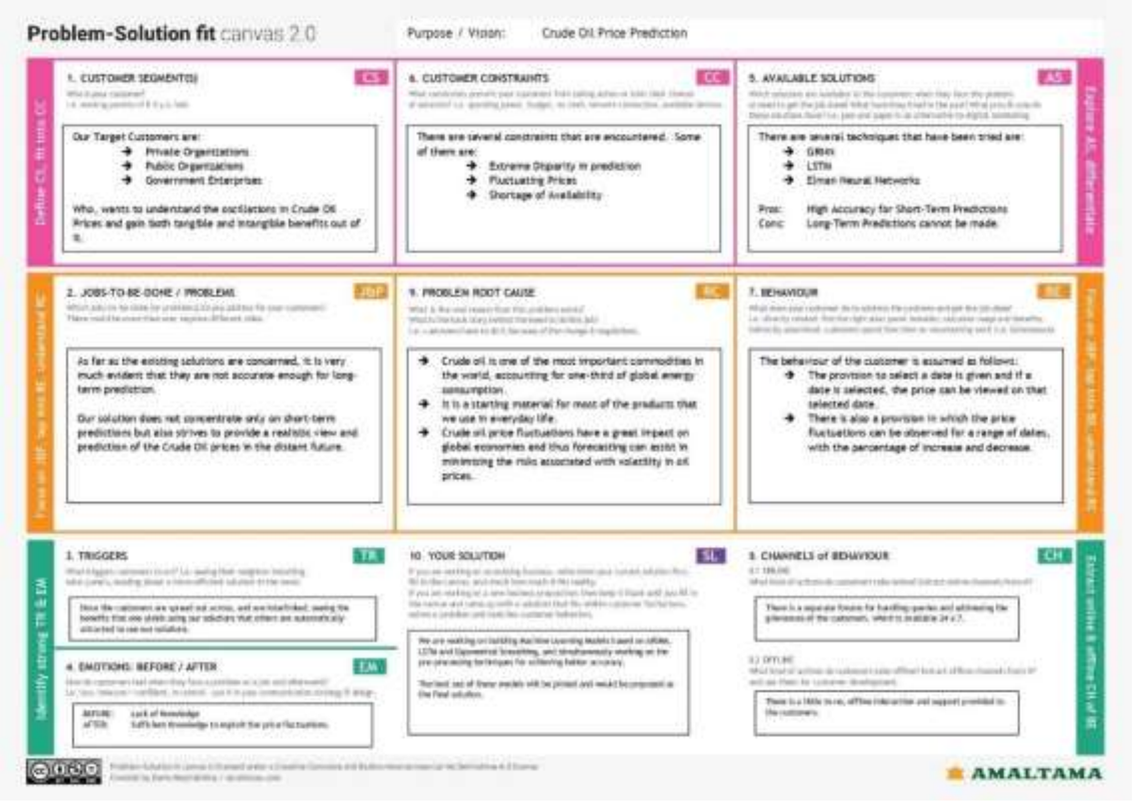
3.2 IDEATION AND BRAINSTORMING:

3.3 PROPOSED SOLUTION:

S. No.	Parameter	Description
1	Problem Statement (Problem to be Solved)	The existing patrimony model of oil price prediction is not capable enough to deliver the accurate predicted prices as expected. Few factors can be described as the conjectural buying and selling , geopolitical, OPEC output, increased demand from important role in the prediction of the oil prices. Now problem arising with the current ANN and CNN models that are used as prediction model's are that they can't provide accurate results when the data is too big.
2	Idea / Solution description	1) LSTM clears about keeping the previous data and prediction which might be encouraging and more accurate. The possible results are comparatively inspiring. 2) The LSTM model will be updated whenever new oil price data are available, and provided to model, so the model continuously evolves over time, and can capture the changing pattern of oil prices.
3	Novelty / Uniqueness	1) Price forecasting can assist in minimising the risks associated with volatility in oil prices. 2) Price forecasts are very important to various stakeholders: governments, public and private enterprises, policymakers, and investors.
4	Social Impact / Customer Satisfaction	1) Brand activation 2) Innovative and schemes 3) Instant reward schemes 4) Personalized consumer purchase exchanges 5) Capability building of sales personnel.

5	Business Model (Revenue Model)	1) The price of crude oil should be easily predictable from the equilibrium between demand and supply. 2) Traders analyze demand and supply factors and take calculated positions. If their prediction comes true, traders close their position to book profits way before expiry. 3) price of crude oil are changeable based from time to time.
6	Scalability of the Solution	1) hydrodynamic conditions in oilfield operations is suggested. 2) Modern refineries typically use a high number of sensors that generate an enormous amount of data. 3) Sustainable Solution for Crude Oil using Concentrated Solar Power Technology.

3.4 PROBLEM SOLUTION FIT:



Problem-Solution fit canvas 2.0

Purpose / Vision: Crude Oil Price Prediction

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Our Target Customers are:
- Private Organizations
- Public Organizations
- Government Enterprises

Who, wants to understand the oscillations in Crude Oil Prices and gain both tangible and intangible benefits out of it.

6. CUSTOMER CONSTRAINTS CC

There are several constraints that are encountered. Some of them are:
- Extreme Disparity in prediction
- Fluctuating Prices
- Shortage of Availability

5. AVAILABLE SOLUTIONS AS

There are several techniques that have been tried are:
- GRNN
- LSTM
- Elman Neural Networks

Pros: High Accuracy for Short-Term Predictions
Cons: Long-Term Predictions cannot be made.

Explore AS, differentiate

2. JOBS-TO-BE-DONE / PROBLEMS J&P

As far as the existing solutions are concerned, it is very much evident that they are not accurate enough for long-term prediction.

Our solution does not concentrate only on short-term predictions but also strives to provide a realistic view and prediction of the Crude Oil prices in the distant future.

9. PROBLEM ROOT CAUSE RC

- Crude oil is one of the most important commodities in the world, accounting for one-third of global energy consumption.
- It is a starting material for most of the products that we use in everyday life.
- Crude oil price fluctuations have a great impact on global economies and thus forecasting can assist in minimizing the risks associated with volatility in oil prices.

7. BEHAVIOUR BE

The behaviour of the customer is assumed as follows:
- The provision to select a date is given and if a date is selected, the price can be viewed on that selected date.
- There is also a provision in which the price fluctuations can be observed for a range of dates, with the percentage of increase and decrease.

Identify strong TR & EM

3. TRIGGERS TR

Once the customers are spread out across, and are benefited, seeing the benefits that one yields using our solutions that others are automatically attracted to use our solution.

4. EMOTIONS: BEFORE / AFTER EM

BEFORE: Lack of knowledge
AFTER: Sufficient knowledge to exploit the price fluctuations.

10. YOUR SOLUTION SL

We are working on building Machine Learning Models based on ARIMA, LSTM and Exponential Smoothing, and simultaneously working on the pre-processing techniques for achieving better accuracy.

The best set of these models will be picked and would be proposed as the final solution.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

There is a separate forum for handling queries and addressing the grievances of the customers, which is available 24 x 7.

8.2 OFFLINE

There is a little to no, offline interaction and support provided to the customers.

Extract online & offline CH of BE

AMALTAMA

4 REQUIREMENT ANALYSIS:

4.1 FUNCTIONAL REQUIREMENT:

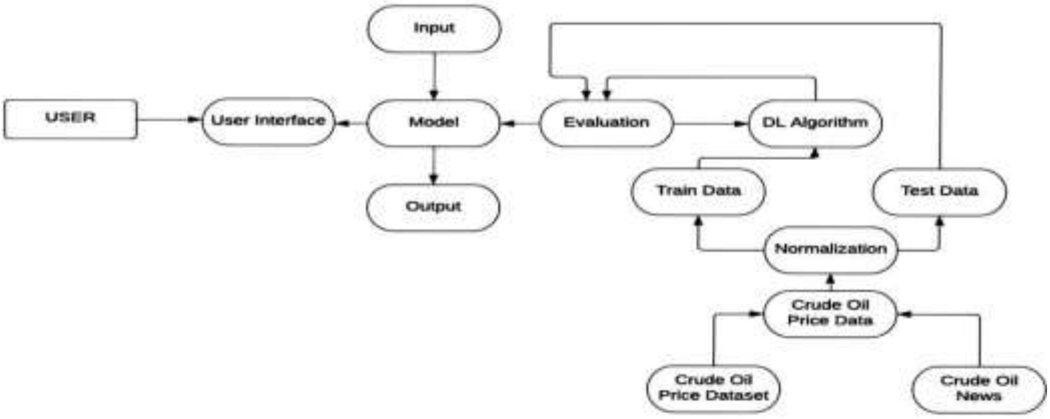
FRNO.	FUNCTIONAL REQUIREMENT	SUB REQUIREMENT
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Enquiry	Enter the date or range of dates
FR-4	User Visualization	Visualize the trend Enquire the prices Analyse the results
FR-5	User Endowment	See the results Gain Knowledge
FR-6	User Utilization	Use it in your idea Close the portal

4.2 NON-FUNCTIONAL REQUIREMENT:

FRNO.	NON-FUNCTIONAL REQUIREMENT	DESCRIPTION
FR-1	**Usability**	The application interface is easy to use and implement.
FR-2	**Security**	The credentials are secured and the result is encrypted.
FR-3	**Reliability**	The accuracy and reliability quotient is quoted to be high.
FR-4	**Performance**	The performance is uninterrupted and undeterred
FR-5	**Availability**	The data is freely available and the trend can be manually analysed
FR-6	**Scalability**	The predictions are scalable and reliable.

5 PROJECT DESIGN

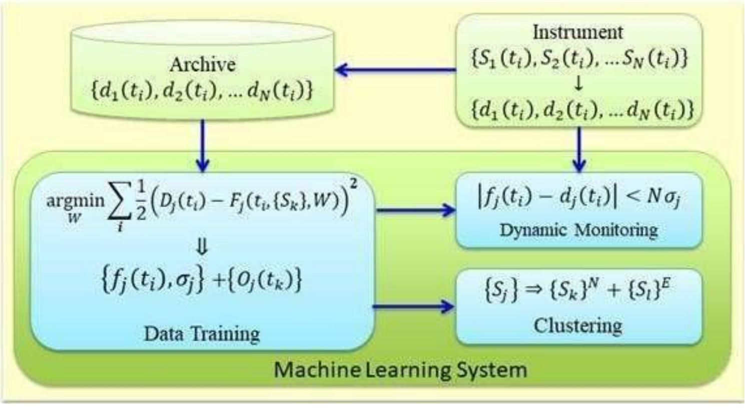
5.1 DATA FLOW DIAGRAM:



5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

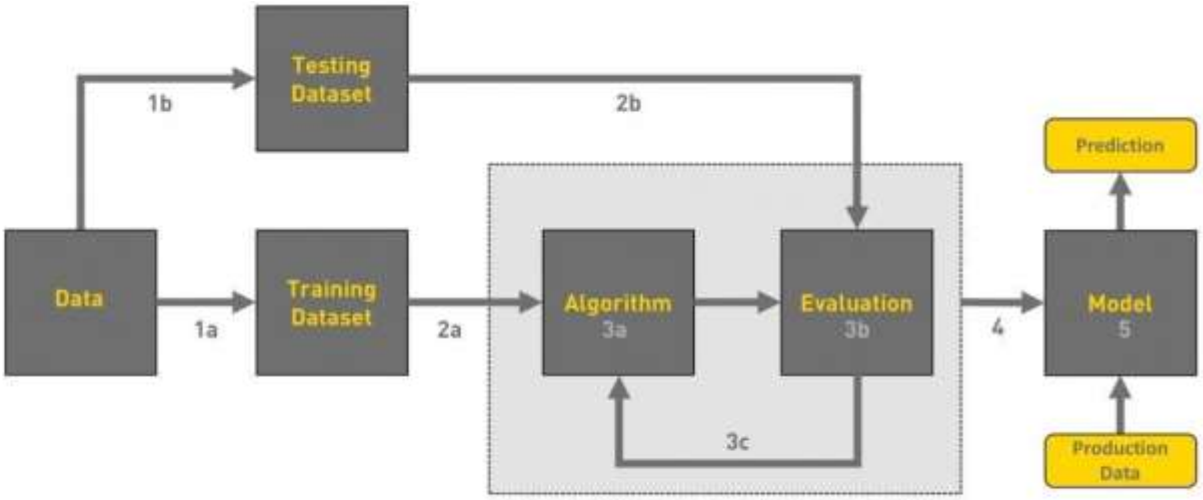
Context View

This view gives a high level representation of the system, the different user types and interactions with external entities. It describes the boundaries of the solution.


Archive
$\{d_1(t_i), d_2(t_i), \ldots d_N(t_i)\}$
Instrument
$\{S_1(t_i), S_2(t_i), \ldots S_N(t_i)\}$
$\downarrow$
$\{d_1(t_i), d_2(t_i), \ldots d_N(t_i)\}$
$\underset{W}{\operatorname{argmin}} \sum_i \frac{1}{2}\left(D_j(t_i) - F_j(t_i, \{S_k\}, W)\right)^2$
$\Downarrow$
$\{f_j(t_i), \sigma_j\} + \{O_j(t_k)\}$
Data Training
$|f_j(t_i) - d_j(t_i)| < N\sigma_j$
Dynamic Monitoring
$\{S_j\} \Rightarrow \{S_k\}^N + \{S_l\}^E$
Clustering
Machine Learning System

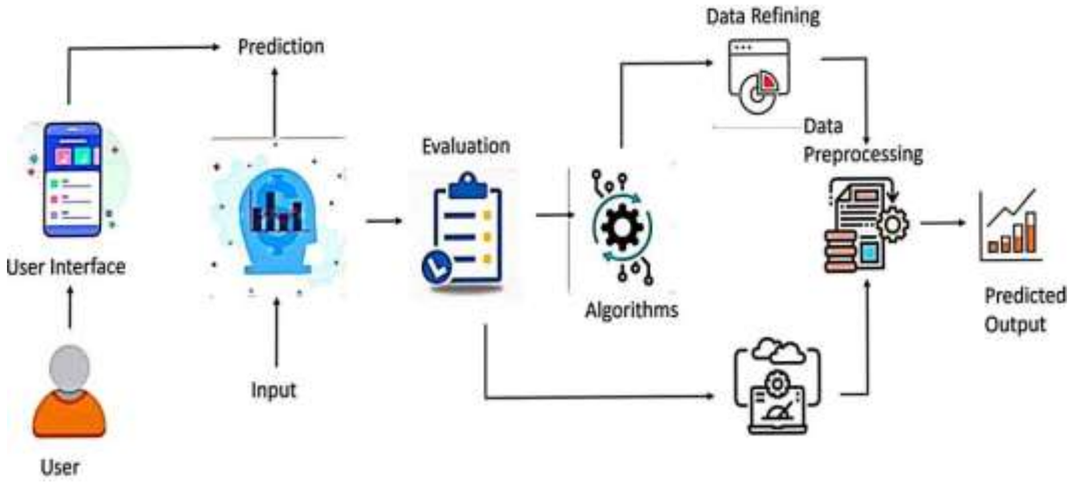
Project View

This section shows how key functionality relevant to the solution architecture maps to releases and milestones.



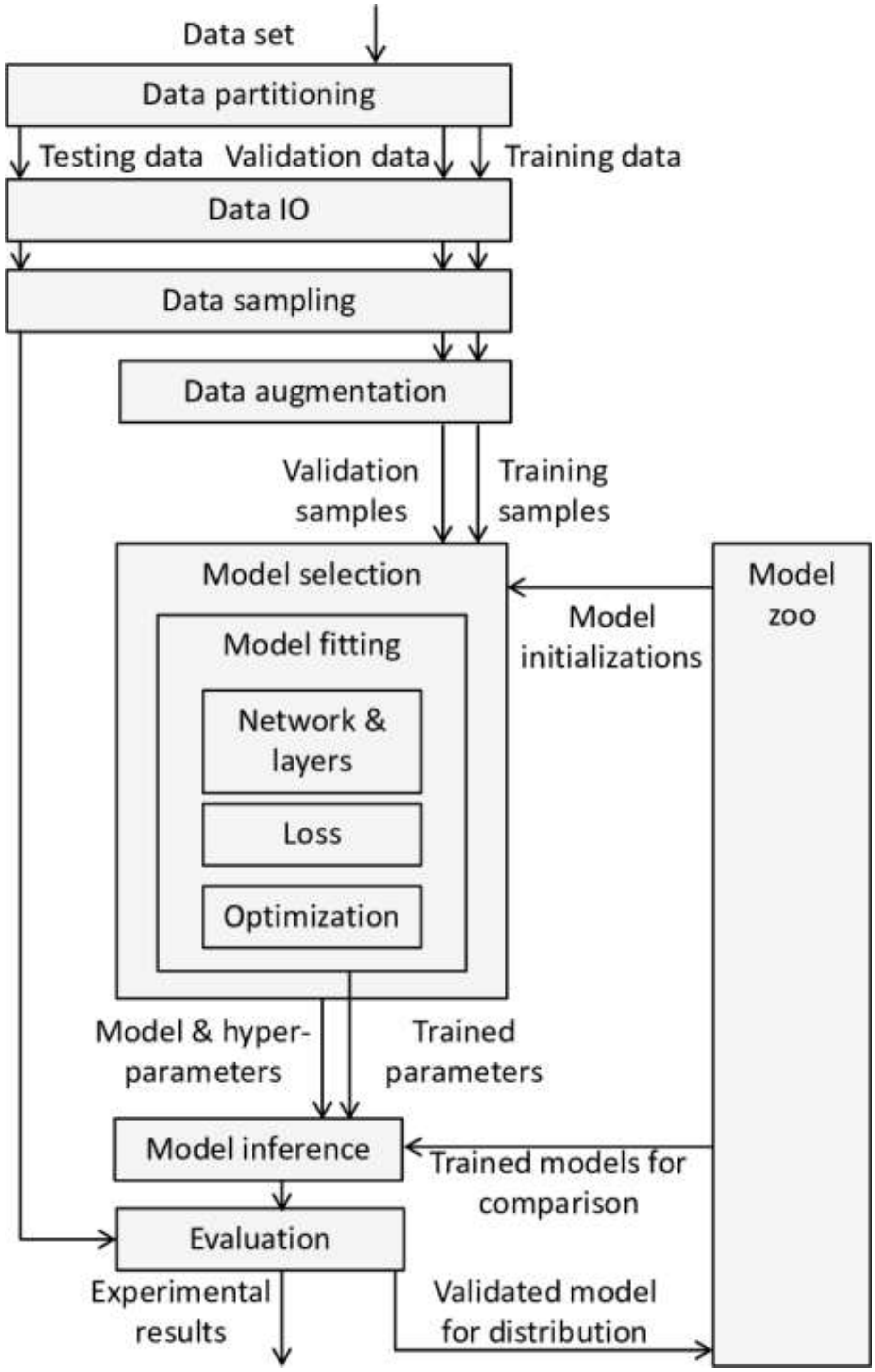
Functional View

This section describes the key functional areas of the project. The goal is to provide context around the architecture – all software performs some functionality and the definition of this functional scope is a very important factor to define the architecture.



Process View

The intent of the process view is to show how the various processing steps within the system fit together to implement the overall functional requirements. This is necessary if the system relies on workflow processes, forked or parallel processing mechanisms. The following processes are significant:



Non-Functional View

This section describes architecturally significant changes that enable the solution to achieve the agreed non-functional requirements (NFRs). Each change is mapped to the corresponding NFR category, which is based on the ISO/IEC 25010-2011 product quality model.

NFRs are documented and maintained in the Non-Functional Requirements Definition and will not be repeated here. In case of duplication, the Non-Functional Requirements Definition takes precedence.

Performance Easy tracking of records and updating can be done. All the requirements relating to performance characteristics of the system are specified in the section below. There are two types of requirements.

1. Static Requirements:

These requirements do not impose any constraints on the execution characteristics of the system. They are:

A) Number of Terminals: The software makes use of an underlying database that will reside at the same system, while the front end will be available to the administrative computer.

B) Number of Users: The number of users can be administrator only, but this software can be extended to applications for almost all staff members of the organization.

2. Dynamic Requirements:

These specify constraints on the execution characteristics of the system. They typically include response time and throughout of the system. Since these factors are not applicable to the proposed software, it will suffice if the response tine is high and the transactions are carried out precisely and quickly. Reliability: The software will not be able to connect to the database in the event of the server being down due to a hardware or software failure.

3. Availability:

The software will be available only to administrator of the organization and the product as well as customer details will be recorded by him. He can add customers, update and delete them as well as add new products and manage them.

4. **Security:**

The security requirements deal with the primary security. The software should be handled only by the administrator and authorized users. Only the administrator has right to create new accounts and generating inventory. Only authorized users can access the system with username and password of administrator

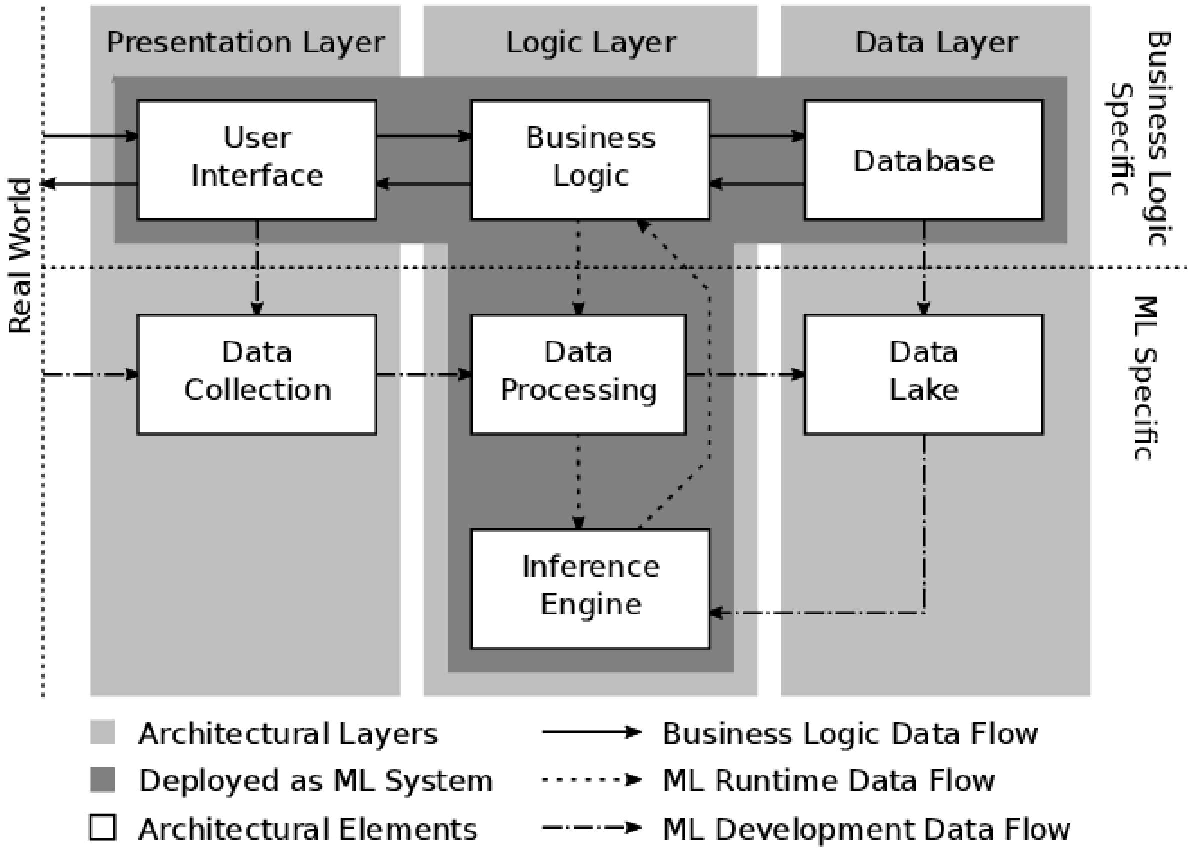
5. **Maintainability:**

Backups for database are available.

6. **Portability:**

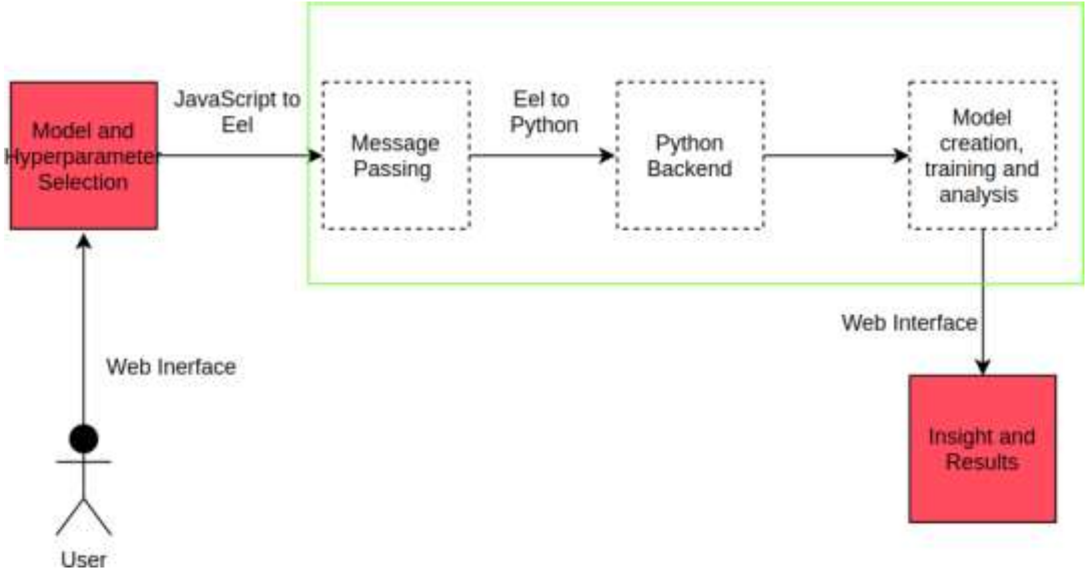
The Software is a web-based application and is built in Python and Nosql so it is platform independent and is independent of operating system.

Logical View



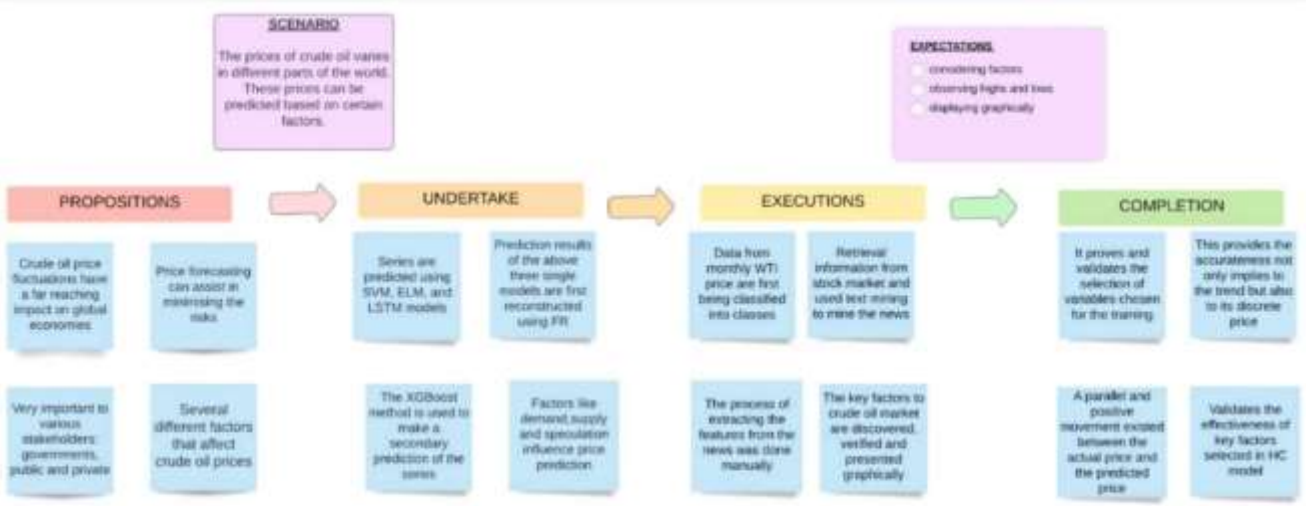
1 Interface View

This section describes the interfaces that will be required to the external system integration touch points



Design View

This section describes and explains any lower-level design concepts arising from the solution if required.



SCENARIO

The prices of crude oil varies in different parts of the world. These prices can be predicted based on certain factors.

EXPECTATIONS

○ considering factors
○ observing highs and lows
○ displaying graphically

PROPOSITIONS

Crude oil price fluctuations have a far reaching impact on global economies

Price forecasting can assist in minimising the risks

Very important to various stakeholders: governments, public and private

Several different factors that affect crude oil prices

UNDERTAKE

Series are predicted using SVM, ELM, and LSTM models

Prediction results of the above three single models are first reconstructed using FR

The XGBoost method is used to make a secondary prediction of the series

Factors like demand,supply and speculation influence price prediction

EXECUTIONS

Data from monthly WTI price are first being classified into classes

Retrieval information from stock market and used text mining to mine the news

The process of extracting the features from the news was done manually

The key factors to crude oil market are discovered, verified and presented graphically

COMPLETION

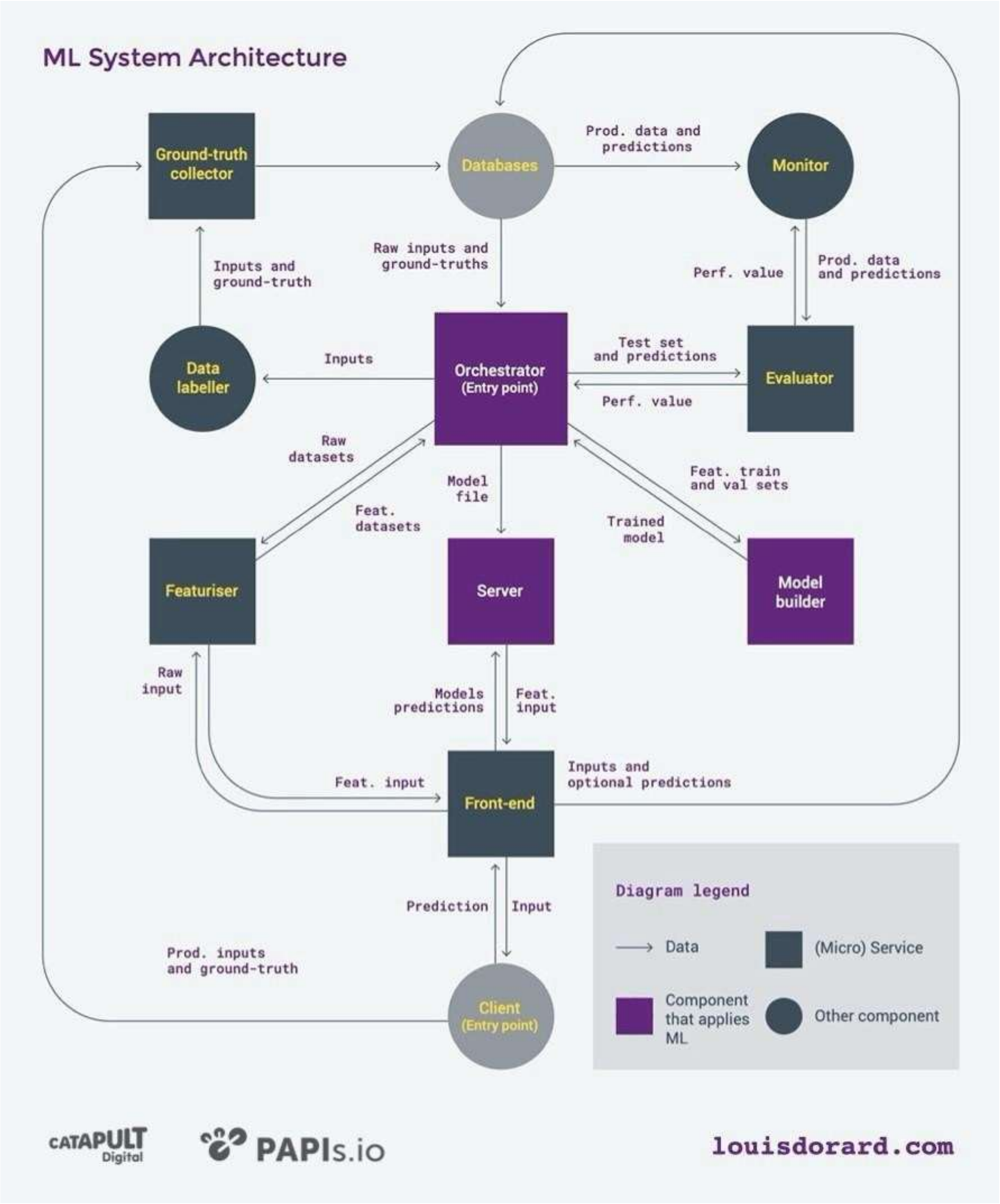
It proves and validates the selection of variables chosen for the training

This provides the accurateness not only implies to the trend but also to its discrete price

A parallel and positive movement existed between the actual price and the predicted price

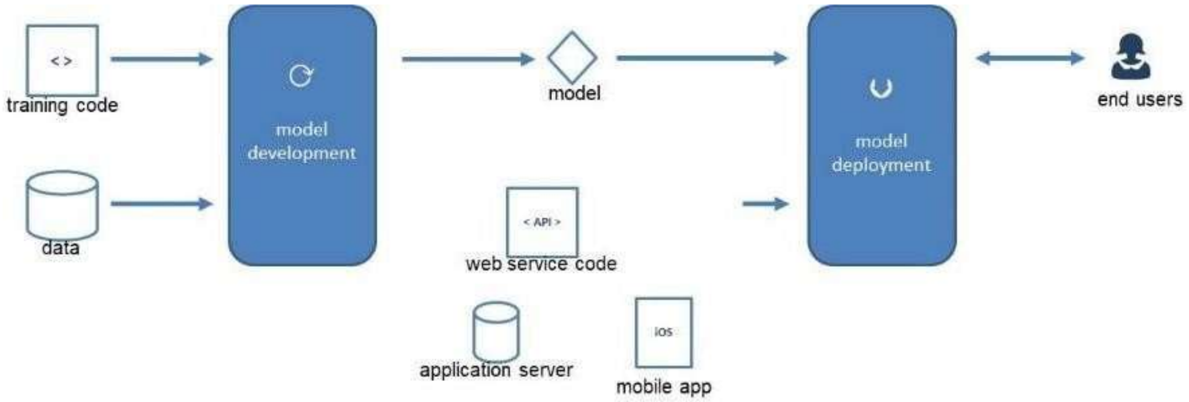
Validates the effectiveness of key factors selected in HC model

Physical View



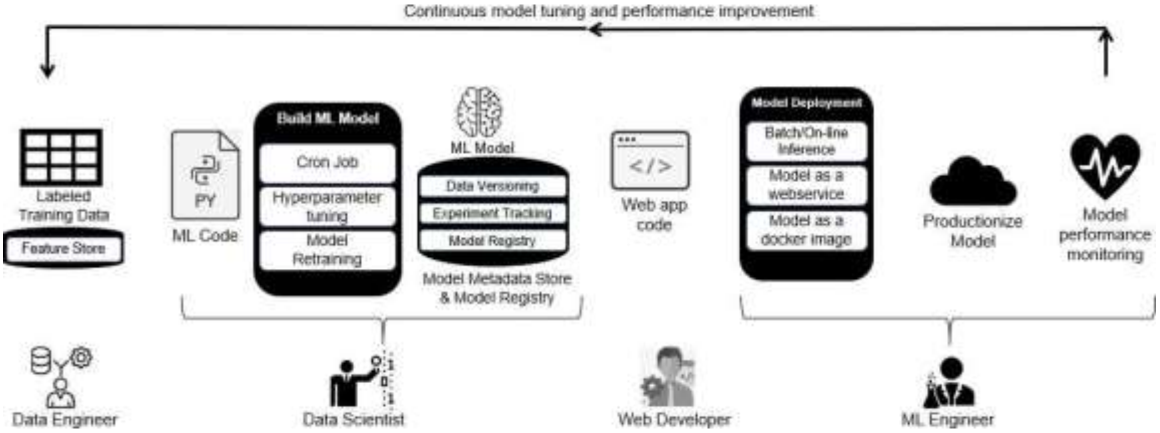
Deployment View

This section describes how code will be deployed in test environments and key considerations for the more complex Production go-live deployment.



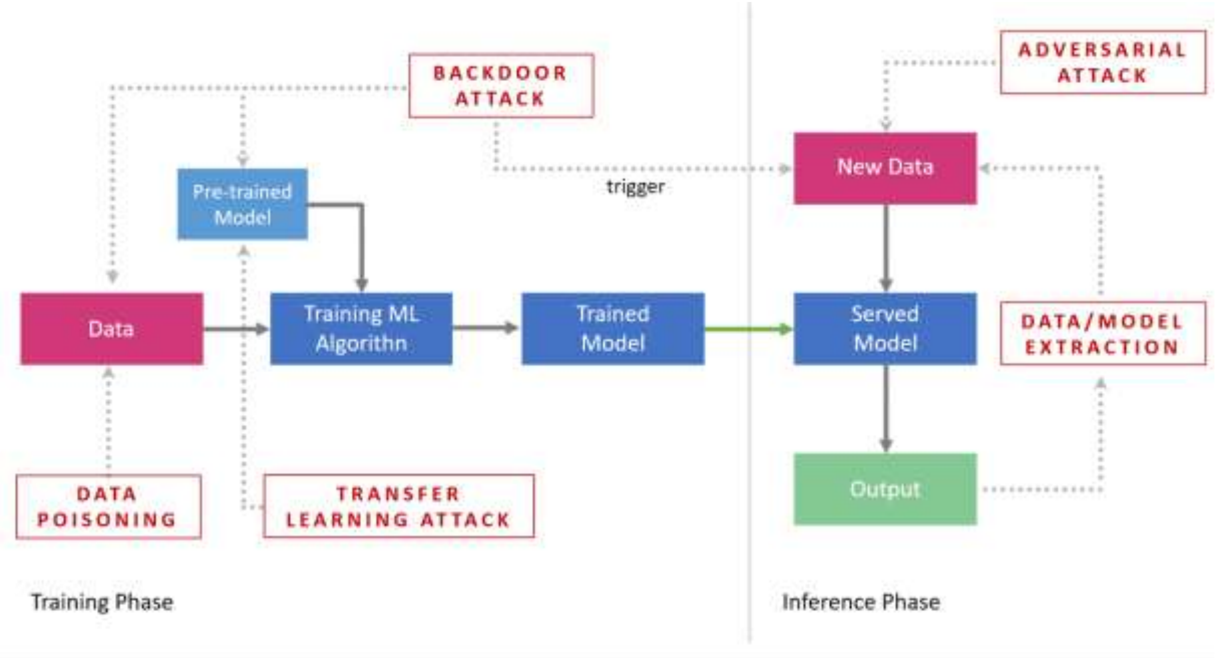
Operational View

This section describes how the architecture will support operational processes and activities.



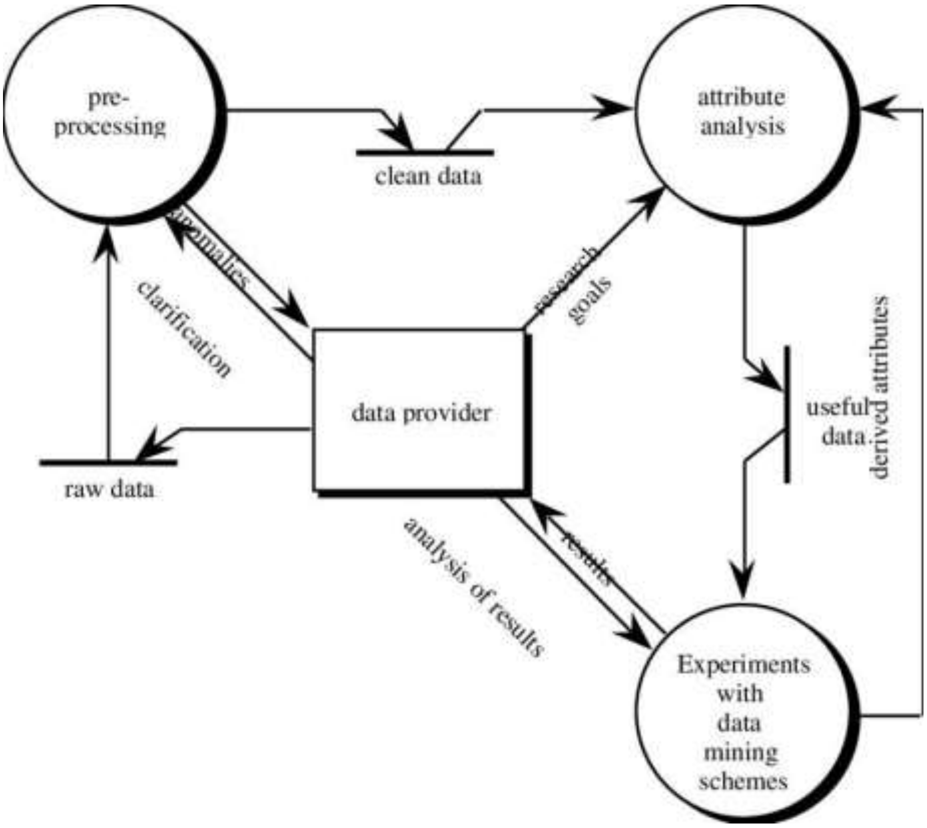
Security View

This section describes how the architecture addresses the different security aspects.



Data View

This section describes the important data model changes required to fulfil the requirements and the associated data flows.



5.3 USER STORIES:

USER TYPE	FUNCTIONAL REQUIREMENT	USER STORY	USER STORY / TASK	ACCEPTANCE CRITERIA	PRIORITY	RELEASE
Customer	Registration	1	Register for the Application through different vendors.	I can access my account / dashboard	High	Sprint-3
	Confirmation	2	Receiving Confirmation Mail	I can receive confirmation email & click confirm	High	Sprint-4
	Login	3	Log in into the application	Access to the account	High	Sprint-2
	Enquiry	4	Enter the range of dates	Plausible Range	High	Sprint-1
	Visualize	5	Visualize the Trend	Accuracy Check	High	Sprint-3
	Endowment	6	See the result	Prediction Check	High	Sprint-1
	Utilization	7	Log Out	Confirmation and Session Closure	High	Sprint-2
Administrator	Authority	1	Verify the imbalances	Session Dryness	High	Sprint-3

6 PROJECT PLANNING AND SCHEDULING:

6.1 SPRINT PLANNING AND ESTIMATION:

SPRINT	FUNCTIONAL REQUIREMENT	USER STORY	USER STORY / TASK	STORY POINTS	PRIORITY	MEMBERS
1	Registration	1	Register for the Application	2	High	2
2	Confirmation	2	Receiving Confirmation Mail	1	Medium	2
2	Login	3	Log in into the application	2	High	2
3	Enquiry	4	Enter the range of dates	2	Medium	2
4	Visualize	5	Visualize the Trend	2	High	2
3	Endowment	6	See the result	2	High	2
4	Utilization	7	Log Out	1	Medium	2

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

1. ***CODING & SOLUTIONING (Explain the features added in the project along with code)***
   1. *A LSTM Price forecasting machine learning model*
   2. *A User Interface for forecasting based on the past 10 days price*

2. ***TESTING***
   1. *Test Cases*
   2. *User Acceptance Testing*

3. ***RESULTS***
   1. *Performance Metrics – RMSE – 2.78*

7.coding and solution

```
LOGIN.html

&lt;!DOCTYPE html&gt;
&lt;html lan=&quot;en&quot; and dir=&quot;Itr&quot;&gt;
    &lt;head&gt;
        &lt;meta charset=&quot;utf-8&quot;&gt;
        &lt;title&gt;login form&lt;/title&gt;
        &lt;link rel=&quot;stylesheet&quot; href=&quot;style.css&quot;&gt;
         &lt;script src =&quot;login.js&quot;&gt;&lt;/script&gt;
    &lt;/head&gt;
    &lt;body&gt;
        &lt;form class=&quot;box&quot; action=&quot;login.html&quot;
method=&quot;POST&quot;&gt;
            &lt;h1&gt;CRUDE OIL PRICE PREDICTION&lt;/h1&gt;
            &lt;h2&gt;
                 LOGIN
            &lt;/h2&gt;
            &lt;input type=&quot;text&quot; name=&quot;&quot; placeholder=&quot;Enter
Username&quot;
id=&quot;username&quot;&gt;
            &lt;input type=&quot;password&quot; name=&quot;&quot; placeholder=&quot;Enter
Password&quot;
id=&quot;password&quot;&gt;
            &lt;input type=&quot;submit&quot; name=&quot;&quot;
value=&quot;Login&quot;  onclick=&quot;validate()&quot;&gt;
            &lt;h3&gt;&lt;a href=&quot;register.html&quot;&gt; New User ? Register
&lt;/a&gt;&lt;/h3&gt;
        &lt;/form&gt;
    &lt;/body&gt;
&lt;/html&gt;

LOGIN.JS
function validate()
{
    var username=document.getElementById(&quot;username&quot;).Value;
    var password=document.getElementById(&quot;password&quot;).Value;
    if(username==&quot;ibm&quot;&amp;&amp;password==&quot;ibm123&quot;)
    {
        alert(&quot;login succesfully&quot;);
        return true;
    }
    else
    {
        document.getElementById(&quot;username&quot;).disabled=true;
        document.getElementById(&quot;password&quot;).disabled=true;
        return false;
    }

}

{
    document.getElementByI
d(&quot;email&quot;).value=&quot;&quot;;
    document.getElementById(&quot;pwd1&quot;).value=&quot;&quot;;
}
```

predict.html

```
<!DOCTYPE html>
<head>
```

```
    <title>Crude Oil Price Prediction </title>
    <link href='https://fonts.googleapis.com/css?family=Roboto' rel='stylesheet'>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/predict.css') }}">
</head>
<body style="text-align:center;background-color: lightsteelblue;">
    <h1 style="color: white;font-size: 50px;font-family: roboto;">
    Crude Oil Price Prediction </h1>
    <h1 style="color: white;font-size: 50px;font-family: roboto;">
        Enter the Oil price for 10 days </h1>
        <form action="/predict" method="POST" enctype = "multipart/form-data">
            <div style="color:green;font-size:50px;font-family:roboto;">
                {{prediction}}
            </div>
        <input type="text" name="val" style="border-radius: 18px;padding: 20px;width: 
300px;height: 15px;text-align: center; align:center;" >
         <br> <br> <br>
        <input type="submit"/ style="border-radius: 9px;;padding: 10px;width: 150px;
        height: 40px;text-align: center;background: #003d66;color: white;">
        </form>
        <br>
        <form action="/predict" method="GET" enctype = "multipart/form-data">
        <input type="submit"/ value="Reset" style="border-radius: 9px;;padding: 
10px;width: 150px;
        height: 40px;text-align: center;background: #003d66;color: white;">
        </form>
</body>
```

REGISTER.CSS

```
body{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
    background: url(ppp.jpg);
    background-size: cover;
}
.box{
    width: 300px;
    padding: 30px;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%,-50%);
    background: rgb(14, 14, 14);
    text-align: center;
}
.box h1
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}
```

```
.box h2
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}

.box input[type="text"],.box input[type="password"] ,.box input[type="date"],.box
input[type="Number"],.box input[type="Email"]
{
    border: 0;
    background: white;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid #2af003;
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: #fff6ff(18, 18, 179);
    border-radius: 24px;
    transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
    width: 270px;
    border-color: rgb(238, 26, 203);

}
.box input[type="submit"]{
    border: 0;
    background: none;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid rgb(211, 15, 152);
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: rgb(73, 31, 224);
    border-radius: 24px;
    transition: 0.25px;
    cursor: pointer;


}
.box input[type="submit"]:hover{
    background: rgb(100, 182, 53);
}
h3{
    color: wheat;
}
```

REGISTER.HTML

```
<!DOCTYPE html>
<html lan="en" and dir="Itr">
    <head>
        <meta charset="utf-8">
        <title>login form</title>
        <link rel="stylesheet" href="register.css">
         <script src ="login.js"></script>
    </head>
    <body>
        <form class="box" action="login.html" method="POST">
            <h1>CRUDE OIL PRICE PREDICTION</h1>
            <h2>
                 Register
            </h2>
            <input type="text" name="" placeholder="Enter Username" id="username">
            <input type="email" name="" placeholder="Enter Your Email Id" id="Email">
            <input type="number" name="" placeholder="Enter Your Number" id="Number">
            <input type="password" name="" placeholder="Enter Password" id="password">
            <input type="submit" name="" value="Register"  onclick="validate()">
            <h3><a href="login.html"> Login </a></h3>
        </form>

    </body>
</html>
```

STYLE.CSS

```
body{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
    background: url(p2.jpg);
    background-size: cover;
}
.box{
    width: 300px;
    padding: 30px;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%,-50%);
    background: rgb(14, 14, 14);
    text-align: center;
}
.box h1
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}
.box h2
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}

.box input[type="text"],.box input[type="password"] ,.box input[type="date"],.box
input[type="Number"],.box input[type="Email"]
{
    border: 0;
    background: white;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid #2af003;
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: #fff6ff(18, 18, 179);
    border-radius: 24px;
    transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
    width: 270px;
    border-color: rgb(238, 26, 203);

}
.box input[type="submit"]{
    border: 0;
    background: none;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid rgb(211, 15, 152);
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: rgb(73, 31, 224);
    border-radius: 24px;
    transition: 0.25px;
    cursor: pointer;


}
.box input[type="submit"]:hover{
    background: rgb(100, 182, 53);
}
h3{
    color: wheat;
}
```

APP.PY

```
from flask import Flask,render_template,request,redirect
import numpy as np
import joblib
```

```
from keras.models import load_model

app = Flask(__name__)

@app.route('/',methods=["GET"])
def index():
    return render('login.html')

@app.route('/predict',methods=["POST","GET"])
def predict():
    if request.method == "POST":
        string = request.form['val']
        if(string ==""):
            return render_template('predict.html')
        string = string.split(',')
        x_input = [eval(i) for i in string]
        sc = joblib.load("scaler.save")
        x_input = sc.fit_transform(np.array(x_input).reshape(-1,1))
        x_input = np.array(x_input).reshape(1,-1)
        x_input = x_input.reshape(1,-1)
        x_input = x_input.reshape((1,10,1))
        model = load_model('model.h5')
        output = model.predict(x_input)
        val = sc.inverse_transform(output)
        return render_template('predict.html', prediction = "The predicted price is
{:.2f}".format(val[0][0]))
    if request.method == "GET":
        return render_template('predict.html')

if __name__=="__main__":
    model = load_model('model.h5')
    app.run(host='0.0.0.0', port=5000)
```

8. TESTING

8.1 TEST CASES

Sprint 1:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

```
data=pd.read_excel("/content/Crude Oil Prices Daily.xlsx")
```

In [3]:

```
data.isnull().any()
```

Out[3]:

```
Date             False
Closing Value     True
dtype: bool
```

In [4]:

```
data.isnull().sum()
```

Out[4]:

```
Date             0
Closing Value    7
dtype: int64
```

In [5]:

```
data.dropna(axis=0,inplace=True)
```

In [6]:

```
data.isnull().sum()
```

Out[6]:

```
Date             0
Closing Value    0
dtype: int64
```

In [7]:

```
data_oil=data.reset_index()['Closing Value']
data_oil
```

Out[7]:

```
0       25.56
1       26.00
2       26.53
3       25.85
4       25.87
        ...
8211    73.89
8212    74.19
8213    73.05
8214    73.78
8215    73.93
Name: Closing Value, Length: 8216, dtype: float64
```

In [8]:

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))
```

In [9]:

```
data_oil
```

Out[9]:

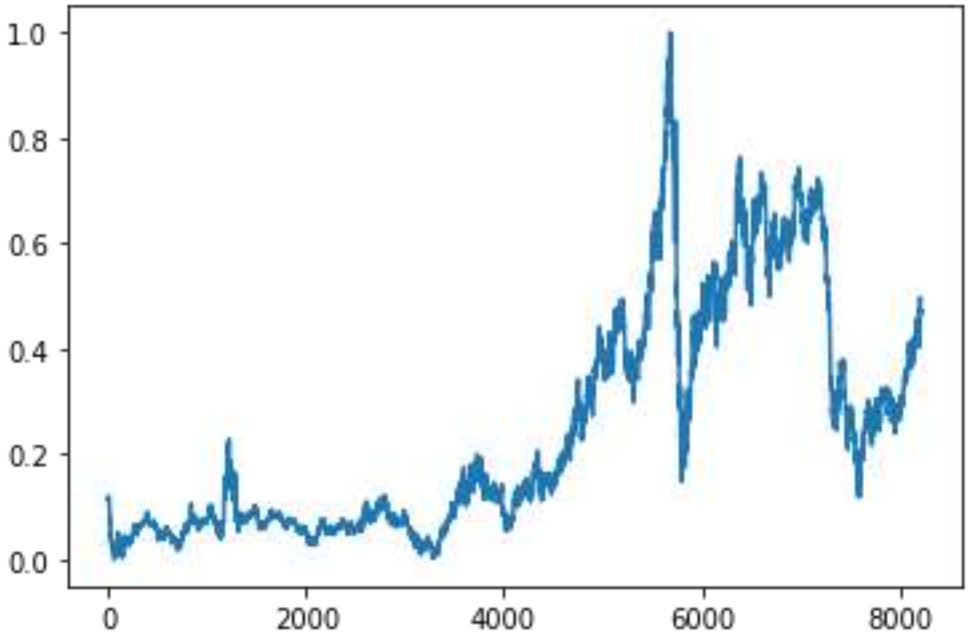
```
array([[0.11335703],
       [0.11661484],
       [0.12053902],
       ...,
       [0.46497853],
       [0.47038353],
       [0.47149415]])
```

In [10]:

```
plt.plot(data_oil)
```

Out[10]:

```
[]
```



In [11]:

```
training_size=int(len(data_oil)*0.65)
test_size=len(data_oil)-training_size
train_data,test_data=data_oil[0:training_size,:],data_oil[training_size:len(data_oi
l),:1]
```

In [12]:

```
training_size,test_size
```

Out[12]:

```
(5340, 2876)
```

In [13]:

```
train_data.shape
```

Out[13]:

```
(5340, 1)
```

In [14]:

```
def create_dataset(dataset,time_step=1):
  dataX,dataY=[],[]
  for i in range(len(dataset)-time_step-1):
    a=dataset[i:(i+time_step),0]
    dataX.append(a)
    dataY.append(dataset[i+time_step,0])
  return np.array(dataX),np.array(dataY)
```

In [15]:

```
time_step=10
x_train,y_train=create_dataset(train_data,time_step)
x_test,y_test=create_dataset(test_data,time_step)
```

In [16]:

```
print(x_train.shape),print(y_train.shape)
(5329, 10)
(5329,)
```

Out[16]:

```
(None, None)
```

In [17]:

```
print(x_test.shape),print(y_test.shape)
(2865, 10)
```

```
(2865,)
```

Out[17]:

```
(None, None)
```

In [18]:

```
x_train
```

```
array([[0.11335703, 0.11661484, 0.12053902, ..., 0.10980305, 0.1089886
        0.11054346],
       [0.11661484, 0.12053902, 0.11550422, ..., 0.1089886 , 0.11054346,
        0.10165852],
       [0.12053902, 0.11550422, 0.1156523 , ..., 0.11054346, 0.10165852,
        0.09906708],
       ...,
       [0.36731823, 0.35176958, 0.36080261, ..., 0.36391234, 0.37042796,
        0.37042796],
       [0.35176958, 0.36080261, 0.35354657, ..., 0.37042796, 0.37042796,
        0.37879461],
       [0.36080261, 0.35354657, 0.35295424, ..., 0.37042796, 0.37879461,
        0.37916482]])
```

In [19]:

```
x_train=x_train.reshape(x_train.shape[0],x_train.shape[1],1)
x_test=x_test.reshape(x_test.shape[0],x_test.shape[1],1)
```

Sprint 2:

crudeoil.html

```
<!DOCTYPE
html>
                <html>
                <head>
                <title>PRICE PREDICTION</title>
                </head>
                <body>
                <p>&nbsp;</p>
                <p>&nbsp;</p>
                <style>
                body {
                  background-image: url('CR.webp');
                  background-repeat: no-repeat;
                    background-attachment: fixed;
                  background-size: cover;
                }
                {
                      min-height: 70%;
                      }
                      body,form {
                      padding: 0;
                      margin: 0;
                      outline: none;
                      font-family: Roboto, Arial, sans-serif;
                      font-size: 14px;
                      color: #FFFFFF;
                      line-height: 22px;
                     }
```

```
     .myDiv {
  border: 5px outset  #000000;
  background-color: #2F4F4F;
  text-align: center;
  font-family: Roboto, Arial, sans-serif;
      font-size: 14px;
      color: #FFFFFF;
}

</style>
<h1 style="text-align:center"><span style="color:#ffffff"><strong><span style="font-family:Arial,Helvetica,sans-serif"><span style="font-size:36px">CRUDE OIL PRICE PREDICTION&nbsp;</span></span></strong></span></h1>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<form style="text-align:center">
  <label for="ENTER PRICE">ENTER PRICE:</label>
  <input type="date" id="PRICE" name="PRICE">
  <input type="submit">
</form>
<p>&nbsp;</p>
<p>&nbsp;</p>
<div class="myDiv">
  <h2>PREDICTED PRICE:</h2>
  <p>{_____________}</P><br>

  </p>
</div>
</body>
</html>
```

DATA VISULAITATION:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
"import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 2,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().any()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "RAUyZB0-Jp0t",
        "outputId": "2cfb34ee-fdfa-4bf4-f8c3-3da341f3a2a9"
      },
      "execution_count": 3,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             False\n",
              "Closing Value     True\n",
              "dtype: bool"
            ]
          },
          "metadata": {},
          "execution_count": 3
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "yorF39lCJsyV",
        "outputId": "35237200-e540-4f63-b787-b5da37cd411a"
```

```
},
      "execution_count": 4,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    7\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 4
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.dropna(axis=0,inplace=True)"
      ],
      "metadata": {
        "id": "g3FWuWVPJwms"
      },
      "execution_count": 5,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "vIHbyOs4JzIf",
        "outputId": "dbf56470-e7ef-4760-974c-f79f484a6bdb"
      },
      "execution_count": 6,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    0\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 6
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil=data.reset_index()['Closing Value']\n",
"data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "IoxUNwrvJ2b6",
        "outputId": "568cb857-db42-4623-9bba-d73857809dc5"
      },
      "execution_count": 7,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "0        25.56\n",
              "1        26.00\n",
              "2        26.53\n",
              "3        25.85\n",
              "4        25.87\n",
              "         ...  \n",
              "8211     73.89\n",
              "8212     74.19\n",
              "8213     73.05\n",
              "8214     73.78\n",
              "8215     73.93\n",
              "Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "metadata": {},
          "execution_count": 7
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from sklearn.preprocessing import MinMaxScaler\n",
        "scaler=MinMaxScaler(feature_range=(0,1))\n",
        "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
      ],
      "metadata": {
        "id": "5m-DUFI9J_WN"
      },
      "execution_count": 8,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "fyMwOo1jKLqL",
        "outputId": "d4483e05-5472-46cc-eb13-14a4f0a4f37b"
      },
      "execution_count": 9,
"outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "metadata": {},
          "execution_count": 9
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "plt.plot(data_oil)"
      ],
      "metadata": {
        "id": "GdNJartuKUfk",
        "outputId": "f635e36e-e38c-4768-ac5b-3ea693cc9c0a",
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 282
        }
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "[<matplotlib.lines.Line2D at 0x7f8b4c0ed150>]"
            ]
          },
          "metadata": {},
          "execution_count": 10
        },
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXQAAAD4CAYAAAD8Zh1EAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO2dd3gU1frHv282DUIIJQklARJ6lRa
KgHSRooId7OXKtXtt94d6sV1UbFdsV1GvyvXasKOgKEV6C70JBAgQakkJJSCD9/P7Ymd2Z2dndyWZ
ny+z7eR4fZ86cnTk7bL5z5j1vISEEGIZhmPAnKtgDYBiGYfwDCzrDMIxFYEFnGIaxCCzoDMwGFoE
FnWEYxiJEB+vCycnJIiMjI1iXZxiGCUvWr19fIIIIRI0TsWNEHPyMhAdnZ2sC7PMAwTlhDRAXfH2OT
CMAxjEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWwaugE9FHRHSCiL5RDRFiLq5O5E9CYR5RDRFiLq5f9
hMgzDMN4wMkP/BMBoD8fHAGgn/TcZwLu1HxbDMAxTU7wKuhbiKYBTHrqMBfBfYWAZZE1MxfA2Q
Yxhw+XLYPeafPBXsYjB/xhw09DcAhxX6e1OYCEU0momVomws7Pz/fDpRmG8YV9+cWYNncnBr20Nh
```

```
DYfxIQBdFhRDvCyGyhBBZKSm6kasMwwSAc+VVwR4CYwL+EPTDAFoo9tOlNoZhQpRqrlRmSfwh6HM
A3Cx5u/QHUCiEOOqH8zIMYxJV1SzoVsRrci4i+gLAUADJRJQH4GkAMQAghHgPwDwAYwHkADgH4Da
zBsswjH/gGbo18SroQohJXo4LAPf6bUQMw5hOVXWwR8CYAUeKMkwEwiYXa8KCzjARCAu6NWFBZ5g
IpIpt6JaEBZ1hIpCVOQXBHgJjAizoDBOBzFy6L9hDYEyABZ1hGMYisKAzDMNYBBZ0hmEYi8CCzjA
MYxFY0BmGYSwCCzrDMIxFYEFnmAhDKIKKWicnBHEkjL9hQWeYCGP+9uOObVsUBXEkjL9hQWeYCON
saYVj+3wFVy6yEizoDBNhxNicf/Z5p88HcSSMv2FBZ5gIg9jKYllY0BmGYSwCCzrDRBjEU3TLwoL
OMBEGy7l1YUFnmAgjSjFDT2tQJ4gjYfwNCzrDRBhKi8vhM+zlYiVY0BkmwjlRVBrsITB+ggWdYSK
MqT9sU+0Xnq9w05MJN1jQGSbCOFlSrtrngtHWgQWdYSKcqmoWdKvAgs4wEQ5P0K0DCzrDRBh3XpS
p2t9fUILyyuogjYbxJyzoDBNhaC0s93+xEY9/tzU4g2H8Cgs6w0QYss28Z8sGjrblOfnBGg7jR1j
QGSbCqBYCDerG4L5hbR1txAkBLAELOsNEGFXVAjYirlZkQQwJOhGNJqJdRJRDRFN0jrckosVEtJG
IthDRWP8PlWEYf/DZmoM4WVKOTYfOBHsojJ/xKuhEZAPwDoAxADoDmEREnTXd/gFgthCiJ4CJAP7
t74EyDONf9pwoDvYQGD9jZIbeF0COEGKfEKIcwJcAxmv6CAD1pe0kAEf8N0SGYcxgQJvGjm1OkW4
NjAh6GoBDiv08qU3JMwBuJKI8APMA3K93IiKaTETZRJSdn8+r6gwTaIQiihqhPRqMgjoQxA38tik4
C8IkQIh3AWACfEpHLuYUQ7wshsoQQWSkpKX66NMMwRlGG+SvXRHmCbg2MCPphAC0U++lSm5I7AMw
GACHEKgDxAJL9MUCGYfxHpUrQWcathhFBXwegHRFlElEs7IueczR9DgIYCQBE1AmAl2QWebCsOEGOV
VzhD/M5w213J4FXQhRCWA+wDMB7ATdm+W7UT0HBFdLnV7V7BMCdRLQZwwBcAbhWCU/4wTKhRocjZopy
fc+FoaxBtpJMQYh7si53KtqcU2zsADPTv0BiG8TdKk0uMjeMKrQb/izJMBKHMqhioSflnaw7g09U
HAnOxCMfQDJ1hGGtQobChN0uqE5BrPvm9veTdTf1bBeR6kQzP0BkmgpBNLqmJcWiUEIvkenGmXm/
JbvaNCCQs6AwTQcgml2kTugIA+mQ0BAAcPnPelOutzCkw5byMPizoDBNBfLshDwCwN78EQHj7ov+
46TC+XZ8X7GGEFGxDZ5gI4uMVuQCAg6ckQQ/jFLoPfrkJAHBV7/QgjyR04Bk6w0QI1QqXxRv62Rc
ozdZzb8EopRVVuOCZ+fh9x3FzBxIhsKAzTISgjBJNiLO/nNtMNrnsOX7W4/G80+dQVFqJO/+bXaP
zctyiPizoDBMhKF0Wo6WpudkRoot3GfdyqdJWr3aDEAKZj8/z2GfjwdO49K1lKr/7SIAFnWEiBKW
4ybbzGFuwbejO6x8+bczTptKA8F/x75XYdrgIu728IVgNFnSGiRDKdWboZi+Ktmpc13Dfj1bsN9S
vuLRSta9889Cy6M8Thq9vBVjQGSZCUM7Q5QLRcdHmSoD3QtTO2fYnK3Nx8O5r+d85OvNqv3Siiq
3fQ8YOJ+VYEFnmAhBKejyDD3axBn6scJS7JP83d2hNZ8cKyr1el7Lgr79SCEypsgzqNFjmP
fbsjDuXL1jN7KsKAzTISgNLkQzF8U/fOYU1hbNNLPG1NZpRZnIzb9tAbxqv1zZXZB/2XrMQDAmDe
WqY5H0sIoCzrDRAgqLxJJN810clE+LNx5GWo9W4yk9B3UVl2+UjbBlFXqm17KWNAZhrEaSu1MqhM
DwFw/dOWZSyuqVYFNALB2/ymH95ZbVaoNcfOA0A+GCZ/qKqUXdIK8CCzjARQnbuKZc
2c2fozu2C4jI89s0Wx/7rv+/GtTNXuXzGiEtiRVXNBJoFnWEYy/HZmoMubWYm5yKozy0nBgOANxb
u0f2MEfGV3RS/nNzf0DiMPCSsAgs6w0QI+wtcPU7MXBT15dRGBL1SEvQ2KfVU7Zd0aeLzOa0CCzr
DRDDBjhOVuSA9CYDRGbq9T4yNMLJTKro0r4/qauGwpWthQWcYxlLM3XLUsZ3VqqFj20yTi56p47M
1+rVF/zq4jdvPaJFNLjG2KMRF21BWWY3WT8xDQXG5qt9dQ+RzspcLwzAW4bqZq3Dv5xsc++/c0Mu
xbcBL0Ge0Xi2As76olnrx9uyPVZL4/rjpMNbpLOICTtGPthFAQM6JYpc+Dwxv63hw8QydYRjLsGa
/Whib1HcG5phpQ6/JYmS9OJv9M5I55cEvN+Ga9+xeMNXVAkcUJfJemb8LABATFaV681Dy8KgOsEl
BSiVl7lMDWA0WdIaJYMx0W6xyY+rQm7nXjZVn6K7H/rN8PwZMX4Qlu/N1M0Zq+fi2PgCAXGkReNI
Hq2s28DCGS9AxTAShzX6odS30J+78xat0gofOldtn0Ut256NrWpLq2NI99pzqt3y0FiM6pnq97rA
O9j6nz1XUaLxWgGfoDGNhtMFE2shQM7Pnugu515uF1421m1xOlZTjopcXq44dVphbFmrS4daPdz8
nvWNQpuGxWgUWdIaxMNpsh9p0tmZ6ucgml4WPDFG1a8P7Z1zXA82T7Mm7+rVu7HIeTxkbX7+uh2p
/6qWdHdtyegPAnmYgEmBBZxgLoxVPraCbaUOXJ+IJsdHo0CTR0V5VLVT7E3qmIVbKy17TzIhaO3r
7JvV0++mlGbAiLOgMY2FOnVP7Zmtn5GbO0OWHSRQBuxSl4KqqhWofgEPQyyqr0CYlwfA1tNWLZNN
NpMKCzjAW5uVfd6n2dyiKPwDm2tDlGbrWNVKviIUtihBjI5RXVmOvZGK5qF2y12toy8/J3jKRiiF
BJ6LRRLSLiHKIaIqbPtcS0Q4i2k5En/t3mAzDmIGZfuiye6L2oaH0C5+hsIHH2qKQf7bMsb/x4Bm
cLdX3VGlSPw4AMKCNWvR5hu4FIrIBeAfAGACdAUwios6aPu0APA5goBCiC4C/mTBWhmH8jFkz9NK
KKuSdttfztEURruyZ5jimfIZMULTHxdjw9XpnRsbiskq8Ol/9hiEzsU9LAEDTJHX1Iu0MPdGDF4w
VMTJD7wsgRwixTwhRDuBLAOM1fe4E8I4Q4jQACCEiq9Q2w4QpZs3QO0791VFwgoiQqohOlRc+J/V
tofpMrE4eglmr9HO/DFKYYx67pINjWztDv0LxwIgEjAh6GoBDiv08qU1JewDtiWgFEa0motF6JyK
iyUSUTUTZ+fn5vo2YYRi/cWUv+59yamKcadeIIkDA6W0jC3rfzEaqfkYKRMv0yXB+9t5hbfHaNd3
RoUmii6A/Oa4TAHO9eUIJfy2KRgNoB2AogEkAPiCiBtpOQoj3hRBZQoisJQU7WGGYQJM3dhojOr
cBI0SYk27RhQRLruguWP/5o/WAgBsUb7Jz60DMlzaruqdjvkPDXZ544iLtgu8gcp2lsDIHT0MQPl
ulC61KckDMEcIUSGE2A9gN+wCzzrnygiQ7U8jaJMpAXYbehd05LwxkR1EFB0DQz4Q9o
7J4BDOvBk0B1GBH0dgHZElElEsQAmApij6fMD7LNzEFEy7CaYfX4cJ8MwPmAkn0lUlLqAdG257n1
1EI88adYGNWn3PdGyUV0MbGuPIq1pYethHVIQFx0ZHtpev6UQohLAfQDmA9gJYLYQYjsRPUdEl0v
d5gM4SUQ7ACwG8JgQ4qRZg2YYxn+Qn2foh06pZ+hy8JJWiA+dOqfa97SAmZIYh/NSAq+auiamJMa
ZalIKJQz59Agh5gGYp2l7SrEt4Sc8Listc2mRB19q3d2uiRT1NvG++sBXmbz8
GwGkXN4otKspt5kerERnvIQwT4ciz39FdmrociyLXnC9G2Ha4ENsOF6razuiYeNxZVlxMMApFf2C
EegkuMT7GYRaq6VpqjI3c5ma3GizoDBMBvH5dD+x/cSzevbGXyzH7omjNz3npW8tx6VvL8XW206t
Zr36nO1937UxbmVfm4Yvbq47ZogjPX9EVWa0aom2qfgIud9iiyFEJyeqwoDOMhemaVt+xmEhEuuJ
KPs7QZV77bbdjuybZEm8fqM5XXuFlFt2rZUN8c/eAGptcYmxRXs9tFVjQGcbCVFUDdWI8L5VFEfn
NT9tdUQs94mLU8vPdBq03tH+wRVHEFIpmQWcYi1JQXIadR4/Ps5uerDV1GGeFZWmG8ILM358M
nxnYEAPz5T93Ac8PERBEqqgREBEQXsaAzjEW57/MMQIaa0b1eRzR4G5Pkp+1J1N9ZAEq+OqrIhKbvt4ndvPaa0
```

```
9yfU8pxuYPLgNcqePQ3xM7TIolkkpdkvKjT9swpXISkXGMBHE6n32smubDp3x2I9qsChaWVWN2z/
Jdmlfl3sKRwtLUak40aS+LbF6n/twFG21IbOYucQe4zhrZS7uHdY2INcMFizoDBPhRBEMmyMqdZS
/Xlw0rnnPtcTbi1d2q9E4fr5/EC59a3mNPlMTXpm/y/KCziYXholwDpw8h4Licq/91h84jcOaPC0
AcImOb7svaHOb+wvZyycS4Bk6w0Q4y3MKANhrfXrKr3LVuyt12+XqQbXFrPqmY7o2w4qcyMhEwjN
0hmEA+O7pomd/H9vNddbuTa7NMqlf1t2eundCj+ZeeoY/LOgMY1G0FYG84aug69nf/31Db5c2b94
qZlVPSqoTg3px0V69aqwACzrDWJCTxWXIO223d+sVhNDDUzClu8CcurE2F0+Wl6+6QLfvRe2S8dS
lnXWPAebN0AF7fdIPl+837wIhAgs6w1iQ3tMWYNmeAnRsmohnLu9i6DOeZugVVfpqbyPC5jx1gq4
GbopqEBFuH5SpeyySqEl6hJrCgs4wYURFVTU+WbEfW/I8+5bL1CQox5PBRc9dsUvz+jhbVunSHuN
jMYnEePuDYLyFbd1fZx9C+3/8goMnz3nv7APs5cIwYcST32/F7Ow8AEDu9HFe+9cxIOh/G9kOMxb
s8TxD15lVupu1x/hYKxQw9p1qQ3llNWKDWL1o3tajAICc/LNo2biu38/PM3SGCSN+2XrMax/lK31
8jPc/cXlm7GlNVC9bobush95yxwQTbVGNQGN2NhkWdIYJI/RMHFqUVYPqGCjXJsuvp2hRvYo/7ma
6vppcAoE/S+2FImxyYZgwYMnufNzy0VpDfZUeKfEGcofL3oKetK5Sx7wSa9MX7mgv7irZ/xgZtII
TZgUvGUW+x2Z9/9B9lDIM4+Cxrzcb7ltW6cwqGG9ghv7VOnvFoewDp3WP5xaUYMaCPS7t7mbo5CW
EKLlenGlh/t7wFAkbCJbszgcATP50vSnnZ0FnmDDghJt0tXoUlzkF/YhO7hUtfx6z25VzThSr2oU
QOF5Uips+WoPvN7oWn3DnfidMtxTXnHdvsJfeC5VSdN7eYnyFBZ1hwhR3Xibfb8hzbB8vMv4g0Ar
xp6sPoN8LC3HolP5Dwd0su11qouFrBgrZfVOv5mmgKCqtQGZyAgDgu3sGmHINFnSGCVPW5Z7Sbe/
UrL5ju44BLxcZrQ191V7PCa2S6ugHEBlZiA000ZLnjZ4/faB4ZPZm7C8oAWBeZkkWdIYJU9wtSp5
XlIFrUDfW5/NvPOg5eCkl0ZkbJatVQwDAaD+l0vU30ZJvfDBNLr/vOO7YNmtxlgWdYUKI6mqB8wZ
LpZ0q0c9hXlzqdG2syWx51spc9Xk8uEj+c0JXTB7c2rE/86be+PzOfpgxsYfh6wUS5ww9eCYXJWY
tzbKgM0wI8dzPO9DpqV8NVamftSpXt7243CnENgMzQTmfuXbh9ZqsdLefual/K8Qo3hCICAPaJNe
6/qdZyIuQwZyhK01UPENnmAjgm/X2Bc3C8xVe+17VS19w5RqaAHBhG+/VeqZfqZ8dcW9+iUvblb3
S0EzH/hu6saF2HCaXINrQlf+m0SZF03JgEcOEELKftOzBMur1JS75WG4fmImPVuz3Ootf/n/DkNa
gjtdrJsbry8BSyWdaybQJXVE31rV/kON1vOIwubjxDDIbbem+GDfrH7WFZ+gME0LIgl4qLWzuPl6
sSk/78MXtcdcQu+26zEsa1vSGdQ0VjYiqgU+0u8jTIE58DRETZC+XQ6fU2RVZ0BkmApAXOoe88of
u8QdGtHNEf5ZWuC6eylGi7lwK9aiJPVcr/g3ryom9QlvRbQ6TS3Bm6O/+sVe1b1bEKptcGCZEqXY
zm4yTQu61M/Q3F+7Bv37fDcCYDV6mNtIivwGEtpw7F0X1koyZybnySsxYsAcr9xYE5HqGZuhENJq
IdhFRDhFN8dDvKiISRJTlvyEyTGSgneW6M6nI/ufa47KY1/i6Pn3KjjNTYy1OEgBkG7oR7yF/MvW
H7Xh/6b6APUi8CjoR2QC8A2AMgM4AJhGRS2FAIkoE8CCANf4eJMNEAlr7rtak8vLVdm8UIkJcdBT
KdEwuvuDOXDK4fYpq/4d7B7r0CfXFUBlnYFFgTS47jhYF9HpGZuh9AeQIIfYJIcoBfAlgvE6/fwJ
4CUCpH8fHMBGDNtnVxkPq7IepisjMsspqjwm7Hh3V3vB13c0dtSafHi0auPR5/boe6N2qIRol+B6
RGgiCtSi6U0fQL+tuXok9Izb0NACHFPt5APopOxBRLwAthBBziegxdycioskAJgNAy5Ytaz5ahrE
wWhPK7Z9kq/a16Wq/33gYTZPisevYWXx0ax/VsT4ZjQxf1525RLmA6M798aJ2KbioXYrusVDCFgK
BRQCw7dlLUC/OvKXLWp+ZiKIA/AvArd76CiHeB/A+AGRlZYW41Y1hAou3avDROrU6Ze8JrZ9zeiP
j9SrdmVyUdt9pE7oaPl8oIrsJ6pXSC+w4zLVRGTG5HAbQQrGfLrXJJALoCuAPIsoF0B/AHF4YZZi
aoSxMUVNW5qi9KIwEFMm4s0Iobfg9W7qaW8IJeYZeFeAZupy0TKY2BbSNYOTs6wC0I6JMIooFMBH
AHPmgEKJQCJEshMgQQmQAWA3gciFEtv7pGIbRY8HOEx6Pe6qH+dg3W3y+buuUBN12paCbFQgTKBx
ui4qn15p9JzHopUVe34xqQ4km0VpNgrh8weu/khCiEsB9AOYD2AlgthBiOxE9R0SXmzo6hokg/vn
zDo/HjbrcfV/D4gnJ9eJ025U2fXfl5sIFIkJ0FKFKMrlsO1yI695fjbzT53HZW8tNu+6e42dNO7c
ehmzoQoh5AOZp2p5y03do7YfFMJHLwxe31/UpN+KhcedFmejZsqHXfkYorXAKulkl0wKJLYoci6K
v/rbL0b7LRNFV/ptdbqJ3i0x4P3YZxiIsUSTCuvSCZrp9+mV691zx1S1vZKcm6KyodARA5eduJCd
MqBNji3Is9BpJK1xbtD7v06/qZvo1WdAZJgS4c5ZzyalefLSqjJyMkVzj7tIFeCM6ilRuiiVllTh
bVolmSfH49m5z6l8GGpvC5JJa35kCeICBFMO+0PbJXxzbozo30c1S6W9Y0BkmxEiMi9ENSFFy99A
2uu2+ztCjbaT67JuL9gAAjhaWoncr/5hwgk2MjRyLoj1aJDnaE0z0CweA6Vd2w/s3B8bpjwWdYYL
IiaJSvL1oDyb1dXoGx+sUdr5HI+CD3QTz+JqrJFphXwaA1sn6ni/hDBGhTFoXUC74avPN+4Ojhc6
4gPIAphvgbIsME0CqqwW2HSnEBel2v+6+LywEAHRPd84Ytfbqb+++EL00C53uKhGd9zG/S7QtSvU
waF4DP/ZwIf9sGb7dkIfXru2Op37c7mg3I6XudxucoTrKlA1mwzN0hgkgL8/fhcvfXoHs3FOqdrm
Y8/X91Ckxpl7aGb1bNdJdlLy4cxOXttFdmvo0rugoclRJAgCS8ii6K3NnFdqm1jM9E6I2yZmZsKA
zTAB5b4k9VH/ToTOq9tX77AL/vBRiv+WZUXj5qgtwx6BMt+d69Zru6JrmXDxd++QIjOmm7yHjjfI
qe7IvOcimSgpi0j5grEbdWJvqQeYvlAvYgVgMlWFBZ5gg0KFpom4OFXkmXj8+Btf2aeFyXElSnRj
8fP9Fjv3YWkRzyiaCr9YdBOB0ubOC/7ke9eKiER1Fkiuj74L+85YjOFHkmmBWPueDI9r5fG5fYBs
6wwQBGxHOlfsnn7mMP3zFK6oEzpwrx7M/2aNWzapOH0yOFp5HcVklmtSPs3u++GByeWX+n3hnsbO
sXO70carjcoHt2z28YZkBz9AZJggQEUrKK/16zprUEXVH7skSPPfzDhyUihqHew4XJVf0TAMAXPj
iIgDA8aIyFJ6vxNr9pzx9TBelmAPAfZ9vUO13aW43hfnj36QmWOdfi2HCCCEEzvt5hu4PThaXo6C
43LFvVjHjYNCqsWtKYdnfv7ZFrn/ectSxXVFVjY0HzwTUu0WGTS4MEwQqqwW+XZ8X7GG4UC0EtuQ
5F2zNTvcaSDy9bZRWVDs8jTxRVS1cFrSd56hCfIwN/V5YiFMl5bp9zMY6/1oME0acr6iCTSOWwzo
Ev/JPtRCqQhpWsqF7WjT25L//6vxdmJ1tL9rWaeqvuOrdlbr9Lnj2NwDOwtnBgAWdYYLAcz/tQMv
G6uCdULFXK4dhJUHXqxb08lX2wtueBP3txTn4u5Rv3lPUZ3llNU4UlQa8bqmS0PgFMUwEoLTTHj5
zHieK1EWefZWBNyb2cAhTbakfr17ESwigD7XZxOjkdI+XzCznyioxZ/MRlwyJSnILSlzahmreqvq
+sBCF5ytqOVLfsc6/FsOEOPnFagF/8Zc/Vfu+rsuN75Hm65BcyMpoiKV7nKl8zU5cFUi0b0A9WzZ
AXSkA6OLXlwIAjozpiLuGOPPm7FeI+KxVuarPT+rbAlvyCt1er09G4JOa8QydYQLAufJK9H1+oe6
```

xIVJoeG09LWrDoLbJAID/+3Zr0Bb0zGbxn+oSfzFRUSgqVc+m88+qH7q7jjmLX3y8Ild1bFDbFGw
/4j4r5oc39/FxpL7Dgs4wAaDgrHuRvKl/KwCea4aazeNjOzq2mybFe+gZvhwtVEd0RtsIJWXqWID
/LN+v2n/gy41uz3fqnOcHX1LdwPqgAyzoDBN0ZF/v4Mk5VJ4th06d99AzfHlgRFvV/qOXdEC7Jok
eP3OJh2RnJ4pKVbl0QgEWdIYJAH8ec/9q3jujIXq2bBDwvB9KLBQ/5Batm2jb1Hro39pztSI9zxi
ZKCJ8d/dAXN3bNSNli0bBST9snRUPhglhlLZYLfXjY/D9PQMDOBpXrFAz1Bv149Vy5y1o6mxphSq
vuZYoIsRGRyFOx3umX6Y5Ze28wTN0hgkAyhqhE3qYX/29puiF+H98W+AX9cykp6ZIiLvZt7w4XVr
hOQtj43qxAIDbBmao2v8+ugOmSWmQAw0LOsMEgN92HHNsRyvc59Y+OSIYwzHEsA6pwR6CqbjLUyM
HD3mrZHR9X3uu+LapidjyzChH+w39Whkq6G0GLOgMEwBmZzvztihzjKcmhoZHSTA9bAKJsuC1OzO
TXG/0yBnn4nByPddEW1GKf0dlXVKtaSeQsKAzTIDp3qJBsIfgQoToOT6/sx9GdPT85vHb9uMAgFk
rDzja9OzkSpQP6WCuR/CiKMOYSEVVNYa9+oeq7Viha4Wb4BMZih4XbcPMm3rr5mQZ2iEFf+zKx6N
fb1Z5rlzTOx1fKzJjLnlsqEtZOVnEg5EyVwkLOsOYyLUzVyHvtPPV/cmxnfDF2oNBHJE+QcwnFXC
ibVGqdQwZbd6aOZuPALBXHZIFvWtafbRqnKB73m/vHoCWjVxzrgcSNrkEgO1HCpH5+FyVTY6JDDY
eVOfOvnNwa1zZy3+5V/xFJPihe2PuVmeRCmWdUOX6wv+N7gh39G7VEClBnqGzoAeA/60+ACGAxbt
OeO/MWJ7LutvdFl++2j8ZEv1Bm5R6wR5C0EiW3A8v7+50J1WaZIQAmta3L14HuqRcTWFBDwBfrLU
nx7dqBXWmZrRqnIDc6eNwbVaLYA/FgXYhz0ql57whf/cnxnZytCkXievG2hwz71BfPDYk6EQ0moh
2EVEOEU3ROf4wEe0goi1EtJCIWvl/qOFPtIXKeTE1R86qGA7UCZIfdTCwkZxLR6BbWhK6ptVHhWK
G3jqlHqZN6Iq+mY3Qoann3C/BxqvCEJENwDsAxgDoDGASEXXWdNsIIEsIcQGAbwC87O+BWoFImvV
Ymb35xVi196THPkIIPPvTdkVacVUAABb7SURBVFXb/cPbuunNBJP4GLsMVgsgMT4a2w4XYeaSfao
+3Vs0wOy/Xhi0gCGjGJky9gWQI4TYJ4QoB/AlgPHKDkKIxUKIc9LuagCu2WoYFvQAIoTADg+5qmv
DiNeWYNIHqz26H+46ftYlf3aoi4GSYC/uBZKPb+uLe4e1QfOkeKyUHtRfSTVEww0jgp4GQPnt8qQ
2d9wB4Be9A0Q0mYiyiSg7Pz9fr4ulYRt64Pjf6gMY++YyrMwpMO0aRwvdey2NnrHMpS2cHuhTL+3
kvZNFyExOwGOXdLREgjK/GnWJ6EYAWQBe0TsuhHhfCJElhMhKSQkfe6K/iAqjP+hwZ9Mhe2mw6z9
cg3s/2+C385ZVOosJN07Qn8X+4cabKdQX1JRoA2eY8MCIoB8GoFyOT5faVBDRSABPArhcCFGmPc4
Af/10fbCHEDEo63cq/Ytri7I8W4Wb5E23frxOtT+wrT2VajjN0MNnpIwSI4/hdQDaEVEm7EI+EcD
1yg5E1BPATACjhRDsbM0Eja15hVi6Jx9Ld5tj0rvvc2dJsnNlVR56Onl7Ui/M23Y05D0klFjB/OA
LKYlxqrqiix4ZEsTR1ByvM3QhRCWA+wDMB7ATwGwhxHYieo6ILpe6vQKgHoCviWgTEc0xbcQOo6G
8shrzt9vT01729nK8Mn+XSx9/FWBef+C0Y9vdDF1Lw4RY3NAvvDx5I1TP8dEtzhzwzZPi0TrMAq4
MGcqEEPMAzNO0PaXYHunncVmWE0WlSK0fGilTrcJLv/6J/yzfj68m93fbp7JaeCwn5gu7jp1FL03
RhO825Kn2P72jr1+vGQi6NK+Prs2Tgj2MoNAt3fm9K8MwwQ1HugSYiR+sDvYQLEfOiWIAwLly9ya
QCp3sejVl48HTqv3Hv9sKACgpq3S4MD48e7Oqz0Xtwm/xf+4DF6FObPi4WJpFUWlFsIdQY3gp22T
KK9VCsi+/BLd+vBaf3BZ+M7dQRV6oPF7k3i+881Pz8e3dF6J3q0Y+X+ftRTkubQNeXIgjkpiP15S
W++vg1j5fiwk+3krQhSI8QzeZ4a/94dL2x67I88E3k62H7S6K7y7Z67HfVe+uqtV19N4AjiiCi37
cdMSxPbFPC0wZ4z4zH8OYAQu6yShzYTP+RwjhyIA3umtTv5+/tKIKfZ9fgIwpc5GZop8HW4/nr+g
WsZ4iTPBgQWfCmn/9vhuF5+22Tm3+DT2qqwVe/GWnIzd9aUWVRw+Yez7bgBOSG9uhU/bsFtdmec9
sEU4+5zIcyRz+sKCbSGmFMT9lxnfe0rFry/z3dtd1ii2HCzFzyT4MmL4IJ4pK0XHqry45VwDgp81
HcOJsKRb96QyrWLanAETAyE5NPI5JTvYUbqx+YgT+eHRosIfB1ILw/OWFCSVllcEeQkQzqG0yerV
UF2Se8M4Kx/amQ/ZqQj9uPqLqU1Ragfu/2IhbPlJHfAJA3RgboryYUj68uY/H46FKcr04ZCQbNyt
ZnVAvZqEHC7qJhJ8Xa3hx/xcbPR6PiiJc46GIxGQpFUOxxj1tjrS4ufOoa7ZGIoI2rf2yvw9zbHd
uVh99M333pGFCh8T48HMCZEE3EaVpNtxCiMOBnzQza8A1wnFS35Zez3PmnFPQ1x84jX/8sM1t3+K
ySpAm00mLRnUdJcpmTOyB2Gj+s2KCQ9j98o4XleK133ahOgyiuJSLbZFQraiiqhoDpy/Ch8u8L07
Wlko3gUJCACumDMfP9w8yfK5LFN4xRee9B5P0zmjo0nZ9P/uDIzWC8ogzoUfYqcztn6zDW4tyHLk
7QhnlIychztqRdz9tPoJuz8zH4TPnMW3uTtOvV+IhKjStQR10TTMeut4k0ZmKQS86cGSnVMd2Vqu
GqB8fgw9uzlL1+euQ1lj292FoUDfW8HWZ0ORxKX6gf+vGQR5JzQk7QZcXGs+GwYJjleItonE968z
cVuQU4PM1Bx37+/KLcf8XG1WRddoIWX9SXS1wsrhmGZqT6sRgdBd9P/UqRZKt2TqVaj5UJGwa260
ZAOCYJio1LtqGFo3q1mhMTGjy1yFtsODhwXj+iq7BHkqNCTtBf/ryLgDslbgDzdr9pzDuzWW67oh
CCLy5cA/yTp9ztMmC/vLVFwAAXr+uu8fzV1WLkHd1PFlchhs+XIMnvt/qMCkNf22JS783F+4xbQy
tn5ine01A32Uwd/o4bH56FN67qTdevLIb3ruxt+p4heLBW6xJifvGxB4AgLkPDEKT+nG4ob/dtHJ
1r3Rc1Ssdm566uFbfhQlN2qYmIi46/N6qw07QOzWtDwAoOh/4GfpTP27D9iNF2Jdf4nJs1spc/Ov
33fjLrGxHmyzocsDGFT09B6Tc/NEadJz6K054yEkSbN5e7PT7XrP/lKF+ZtIoIRYvXNENHaVc419
OvtBj/0l9W2J016b45cGL8NaknrBFEQ4ronlPlahn/p2a2X9vXZonYc0TIx1/5HVibXjt2u5sYmF
CirATdNmVqLgs8JnQZP/jap3Iwmd+2gEA+PPYWUdbldRPL2pQLzpxRY69QO2/ft+taq+uFvh09QG
c92A3DhTKMazMKXBr+riql+vD68TZUlXFH3/w39v74vp+LfHujb3x0Mj26J5uzHbeqVl9XNa9Oaq
qBeZsPuJ4iCZLprEm9eOw/P+GoX2T8ClKwTBhJ+iyS1hNbbRV1QIPz96E9Qfczyq9YdSvRvbAkD1
x9AJRPly237G9au9J9Prn7479L9ep7bitn5iHqT9sw6PfbMaslbl47OvNjnD3QPOrYjH6zUU56D1
tger4FT3t9cO/3ZAHIYRqnH2fX6j6nv6gYYJ9hpyZnIAHR7bzOX+KXO39zLkKXNi6MRY8PATpDdk
mzoQXYSfo0VEEopoL+qZDZ/DdhsO1yrgn59T+dNUBVbt25rxHys/taYb+/LydyJgyF8/M2Y5JH6w
2NHOdu+Uonp6zHV+vz8M7ATJpaFH6bCvpk9EQL1zRDa9I6wUAkPn4PHR/9jccPHlO9zO+MGOB+u2
lqZ+Khfztq03ILSjB/oIStE2th8T48IsSZJiwE3QighDAwVM1E4kz55yC6Ws5MrmQwlfZh/D0j/b
gkz3Hz2Jz3hlVvzFvLAMAVFapbeh6fLIyV7ddnt0v2HFc97jsjbFyr93jxOh3qqyq9jklgadrdGm
ehOv7tUS0zfUn9cjXm5BzwmmKypgy16frA8CMBerFVn8mwRr66h8AgBV7C/x2ToYJJGEn6DI/Bhk

NEvTEHYrFyt92HFe53RlB630ya9UBPPvTdlz8+lJMfN9ehahdqrP+YHFZJQok+7LRyMGf7x/kWNw
rPF+Bo4Xn8Zf/Zuv2PXOuAu8szsH1H9g9TjIfn4e/fek5FB4AnpqzHV2eno+MKXN1BbrDP35BxpS
5LhV+zpdXOb6nHp5yf6/LPY2R/1rqdWzeWKtZhL1naJvan/PJES5teoveDBMOhK2g14Sb/rNGtf/
XT9fjie+3OlKoGuHh2Ztc2rRZ+l5SmBsGv7wY93y2AQCQW+BdIHKnj0PXtCRHJr+e//wdF764yHH
8oZHtkTt9HHKnj3O0aYsh/7DpiFcPGeWDrKhUPVMvq6xCmWTKum7mKsxamYtuT8+HEAKvzN/l1qu
lb2YjxMc4XbxeuqqbxzEAvr0lTf7U+XBb++QIPDKqQ43PoSU10dVkM8aEvOoMEwgsK+jj3lyGEa/
9gYwpc7Fsj/0VelRnddrTAdMX6X3UhdEzlmLeVu+Rqd3TnZn9TpWUOyrc9FEka1r7hHpGeEF6Em4
dkOHYj9ExWdxyYSs8MKKtY3/3tDFux/D6gt1Yve8kftt+DGWV6rcKra3/WGEpPl19AAdPnsObC/e
gwz9+dRzbcPAMnp6zHWfLKrE8pwAfrdiv+uwP9w4EAPzvjn743x39VMeu6+M9f8rxIuOBQVXVAh8
u2+cwQ029tDNSE+NNyzn+7xt6mXJehjGb8EsnBuDC1o1x/Kz7magQAtuPqDPldU9Pwrs39kabJ+a
p2vflFyOjcQJaS+3KGbCM0hXx5/sHYW9+MR780jlj/+PRoSguq4QtivDNXRfi6vfUC69dFBXUUxW
LePP/Nhgdmqrd4u4a2hqvaxb+nh2vjljTmnC2PjMK367PwzM/7cAXaw/hi7VOLxnl93lQY5K5ZIY
xM8hN/1mr2h/SPgU9WjTQvVcyvz80GK/+tgvzt9vXAKKjCGO6NYON7G8S/V9c6PHzSlbkFKjSCdw
+MMPQ53yFKw0x4UpYztCPFJ7HvvwSt6/th3VMKd/fMxC2KHIxWwx/bQk+W+P0WnlhnjoPidbNsWt
aEsb3SHPsr5gyHBnJCY7cIVqBbu0hv7S2LwCX6LTLuzd36QMAqx4f7thOjI/BrQMz3V4HsHsF/eZ
mgVXL30d7NmVM6us+Ja1MuyaJmHlTFhY9MgRTxnREzgtj8daknnj6si6OPn+Z5ZpvXI9//rxDte9
vwW2e5HzIDmqb7NdzM0wgCUtBPyC5wVVUCWRMmYu7pLzWMloPmM1Pj0KU5vVcNhkAwNQftzu231+
qzhSodHNspvjDnzahK24dkIG0BnVU/evFqV969unYz2Ojo5Bcz32E4donRuDj2/pgz/Nj8Oaknrp
9miXVcXk4ZbVyzQIoZz48rfDycXfOR0e1x7wHLlI9sPTo0cL1Ou5onVIPdw1xLl42TIh1iOaCnSd
UphR3yG6gAHBlL89j84XJg1sDAH68dyBm3tTbS2+GCV3CUtBlSiUb8a/bj6GguAyXvbUcuQUlLn7
iepVHtMKrJGPKXBTruPbNf2iwY/vG/q3wzOVdXPoQES69oJnHcW99ZhRWTnH1rpBJrR+PYR1Sde3
pnvj6rgsxvGOqqm3a3J3YfOiMakHT3az/vuHt0Ll5faQ1qIM4yawjR3xunHoxtj4zCsv+PgxNk2r
n+708x+kWOG3uTjykWHDecPA0ftx02LGvTZP70Mj2tbq2HrcOzETu9HHo3qIBEjz8Lhgm1AnrX++
BAudMPEuKWJR9iQF7Miw3abN1k3vdNjDD4bnS9en5mPuAM6e2UXsvALx9fS/8vMXua31jf9cFQrO
S/hARPrq1j8PFsuNU+yLneEXZtXeuty/47X9xLAqKy5GSGIeteYVo2VgdFblLsfD62rXOpGL+CLi
Z+8AgjHtzuWP/x01HMPXSzkiuF4cr/70SgP2hQ0Q4qQi4Gt+jOZpr3ogYhnESljP0we1TAABr9p/
02O+Knum4urd+QizZzU4p7PcNa6vqoxSdmjLjOnuWvpv6Z/h8Dl+Jj7Gp3AiVXCx5+hARUqRiDN3
SkwJaP7GzlPBKSda0BSqXSjly9ou19rYreqbhjYk9TfNsYRgrEJYz9Kt7p2Pp7nyPhRSaeTELNEq
IxYzremBA28aIIkLDurFwpxWPXVJzf+cJPdMwoaf/7b21JRTKoxER9r4w1sXj6Invtzq2e09bgNz
p4xyRof8Y1ymgY2SYcCT4f90+YGSOtupx9zZqmQk905CaGI/kenGwRRGICDNv6o2pl3ZW9evS3HV
GGQ68eKU6wEe7gBtMjMy0v84+hATpDcpKBUIYxizCcoZ+iZvKM/Xjo3H30Lbo2bKB7vGanFvpKte
qsXvXw1BmUt+WGNExFVVCoFqElqADdtPZ0t35bo8/9s0WAECblPC8/wwTaMJyhu7ObNAsqQ7uHtr
GL7UA5ejNT27rg0wPvuShTmr9eDRLqhNyYg7Yc5lrizmnNaiDhy9We7Ls5dwqDGMIQ4JORKOJaBc
R5RDRFJ3jcUT0lXR8DRFl+HugRth1/Kz3TgaZMqYjVkwZjqEdUr13ZnxGWcz5+n4tsWLKcHRv4fs
bFsNEMl4FnYhsAN4BMAZAZwCTiKizptsdAE4LIdoCeB3AS/4eqJZHR7n6I/vz1Tw+xhaSs1orsvm
pUbikSxNMk1IcaIOuONiHYYxhZIbeF0COEGKfEKIcwJcAxmv6jAcwS9r+BsAIMjkhRp1Yu/n//uF
t8fmd9uRQ/9UkiWLCg6S6MZh5U5YjmldZ9i13+ji3ayYMw6gxsiiaBkBZEy0PgFY5HX2EEJVEVAi
gMQBVpQAimgxgMgC0bOk9I58nbujXEseLSnHXkDZIiIuuUeAPE9rE2KL435NhfCCgi6JCiPeFEFl
CiKyUlJRanSs+xoYnxnbiUG2GYRgJI4J+GIAyvV661Kbbh4iiASQB8BzGyTAMw/gVI4K+osgAEwHM0fSZA+AWaqgAIuEr4U7GYZhgAbgIyHEdiJ6DkC2EGIOgP8A+JS
IcgCcgl30GYZhmABiyAAthJgHYJ6m7SnFdimAa/w7NIZhGKYmhGWkMMwDOMKCzrDMIxFYEFnGIaAnGIa
xCCzoDMMwFoGC5V1IRPkADnjtqE8yNFGojC58n4zB98k4fK+MYeZ9aiWE0I3MDJqg1wYiyhZCZAV
7HKEO3ydj8H0yDt8rYwTrPrHJhWEYxiKwoDMMw1iEcBX094M9gDCB75Mx+D4Zh++VMYJyn8Shs4
wDMO4Eq4zdIZhGEYDCzrDMIxFCDtB91aw2uoQUQsi+pmINhLRBiK6X2o/lYgWENE26f+GUjsR0RvS/VlLRN0V17pF6r+NiG5R
S/dpCRL0U57pF6r+HiG5xd81whYhsRLSRiH6W9jOjZFzInpcat9CRKPrbMNHIb0z3+KHQUF6R2Yt2QCoNjfVXy8
1IKJviOhPItpJRBfy78kB9rvlRcB0ow1NMmTTEDNYJQdDo6b6xCZ6P1E2IiCFXCZA
6SduJAHbDXrz7ZQBTpPYpAF6StscC+AUAAegPYI3U3gjAPun/DaXthsH+fn6+Vw8D+BzAz9L+bAA
Tpe33ANwtbd8D4D1peyKAr6TtzJvLA5ApvTbswX7e5lwn2YB+Iu0HQugAf+eXO5RGoD9AOoofku
3htpvKtxm6EYKVlsaIcRRIcQGafssgJ2w9/9iUhbpnAZggbY8H8F9hZzWABkTUDMAlAH4XQpwSQpw
G8DuA0QH8KqZCROPE0LsB5AD+2/QMhBREoDwBhNbnAMBhL2IOuN4jvSLn4wF8KYQoE0LsB5AD+2/QMhBREoDBsNc0gBC
iXAhxBvx70nsVjA0M0vYLvEaS9CRXuN6A1goDoIkQ4qh06BiAJtK2vfsLpfsRVgpHDaPcJ3DPS7wU3
5A8DfAVRL+40BnBFCVEr7+yknIOQC5ybbvV7BNhnifkAPpbMUx8SUQL496RCCJEH4F8ADgI4CtBC
A1iPEflPhJuiMBBHVA/AtgL8JIYqUx4T9VclWwuhAu9HrXhhYN/QxuAV+WQ9Wh5eSJSaYT93S5i/VGJ6F1MjGHkJkDyxQXCW0SdYnXdwI
g0n9PACCtIYyH/QHYHEACQvANJNwE3UjBastDRDGwi/lnQojvpObj0qsvpP+fkNrd3TMr38uBAC4
nolzYzXLDAbwBu3lArtKl/L7uipxb+R7J5AHIE0Kskfa/gV3g+fekZiSA/UKIfCFEBYDvYP+dhdR
vKtwE3UjBaksj2eH+A2CnEOJfikPKQt23APhR0X6zN3ZRzO+8wDwHOSATXC9o9H8AoImoozT5GSW1hjxDicSF
EuhAiA/bfyCIhxA0AFsNexBxwAFsNexBxwvUd6Rc7nAJgoeSxkAmgHYG2AvYEZiEiRCAdiSpIHTSMA7AD/nrQ
cBNCfiOpKf4PyfQqt31SwV499WG0eC7tnx14ATwZ7PEH4/oNgf/3dAmCT9N9Y2O1zCwHsAbAAQCO
pPwF4R7pfWwFkKc51O+yLMjkAbgv2dzPpfg2F08ultfTHkwPgawBxUnu8tJ8jHW+t+PyT0r3bBWB
MsL+PSfeoB4Bs6Tf1A+xeKvx7cr1PzwL4E8A2AJ/C7qkSqS8pDv1nGIaxCOmcmEYhmHcwILOMAxHR/bQMA

```
jEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWgQWdYRjGIvw/IlbURK54GdgAAAAASUVORK5CYII
=\n"
          },
          "metadata": {
            "needs_background": "light"
          }
        }
      ]
    }
  ]
}
```

DATASET WITH SLIDING:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": [],
      "toc_visible": true
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 2,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().any()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "RAUyZB0-Jp0t",
        "outputId": "3b25d177-1f91-455b-9f51-25b1c7debe06"
      },
      "execution_count": 3,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             False\n",
              "Closing Value     True\n",
              "dtype: bool"
            ]
          },
          "metadata": {},
          "execution_count": 3
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "yorF39lCJsyV",
        "outputId": "e7e4bf8b-9225-4971-fdb5-fbc6fc762cd1"
      },
      "execution_count": 4,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    7\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 4
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.dropna(axis=0,inplace=True)"
      ],
      "metadata": {
        "id": "g3FWuWVPJwms"
      },
      "execution_count": 5,
"outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "vIHbyOs4JzIf",
        "outputId": "dd09c217-307c-42cf-bad2-ff1789b31bb0"
      },
      "execution_count": 6,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    0\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 6
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil=data.reset_index()['Closing Value']\n",
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "IoxUNwrvJ2b6",
        "outputId": "21724353-c459-4b38-e5f7-88a335ddbc09"
      },
      "execution_count": 7,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "0       25.56\n",
              "1       26.00\n",
              "2       26.53\n",
              "3       25.85\n",
              "4       25.87\n",
              "        ...  \n",
              "8211    73.89\n",
              "8212    74.19\n",
              "8213    73.05\n",
              "8214    73.78\n",
              "8215    73.93\n",
"Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "metadata": {},
          "execution_count": 7
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from sklearn.preprocessing import MinMaxScaler\n",
        "scaler=MinMaxScaler(feature_range=(0,1))\n",
        "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
      ],
      "metadata": {
        "id": "5m-DUFI9J_WN"
      },
      "execution_count": 8,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "fyMwOo1jKLqL",
        "outputId": "0941fd8f-2bc0-4dea-f85d-d92bfcee4d2e"
      },
      "execution_count": 9,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "metadata": {},
          "execution_count": 9
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "plt.plot(data_oil)"
      ],
      "metadata": {
        "colab": {
```

```
"base_uri": "https://localhost:8080/",
          "height": 282
        },
        "id": "GdNJartuKUfk",
        "outputId": "12f7d47d-525b-4282-c77a-b403c5e39639"
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "[<matplotlib.lines.Line2D at 0x7fe7c7b0e2d0>]"
            ]
          },
          "metadata": {},
          "execution_count": 10
        },
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXQAAAD4CAYAAAD8Zh1EAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO2dd3gU1frHv282DUIIJQklARJ6lRa
KgHSRooId7OXKtXtt94d6sV1UbFdsV1GvyvXasKOgKEV6C70JBAgQakJJSCD9/P7Ymd2Z2dndyWZ
ny+z7eR4fZ86cnTk7bL5z5j1vISEEGIZhmPAnKtgDYBiGYfwDCzrDMIxFYEFnGIaxCCzoDMMwFoE
FnWEYxiJEB+vCycnJIiMjI1liXZxiGCUvWr19fIIIRI0TsWNEHPyMhAdnZ2sC7PMAwTlhDRAXfH2OT
CMAxjEVjQGYZhLAILOsMwEVgQWcYhrEILOgMwzAWwaugE9FHRHSCiLa5OU5E9CYRZRERFiLa5OU5E9CYR5RDRFiLq5f9
hMgzDMN4wMkP/BMBoD8fHAGn/TcZwLuxT7wKuhbiKYBTHrqMB/FfYWc1gAZE1Mxf4
Yxhw+XLYPeafPBXsYjB/xhw09DcAhxX6e1OYCEU0momwiys7Pz/fDpRmG8YV9+cWYNncnr2ONh
DYfxIQBdFhRDvCyGyhBBZKSm6kasMwwSAc+VVwR4CYwL+EPTOAMoo9tOZqhlRmSfwh6HM
A3Cx5u/QHUCiEOOqH8zIMGVLH4GkAMQAgJ8GeJhPwDwAYwHkADgH4Da
zBsswjH/gGbo18SroQohJXo4LAHf4bUQMwzCMT3CkKMMwjEVgQWcYhrEILOgMwzAWgQWdYSKSm6kasMwwSAc+VVwR4CYwL+EPTDAFoo9tOlNoZhQpRqrlRmSfwh6HM
A3Cx5u/QHUCiEOOqH8zIMYxJV1SzoVsRrci4i+gLAUADJRJQH4GkAMQAgJ8GeJhPwDwAYwHkADgH4Da
zBsswjH/gGbo18SroQohJXo4LAPf6bUQMw5hOVXWwR8CYAUeKMkwEwiYAaZMwEwiYXa8KCzjARCAu6NWFBZ5g
IpIpt6JaEBZ1hIpCVOQBHgJjAizoDMBOBzFy6L9hDYEyABZ1hGMYisKAzDMNYBBZ0homwgAQ
MYxFY0BmGYSwCCzrDMIxFYEFnmAhDKIKWicnBHEkjL9hQWeYCGP+9uONCON
saYVj+3wFVy6yEizoDBNhxNicf/Z5p88HcSSMv2FBZ5gIg9jKYllY0BmGYSwCCzrDRBjEU3TLwoo
OMBEGy7l1YUFnmAgjSjHCFT4tQJ4gjYfwNCzrDRBhKi8Ka7FORbT2UXMmYvsGjrlOfnBGg7jR1j
MqT9sU+0Xnq9w05MJN1jQGSbCOFrrtrngtHWHWQdYYSKcqmoWdDwDKbyk+M4j2bwu8+FhtQVlE1
p2t9fUILyyuogjYBWbzqnyO5MJeywcU9/tzBsG7jR1j
QGSbCqBYCDCerG4L5hbR1txAkBLAELOsNEGCyyBFXOnkEy9HYkGrgYn2QjWIGLL1MiLCA5cwTHkRCrdmcUSXhXijGhhBzgE1Qpkvg10dL1Oa
IthDRWP8PlWEYf/DZmoM4WVKOTYfsojAGxlBm4JbIUjGZHeQpbfRHxAEP9gpCBwVl9gY5M0IoShQdA4cJM4ViKc
t74EyDONf9pwoDvYQGD9jZIbeF0COEGKfEKIcwJcAxmv6v6CAD1pe0kAEf8N0SGYcxgQJvGjm1OkW4
NjAh6GoBDiv08qU3JMwBuJKI8APMA3K93IiKaTQohllhvGUkqnkX66NMMwR1GG+SvXRHmCbg2MCPw
C8IkQIh3AWACfpHLuYUQ7wshmoQkRgoTBsCaxwQDzTBfWAcYdo
FBXwegHRFlElEsEs7IueczR9DgIYAQBE1AlWCsOEGOV
VzhD/M5w213J4FXQhRCWA+wDMB7ATdm+lQZwBcAbhWCU/4wTKhRocjZopy
fc+FoaxBtpJMQYh7si53KtqcU2zsADPTv0BiG8TdKk0uMjeMKrQb/izJMBKHMqhioSflnaw7g
HAnOxCMfQDJ1hGGtQobChN0uqmEvE5BrPvm9veTdTf1bBeR6kQkhnXm/
JbvaNCCQs6AwTgQ2kTugIA+mQ0BAAcPnPelOutzCMPizoDBwMBfLcwN78EQHj7ov+
46TC+XZ8X7GGGEFxDZ5gIg4uoWLsIOuMwzAWgQWdYSKSm6kasMwwSAc+VVwR4CYwL+EPTDAFoo9tOlNoZhQpRqrlRmSfwh6HM
ozdZzb8EopRVVuOCZgZ+fh9BxIuLO/nsOX7X7W4/
zctyiPizoDBMhKF0g+bFkYZjvkIASj0sclxmRYWlbigAcf
4ybbzGFuwbejO6x8+bcztTKTAb8h8+bczTptKA8F/x75XYdrgI2bE3oDE0aJ+xzrwJwq3bPzR
vuLRSta9889Cy6M8Thq9vBVjQGSYCGS3mNavsb6irSbLv4vDgIbmRFsHtWZ2nYhuu9DCe
3fQ8YOJ+VYEFnmAhDKIKWicnBHEkjL9hQWeYCGP+9uONCON
fbsjDuXL1jN1m0Dsc7x6Ou0L9yhQ5Wvow7HHtu/rUkp7/+g4OTgxQJ2Q7zgy5kyAZ9ZSjqRQQ4ZhvFIDjTBLiU9MXDbYI0IhH8fqEgbFtFUT0E1PrcTsJ5Tu/MZwNuKjQDOhfCqbZqZuvA9mMq2pphFQBhZdAfDpUQ7D/HrpPyKEwvCoS
```

WqY5H0sIoCzrDRAgqLxJJN810clE+LNx5GWo9W4yk9B3UVl2+UjbBlFXqm17KWNAZhrEaSu1MqhM
DwFw/dOWZSyuqVYFNALB2/ymMf2eFqq3agH95ZbVaoNcfOA0A+GCZ/qKqUXdIK8CCzjARQnbuKZc
2c2fozu2C4jI89s0Wx/7rv+/GtTNXuXzGiEtiRVXNBJoFnWEYy/HZmoMubWYm5yKozy0nBgOANxb
u0f2MEfGV3RS/nNzf0DiMPCSsAgs6w0QI+wtcPU7MXBT15dRGBL1SEvQ2KfVU7Zd0aeLzOa0CCzr
DRDDBjhOVuSA9CYDRGbq9T4yNMLJTKro0r4/qauGwpWthQWcYxlLM3XLUsZ3VqqFj20yTi56p47M
1+rVF/zq4jdvPaJFNLjG2KMRF21BWWY3WT8xDQXG5qt9dQ+RzspcLwzAW4bqZq3Dv5xsc++/c0Mu
xbcBL0Ge0Xi2As76olnrx9uyPVZL4/rjpMNbpLOICTtGPthFAQM6JYpc+Dwxv63hw8QydYRjLsGa
/Whib1HcG5phpQ6/JYmS9OJv9M5I55cEvN+Ga9+xeMNXVAkcUJfJemb8LABATFaV681Dy8KgOsEl
BSiVl7lMDWA0WdIaJYMx0W6xyY+rQm7nXjZVn6K7H/rN8PwZMX4Qlu/N1M0Zq+fi2PgCAXGkReNI
Hq2s28DCGS9AxTAShzX6odS30J+78xat0gofOldtn0Ut256NrWpLq2NI99pzqt3y0FiM6pnq97rA
O9j6nz1XUaLxWgGfoDGNhtMFE2shQM7Pnugu515uF1421m1xOlZTjopcXq44dVphbFmrS4daPdz8
nvWNQpuGxWgUWdIaxMNpsh9p0tmZ6ucgml4WPDFG1a8P7Z1zXA82T7Mm7+rVu7HIeTxkbX7+uh2p
/6qWdHdtyegPAnmYgEmBBZxgLoxVPraCbaUOXJ+IJsdHo0CTR0V5VLVT7E3qmIVbKy17TzIhaO3r
7JvV0++mlGbAiLOgMY2FOnVP7Zmtn5GbO0OWHSRQBuxSl4KqqhWofgEPQyyqr0CYlwfA1tNWLZNN
NpMKCzjAW5uVfd6n2dyiKPwDm2tDlGbrWNVKviIUtihBjI5RXVmOvZGK5qF2y12toy8/J3jKRiiF
BJ6LRRLSLiHKIaIqbPtcS0Q4i2k5En/t3mAzDmIGZfuiye6L2oaH0C5+hsIHH2qKQf7bMsb/x4Bm
cLdX3VGlSPw4AMKCNWvR5hu4FIrIBeAfAGACdAUwios6aPu0APA5goBCiC4C/mTBWhmH8jFkz9NK
KKuSdttfztEURruyZ5jimfIZMULTHxdjw9XpnRsbiskq8Ol/9hiEzsU9LAEDTJHX1Iu0MPdGDF4w
VMTJD7wsgRwixTwhRDuBLAOM1fe4E8I4Q4jQACCEiq9Q2w4QpZs3QO0791VFwgoiQqohOlRc+J/V
tofpMrE4eglmr9HO/DFKYYx67pINjWztDv0LxwIgEjAh6GoBDiv08qU1JewDtiWgFEa0motF6JyK
iyUSUTUTZ+fn5vo2YYRi/cWUv+59yamKcadeIIkDA6W0jC3rfzEaqfkYKRMv0yXB+9t5hbfHaNd3
RoUmii6A/Oa4TAHO9eUIJfy2KRgNoB2AogEkAPiCiBtpOQoj3hRBZQoislJQU7WGGYQJM3dhojOr
cBI0SYk27RhQRLruguWP/5o/WAgBsUb7Jz60DMlzaruqdjvkPDXZ544iLtgu8gcp2lsDIHT0MQPl
ulC61KckDMEcIUSGE2A9gN+wCzzBMEGlYN8ZrnygiQ7U8jaJMpAXYbehd05LwxkR1EFB0DQz4Q9o
7J4BDOvBk0B1GBH0dgHZElElEsQAmApij6fMD7LNzEFEy7CaYfX4cJ8MwPmAkn0lUlLqAdG257n1
1EI88adYGNWn3PdGyUV0MbGuPIq1pYethHVIQFx0ZHtpev6UQohLAfQDmA9gJYLYQYjsRPUdEl0v
d5gM4SUQ7ACwG8JgQ4qRZg2YYxn+Qn2foh06pZ+hy8JJWiA+dOqfa97SAmZIYh/NSAq+auiamJMa
ZalIKJQz59Agh5gGYp2l7SrEtADws/ccwTBgRRQT4Sc8Listc2mRB19q3d2uiRT1NvG++sBXmbz8
GwGkXN4otKspt5kerERnvIQwT4ciz39FdmrociyLXnC9G2Ha4ENsOF6razuiYeNxZVlxMMApFf2C
EegkuMT7GYRaq6VpqjI3c5ma3GizoDBMBvH5dD+x/cSzevbGXyzH7omjNz3npW8tx6VvL8XW206t
Zr36nO1937UxbmVfm4Yvbq47ZogjPX9EVWa0aom2qfgIud9iiyFEJyeqwoDOMhemaVt+xmEhEuuJ
KPs7QZV77bbdjuybZEm8fqM5XXuFlFt2rZUN8c/eAGptcYmxRXs9tFVjQGcbCVFUDdWI8L5VFEfn
NT9tdUQs94mLU8vPdBq03tH+wRVHEFIpmQWcYi1JQXIadR4sQY/Ps5uerDV1GGeFZWmG8ILM358M
nxnYEAPz5T93Ac8PERBEqqgREBEQXsaAzjEW57/MNAICjhZ5D6msaWLToz+OqrIhKbvt4ndvPaa0
9yfU8pxuYPLgNcqePQ3xM7TIolkkpdkvKjT9swpXISkXGMBHE6n32smubDp3x2I9qsChaWVWN2z/
Jdmlfl3sKRwtLUak40aS+LbF6n/twFG21IbOYucQe4zhrZS7uHdY2INcMFizoDBPhRBEMmyMqdZS
/Xlw0rnnPtcTbi1d2q9E4fr5/EC59a3mNPlMTXpm/y/KCziYXholwDpw8h4Licq/91h84jcOaPC0
AcImOb7svaHOb+wvZyycS4Bk6w0Q4y3MKANhrfXrKr3LVuyt12+XqQbXFrPqmY7o2w4qcyMhEwjN
0hmEA+O7pomd/H9vNddbuTa7NMqlf1t2eundCj+ZeeoY/LOgMY1G0FYG84aug69nf/31Db5c2b94
qZlVPSqoTg3px0V69aqwACzrDWJCTxWXIO223d+sVhNDDUzClu8CcurE2F0+Wl6+6QLfvRe2S8dS
lnXWPAebN0AF7fdIPl+837wIhAgs6w1iQ3tMWYNmeAnRsmohnLu9i6DOeZugVVfpqbyPC5jx1gq4
GbopqEBFuH5SpeyySqEl6hJrCgs4wYURFVTU+WbEfW/I8+5bL1CQox5PBRc9dsUvz+jhbVunSHuN
jMYnEePuDYLyFbd1fZx9C+3/8goMnz3nv7APs5cIwYcST32/F7Ow8AEDu9HFe+9cxIOh/G9kOMxb
s8TxD15lVupu1x/hYKxQw9p1qQ3llNWKDWL1o3tajAICc/LNo2biu38/PM3SGCSN+2XrMax/lK31
8jPc/cXlm7GlNVC9bobush95yxwQTbVGNQGN2NhkWdIYJI/RMHFqUVYPqGCjXJsuvp2hRvYo/7ma
6vppcAoE/S+2FImxyYZgwYMnufNzy0VpDfZUeKfEGcofL3oKetK5Sx7wSa9MX7mgv7irZ/xgZtII
TZgUvGUW+x2Z9/9B9lDIM4+Cxrzcb7ltW6cwqGG9ghv7VOnvFoewDp3WP5xaUYMaCPS7t7mbo5CW
EKLlenGlh/t7wFAkbCJbszgcATP50vSnnZ0FnmDDghJt0tXoUlzkF/YhO7hUtfx6z25VzThSr2oU
QOF5Uips+WoPvN7oWn3DnfidMtxTXnHdvsJfeC5VSdN7eYnyFBZ1hwhR3Xibfb8hzbB8vMv4g0Ar
xp6sPoN8LC3HolP5Dwd0su11qouFrBgrZfVOv5mmgKCqtQGZyAgDgu3sGmHINFnSGCVPW5Z7Sbe/
UrL5ju44BLxcZrQ191V7PCa2S6ugHEBlZiA000ZLnjZ4/faB4ZPZm7C8oAWBeZkkWdIYJU9wtSp5
XlIFrUDfW5/NvPOg5eCkl0ZkbJatVQwDAaD+l0vU30ZJvfDBNLr/vOO7YNmtxlgWdYUKI6mqB8wZ
LpZ0q0c9hXlzqdG2syWx51spc9Xk8uEj+c0JXTB7c2rE/86be+PzOfpgxsYfh6wUS5ww9eCYXJWY
tzbKgM0wI8dzPO9DpqV8NVamftSpXt7243CnENgMzQTmfuXbh9ZqsdLefual/K8Qo3hCICAPaJNe
6/qdZyIuQwZyhK01UPENnmAjgm/X2Bc3C8xVe+17VS19w5RqaAHBhG+/VeqZfqZ8dcW9+iUvblb3
S0EzH/hu6saF2HCaXINrQlf+m0SZF03JgEcOEELKftOzBMur1JS75WG4fmImPVuz3Ootf/n/DkNa
gjtdrJsbry8BSyWdaybQJXVE31rV/kON1vOIwubjxDDIbbem+GDfrH7WFZ+gME0LIgl4qLWzuPl6
sSk/78MXtcdcQu+26zEsa1vSGdQ0VjYiqgU+0u8jTIE58DRETZC+XQ6fU2RVZ0BkmApAXOoe88of
u8QdGtHNEf5ZWuC6eylGi7lwK9aiJPVcr/g3ryom9QlvRbQ6TS3Bm6O/+sVe1b1bEKptcGCZEqXY
zm4yTQu61M/Q3F+7Bv37fDcCYDV6mNtIivwGEtpw7F0X1koyZybnySsxYsAcr9xYE5HqGZuhENJq

```
IdhFRDhFN8dDvKiISRJTlvyEyTGSgneW6M6nI/ufa47KY1/i6Pn3KjjNTYy1OEgBkG7oR7yF/MvW
H7Xh/6b6APUi8CjoR2QC8A2AMgM4AJhGRS2FAIkoE8CCANf4eJMNEAlr7rtak8vLVdm8UIkJcdBT
KdEwuvuDOXDK4fYpq/4d7B7r0CfXFUBlnYFFgTS47jhYF9HpGZuh9AeQIIfYJIcoBfAlgvE6/fwJ
4CUCpH8fHMBGDNtnVxkPq7IepisjMsspqjwm7Hh3V3vB13c0dtSafHi0auPR5/boe6N2qIRol+B6
RGgiCtSi6U0fQL+tuXok9Izb0NACHFPt5APopOxBRLwAthBBziegxdycioskAJgNAy5Ytaz5ahrE
wWhPK7Z9kq/a16Wq/33gYTZPisevYWXx0ax/VsT4ZjQxf1525RLmA6M798aJ2KbioXYrusVDCFgK
BRQCw7dlLUC/OvKXLWp+ZiKIA/AvArd76CiHeB/A+AGRlZYW41Y1hAou3avDROrU6Ze8JrZ9zeiP
j9SrdmVyUdt9pE7oaPl8oIrsJ6pXSC+w4zLVRGTG5HAbQQrGfLrXJJALoCuAPIsoF0B/AHF4YZZi
aoSxMUVNW5qi9KIwEFMm4s0Iobfg9W7qaW8IJeYZeFeAZupy0TKY2BbSNYOTs6wC0I6JMIooFMBH
AHPmgEKJQCJEshMgQQmQAWA3gciFEtv7pGIbRY8HOEx6Pe6qH+dg3W3y+buuUBN12paCbFQgTKBx
ui4qn15p9JzHopUVe34xqQ4km0VpNgrh8weu/khCiEsB9AOYD2AlgthBiOxE9R0SXmzo6hokg/vn
zDo/HjbrcfV/D4gnJ9eJ025U2fXfl5sIFIkJ0FKFKMrlsO1yI695fjbzT53HZW8tNu+6e42dNO7c
ehmzoQoh5AOZp2p5y03do7YfFMJHLwxe31/UpN+KhcedFmejZsqHXfkYorXAKulkl0wKJLYoci6K
v/rbL0b7LRNFV/ptdbqJ3i0x4P3YZxiIsUSTCuvSCZrp9+mV691zx1S1vZKcm6KyodARA5eduJCd
MqBNji3Is9BpJK1xbtD7v06/qZvo1WdAZJgS4c5ZzyalefLSqjJyMkVzj7tIFeCM6ilRuiiVllTh
bVolmSfH49m5z6l8GGpvC5JJa35kCeICBFMO+0PbJXxzbozo30c1S6W9Y0BkmxEiMi9ENSFFy99A
2uu2+ztCjbaT67JuL9gAAjhaWoncr/5hwgk2MjRyLoj1aJDnaE0z0CweA6Vd2w/s3B8bpjwWdYYL
IiaJSvL1oDyb1dXoGx+sUdr5HI+CD3QTz+JqrJFphXwaA1sn6ni/hDBGhTFoXUC74avPN+4Ojhc6
4gPIAphvgbIsME0CqqwW2HSnEBel2v+6+LywEAHRPd84Ytfbqb+++EL00C53uKhGd9zG/S7QtSvU
waF4DP/ZwIf9sGb7dkIfXru2Op37c7mg3I6XudxucoTrKlA1mwzN0hgkgL8/fhcvfXoHs3FOqdrm
Y8/X91Ckxpl7aGb1bNdJdlLy4cxOXttFdmvo0rugoclRJAgCS8ii6K3NnFdqm1jM9E6I2yZmZsKA
zTAB5b4k9VH/ToTOq9tX77AL/vBRiv+WZUXj5qgtwx6BMt+d69Zru6JrmXDxd++QIjOmm7yHjjfI
qe7IvOcimSgpi0j5grEbdWJvqQeYvlAvYgVgMlWFBZ5gg0KFpom4OFXkmXj8+Btf2aeFyXElSnRj
8fP9Fjv3YWkRzyiaCr9YdBOB0ubOC/7ke9eKiER1Fkiuj74L+85YjOFHkmmBWPueDI9r5fG5fYBs
6wwQBGxHOlfsnn7mMP3zFK6oEzpwrx7M/2aNWzapOH0yOFp5HcVklmtSPs3u++GByeWX+n3hnsbO
sXO70carjcoHt2z28YZkBz9AZJggQEUrKK/16zprUEXVH7skSPPfzDhyUihqHew4XJVf0TAMAXPj
iIgDA8aIyFJ6vxNr9pzx9TBelmAPAfZ9vUO13aW43hfnj36QmWOdfi2HCCCEEzvt5hu4PThaXo6C
43LFvVjHjYNCqsWtKYdnfv7ZFrn/ectSxXVFVjY0HzwTUu0WGTS4MEwQqqwW+XZ8X7GG4UC0EtuQ
5F2zNTvcaSDy9bZRWVDs8jTxRVS1cFrSd56hCfIwN/V5YiFMl5bp9zMY6/1oME0acr6iCTSOWwzo
Ev/JPtRCqQhpWsqF7WjT25L//6vxdmJ1tL9rWaeqvuOrdlbr9Lnj2NwDOwtnBgAWdYYLAcz/tQMv
G6uCdULFXK4dhJUHXqxb08lX2wtueBP3txTn4u5Rv3lPUZ3llNU4UlQa8bqmS0PgFMUwEoLTTHj5
zHieK1EWefZWBNyb2cAhTbakfr17ESwigD7XZxOjkdI+XzCznyioxZ/MRlwyJSnILSlzahmreqvq
+sBCF5ytqOVLfsc6/FsOEOPnFagF/8Zc/Vfu+rsuN75Hm65BcyMpoiKV7nKl8zU5cFUi0b0A9WzZ
AXSkA6OLXlwIAjozpiLuGOPPm7FeI+KxVuarPT+rbAlvyCt1er09G4JOa8QydYQLAufJK9H1+oe6
xIVJoeG09LWrDoLbJAID/+3Zr0Bb0zGbxn+oSfzFRUSgqVc+m88+qH7q7jjmLX3y8Ild1bFDbFGw
/4j4r5oc39/FxpL7Dgs4wAaDgrHuRvKl/KwCea4aazeNjOzq2mybFe+gZvhwtVEd0RtsIJWXqWID
/LN+v2n/gy41uz3fqnOcHX1LdwPqgAyzoDBN0ZF/v4Mk5VJ4th06d99AzfHlgRFvV/qOXdEC7Jok
eP3OJh2RnJ4pKVbl0QgEWdIYJAH8ec/9q3jujIXq2bBDwvB9KLBQ/5Batm2jb1Hro39pztSI9zxi
ZKCJ8d/dAXN3bNSNli0bBST9snRUPhglhlLZYLfXjY/D9PQMDOBpXrFAz1Bv149Vy5y1o6mxphSq
vuZYoIsRGRyFOx3umX6Y5Ze28wTN0hgkAyhqhE3qYX/29puiF+H98W+AX9cykp6ZIiLvZt7w4XVr
hOQtj43qxAIDbBmao2v8+ugOmSWmQAw0LOsMEgN92HHNsRyvc59Y+OSIYwzHEsA6pwR6CqbjLUyM
HD3mrZHR9X3uu+LapidjyzChH+w39Whkq6G0GLOgMEwBmZzvztihzjKcmhoZHSTA9bAKJsuC1OzO
TXG/0yBnn4nByPddEW1GKf0dlXVKtaSeQsKAzTIDp3qJBsIfgQoToOT6/sx9GdPT85vHb9uMAgFk
rDzja9OzkSpQP6WCuR/CiKMOYSEVVNYa9+oeq7Viha4Wb4BMZih4XbcPMm3rr5mQZ2iEFf+zKx6N
fb1Z5rlzTOx1fKzJjLnlsqEtZOVnEg5EyVwkLOsOYyLUzVyHvtPPV/cmxnfDF2oNBHJE+QcwnFXC
ibVGqdQwZbd6aOZuPALBXHZIFvWtafbRqnKB73m/vHoCWjVxzrgcSNrkEgO1HCpH5+FyVTY6JDDY
eVOfOvnNwa1zZy3+5V/xFJPihe2PuVmeRCmWdUOX6wv+N7gh39G7VEClBnqGzoAeA/60+ACGAxbt
OeO/MWJ7LutvdFl++2j8ZEv1Bm5R6wR5C0EiW3A8v7+50J1WaZIQAmta3L14HuqRcTWFBDwBfrLU
nx7dqBXWmZrRqnIDc6eNwbVaLYA/FgXYhz0ql57whf/cnxnZytCkXievG2hwz71BfPDYk6EQ0moh
2EVEOEU3ROf4wEe0goi1EtJCIWvl/qOFPtIXKeTE1R86qGA7UCZIfdTCwkZxLR6BbWhK6ptVHhWK
G3jqlHqZN6Iq+mY3Qoann3C/BxqvCEJENwDsAxgDoDGASEXXWdNsIIEsIcQGAbwC87O+BWoFImvV
Ymb35xVi196THPkIIPPvTdkVacVUAABb7SURBVFXb/cPbuunNBJP4GLsMVgsgMT4a2w4XYeaSfao
+3Vs0wOy/Xhi0gCGjGJky9gWQI4TYJ4QoB/AlgPHKDkKIxUKIc9LuagCu2WoYFvQAIoTADg+5qmv
DiNeWYNIHqz26H+46ftYlf3aoi4GSYC/uBZKPb+uLe4e1QfOkeKyUHtRfSTVEww0jgp4GQPnt8q8
2d9wB4Be9A0Q0mYiyg7Pz9fr4uYRt64Pjf6gMY++YyrMwpMO0aRwvdey2NnrHMpS2cHuhTL+3
kvZNFyExOwGOXdLREgjK/GnWJ6EYAWQBe0TsuhHhCJElhMhKSQ6kf6K/iAqjP+hwZ9Mhe2mw6z9
cg3s/2+C385ZVOosJN07Qn8X+4cabKdQX1JRoA2eY8MCIoB8GoB8GoFyOT5faVBDRSrSAPArhc
Af/10fbCHEDEo63cq/Ytri7I8W4Wb5E23frxOtT+wrT2VajjN0MNnpIwSI4/hdQDaEVm7EI+EcD
1yg5E1BPPPATACjhRDsbM0Eja15hVi6Jx9Ld5tj0rvvvc2dJsnNlVR56Onl7Ui/M23Y05D0klFjB/OA
LKYlxqrqiix4ZEsTR1ByvM3QhRCWA+wDMB7ATwlRBiOxE9R0SXmzo6hokg/vn
```

8shrzt9vT01729nK8Mn+XSx9/FWBef+C0Y9vdDF1Lw4RY3NAvvDx5I1TP8dEtzhzwzZPi0TrMAq4
MGcqEEPMAzNO0PaXYHunncVmWE0WlSK0fGilTrcJLv/6J/yzfj68m93fbp7JaeCwn5gu7jp1FL03
RhO825Kn2P72jr1+vGQi6NK+Prs2Tgj2MoNAt3fm9K8MwwQ1HugSYiR+sDvYQLEfOiWIAwLly9ya
QCp3sejVl48HTqv3Hv9sKACgpq3S4MD48e7Oqz0Xtwm/xf+4DF6FObPi4WJpFUWlFsIdQY3gp22T
KK9VCsi+/BLd+vBaf3BZ+M7dQRV6oPF7k3i+881Pz8e3dF6J3q0Y+X+ftRTkubQNeXIgjkpiP15S
W++vg1j5fiwk+3krQhSI8QzeZ4a/94dL2x67I88E3k62H7S6K7y7Z67HfVe+uqtV19N4AjiiCi37
cdMSxPbFPC0wZ4z4zH8OYAQu6yShzYTP+RwjhyIA3umtTv5+/tKIKfZ9fgIwpc5GZop8HW4/nr+g
WsZ4iTPBgQWfCmn/9vhuF5+22Tm3+DT2qqwVe/GWnIzd9aUWVRw+Yez7bgBOSG9uhU/bsFtdmec9
sEU4+5zIcyRz+sKCbSGmFMT9lxnfe0rFry/z3dtd1ii2HCzFzyT4MmL4IJ4pK0XHqry45VwDgp81
HcOJsKRb96QyrWLanAETAyE5NPI5JTvYUbqx+YgT+eHRosIfB1ILw/OWFCSVllcEeQkQzqG0yerV
UF2Se8M4Kx/amQ/ZqQj9uPqLqU1Ragfu/2IhbPlJHfAJA3RgboryYUj68uY/H46FKcr04ZCQbNyt
ZnVAvZqEHC7qJhJ8Xa3hx/xcbPR6PiiJc46GIxGQpFUOxxj1tjrS4ufOoa7ZGIoI2rf2yvw9zbHd
uVh99M333pGFCh8T48HMCZEE3EaVpNtxCiMOBnzQza8A1wnFS35Zez3PmnFPQ1x84jX/8sM1t3+K
ySpAm00mLRnUdJcpmTOyB2Gj+s2KCQ9j98o4XleK133ahOgyiuJSLbZFQraiiqhoDpy/Ch8u8L07
Wlko3gUJCACumDMfP9w8yfK5LFN4xRee9B5P0zmjo0nZ9P/uDIzWC8ogzoUfYqcztn6zDW4tyHLk
7QhnlIychztqRdz9tPoJuz8zH4TPnMW3uTtOvV+IhKjStQR10TTMeut4k0ZmKQS86cGSnVMd2Vqu
GqB8fgw9uzlL1+euQ1lj292FoUDfW8HWZ0ORxKX6gf+vGQR5JzQk7QZcXGs+GwYJjleItonE968z
cVuQU4PM1Bx37+/KLcf8XG1WRddoIWX9SXS1wsrhmGZqT6sRgdBd9P/UqRZKt2TqVaj5UJGwa260
ZAOCYJio1LtqGFo3q1mhMTGjy1yFtsODhwXj+iq7BHkqNCTtBf/ryLgDslbgDzdr9pzDuzWW67oh
CCLy5cA/yTp9ztMmC/vLVFwAAXr+uu8fzV1WLkHd1PFlchhs+XIMnvt/qMCkNf22JS783F+4xbQy
tn5ine01A32Uwd/o4bH56FN67qTdevLIb3ruxt+p4heLBW6xJifvGxB4AgLkPDEKT+nG4ob/dtHJ
1r3Rc1Ssdm566uFbfhQlN2qYmIi46/N6qw07QOzWtDwAoOh/4GfpTP27D9iNF2Jdf4nJs1spc/Ov
33fjLrGxHmyzocsDGFT09B6Tc/NEadJz6K054yEkSbN5e7PT7XrP/lKF+ZtIoIRYvXNENHaVc419
OvtBj/0l9W2J016b45cGL8NaknrBFEQ4ronlPlahn/p2a2X9vXZonYc0TIx1/5HVibXjt2u5sYmF
CirATdNmVqLgs8JnQZP/jap3Iwmd+2gEA+PPYWUdbldRPL2pQLzpxRY69QO2/ft+taq+uFvh09QG
c92A3DhTKMazMKXBr+riql+vD68TZUlXFH3/w39v74vp+LfHujb3x0Mj26J5uzHbeqVl9XNa9Oaq
qBeZsPuJ4iCZLprEm9eOw/P+GoX2T8ClKwTBhJ+iyS1hNbbRV1QIPz96E9Qfczyq9YdSvRvbAkD1
x9AJRPly237G9au9J9Prn7479L9ep7bitn5iHqT9sw6PfbMaslbl47OvNjnD3QPOrYjH6zUU56D1
tger4FT3t9cO/3ZAHIYRqnH2fX6j6nv6gYYJ9hpyZnIAHR7bzOX+KXO39zLkKXNi6MRY8PATpDdk
mzoQXYSfo0VEEopoL+qZDZ/DdhsO1yrgn59T+dNUBVbt25rxHys/taYb+/LydyJgyF8/M2Y5JH6w
2NHOdu+Uonp6zHV+vz8M7ATJpaFH6bCvpk9EQL1zRDa9I6wUAkPn4PHR/9jccPHlO9zO+MGOB+u2
lqZ+Khfztq03ILSjB/oIStE2th8T48IsSZJiwE3QighDAwVM1E4kz55yC6Ws5MrmQwlfZh/D0j/b
gkz3Hz2Jz3hlVvzFvLAMAVFapbeh6fLIyV7ddnt0v2HFc97jsjbFyr93jxOh3qqyq9jklgadrdGm
ehOv7tUS0zfUn9cjXm5BzwmmKypgy16frA8CMBerFVn8mwRr66h8AgBV7C/x2ToYJJGEn6DI/bHK
NEvTEHYrFyt92HFe53RlB630ya9UBPPvTdlz8+lJMfN9ehahdqrP+YHFZJQok+7LRyMGf7x/kWNw
rPF+Bo4Xn8Zf/Zuv2PXOuAu8szsH1H9g9TjIfn4e/fek5FB4AnpqzHV2eno+MKXN1BbrDP35BxpS
5LhV+zpdXOb6nHp5yf6/LPY2R/1rqdWzeWKtZhL1naJvan/PJES5teoveDBMOhK2g14Sb/rNGtf/
XT9fjie+3OlKoGuHh2Ztc2rRZ+l5SmBsGv7wY93y2AQCQW+BdIHKnj0PXtCRHJr+e//wdF764yHH
8oZHtkTt9HHKnj3O0aYsh/7DpiFcPGeWDrKhUPVMvq6xCmWTKum7mKsxamYtuT8+HEAKvzN/l1qu
lb2YjxMc4XbxeuqqbxzEAvr0lTf7U+XBb++QIPDKqQ43PoSU10dVkM8aEvOoMEwgsK+jj3lyGEa/
9gYwpc7Fsj/0VelRnddrTAdMX6X3UhdEzlmLeVu+Rqd3TnZn9TpWUOyrc9FEka1r7hHpGeEF6Em4
dkOHYj9ExWdxyYSs8MKKtY3/3tDFux/D6gt1Yve8kftt+DGWV6rcKra3/WGEpPl19AAdPnsObC/e
gwz9+dRzbcPAMnp6zHWfLKrE8pwAfrdiv+uwP9w4EAPzvjn743x39VMeu6+M9f8rxIuOBQVXVAh8
u2+cwQ029tDNSE+NNyzn+7xt6mXJehjGb8EsnBuDC1o1x/Kz7magQAtuPqDPldU9Pwrs39kabJ+a
p2vflFyOjcQJaS+3KGbCM0hXx5/sHYW9+MR780jlj/+PRoSguq4QtivDNXRfi6vfUC69dFBXUUxW
LePP/Nhgdmqrd4u4a2hqvaxb+nh2vjljTmnC2PjMK367PwzM/7cAXaw/hi7VOLxnl93lQY5K5ZIY
xM8hN/1mr2h/SPgU9WjTQvVcyvz80GK/+tgvzt9vXAKKjCGO6NYON7G8S/V9c6PHzSlbkFKjSCdw
+MMPQ53yFKw0x4UpYztCPFJ7HvvwSt6/th3VMKd/fMxC2KHIxWwx/bQk+W+P0WnlhnjoPidbNsWt
aEsb3SHPsr5gyHBnJCY7cIVqBbu0hv7S2LwCX6LTLuzd36QMAqx4f7thOjI/BrQMz3V4HsHsF/eZ
mgVXL30d7NmVM6us+Ja1MuyaJmHlTFhY9MgRTxnREzgtj8daknnj6si6OPn+Z5ZpvXI9//rxDte9
vwW2e5HzIDmqb7NdzM0wgCUtBPyC5wVVUCWRMmYu7pLzWMloPmM1Pj0KU5vVcNhkAwNQftzu231+
qzhSodHNspvjDnzahK24dkIG0BnVU/evFqV969unYz2Ojo5Bcz32E4donRuDj2/pgz/Nj8Oaknrp
9miXVcXk4ZbVyzQIoZz48rfDycXfOR0e1x7wHLlI9sPTo0cL1Ou5onVIPdw1xLl42TIh1iOaCnSd
UphR3yG6gAHBlL89j84XJg1sDAH68dyBm3tTbS2+GCV3CUtBlSiUb8a/bj6GguAyXvbUcuQUlLn7
iepVHtMKrJGPKXBTruPbNf2iwY/vG/q3wzOVdXPoQES69oJnHcW99ZhRWTnH1rpBJrR+PYR1Sde3
pnvj6rgsxvGOqqm3a3J3YfOiMakHT3az/vuHt0Ll5faQ1qIM4yawjR3xunHoxtj4zCsv+PgxNk2r
n+708x+kWOG3uTjykWHDecPA0ftx02LGvTZP70Mj2tbq2HrcOzETu9HHo3qIBEjz8Lhgm1AnrX++
BAudMPEuKWJR9iQF7Miw3abN1k3vdNjDD4bnS9en5mPuAM6e2UXsvALx9fS/8vMXua31jf9cFQrO
S/hARPrq1j8PFsuNU+yLneEXZtXeuty/47X9xLAqKy5GSGIeteYVo2VgdFblLsfD62rXOpGL+CLi
Z+8AgjHtzuWP/x01HMPXSzkiuF4cr/70SgP2hQ0Q4qQi4Gt+jOZpr3ogYhnESljP0we1TAABr9p/

```
02O+Knum4urd+QizZzU4p7PcNa6vqoxSdmjLjOnuWvpv6Z/h8Dl+Jj7Gp3AiVXCx5+hARUqRiDN3
SkwJaP7GzlPBKSda0BSqXSjly9ou19rYreqbhjYk9TfNsYRgrEJYz9Kt7p2Pp7nyPhRSaeTELNEq
IxYzremBA28aIIkLDurFwpxWPXVJzf+cJPdMwoaf/7b21JRTKoxER9r4w1sXj6Invtzq2e09bgNz
p4xyRof8Y1ymgY2SYcCT4f90+YGSOtupx9zZqmQk905CaGI/kenGwRRGICDNv6o2pl3ZW9evS3HV
GGQ68eKU6wEe7gBtMjMy0v84+hATpDcpKBUIYxizCcoZ+iZvKM/Xjo3H30Lbo2bKB7vGanFvpKte
qsXvXw1BmUt+WGNExFVVCoFqElqADdtPZ0t35bo8/9s0WAECblPC8/wwTaMJyhu7ObNAsqQ7uHtr
GL7UA5ejNT27rg0wPvuShTmr9eDRLqhNyYg7Yc5lrizmnNaiDhy9We7Ls5dwqDGMIQ4JORKOJaBc
R5RDRFJ3jcUT0lXR8DRFl+HugRth1/Kz3TgaZMqYjVkwZjqEdUr13ZnxGWcz5+n4tsWLKcHRv4fs
bFsNEMl4FnYhsAN4BMAZAZwCTiKizptsdAE4LIdoCeB3AS/4eqJZHR7n6I/vz1Tw+xhaSs1orsvm
pUbikSxNMk1IcaIOuONiHYYxhZIbeF0COEGKfEKIcwJcAxmv6jAcwS9r+BsAIMjkhRp1Yu/n//uF
t8fmd9uRQ/9UkiWLCg6S6MZh5U5YjmldZ9i13+ji3ayYMw6gxsiiaBkBZEy0PgFY5HX2EEJVEVAi
gMQBVpQAimgxgMgC0bOk9I58nbujXEseLSnHXkDZIiIuuUeAPE9rE2KL435NhfCCgi6JCiPeFEFl
CiKyUlJRanSs+xoYnxnbiUG2GYRgJI4J+GIAyvV661Kbbh4iiASQB8BzGyTAMw/gVI4K+DkA7Iso
kolgAEwHM0fSZA+AWaftqAIuEr4U7GYZhGJ/waq+QbOL3AZgPwAbgIyHEdiJ6DkC2EGIOgP8A+JS
IcgCcgl30GYZhmABiyAAthJgHYJ6m7SnFdimAa/w7NIZhGKYmhGWkKMMwDOMKCzrDMIxFYEFnGIa
xCCzoDMMwFoGC5V1IRPkADnjtqE8yNFGojC58n4zB98k4fK+MYeZ9aiWE0I3MDJqg1wYiyhZCZAV
7HKEO3ydj8H0yDt8rYwTrPrHJhWEYxiKwoDMMw1iEcBX094M9gDCB75Mx+D4Zh++VMcJyn8Lyhs4
wDMO4Eq4zdIZhGEYDCzrDMIxFCDtB91aw2uoQUQsiWkxEO4hoOxE9KLU3IqIFRLRH+p0stRMR/Vu6
X0v
S/dpCRL0U57pF6r+HiG5xd81whYhsRLSRiH6W9j0lIuY5UlHzWKndbZFzInpcat9FRJcE55uYCxE
1IKJviOhPItpJRBfy78/wBZy78kVInpI+pvbRkRfEFF8yP2h2mhBBh8x/s6Xv3AmgNIBbAZgCdgz2uAN+DZgB
6SduJAHbDXrz7ZQBTpPYpAF6StscC+AUAAegPYI3U3gjAPun/DaXthsH+fn6+Vw8D+BzAz9L+bAA
Tpe33ANwtbd8D4D1peyKAr6TtztJvLA5ApvTbswX7e5lwn2YB+Iu0HQugAf+eXO5RGoD9AOoofku
3htpvKtxm6EYKVlsaIcRRIcQGafssgJ2w/9iUhbpnAZggbY8H8F9hZzWABkTUDMAlAH4XQpwSQpw
G8DuA0QH8KqZCROkAxgH4UNonAMMNhL2IOuN4jvSLn4wF8KYQoE0LsB5AD+2/QMhBREoDoDsNc0gBC
iXAhxBvx70iMaQB2pKltdAEcRYr+pcBN0vYLVaUEaS9CRXuN6AlgDoIkQ4qh06BiAJtK2u3tm9Xs
5A8DfAVRL+40BnBFCVEr7yu+rKnIOQC5ybbvV7BNhnifkAPpbMUx8SUQL496RCCHEYwKsADsIu5IU
A1iPEflPhJuiMBBHVA/AtgL8JIYqUx4T93S5i/VGJ6FIAJ4QQ64M9lljAKIXoCKIHdxOI
g0n9PACCtIYyH/QHYHEACQvANJNwE3UjBastDRDGwi/lnQojvpObj0qsvpP+fkNrd3TMr38uBAC4
nolzYzXLDAbwBu3lArtKl/L7uipxb+R7J5AHIE0Kskfa/gV3g+fekZiSA/UKIfCFEBYDvYP+dhdR
vKtwE3UjBaksj2eH+A2CnEOJfikPKQt23APhR0X6z5J3QH0Ch9Co9H8AoImz5GSW1hjxDicSF
EuhAiA/bfyCIhxA0AFsNexBxwvUd6Rc7nAJgoeSxkAmgHYG2AvkZAEEIcA3CIiDpITSMA7AD/nrQ
cBNCfiOpKf4PyfQqt31SwV499WG0eC7tnx14ATwZ7PEH4/oNgf/3dAmCT9N9Y2O1zCwHsAbAAQCO
pPwF4R7pfWwFkKc51O+yLMjkAbgv2dzPpfg2F08ultfTHkwPgawBxUnu8tJ8jHW+t+PyT0r3bBWB
MsL+PSfeoB4Bs6Tf1A+xeKvx7cr1PzwL4E8A2AJ/C7qkSUr8pDv1nGIaxCOFmcmEYhmHcwILOMAx
jEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWgQWdYRjGIvw/IlbURK54GdgAAAAASUVORK5CYII
=\n"
          },
          "metadata": {
            "needs_background": "light"
          }
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "training_size=int(len(data_oil)*0.65)\n",
        "test_size=len(data_oil)-training_size\n",

"train_data,test_data=data_oil[0:training_size,:],data_oil[training_size:len
(data_oil),:1]"
      ],
      "metadata": {
        "id": "S9THgFu3KaKH"
      },
      "execution_count": 11,
      "outputs": []
    },
    {
      "cell_type": "code",
  "source": [
        "training_size,test_size"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "mbvhl2HCKmSG",
        "outputId": "2a25053c-8229-4d3f-c3b3-c6da647f1bfc"
      },
      "execution_count": 12,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(5340, 2876)"
            ]
          },
          "metadata": {},
          "execution_count": 12
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "train_data.shape"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "M14MiYVzKrJg",
        "outputId": "f4500139-da49-40a4-b667-3644152a5d1f"
      },
      "execution_count": 13,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(5340, 1)"
            ]
          },
          "metadata": {},
          "execution_count": 13
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "def create_dataset(dataset,time_step=1):\n",
        "  dataX,dataY=[],[]\n",
        "  for i in range(len(dataset)-time_step-1):\n",
        "    a=dataset[i:(i+time_step),0]\n",
        "    dataX.append(a)\n",
        "    dataY.append(dataset[i+time_step,0])\n",
        "  return np.array(dataX),np.array(dataY)"
  ],
      "metadata": {
        "id": "vrH4kcM5K9S8"
      },
      "execution_count": 14,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "time_step=10\n",
        "x_train,y_train=create_dataset(train_data,time_step)\n",
        "x_test,y_test=create_dataset(test_data,time_step)"
      ],
      "metadata": {
        "id": "dv2OIjslLA3v"
      },
      "execution_count": 15,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "print(x_train.shape),print(y_train.shape)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "ApfZGkVhLD2p",
        "outputId": "7c4a6632-d25b-4098-a7f2-1d6ca77ac037"
      },
      "execution_count": 16,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "(5329, 10)\n",
            "(5329,)\n"
          ]
        },
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(None, None)"
            ]
          },
          "metadata": {},
          "execution_count": 16
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "print(x_test.shape),print(y_test.shape)"
      ],
      "metadata": {
```

```
"colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "iLQiXr21LE35",
        "outputId": "318f7808-d48b-41f6-c120-50f3f89a1db4"
      },
      "execution_count": 17,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "(2865, 10)\n",
            "(2865,)\n"
          ]
        },
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(None, None)"
            ]
          },
          "metadata": {},
          "execution_count": 17
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "x_train"
      ],
      "metadata": {
        "id": "w95tBQ4gLJWj",
        "outputId": "a4438245-7fb3-4b13-977e-5bb7c4178e22",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 18,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703, 0.11661484, 0.12053902, ..., 0.10980305,
0.1089886 ,\n",
              "        0.11054346],\n",
              "       [0.11661484, 0.12053902, 0.11550422, ..., 0.1089886 ,
0.11054346,\n",
              "        0.10165852],\n",
              "       [0.12053902, 0.11550422, 0.1156523 , ..., 0.11054346,
0.10165852,\n",
              "        0.09906708],\n",
              "       ...,\n",
              "       [0.36731823, 0.35176958, 0.36080261, ..., 0.36391234,
0.37042796,\n",
              "        0.37042796],\n",
              "       [0.35176958, 0.36080261, 0.35354657, ..., 0.37042796,
0.37042796,\n",
              "        0.37879461],\n",
              "       [0.36080261, 0.35354657, 0.35295424, ..., 0.37042796,
0.37879461,\n",
              "        0.37916482]])"
            ]
          },
          "metadata": {},
          "execution_count": 18
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "x_train=x_train.reshape(x_train.shape[0],x_train.shape[1],1)\n",
        "x_test=x_test.reshape(x_test.shape[0],x_test.shape[1],1)"
      ],
      "metadata": {
        "id": "59VQDX1LLL3D"
      },
      "execution_count": 19,
      "outputs": []
    }
  ]
}
```

FUTURE SCALING:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 2,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().any()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "RAUyZB0-Jp0t",
        "outputId": "7b396aa2-0bdc-4947-c627-34d1260cf12a"
      },
      "execution_count": 3,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date              False\n",
              "Closing Value      True\n",
              "dtype: bool"
            ]
          },
          "metadata": {},
          "execution_count": 3
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "yorF39lCJsyV",
        "outputId": "94c4ab77-12d0-49ea-9fc7-9a40dc1532d0"
      },
      "execution_count": 5,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date              0\n",
              "Closing Value     7\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 5
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.dropna(axis=0,inplace=True)"
      ],
      "metadata": {
        "id": "g3FWuWVPJwms"
      },
      "execution_count": 6,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "vIHbyOs4JzIf",
        "outputId": "2a7a4f4d-a27c-482f-d791-a2687f620cc9"
      },
      "execution_count": 7,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    0\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 7
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil=data.reset_index()['Closing Value']\n",
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "IoxUNwrvJ2b6",
        "outputId": "c8566900-c503-4f97-d55b-637fefb05fa6"
      },
      "execution_count": 8,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "0       25.56\n",
              "1       26.00\n",
              "2       26.53\n",
              "3       25.85\n",
              "4       25.87\n",
              "        ...  \n",
              "8211    73.89\n",
              "8212    74.19\n",
              "8213    73.05\n",
              "8214    73.78\n",
              "8215    73.93\n",
              "Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "metadata": {},
          "execution_count": 8
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from sklearn.preprocessing import MinMaxScaler\n",
        "scaler=MinMaxScaler(feature_range=(0,1))\n",
        "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
      ],
      "metadata": {
        "id": "5m-DUFI9J_WN"
      },
      "execution_count": 9,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil"
      ],
      "metadata": {
        "id": "fyMwOo1jKLqL",
        "outputId": "cbe459af-a1ce-41ff-9dd0-1bc24769a668",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "metadata": {},
          "execution_count": 10
        }
      ]
    }
  ]
}
```

IMPORT LIBRARIES:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 2,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().any()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "RAUyZB0-Jp0t",
        "outputId": "7b396aa2-0bdc-4947-c627-34d1260cf12a"
      },
      "execution_count": 3,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             False\n",
              "Closing Value     True\n",
              "dtype: bool"
        ]
      },
      "metadata": {},
      "execution_count": 3
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "data.isnull().sum()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "yorF39lCJsyV",
    "outputId": "94c4ab77-12d0-49ea-9fc7-9a40dc1532d0"
  },
  "execution_count": 5,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "Date             0\n",
          "Closing Value    7\n",
          "dtype: int64"
        ]
      },
      "metadata": {},
      "execution_count": 5
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "data.dropna(axis=0,inplace=True)"
  ],
  "metadata": {
    "id": "g3FWuWVPJwms"
  },
  "execution_count": 6,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "data.isnull().sum()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "vIHbyOs4JzIf",
    "outputId": "2a7a4f4d-a27c-482f-d791-a2687f620cc9"
  },
  "execution_count": 7,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "Date             0\n",
          "Closing Value    0\n",
          "dtype: int64"
        ]
      },
      "metadata": {},
      "execution_count": 7
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "data_oil=data.reset_index()['Closing Value']\n",
    "data_oil"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "IoxUNwrvJ2b6",
    "outputId": "c8566900-c503-4f97-d55b-637fefb05fa6"
  },
  "execution_count": 8,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "0       25.56\n",
          "1       26.00\n",
          "2       26.53\n",
          "3       25.85\n",
          "4       25.87\n",
          "         ...  \n",
          "8211    73.89\n",
          "8212    74.19\n",
          "8213    73.05\n",
          "8214    73.78\n",
          "8215    73.93\n",
          "Name: Closing Value, Length: 8216, dtype: float64"
        ]
      },
      "metadata": {},
      "execution_count": 8
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "from sklearn.preprocessing import MinMaxScaler\n",
    "scaler=MinMaxScaler(feature_range=(0,1))\n",
    "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
  ],
  "metadata": {
    "id": "5m-DUFI9J_WN"
  },
  "execution_count": 9,
  "outputs": []
},
{
      "cell_type": "code",
      "source": [
        "data_oil"
      ],
      "metadata": {
        "id": "fyMwOo1jKLqL",
        "outputId": "cbe459af-a1ce-41ff-9dd0-1bc24769a668",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "metadata": {},
          "execution_count": 10
        }
      ]
    }
  ]
}
```

IMPORTING DATASETS:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 2,
      "outputs": []
    }
  ]
}
```

MODEL BUILDING:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": [],
      "collapsed_sections": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "execution_count": null,
      "metadata": {
        "id": "gMYdEoHq-oKE"
      },
      "outputs": [],
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt\n"
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "gmhtVZLW_Ghh"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
    "data.isnull().any()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "N-TTvICd_RdY",
    "outputId": "553b761b-7f2e-4141-d253-ccca37df751a"
  },
  "execution_count": null,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "Date             False\n",
          "Closing Value     True\n",
          "dtype: bool"
        ]
      },
      "metadata": {},
      "execution_count": 5
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "data.isnull().sum()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "XZ2t4xPq_Wan",
    "outputId": "12bd5fba-2625-43b1-85d2-48ed44149251"
  },
  "execution_count": null,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "Date             0\n",
          "Closing Value    7\n",
          "dtype: int64"
        ]
      },
      "metadata": {},
      "execution_count": 6
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "data.dropna(axis=0,inplace=True)"
  ],
  "metadata": {
    "id": "5Td0P9d2_aKw"
  },
```

```
  },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "E4xiOaaP_hLy",
        "outputId": "5a082030-aeb2-4686-b991-0c47f814d67b"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    0\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 8
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil=data.reset_index()['Closing Value']\n",
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "hCIMsLTG_mLH",
        "outputId": "e2174bb8-b283-419b-c5d3-d7e9ffa55e5a"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "0       25.56\n",
              "1       26.00\n",
              "2       26.53\n",
              "3       25.85\n",
              "4       25.87\n",
              "        ...  \n",
              "8211    73.89\n",
              "8212    74.19\n",
              "8213    73.05\n",
              "8214    73.78\n",
              "8215    73.93\n",
              "Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "metadata": {},
          "execution_count": 9
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from sklearn.preprocessing import MinMaxScaler\n",
        "scaler=MinMaxScaler(feature_range=(0,1))\n",
        "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
      ],
      "metadata": {
        "id": "nJPq_3ep_xE-"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "SYKkDj5vAHem",
        "outputId": "1a0cd460-f0fa-4048-b747-bd3ed93e1fac"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "metadata": {},
          "execution_count": 11
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "plt.plot(data_oil)"
      ],
      "metadata": {
        "colab": {
```

```
"base_uri": "https://localhost:8080/",
         "height": 282
       },
       "id": "NEJmc3gQAJ5F",
       "outputId": "43f960c9-7748-47e1-d218-5f4c4d91cb16"
     },
     "execution_count": null,
     "outputs": [
       {
         "output_type": "execute_result",
         "data": {
           "text/plain": [
             "[<matplotlib.lines.Line2D at 0x7f2e8aae7250>]"
           ]
         },
         "metadata": {},
         "execution_count": 12
       },
       {
         "output_type": "display_data",
         "data": {
           "text/plain": [
             "<Figure size 432x288 with 1 Axes>"
           ],
           "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXQAAAD4CAYAAAD8Zh1EAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO2dd3gU1frHv282DUIIJQklARJ6lRa
KgHSRooId7OXKtXtt94d6sV1UbFdsV1GvyvXasKOgKEV6C70JBAgQakJJSCD9/9d6d6sV1UbFdsV1GvyvXasKOgKEV6C70JBAgQakJJSCD9/P7Ymd2Z2dndyWZ
ny+z7eR4fZ86cnTk7bL5z5j1vISEEGIZhmPAnKtgDYBiGYfwDCzrDMIxFYEFnGIaxCCzoDMMwFoE
FnWEYxiJEB+vCycnJIiMjI1iXZxiGCUvWr19fIIRI0TsWNEHPyMzMzMzMzMzMRAXfH2OT
CMAxjEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWwaugE9FHRSCiLa5Oo5E9CYR5RDRFiLq5f9
hMgzDMN4wMkP/BMoD8HAGgn/TcZwLu1HxbDMAxTU7wKuhiBiKYBTHrqMB/BfYWc1gAZE1MxfA2
Yxhw+XLYPeafPBXsYjB/xhw09Ddc2AhxX6Ov7YYCEU0momwiys7P6Pz/fDpRmG8YV9+cWYNncnBr20ONh
DYfxIQBdFhRDvCyGyhBBZKSm6kasMwSAc+VWwR4CYwL+EPTDAFoo9tOlNoZhQpRqrlRmSfwh6Z
A3Cx5u/QHUCiEOqH8zIMYxJV1SzoVsRi4gLAUADJRQH4GkAMQAgHgPwDgH4Da
zBsswjH/gGbo18SroQohJsoa8KCzjARCAu6NWFBZ5g
IpIpt6JaEBZ1hIpCVOQXBHgJjAizoDBOBzFy6L9hDhYEyABZ1hGMYisKCzjAIKKWicnBHEkjL9hQWeYCON
MYxFY0BmGYSwCCzrDMIxFYEFnmAhDKKWicnBHEkjL9hQWeYCGP+9uOObVsUBXEkjL9hQWeYCON
saYVj+3wFVy6yEizoDBNhxNicf/Z5p88HcSSMv2FBZ5gIg9jKYllY0BmGYSwCCzrDRBg8Q7cuLOgMwzAWgQWdYRjGIrCgM0yEwRYX68KCzjAMYxFY0BmGYSwCCzrDRBi7jxYHewiMSbCgM0yEsfngqWAPgTEJFnSGiUDiovlP34rwt8owEUb/ds7CL+UVXFvUSrCgM0yEcXW/XnqnOkz6tHJqZcyLEv0hNtt/CtCtS2JnLIkkgwQJGKhZV5dV7+ONhA2JhbTsXYWEE2qLGXIBgdMdxStAqNLdpzTk+G3rl1QhwYuj0UQl+EMV7yG0D6M89vCy8x1b4wiyS35bSp+W0/FhZSlb+/3XO2vfYzVVo9E4vJfH2q8C2JOGc8X9CK0ZcTXb/dOcTN4gu4xV5WLXzIVf5iZWTRl0wZr6PmyhmTs+KdXgxH3tbpsubEK3lqOR76YjgL9dMwahYpgJ1MuA05yPvyA7HPRaIwzxcWY66r7PnKsrJ1Kj/rWn8kTBk6dPHCi98KAZ+8ZdvaX7PAYbPJFPDW+R2C1b6VF9zX6mAcUovM4kybpu8nQxyZOW0f/Puoa80Rn6TORsFbOBVxbCMTb5vXxcWu28DIYDZudqg5bCzo7YjRNR9k2U0rvbnsY4L3mBgQG2UYN6nK87Sw/r3v4KLXs8TZU5tUwbfVpq7UzKX7P8/KyeprNPm3v/CW7QzD1mf1K2ZNNP0pa62dHSG0NsBaPtKUgNW7N2lpkE5+Bt7y3lqsALnnazyZPzLzjFpKpUpf9hLOpiTHrqD8vjRaFGgLRmzQ8IWswXujoRnVo9MKgu1MtlzLhKE3gkfV7vJvaRpVnfU4Fb1rBPfGSVaYa4FRZcvzYK5jYYmjtOcpjVqFvlRjT7RVfaQiNNGUxDTzWbA4zE5I4ZLxz2qqLmjIuHcGobPKIxHdgE4qYo/Q5TKcG2e5gdGu89SHbvwOr8vx+r3KcaLGvezOvD6pl9rotDmPEmc+wt/36n2uDQtZvKk1zwGruC7SPw6+06+vPG2N+pTevZpd/0qZ3FhszOSJ/2S2f4Wg5Bg/vlmqKw+XSWhvPSzfanh6OS0VZGYFm+FasXHzQsCKLhSRDKSpwNY7+9rcZt9V+nNjSe84PRQsQWtnqFrFPJOLW6SxFPq92evvZP2bXmOKKYwFPQLo0MS5ZpbhOsNjbO+3z13PPmsobCzoEUCzpNrbcKgXqrL6vNGeRlQUxNtKF9a4WmYQSYbtkUR/c1KSsdmSUrFzxGzsnt7wdhIcbCzoEcCIF9ZjAtW1K1fhbXnRqEfVqEk9iW+dWGs/2qt1FaT/WvC3F+S29Ncn0vg6zKeF8JLcPI6RrwoO5LOj66M5LY9DzhnKzfn9TKrQSU8zQN2Rff+SEw0Fg9JV5P0bxB+2gnl/BF9PIuhkWEDO7+0P/C5Z0C23v2zZ9Yj2Vno63DjHUS4qBJdBhz1HfWnTu+wWjb6pWFQ2lcy7pKNuDnnR1tXTMpUSc/0g3VDBI8DYQGJVX2awRDLdxtpsWxJoD+PpL4vQa8MWnI6hYktNhzdRZHcf6ZkhJqkEV6kGGz00k4IAXIbJDUqbGoz5cKHtQJ4qbuu2uSkaQFrxRzXPAjAgLtjNupvzGyIH8b7cS5YwVpmcBZn9BYIwjF0U+mWp7L60lZp31Zj0a/MEvvLMEHSybQd8y4B4dh6qWaXc8ULKKWsiXp5aNP1jTmCjB3+H2ZGs1W4m+vTJgf6jPmUTyOV6HiXbzZ1FDexp+f/9qLGrCFuT7pm8MrGJg8oUPLsDH/hHUzMCtJSWl7P8ggJzn+jqnXc6C9LK9aIC0BbLd5ns1VRSd1N5PhI0aBTlu77hV6tZjkI9YLbjvyy+WjhdVlPP4cCXnL7BZpJMnU4CqZxjqPHnfKZJFOfkuQC/Ahqs5FmyM5BHaQ5JZM2YiRR75s2rSk6DL6iRXG8SgfLyD7UFoeQYr5X4rKjnMxpDaExd1LMUxRjHaa/IZlJkLDvzR/nCPGr01YDhhQ5VhySsRI8AhhY2+8JGqd+JKDfhXDjp+i8k8i5CCiiaY+jqw3OBPrA+0bkx/hd4N8mTTPgkkpyOeXubMi8plWjy6ubhjIY13RfMj13d9pOJ+/i15ljpmWvh0rpNaWNBIQlvQGNxqHSzwUffTk6zqhDqvvaZ3zjRrRZcXxkpuwHg4hljF4xHnWK+tNGUGoTV+FSwRqvkeRZOzZ7F9TuB/yS+4pCb+e5o6o43eD1X4mZ6OkVdIbjHN9eHDsnLe2w8O7f1/VJMqk2+AKpMo7gPyE0vDVGTOl+S4emOl0BDajHjaiJJ/O0JTJmdYmJCMhRk/p++hSzZPSD1Yhvt4RO6wSHMNhh7W8bXnHdSG/uIS2uaIvjoafX6o3sdG6E5T6bVQXmNPQaw5G6vvvzGVhPJAWLCpKkd+W3rqM6LEsjZ6hjqCtAy6RrNzrWQs0tHG3hLOiQLLh3YkNhj3QrmXcGrvm71tWSsZhBa6jtlbDXgRcTxP76GG7Lrsld/gYwGeYAyWbJ7h3YSVI1Vpsx9OoJnVzsd0FyuI+G7SaF6wWmK3GvXEZp0rxhYEEPBA+Nch1DgfWAi7bpAU2AC8dUmbOcBiNSkSBbGr61udYWwqqqsx93dsP/mtOtY/SANzGVZvpdg3GQoIWjbKHmkcrxmdJtiuJMWVq8YvgQOeasu0+Z3BRxPRPS4xmjh8Kvp6eAgoDMfuAuOo8bjhdUbtxxe1EPNahEm+Tt/bNCO4Ly1Ng0OWFVuNoKv3UxNqCyuF7Ug4WIpdu4yeCT4QP6TacqWRyXxdBdrV5cTxYpVRgPNFj4AZvmQNUTNr4RMzMtS6sGpnVsnCiHQmQ9LahiS8dWgsZOHNeDL6zE97tM8yDrpmPk7AN/AhfbhCrOqPCyVgmi5MVHWRYjlYQk7Ry9EfUxmFy8fO0Mf2o4UPcxv7mQNxTB0UIbQpOpfJm/PM1+aRg6crsdsxkDWahzBaskiy3xqdVa24PyPUvBfkRPjQMU5p2hEMxPpaCAtWuSZ8spfZqa4KFH85ae6cCc/gtuXeMGR6lymv4zJvMqUqEYmZj0O17uXqV4W/MFJHMYMhrMbCr5WC1Bp1U0g1t5cZ6W6Sl1gDdfN/qMhffr7jHJRtmyP+0i6q5nnazXUWfY3p0VYc6ucEu+B0mWW9Rbp0Oo5ANsc1qjS0VRBxJPkVjppY27ifX/PFNuTyWSIdhOOV1m2FLCcl3fHTa1jZZLzJrQbEXG1rzedX0w80K9ai/jD7RuLi3dLwrSnYgSc9+6G97Q4dHKYdTwp3pTLQQCJvCNjsP8uZqkzCBg4AwIAnDJEOYNZHpIUXhUHMVxC4zTaR/Kz/wPbtkYzg2oqmMEUp12EZfp9q39gIl9qQPfOyDaVZ6p6JWTlyLc+qd38tlu7+ibCmkQIxcbaSXydfrKQ/SMswhJk4KWYn3qCgj6KHsJhlhQjy7AC1F9XktmoR2AXDMLkaTGhYfBXnkMs3rPdVrXX4/JILeMZzVRzFJtn+uI1RpKalOzF0m4BwRytBIdR9d8V/mqPA6iCaTlq4IDhjv5k0xeP19f+yuhiWfA7rj1qUE2UnJnYjDVdoU/JsqdiEx7atjJhGqcmCSSVIBm/JLnJXqo1kUgmw8wgE+CK+kDfA5yjHM+iKwWCrKtS2/QjDt6bP5xV+HTmE3G6ILpmNajaaJqHE7ZPPB6MMRW02QWunrfkWyhZB1wnvQwPPXYaUNoCn8e0PIqvjSjhtCKt7j8AxfYBlj2FdgS6FOjK+zUybjIQKaORJqJbXAKqzJlCD6lTJEIC/bYsmiXfGMv8VFLzf+CLx0R6klhnyNJEfQPyllgxjbSMrdWwx9n8gMpDbUxmn5RWdeLOi1uwRB4cvZS/WmrH0GZGo6ab+xB1ujCsnefyUePMd7JKjYg3sRmfOSOgTT/NcTEnK4nGoOHeiNFChhZQEIHb6K00v5FWTp3jl6/6nfDBTwOSjUpnrm9idA9aTALZCqRRR3J8OLFMvDcdRFvsHRrpRlDK/g8TkACvNvoZlhH5zAr3aSsvXcSsgxwdgDO+a1tuhiGBzKFgRgEnEKvGBeAFqUNG1wEcTaXlLaeUR2WTJFB/A7CRqgq4Xj+h/3oyTpoAJT8CHX4s3JVlsnxB3V1R/uzuS4BymBpiKLK7d80zEM5uqWgEOHLI0NPupZUhZIanuJPYDjZXObDtR2MM7cfFrCSMl3NX8yIuNkB9krEEb0IjZP3rjNBoY1y3dbKd/nbGCy+i0oKJo40VgFAvxe3lS8AOUrlGl+IF7qJ2cl6F9P4ahZ+2ltIgFL4r6QIXbYzQeC9sqXVigOUcIO9TtJlJ2Ovbz9dOUOHIQovGUX2xRnzCSKSKrnhqYEbNG0pwdOyw3/G71P/cmqwoZqazQNPU1IXnHa5Kc94wS4IHFt/ccfXpEK2gEJmFH4e+Lz+8dBiC2rr2ko+R+NT5mUcijI3
w9vz5pUKF5EuXyOdAn7Tw9f1zrycrr5QGJkP6hZXUBhjxfuPcaAzqD+cPjRUEYjoWpD7UYj5
9RiFjtjULMBmxdbEzJo8VSIo88PWoAPOSQWSdbKN7zB+yGiWL4o5Ke+CvfESyYDmGBtgm/+JqbZKVh
Mx7rNrezg1jXqq+4C4gNo43ERNiBwhDT2ggRn82n1a3yJ7lASgb8+EfQm4cnq1t8YZIH2/P9ct9WkI7t
1cRwCMWwiNKuOd5pCpWzOrZ4vQMNNNmGNHuMsxQOF3HCTbFfMzYnFSlFgmz/T+n7dzrA1yGAwh8BmaoT
L+3Zw8vL1C5tglMnqmk2cxNDtTkyQnoIV/AUtrwGq3Zk1kZ9lhS2HVu4RbeyrCvb8ac1QyD9xS0BtJ
93AnzTJkKAoHw+d7NRB5Mxrx86wYwyibNbOfmRS7ct58rsgBRcS+9hPx+vZCP8Aa3rW2FVVVBNc42CVOo
yRHHQuKkWYVOoEFSOJb4qSCaWxsHNZsKOfSv5rYXVeakAYh6oAuYw+hzLqkx4x0bcQHiRUtAIYD3o+EK
xiTrNsNV9IEI84fX
```

```
WqY5H0sIoCzrDRAgqLxJJN810clE+LNx5GWo9W4yk9B3UVl2+UjbBlFXqm17KWNAZhrEaSu1MqhM
DwFw/dOWZSyuqVYFNALB2/ymMf2eFqq3agH95ZbVaoNcfOA0A+GCZ/qKqUXdIK8CCzjARQnbuKZc
2c2fozu2C4jI89s0Wx/7rv+/GtTNXuXzGiEtiRVXNBJoFnWEYy/HZmoMubWYm5yKozy0nBgOANxb
u0f2MEfGV3RS/nNzf0DiMPCSsAgs6w0QI+wtcPU7MXBT15dRGBL1SEvQ2KfVU7Zd0aeLzOa0CCzr
DRDDBjhOVuSA9CYDRGbq9T4yNMLJTKro0r4/qauGwpWthQWcYxlLM3XLUsZ3VqqFj20yTi56p47M
1+rVF/zq4jdvPaJFNLjG2KMRF21BWWY3WT8xDQXG5qt9dQ+RzspcLwzAW4bqZq3Dv5xsc++/c0Mu
xbcBL0Ge0Xi2As76olnrx9uyPVZL4/rjpMNbpLOICTtGPthFAQM6JYpc+Dwxv63hw8QydYRjLsGa
/Whib1HcG5phpQ6/JYmS9OJv9M5I55cEvN+Ga9+xeMNXVAkcUJfJemb8LABATFaV681Dy8KgOsEl
BSiVl7lMDWA0WdIaJYMx0W6xyY+rQm7nXjZVn6K7H/rN8PwZMX4Qlu/N1M0Zq+fi2PgCAXGkReNI
Hq2s28DCGS9AxTAShzX6odS30J+78xat0gofOldtn0Ut256NrWpLq2NI99pzqt3y0FiM6pnq97rA
O9j6nz1XUaLxWgGfoDGNhtMFE2shQM7Pnugu515uF1421m1xOlZTjopcXq44dVphbFmrS4daPdz8
nvWNQpuGxWgUWdIaxMNpsh9p0tmZ6ucgml4WPDFG1a8P7Z1zXA82T7Mm7+rVu7HIeTxkbX7+uh2p
/6qWdHdtyegPAnmYgEmBBZxgLoxVPraCbaUOXJ+IJsdHo0CTR0V5VLVT7E3qmIVbKy17TzIhaO3r
7JvV0++mlGbAiLOgMY2FOnVP7Zmtn5GbO0OWHSRQBuxSl4KqqhWofgEPQyyqr0CYlwfA1tNWLZNN
NpMKCzjAW5uVfd6n2dyiKPwDm2tDlGbrWNVKviIUtihBjI5RXVmOvZGK5qF2y12toy8/J3jKRiiF
BJ6LRRLSLiHKIaIqbPtcS0Q4i2k5En/t3mAzDmIGZfuiye6L2oaH0C5+hsIHH2qKQf7bMsb/x4Bm
cLdX3VGlSPw4AMKCNWvR5hu4FIrIBeAfAGACdAUwios6aPu0APA5goBCiC4C/mTBWhmH8jFkz9NK
KKuSdttfztEURruyZ5jimfIZMULTHxdjw9XpnRsbiskq8Ol/9hiEzsU9LAEDTJHX1Iu0MPdGDF4w
VMTJD7wsgRwixTwhRDuBLAOM1fe4E8I4Q4jQACCEiq9Q2w4QpZs3QO0791VFwgoiQqohOlRc+J/V
tofpMrE4eglmr9HO/DFKYYx67pINjWztDv0LxwIgEjAh6GoBDiv08qU1JewDtiWgFEa0motF6JyK
iyUSUTUTZ+fn5vo2YYRi/cWUv+59yamKcadeIIkDA6W0jC3rfzEaqfkYKRMv0yXB+9t5hbfHaNd3
RoUmii6A/Oa4TAHO9eUIJfy2KRgNoB2AogEkAPiCiBtpOQoj3hRBZQoislJQU7WGGYQJM3dhojOr
cBI0SYk27RhQRLruguWP/5o/WAgBsUb7Jz60DMlzaruqdjvkPDXZ544iLtgu8gcp2lsDIHT0MQPl
ulC61KckDMEcIUSGE2A9gN+wCzzBMEGlYN8ZrnygiQ7U8jaJMpAXYbehd05LwxkR1EFB0DQz4Q9o
7J4BDOvBk0B1GBH0dgHZElElEsQAmApij6fMD7LNzEFEy7CaYfX4cJ8MwPmAkn0lUlLqAdG257n1
1EI88adYGNWn3PdGyUV0MbGuPIq1pYethHVIQFx0ZHtpev6UQohLAfQDmA9gJYLYQYjsRPUdEl0v
d5gM4SUQ7ACwG8JgQ4qRZg2YYxn+Qn2foh06pZ+hy8JJWiA+dOqfa97SAmZIYh/NSAq+auiamJMa
ZalIKJQz59Agh5gGYp2l7SrEtADws/ccwTBgRRQT4Sc8Listc2mRB19q3d2uiRT1NvG++sBXmbz8
GwGkXN4otKspt5kerERnvIQwT4ciz39FdmrociyLXnC9G2Ha4ENsOF6razuiYeNxZVlxMMApFf2C
EegkuMT7GYRaq6VpqjI3c5ma3GizoDBMBvH5dD+x/cSzevbGXyzH7omjNz3npW8tx6VvL8XW206t
Zr36nO1937UxbmVfm4Yvbq47ZogjPX9EVWa0aom2qfgIud9iiyFEJyeqwoDOMhemaVt+xmEhEuuJ
KPs7QZV77bbdjuybZEm8fqM5XXuFlFt2rZUN8c/eAGptcYmxRXs9tFVjQGcbCVFUDdWI8L5VFEfn
NT9tdUQs94mLU8vPdBq03tH+wRVHEFIpmQWcYi1JQXIadR4sQY/Ps5uerDV1GGeFZWmG8ILM358M
nxnYEAPz5T93Ac8PERBEqqgREBEQXsaAzjEW57/MNAICjhZ5D6msaWLToz+OqrIhKbvt4ndvPaa0
9yfU8pxuYPLgNcqePQ3xM7TIolkkpdkvKjT9swpXISkXGMBHE6n32smubDp3x2I9qsChaWVWN2z/
Jdmlfl3sKRwtLUak40aS+LbF6n/twFG21IbOYucQe4zhrZS7uHdY2INcMFizoDBPhRBEMmyMqdZS
/Xlw0rnnPtcTbi1d2q9E4fr5/EC59a3mNPlMTXpm/y/KCziYXholwDpw8h4Licq/91h84jcOaPC0
AcImOb7svaHOb+wvZyycS4Bk6w0Q4y3MKANhrfXrKr3LVuyt12+XqQbXFrPqmY7o2w4qcyMhEwjN
0hmEA+O7pomd/H9vNddbuTa7NMqlf1t2eundCj+ZeeoY/LOgMY1G0FYG84aug69nf/31Db5c2b94
qZlVPSqoTg3px0V69aqwACzrDWJCTxWXIO223d+sVhNDDUzClu8CcurE2F0+Wl6+6QLfvRe2S8dS
lnXWPAebN0AF7fdIPl+837wIhAgs6w1iQ3tMWYNmeAnRsmohnLu9i6DOeZugVVfpqbyPC5jx1gq4
GbopqEBFuH5SpeyySqEl6hJrCgs4wYURFVTU+WbEf/I8+5bL1CQox5PBRc9dsUvz+jhbVunSHSuN
jMYnEePuDYLyFbd1fZx9C+3/8goMnz3cIwYcST32/F7Ow8AEDu9HFe+9cxIOh/G9kOMxb
s8TxD15lVupu1x/hYKxQw9p1qQ3llNWKDWL1o3tajAICc/LNo2biu38/PM3SGCSN+2XrMax/lK31
8jPc/cXlm7GlNVC9bobush95yxwQTbVGNQGN2NhkWdIYJI/RMHFqUVYPqGCjXJsuvp2hRvYo/7ma
6vppcAoE/S+2FImxyYZgwYMnufNzy0VpDfZUeKfEGcofL3oKetK5Sx7wSa9MX7mgv7irZ/xgZtII
TZgUvGUW+x2Z9/9B9lDIM4+Cxrzcb7ltW6cwqGG9ghv7VOnvFoewDp3WP5xaUYMaCPS7t7mbo5CW
EKLlenGlh/t7wFAkbCJbszgcATP50vSnnZ0FnmDDghJt0tXoUlzkF/YhO7hUtfx6z25VzThSr2oU
QOF5Uips+WoPvN7oWn3DnfidMtxTXnHdvsJfeC5VSdN7eYnyFBZ1hwhR3Xibfb8hzbB8vMv4g0Ar
xp6sPoN8LC3HolP5Dwd0su11qouFrBgrZfVOv5mmgKCqtQGZyAgDgu3sGmHINFnSGCVPW5Z7Sbe/
UrL5ju44BLxcZrQ191V7PCa2S6ugHEBlZiA000ZLnjZ4/faB4ZPZm7C8oAWBeZkkWdIYJU9wtSp5
XlIFrUDfW5/NvPOg5eCkl0ZkbJatVQwDAaD+l0vU30ZJvfDBNLr/vOO7YNmtxlgWdYUKI6mqB8wZ
LpZ0q0c9hXlzqdG2syWx51spc9Xk8uEj+c0JXTB7c2rE/86be+PzOfpgxsYfh6wUS5ww9eCYXJWY
tzbKgM0wI8dzPO9DpqV8NVamftSpXt7243CnENgMzQTmfuXbh9ZqsdLefual/K8Qo3hCICAPaJNe
6/qdZyIuQwZyhK01UPENnmAjgm/X2Bc3C8xVe+17VS19w5RqaAHBhG+/VeqZfqZ8dcW9+iUvblb3
S0EzH/hu6saF2HCaXINrQlf+m0SZF03JgEcOEELKftOzBMur1JS75WG4fmImPVuz3Ootf/n/DkNa
gjtdrJsbry8BSyWdaybQJXVE31rV/kON1vOIwubjxDDIbbem+GDfrH7WFZ+gME0LIgl4qLWzuPl6
sSk/78MXtcdcQu+26zEsa1vSGdQ0VjYiqgU+0u8jTIE58DRETZC+XQ6fU2RVZ0BkmApAXOoe88of
u8QdGtHNEf5ZWuC6eylGi7lwK9aiJPVcr/g3ryom9QlvRbQ6TS3Bm6O/+sVe1b1bEKptcGCZEqXY
zm4yTQu61M/Q3F+7Bv37fDcCYDV6mNtIivwGEtpw7F0X1koyZybnySsxYsAcr9xYE5HqGZuhENJq
```

IdhFRDhFN8dDvKiISRJTlvyEyTGSgneW6M6nI/ufa47KY1/i6Pn3KjjNTYy1OEgBkG7oR7yF/MvW
H7Xh/6b6APUi8CjoR2QC8A2AMgM4AJhGRS2FAIkoE8CCANf4eJMNEAlr7rtak8vLVdm8UIkJcdBT
KdEwuvuDOXDK4fYpq/4d7B7r0CfXFUBlnYFFgTS47jhYF9HpGZuh9AeQIIfYJIcoBfAlgvE6/fwJ
4CUCpH8fHMBGDNtnVxkPq7IepisjMsspqjwm7Hh3V3vB13c0dtSafHi0auPR5/boe6N2qIRol+B6
RGgiCtSi6U0fQL+tuXok9Izb0NACHFPt5APopOxBRLwAthBBziegxdycioskAJgNAy5Ytaz5ahrE
wWhPK7Z9kq/a16Wq/33gYTZPisevYWXx0ax/VsT4ZjQxf1525RLmA6M798aJ2KbioXYrusVDCFgK
BRQCw7dlLUC/OvKXLWp+ZiKIA/AvArd76CiHeB/A+AGRlZYW41Y1hAou3avDROrU6Ze8JrZ9zeiP
j9SrdmVyUdt9pE7oaPl8oIrsJ6pXSC+w4zLVRGTG5HAbQQrGfLrXJJALoCuAPIsoF0B/AHF4YZZi
aoSxMUVNW5qi9KIwEFMm4s0Iobfg9W7qaW8IJeYZeFeAZupy0TKY2BbSNYOTs6wC0I6JMIooFMBH
AHPmgEKJQCJEshMgQQmQAWA3gciFEtv7pGIbRY8HOEx6Pe6qH+dg3W3y+buuUBN12paCbFQgTKBx
ui4qn15p9JzHopUVe34xqQ4km0VpNgrh8weu/khCiEsB9AOYD2AlgthBiOxE9R0SXmzo6hokg/vn
zDo/HjbrcfV/D4gnJ9eJ025U2fXfl5sIFIkJ0FKFKMrlsO1yI695fjbzT53HZW8tNu+6e42dNO7c
ehmzoQoh5AOZp2p5y03do7YfFMJHLwxe31/UpN+KhcedFmejZsqHXfkYorXAKulkl0wKJLYoci6K
v/rbL0b7LRNFV/ptdbqJ3i0x4P3YZxiIsUSTCuvSCZrp9+mV691zx1S1vZKcm6KyodARA5eduJCd
MqBNji3Is9BpJK1xbtD7v06/qZvo1WdAZJgS4c5ZzyalefLSqjJyMkVzj7tIFeCM6ilRuiiVllTh
bVolmSfH49m5z6l8GGpvC5JJa35kCeICBFMO+0PbJXxzbozo30c1S6W9Y0BkmxEiMi9ENSFFy99A
2uu2+ztCjbaT67JuL9gAAjhaWoncr/5hwgk2MjRyLoj1aJDnaE0z0CweA6Vd2w/s3B8bpjwWdYYL
IiaJSvL1oDyb1dXoGx+sUdr5HI+CD3QTz+JqrJFphXwaA1sn6ni/hDBGhTFoXUC74avPN+4Ojhc6
4gPIAphvgbIsME0CqqwW2HSnEBel2v+6+LywEAHRPd84Ytfbqb+++EL00C53uKhGd9zG/S7QtSvU
waF4DP/ZwIf9sGb7dkIfXru2Op37c7mg3I6XudxucoTrKlA1mwzN0hgkgL8/fhcvfXoHs3FOqdrm
Y8/X91Ckxpl7aGb1bNdJdlLy4cxOXttFdmvo0rugoclRJAgCS8ii6K3NnFdqm1jM9E6I2yZmZsKA
zTAB5b4k9VH/ToTOq9tX77AL/vBRiv+WZUXj5qgtwx6BMt+d69Zru6JrmXDxd++QIjOmm7yHjjfI
qe7IvOcimSgpi0j5grEbdWJvqQeYvlAvYgVgMlWFBZ5gg0KFpom4OFXkmXj8+Btf2aeFyXElSnRj
8fP9Fjv3YWkRzyiaCr9YdBOB0ubOC/7ke9eKiER1Fkiuj74L+85YjOFHkmmBWPueDI9r5fG5fYBs
6wwQBGxHOlfsnn7mMP3zFK6oEzpwrx7M/2aNWzapOH0yOFp5HcVklmtSPs3u++GByeWX+n3hnsbO
sXO70carjcoHt2z28YZkBz9AZJggQEUrKK/16zprUEXVH7skSPPfzDhyUihqHew4XJVf0TAMAXPj
iIgDA8aIyFJ6vxNr9pzx9TBelmAPAfZ9vUO13aW43hfnj36QmWOdfi2HCCCEEzvt5hu4PThaXo6C
43LFvVjHjYNCqsWtKYdnfv7ZFrn/ectSxXVFVjY0HzwTUu0WGTS4MEwQqqwW+XZ8X7GG4UC0EtuQ
5F2zNTvcaSDy9bZRWVDs8jTxRVS1cFrSd56hCfIwN/V5YiFMl5bp9zMY6/1oME0acr6iCTSOWwzo
Ev/JPtRCqQhpWsqF7WjT25L//6vxdmJ1tL9rWaeqvuOrdlbr9Lnj2NwDOwtnBgAWdYYLAcz/tQMv
G6uCdULFXK4dhJUHXqxb08lX2wtueBP3txTn4u5Rv3lPUZ3llNU4UlQa8bqmS0PgFMUwEoLTTHj5
zHieK1EWefZWBNyb2cAhTbakfr17ESwigD7XZxOjkdI+XzCznyioxZ/MRlwyJSnILSlzahmreqvq
+sBCF5ytqOVLfsc6/FsOEOPnFagF/8Zc/Vfu+rsuN75Hm65BcyMpoiKV7nKl8zU5cFUi0b0A9WzZ
AXSkA6OLXlwIAjozpiLuGOPPm7FeI+KxVuarPT+rbAlvyCt1er09G4JOa8QydYQLAufJK9H1+oe6
xIVJoeG09LWrDoLbJAID/+3Zr0Bb0zGbxn+oSfzFRUSgqVc+m88+qH7q7jjmLX3y8Ild1bFDbFGw
/4j4r5oc39/FxpL7Dgs4wAaDgrHuRvKl/KwCea4aazeNjOzq2mybFe+gZvhwtVEd0RtsIJWXqWID
/LN+v2n/gy41uz3fqnOcHX1LdwPqgAyzoDBN0ZF/v4Mk5VJ4th06d99AzfHlgRFvV/qOXdEC7Jok
eP3OJh2RnJ4pKVbl0QgEWdIYJAH8ec/9q3jujIXq2bBDwvB9KLBQ/5Batm2jb1Hro39pztSI9zxi
ZKCJ8d/dAXN3bNSNli0bBST9snRUPhglhlLZYLfXjY/D9PQMDOBpXrFAz1Bv149Vy5y1o6mxphSq
vuZYoIsRGRyFOx3umX6Y5Ze28wTN0hgkAyhqhE3qYX/29puiF+H98W+AX9cykp6ZIiLvZt7w4XVr
hOQtj43qxAIDbBmao2v8+ugOmSWmQAw0LOsMEgN92HHNsRyvc59Y+OSIYwzHEsA6pwR6CqbjLUyM
HD3mrZHR9X3uu+LapidjyzChH+w39Whkq6G0GLOgMEwBmZzvztihzjKcmhoZHSTA9bAKJsuC1OzO
TXG/0yBnn4nByPddEW1GKf0dlXVKtaSeQsKAzTIDp3qJBsIfgQoToOT6/sx9GdPT85vHb9uMAgFk
rDzja9OzkSpQP6WCuR/CiKMOYSEVVNYa9+oeq7Viha4Wb4BMZih4XbcPMm3rr5mQZ2iEFf+zKx6N
fb1Z5rlzTOx1fKzJjLnlsqEtZOVnEg5EyVwkLOsOYyLUzVyHvtPPV/cmxnfDF2oNBHJE+QcwnFXC
ibVGqdQwZbd6aOZuPALBXHZIFvWtafbRqnKB73m/vHoCWjVxzrgcSNrkEgO1HCpH5+FyVTY6JDDY
eVOfOvnNwa1zZy3+5V/xFJPihe2PuVmeRCmWdUOX6wv+N7gh39G7VEClBnqGzoAeA/60+ACGAxbt
OeO/MWJ7LutvdFl++2j8ZEv1Bm5R6wR5C0EiW3A8v7+50J1WaZIQAmta3L14HuqRcTWFBDwBfrLU
nx7dqBXWmZrRqnIDc6eNwbVaLYA/FgXYhz0ql57whf/cnxnZytCkXievG2hwz71BfPDYk6EQ0moh
2EVEOEU3ROf4wEe0goi1EtJCIWvl/qOFPtIXKeTE1R86qGA7UCZIfdTCwkZxLR6BbWhK6ptVHhWK
G3jqlHqZN6Iq+mY3Qoann3C/BxqvCEJENwDsAxgDoDGASEXXWdNsIIEsIcQGAbwC87O+BWoFImvV
Ymb35xVi196THPkIIPPvTdkVacVUAABb7SURBVFXb/cPbuunNBJP4GLsMVgsgMT4a2w4XYeaSfao
+3Vs0wOy/Xhi0gCGjGJky9gWQI4TYJ4QoB/AlgPHKDkKIxUKIc9LuagCu2WoYFvQAIoTADg+5qmv
DiNeWYNIHqz26H+46ftYlf3aoi4GSYC/uBZKPb+uLe4e1QfOkeKyUHtRfSTVEww0jgp4GQPnt8qQ
2d9wB4Be9A0Q0mYiyiSg7Pz9fr4ulYRt64Pjf6gMY++YyrMwpMO0aRwvdey2NnrHMpS2cHuhTL+3
kvZNFyExOwGOXdLREgjK/GnWJ6EYAWQBe0TsuhHhCJElhMhKSkqe6K/iAqjP+hwZ9Mhe2mw6z9
cg3s/2+C385ZVOosJN07Qn8X+4cabKdQX1JRoA2eY8MCIoB8GoFyOT5faVBDRSABPArhcCFGmPc4
Af/10fbCHEDEo63cq/Ytri7I8W4Wb5E23frxOtT+wrT2VajjN0MNnpIwSI4/hdQDaEVEm7EI+EcD
1yg5E1BPATABjhBAnfO6lYR1PATACjhRDsbM0Eja15hVi6Jx9Ld5tj0rvvc2dJsnNlVR56Onl7Ui/M23Y05D0klFjB/OA
LKYlxqrqiix4ZEsTR1ByvM3QhRCWA+wDMB7ATwGwhxDYimk5El0rdXgGQAOB/ItpERHNDZnEYhmEYQ3x9fh1h6vQq07/wvUvFfv5ZZ9bmddp5IzzaUVfunpyQzi11fwcFdqerlAz5+CU1DZ4G

```
8shrzt9vT01729nK8Mn+XSx9/FWBef+C0Y9vdDF1Lw4RY3NAvvDx5I1TP8dEtzhzwzZPi0TrMAq4
MGcqEEPMAzNO0PaXYHunncVmWE0WlSK0fGilTrcJLv/6J/yzfj68m93fbp7JaeCwn5gu7jp1FL03
RhO825Kn2P72jr1+vGQi6NK+Prs2Tgj2MoNAt3fm9K8MwwQ1HugSYiR+sDvYQLEfOiWIAwLly9ya
QCp3sejVl48HTqv3Hv9sKACgpq3S4MD48e7Oqz0Xtwm/xf+4DF6FObPi4WJpFUWlFsIdQY3gp22T
KK9VCsi+/BLd+vBaf3BZ+M7dQRV6oPF7k3i+881Pz8e3dF6J3q0Y+X+ftRTkubQNeXIgjkpiP15S
W++vg1j5fiwk+3krQhSI8QzeZ4a/94dL2x67I88E3k62H7S6K7y7Z67HfVe+uqtV19N4AjiiCi37
cdMSxPbFPC0wZ4z4zH8OYAQu6yShzYTP+RwjhyIA3umtTv5+/tKIKfZ9fgIwpc5GZop8HW4/nr+g
WsZ4iTPBgQWfCmn/9vhuF5+22Tm3+DT2qqwVe/GWnIzd9aUWVRw+Yez7bgBOSG9uhU/bsFtdmec9
sEU4+5zIcyRz+sKCbSGmFMT9lxnfe0rFry/z3dtd1ii2HCzFzyT4MmL4IJ4pK0XHqry45VwDgp81
HcOJsKRb96QyrWLanAETAyE5NPI5JTvYUbqx+YgT+eHRosIfB1ILw/OWFCSVllcEeQkQzqG0yerV
UF2Se8M4Kx/amQ/ZqQj9uPqLqU1Ragfu/2IhbPlJHfAJA3RgboryYUj68uY/H46FKcr04ZCQbNyt
ZnVAvZqEHC7qJhJ8Xa3hx/xcbPR6PiiJc46GIxGQpFUOxxj1tjrS4ufOoa7ZGIoI2rf2yvw9zbHd
uVh99M333pGFCh8T48HMCZEE3EaVpNtxCiMOBnzQza8A1wnFS35Zez3PmnFPQ1x84jX/8sM1t3+K
ySpAm00mLRnUdJcpmTOyB2Gj+s2KCQ9j98o4XleK133ahOgyiuJSLbZFQraiiqhoDpy/Ch8u8L07
Wlko3gUJCACumDMfP9w8yfK5LFN4xRee9B5P0zmjo0nZ9P/uDIzWC8ogzoUfYqcztn6zDW4tyHLk
7QhnlIychztqRdz9tPoJuz8zH4TPnMW3uTtOvV+IhKjStQR10TTMeut4k0ZmKQS86cGSnVMd2Vqu
GqB8fgw9uzlL1+euQ1lj292FoUDfW8HWZ0ORxKX6gf+vGQR5JzQk7QZcXGs+GwYJjleItonE968z
cVuQU4PM1Bx37+/KLcf8XG1WRddoIWX9SXS1wsrhmGZqT6sRgdBd9P/UqRZKt2TqVaj5UJGwa260
ZAOCYJio1LtqGFo3q1mhMTGjy1yFtsODhwXj+iq7BHkqNCTtBf/ryLgDslbgDzdr9pzDuzWW67oh
CCLy5cA/yTp9ztMmC/vLVFwAAXr+uu8fzV1WLkHd1PFlchhs+XIMnvt/qMCkNf22JS783F+4xbQy
tn5ine01A32Uwd/o4bH56FN67qTdevLIb3ruxt+p4heLBW6xJifvGxB4AgLkPDEKT+nG4ob/dtHJ
1r3Rc1Ssdm566uFbfhQlN2qYmIi46/N6qw07QOzWtDwAoOh/4GfpTP27D9iNF2Jdf4nJs1spc/Ov
33fjLrGxHmyzocsDGFT09B6Tc/NEadJz6K054yEkSbN5e7PT7XrP/lKF+ZtIoIRYvXNENHaVc419
OvtBj/0l9W2J016b45cGL8NaknrBFEQ4ronlPlahn/p2a2X9vXZonYc0TIx1/5HVibXjt2u5sYmF
CirATdNmVqLgs8JnQZP/jap3Iwmd+2gEA+PPYWUdbldRPL2pQLzpxRY69QO2/ft+taq+uFvh09QG
c92A3DhTKMazMKXBr+riql+vD68TZUlXFH3/w39v74vp+LfHujb3x0Mj26J5uzHbeqVl9XNa9Oaq
qBeZsPuJ4iCZLprEm9eOw/P+GoX2T8ClKwTBhJ+iyS1hNbbRV1QIPz96E9Qfczyq9YdSvRvbAkD1
x9AJRPly237G9au9J9Prn7479L9ep7bitn5iHqT9sw6PfbMaslbl47OvNjnD3QPOrYjH6zUU56D1
tger4FT3t9cO/3ZAHIYRqnH2fX6j6nv6gYYJ9hpyZnIAHR7bzOX+KXO39zLkKXNi6MRY8PATpDdk
mzoQXYSfo0VEEopoL+qZDZ/DdhsO1yrgn59T+dNUBVbt25rxHys/taYb+/LydyJgyF8/M2Y5JH6w
2NHOdu+Uonp6zHV+vz8M7ATJpaFH6bCvpk9EQL1zRDa9I6wUAkPn4PHR/9jccPHlO9zO+MGOB+u2
lqZ+Khfztq03ILSjB/oIStE2th8T48IsSZJiwE3QighDAwVM1E4kz55yC6Ws5MrmQwlfZh/D0j/b
gkz3Hz2Jz3hlVvzFvLAMAVFapbeh6fLIyV7ddnt0v2HFc97jsjbFyr93jxOh3qqyq9jklgadrdGm
ehOv7tUS0zfUn9cjXm5BzwmmKypgy16frA8CMBerFVn8mwRr66h8AgBV7C/x2ToYJJGEn6DI/bHK
NEvTEHYrFyt92HFe53RlB630ya9UBPPvTdlz8+lJMfN9ehahdqrP+YHFZJQok+7LRyMGf7x/kWNw
rPF+Bo4Xn8Zf/Zuv2PXOuAu8szsH1H9g9TjIfn4e/fek5FB4AnpqzHV2eno+MKXN1BbrDP35BxpS
5LhV+zpdXOb6nHp5yf6/LPY2R/1rqdWzeWKtZhL1naJvan/PJES5teoveDBMOhK2g14Sb/rNGtf/
XT9fjie+3OlKoGuHh2Ztc2rRZ+l5SmBsGv7wY93y2AQCQW+BdIHKnj0PXtCRHJr+e//wdF764yHH
8oZHtkTt9HHKnj3O0aYsh/7DpiFcPGeWDrKhUPVMvq6xCmWTKum7mKsxamYtuT8+HEAKvzN/l1qu
lb2YjxMc4XbxeuqqbxzEAvr0lTf7U+XBb++QIPDKqQ43PoSU10dVkM8aEvOoMEwgsK+jj3lyGEa/
9gYwpc7Fsj/0VelRnddrTAdMX6X3UhdEzlmLeVu+Rqd3TnZn9TpWUOyrc9FEka1r7hHpGeEF6Em4
dkOHYj9ExWdxyYSs8MKKtY3/3tDFux/D6gt1Yve8kftt+DGWV6rcKra3/WGEpPl19AAdPnsObC/e
gwz9+dRzbcPAMnp6zHWfLKrE8pwAfrdiv+uwP9w4EAPzvjn743x39VMeu6+M9f8rxIuOBQVXVAh8
u2+cwQ029tDNSE+NNyzn+7xt6mXJehjGb8EsnBuDC1o1x/Kz7magQAtuPqDPldU9Pwrs39kabJ+a
p2vflFyOjcQJaS+3KGbCM0hXx5/sHYW9+MR780jlj/+PRoSguq4QtivDNXRfi6vfUC69dFBXUUxW
LePP/Nhgdmqrd4u4a2hqvaxb+nh2vjljTmnC2PjMK367PwzM/7cAXaw/hi7VOLxnl93lQY5K5ZIY
xM8hN/1mr2h/SPgU9WjTQvVcyvz80GK/+tgvzt9vXAKKjCGO6NYON7G8S/V9c6PHzSlbkFKjSCdw
+MMPQ53yFKw0x4UpYztCPFJ7HvvwSt6/th3VMKd/fMxC2KHIxWwx/bQk+W+P0WnlhnjoPidbNsWt
aEsb3SHPsr5gyHBnJCY7cIVqBbu0hv7S2LwCX6LTLuzd36QMAqx4f7thOjI/BrQMz3V4HsHsF/eZ
mgVXL30d7NmVM6us+Ja1MuyaJmHlTFhY9MgRTxnREzgtj8daknnj6si6OPn+Z5ZpvXI9//rxDte9
vwW2e5HzIDmqb7NdzM0wgCUtBPyC5wVVUCWRMmYu7pLzWMloPmM1Pj0KU5vVcNhkAwNQftzu231+
qzhSodHNspvjDnzahK24dkIG0BnVU/evFqV969unYz2Ojo5Bcz32E4donRuDj2/pgz/Nj8Oaknrp
9miXVcXk4ZbVyzQIoZz48rfDycXfOR0e1x7wHLlI9sPTo0cL1Ou5onVIPdw1xLl42TIh1iOaCnSd
UphR3yG6gAHBlL89j84XJg1sDAH68dyBm3tTbS2+GCV3CUtBlSiUb8a/bj6GguAyXvbUcuQUlLn7
iepVHtMKrJGPKXBTruPbNf2iwY/vG/q3wzOVdXPoQES69oJnHcW99ZhRWTnH1rpBJrR+PYR1Sde3
pnvj6rgsxvGOqqm3a3J3YfOiMakHT3az/vuHt0Ll5faQ1qIM4yawjR3xunHoxtj4zCsv+PgxNk2r
n+708x+kWOG3uTjykWHDecPA0ftx02LGvTZP70Mj2tbq2HrcOzETu9HHo3qIBEjz8Lhgm1AnrX++
BAudMPEuKWJR9iQF7Miw3abN1k3vdNjDD4bnS9en5mPuAM6e2UXsvALx9fS/8vMXua31jf9cFQrO
S/hARPrq1j8PFsuNU+yLneEXZtXeuty/47X9xLAqKy5GSGIeteYVo2VgdFblLsfD62rXOpGL+CLi
Z+8AgjHtzuWP/x01HMPXSzkiuF4cr/70SgP2hQ0Q4qQi4Gt+jOZpr3ogYhnESljP0we1TAABr9p/
```

```
02O+Knum4urd+QizZzU4p7PcNa6vqoxSdmjLjOnuWvpv6Z/h8Dl+Jj7Gp3AiVXCx5+hARUqRiDN3
SkwJaP7GzlPBKSda0BSqXSjly9ou19rYreqbhjYk9TfNsYRgrEJYz9Kt7p2Pp7nyPhRSaeTELNEq
IxYzremBA28aIIkLDurFwpxWPXVJzf+cJPdMwoaf/7b21JRTKoxER9r4w1sXj6Invtzq2e09bgNz
p4xyRof8Y1ymgY2SYcCT4f90+YGSOtupx9zZqmQk905CaGI/kenGwRRGICDNv6o2pl3ZW9evS3HV
GGQ68eKU6wEe7gBtMjMy0v84+hATpDcpKBUIYxizCcoZ+iZvKM/Xjo3H30Lbo2bKB7vGanFvpKte
qsXvXw1BmUt+WGNExFVVCoFqElqADdtPZ0t35bo8/9s0WAECblPC8/wwTaMJyhu7ObNAsqQ7uHtr
GL7UA5ejNT27rg0wPvuShTmr9eDRLqhNyYg7Yc5lrizmnNaiDhy9We7Ls5dwqDGMIQ4JORKOJaBc
R5RDRFJ3jcUT0lXR8DRFl+HugRth1/Kz3TgaZMqYjVkwZjqEdUr13ZnxGWcz5+n4tsWLKcHRv4fs
bFsNEMl4FnYhsAN4BMAZAZwCTiKizptsdAE4LIdoCeB3AS/4eqJZHR7n6I/vz1Tw+xhaSs1orsvm
pUbikSxNMk1IcaIOuONiHYYxhZIbeF0COEGKfEKIcwJcAxmv6jAcwS9r+BsAIMjkhRp1Yu/n//uF
t8fmd9uRQ/9UkiWLCg6S6MZh5U5YjmldZ9i13+ji3ayYMw6gxsiiaBkBZEy0PgFY5HX2EEJVEVAi
gMQBVpQAimgxgMgC0bOk9I58nbujXEseLSnHXkDZIiIuuUeAPE9rE2KL435NhfCCgi6JCiPeFEFl
CiKyUlJRanSs+xoYnxnbiUG2GYRgJI4J+GIAyvV661Kbbh4iiASQB8BzGyTAMw/gVI4K+DkA7Iso
kolgAEwHM0fSZA+AWaftqAIuEr4U7GYZhGJ/waq+QbOL3AZgPwAbgIyHEdiJ6DkC2EGIOgP8A+JS
IcgCcgl30GYZhmABiyAAthJgHYJ6m7SnFdimAa/w7NIZhGKYmhGWkKMMwDOMKCzrDMIxFYEFnGIa
xCCzoDMMwFoGC5V1IRPkADnjtqE8yNFGojC58n4zB98k4fK+MYeZ9aiWE0I3MDJqg1wYiyhZCZAV
7HKEO3ydj8H0yDt8rYwTrPrHJhWEYxiKwoDMMw1iEcBX094M9gDCB75Mx+D4Zh++VMYJyn8LShs4
wDMO4Eq4zdIZhGEYDCzrDMIxFCDtB91aw2uoQUQsiWkxEO4hoOxE9KLU3IqLfiWiP9P9P+GUjsR0Zv
S/dpCRL0U57pF6r+HiG5xd81whYhsRLSRiH6W9jOlIuY5UlHzWKndbZFzInpcat9FRJcE55uYCxE
1IKJviOhPItpJRBfy78kVInpI+pvbRkRfEFF8yP2mhBBh8x/s6Xv3AmgNIBbAZgCdgz2uAN+DZgB
6SduJAHbDXrz7ZQBTpPYpAF6StscC+AUAAegPYI3U3gjAPun/DaXthsH+fn6+Vw8D+BzAz9L+bAA
Tpe33ANwtbd8D4D1peyKAr6TtztJvLA5ApvTbswX7e5lwn2YB+Iu0HQugAf+eXO5RGoD9AOoofku
3htpvKtxm6EYKVlsaIcRRIcQGafssgJ2w/9iUhbpnAZggbY8H8F9hZzWABkTUDMAlAH4XQpwSQpw
G8DuA0QH8KqZCROkAxgH4UNonAMNhL2L2IOuN4jvSLn4wF8KYQoE0LsB5AD+2/QMhBREoDBsNc0gBC
iXAhxBvx70iMaQB2pKltdAEcRYr+pcBN0vYLVaUEaS9CRXuN6AlgDoIkQ4qh06BiAJtK2q/tmCBiAJtK2u3tm9Xs
5A8DfAVRL+40BnBFCVEr7yu+rKnIOQC5ybvV7BNhnifkAPpbMUx8SUQL496RCCHEYwL8AHARwFA4A
A1iPEflPhJuiMBBHVA/AtgL8JIYqUx4T93S5i/VGJ6FIAJ4QQ64M9lj

g0n9PACCtIYyH/QHYHEACQvANJNwE3UjBastDRDGwi/lnQojvpObj0qsvpP+fkNrd3TMr38uBAC4
nolzYzXLDAbwBu3lArtKl/L7uipxb+R7J5AHIE0Kskfa/gV3g+fekZiSA/UKIfCFEBYDvYP+dhdR
vKtwE3UjBaksj2eH+A2CnEOJfikPKQt23APhR0X6z5J3QH0Ch9Co9H8AoImoozT5GSW1hjxDicSF
EuhAiA/bfyCIhxA0AFsNexBxBxwvUd6Rc7nAJgoeSxkAmgHYG2AvkZAEEIcA3CIiDpITSMA7AD/nrQ
cBNCfiOpKf4PyfQqt31SwV499WG0eC7tnx14ATwZ7PEH4/oNgf/3dAmCT9N9Y2O1zCwHsAbAAQCO
pPwF4R7pfWwFkKc51O+yLMjkAbgv2dzPpfg2F08ultfTHkwPgawBxUnu8tJ8jHW+t+PyT0r3bBWB
MsL+PSfeoB4Bs6Tf1A+xeKvx7cr1PzwL4E8A2AJ/C7qkSUr8pDv1nGIaxCOFmcmEYhmHcwILOMAx
jEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWgQWdYRjGIvw/IlbURK54GdgAAAAASUVORK5CYII
=\n"
          },
          "metadata": {
            "needs_background": "light"
          }
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "training_size=int(len(data_oil)*0.65)\n",
        "test_size=len(data_oil)-training_size\n",

"train_data,test_data=data_oil[0:training_size,:],data_oil[training_size:len
(data_oil),:1]"
      ],
      "metadata": {
        "id": "C9pYgejLAPns"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
 "source": [
        "training_size,test_size"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "k89wWlnmAo6U",
        "outputId": "af336bcf-8a8e-4389-c233-ac26c57a61c1"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(5340, 2876)"
            ]
          },
          "metadata": {},
          "execution_count": 14
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "train_data.shape"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "hvzEBVyUAr-T",
        "outputId": "a356f237-c919-4164-e01f-a7463275449c"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(5340, 1)"
            ]
          },
          "metadata": {},
          "execution_count": 15
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "def create_dataset(dataset,time_step=1):\n",
        "  dataX,dataY=[],[]\n",
        "  for i in range(len(dataset)-time_step-1):\n",
        "    a=dataset[i:(i+time_step),0]\n",
        "    dataX.append(a)\n",
        "    dataY.append(dataset[i+time_step,0])\n",
        "  return np.array(dataX),np.array(dataY)"
 ],
      "metadata": {
        "id": "-YpjHQdPAu1r"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "time_step=10\n",
        "x_train,y_train=create_dataset(train_data,time_step)\n",
        "x_test,y_test=create_dataset(test_data,time_step)"
      ],
      "metadata": {
        "id": "DDWunzVjBOs0"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "print(x_train.shape),print(y_train.shape)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "5hHUEc_WBhOZ",
        "outputId": "aea25456-6e97-40fe-ebae-33153be8ec20"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "(5329, 10)\n",
            "(5329,)\n"
          ]
        },
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(None, None)"
            ]
          },
          "metadata": {},
          "execution_count": 18
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "print(x_test.shape),print(y_test.shape)"
      ],
      "metadata": {
```

```
"colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "-rAmRqsbBnlg",
        "outputId": "79d03fcb-a209-423c-c2a1-d240936035db"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "(2865, 10)\n",
            "(2865,)\n"
          ]
        },
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(None, None)"
            ]
          },
          "metadata": {},
          "execution_count": 19
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "x_train"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "zd-z5Ib8BwGP",
        "outputId": "90542887-8fc2-458c-b22e-929f8c0bd41c"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703, 0.11661484, 0.12053902, ..., 0.10980305,
0.1089886 ,\n",
              "        0.11054346],\n",
              "       [0.11661484, 0.12053902, 0.11550422, ..., 0.1089886 ,
0.11054346,\n",
              "        0.10165852],\n",
              "       [0.12053902, 0.11550422, 0.1156523 , ..., 0.11054346,
0.10165852,\n",
              "        0.09906708],\n",
              "       ...,\n",
              "       [0.36731823, 0.35176958, 0.36080261, ..., 0.36391234,
0.37042796,\n",
              "        0.37042796],\n",
              "       [0.35176958, 0.36080261, 0.35354657, ..., 0.37042796,
0.37042796,\n",
              "        0.37879461],\n",
              "       [0.36080261, 0.35354657, 0.35295424, ..., 0.37042796,
0.37879461,\n",
              "        0.37916482]])"
            ]
          },
          "metadata": {},
          "execution_count": 20
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "x_train=x_train.reshape(x_train.shape[0],x_train.shape[1],1)\n",
        "x_test=x_test.reshape(x_test.shape[0],x_test.shape[1],1)"
      ],
      "metadata": {
        "id": "BsJsc-dzBzPI"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "from tensorflow.keras.models import Sequential\n",
        "from tensorflow.keras.layers import Dense\n",
        "from tensorflow.keras.layers import LSTM"
      ],
      "metadata": {
        "id": "Nti1iMdXCFHH"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model=Sequential()"
      ],
      "metadata": {
        "id": "5Bbk8sA6CWWd"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model.add(LSTM(50,return_sequences=True,input_shape=(10,1)))\n",
        "model.add(LSTM(50,return_sequences=True))\n",
        "model.add(LSTM(50))"
      ],
      "metadata": {
        "id": "gciW66VECcKl"
      },
      "execution_count": null,
      "outputs": []
```

```
 },
    {
      "cell_type": "code",
      "source": [
        "model.add(Dense(1))"
      ],
      "metadata": {
        "id": "PI_o_1SNC0jV"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model.summary()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "y_QQbjo2C7z0",
        "outputId": "ecd997a8-231d-4d4c-c495-fc020a381466"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "Model: \"sequential\"\n",

"_________________________________________________________________\n",
            " Layer (type)                Output Shape              Param #   
\n",

"=================================================================\n",
            " lstm (LSTM)                 (None, 10, 50)            10400     
\n",
            "                                                                 
\n",
            " lstm_1 (LSTM)               (None, 10, 50)            20200     
\n",
            "                                                                 
\n",
            " lstm_2 (LSTM)               (None, 50)                20200     
\n",
            "                                                                 
\n",
            " dense (Dense)               (None, 1)                 51        
\n",
            "                                                                 
\n",

"=================================================================\n",
            "Total params: 50,851\n",
            "Trainable params: 50,851\n",
            "Non-trainable params: 0\n",

"_________________________________________________________________\n"
]
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "model.compile(loss='mean_squared_error',optimizer='adam')"
      ],
      "metadata": {
        "id": "9-EcjJXFDAvc"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [

"model.fit(x_train,y_train,validation_data=(x_test,y_test),epochs=3,batch_si
ze=64,verbose=1)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "Jmk2fEa4DItV",
        "outputId": "5533ed7e-ab81-4b5a-a0bb-e86e0399a05d"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "Epoch 1/3\n",
            "84/84 [==============================] - 6s 25ms/step - loss:
0.0017 - val_loss: 0.0011\n",
            "Epoch 2/3\n",
            "84/84 [==============================] - 1s 16ms/step - loss:
1.2375e-04 - val_loss: 7.8338e-04\n",
            "Epoch 3/3\n",
            "84/84 [==============================] - 1s 16ms/step - loss:
1.2058e-04 - val_loss: 7.5010e-04\n"
          ]
        },
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "<keras.callbacks.History at 0x7f2e2b3da490>"
            ]
          },
          "metadata": {},
          "execution_count": 28
        }
      ]
    },
    {
      "cell_type": "code",
"source": [
        "##Transformback to original form\n",
        "train_predict=scaler.inverse_transform(train_data) \n",
        "test_predict=scaler.inverse_transform(test_data)\n",
        "### Calculate RMSE performance metrics\n",
        "import math \n",
        "from sklearn.metrics import mean_squared_error\n",
        "math.sqrt(mean_squared_error(train_data,train_predict))"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "vtdx97vxH4hF",
        "outputId": "dd7ecc10-c73c-4005-f15b-4bdfc7d710f7"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "29.347830443269938"
            ]
          },
          "metadata": {},
          "execution_count": 29
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from tensorflow.keras.models import load_model"
      ],
      "metadata": {
        "id": "BBanpJ69H6Hh"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model.save(\"crude_oil.hs\")"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "jYCE9HozIqOa",
        "outputId": "8d32e61b-d127-4314-9645-234793b3be28"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stderr",
          "text": [
            "WARNING:absl:Found untraced functions such as
```

```
lstm_cell_layer_call_fn, lstm_cell_layer_call_and_return_conditional_losses,
lstm_cell_1_layer_call_fn,
lstm_cell_1_layer_call_and_return_conditional_losses,
lstm_cell_2_layer_call_fn while saving (showing 5 of 6). These functions
will not be directly callable after loading.\n",
            "WARNING:absl:<keras.layers.recurrent.LSTMCell object at
0x7f2e2f939a10> has the same name 'LSTMCell' as a built-in Keras object.
Consider renaming <class 'keras.layers.recurrent.LSTMCell'> to avoid naming
conflicts when loading with `tf.keras.models.load_model`. If renaming is not
possible, pass the object in the `custom_objects` parameter of the load
function.\n",
            "WARNING:absl:<keras.layers.recurrent.LSTMCell object at
0x7f2e2b60bf90> has the same name 'LSTMCell' as a built-in Keras object.
Consider renaming <class 'keras.layers.recurrent.LSTMCell'> to avoid naming
conflicts when loading with `tf.keras.models.load_model`. If renaming is not
possible, pass the object in the `custom_objects` parameter of the load
function.\n",
            "WARNING:absl:<keras.layers.recurrent.LSTMCell object at
0x7f2e2b510cd0> has the same name 'LSTMCell' as a built-in Keras object.
Consider renaming <class 'keras.layers.recurrent.LSTMCell'> to avoid naming
conflicts when loading with `tf.keras.models.load_model`. If renaming is not
possible, pass the object in the `custom_objects` parameter of the load
function.\n"
          ]
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "### Plotting\n",
        "look_back=10\n",
        "trainpredictPlot = np.empty_like(data_oil)\n",
        "trainpredictPlot[:, :]= np.nan\n",
        "trainpredictPlot[look_back:len(train_predict)+look_back, :] =
train_predict\n",
        "# shift test predictions for plotting\n",
        "testPredictplot = np.empty_like(data_oil)\n",
        "testPredictplot[:,: ] = np.nan\n",
        "testPredictplot[look_back:len(test_predict)+look_back, :] =
test_predict\n",
        "# plot baseline and predictions\n",
        "plt.plot(scaler.inverse_transform(data_oil))\n",
        "plt.show()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 265
        },
        "id": "L-6vXuAYJkcu",
        "outputId": "c930fefd-934e-4926-89ab-930666200083"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
```

```
  ],
          "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXcAAAD4CAYAAAAXUaZHAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO2dd3wUZf7HP9/dTQ+QBELoJHRpAoY
mAtKkeWI7X9ixoaee56Gn2M92h738bMehiKfiKaeColRBpBt6hxACCTWUBEJI3ef3x8xsZmZns5v
tO/t9v155ZeaZZ2eenWw++8z3+RYSQoBhGIYxF5ZQD4BhGIbxPyzuDMMwJoTFnWEYxoxuDMMw5g
QFneGYRgTYgv1AACgSZMmIjMzM9TDYBiGiSg2bNhwUgiRbnQsLMQ9MzTk5oR4GwzBMREFEB10
dY7MMwzCMCWFxZxiGMSEs7gzDMCaEExZ1hGMaEsLgzDMOYEBZ3hmEYE8LizjAMY0JY3BmGwZwNhdh
+uCTUw2D8SFgEMTEME1oe/WYLACB/2vgQj4TxFzxzZxiGMSEs7gwT5XA1NnPC4k4wUU5VDDYu7GWF
xZ5gop9puD/UQmADgVtyJ6BMiOkFE2w2OPUJEgoiayPtERO8SUS4RbSWiPoEYEYNMMw/qOqmmfuZsS
TmfunAMboG4moNYArABxSNY8F0FH+mQzgQ9+HyDBMIKnimbspcSvuQojfAJw2OPQ2gMcBqF/cCQA
8JiTWAkghouZ+GSnDMAxTL3yyuRPRBACHhRBb2T2GXsIkuzDoVJsnnDMAGhqobF3Yx4ZXMnogkADgshtugOtQRQoNovlNuMzjGZiHKIKeoqMibYTA
M4weqeUHVlNRb3IkoEcCTAJ715cJCiOlCiGwhRHZ6umGVKIZhgsCmguJQD4EJAN5EqLYHHkLYHkAVgCxE
BQCsAG4moH4DDAFqr+raS2xiGCVMemr0p1ENGgAkC9Z+5CiG1CiKZCiEwhRCYk00sfIcQxAPMA3CZ
7zQwAUCKEO0OrfITMMwzDu8MQVcjaANQA6E1EhEd1VR/efAOQByAwbwD3+2WUDMMMwTL1wa5YRQtz
o5nimalsAeMD3YTEMwzC+wK6QEMwzC+wBGqDMMwJoTFnWEYxoxoSwuDMMw5goxs7qblpY3BkmiqlhcZxZ5goxs7qblpY3BkmiqlhcTctLO4ME8X
UCBZ3s8LizjBRjNCJu36fiVxY3BkmivaltFX2GMzjBkitFX2GMzjXlgcWeYKEYv5r/u5ZKXZoHFnWWimJmrDmj2V+07CQBYtGSwsfqtFnIOg2Ggbk2YM5ELizjBRzDu/5DodI5JsSvs2HOmtFa
Oh2gkjL9hcWeYKGb+Nm2htCq73UPPJtJgcWeYKKZKZ3wGmvKzUPCBWXkKGa9qqLjsVKBa9Cq73UVPJtJgcWeYKKZKZ3Svqmabu1lgcWeYKGZg+yaafb3YM5ELizv
DRDEjL2qq2T90ugznK6pDNBrGn7C4M0wUo/eWmLV8dgDkPoakSV6cH9SMFb7nAvyCO9OQ0wVkCwSiDHETJVIWWRVcAPutUx8dkPTcSfJVAw1ZSbgXWYD7LtHnS
T0VYi+o6IUlTHniCiXCLaQ0SjAzVwhmVF8RxH3EV1qZ/C7j50L1XAYP+LJzP1TAGN0bYsBdBdC9AS
wF8ATAEBEXQFMBNBNfs0HRGT122gZhvEdjki9f5hUI8EsbfuBV3IcQKAKAKd1bYuEEIphJn
2BABfCSEqhBAHAOQC6Of8H8TIM40eU9dM4G1tozYY//qJ3Avh3m4JoEB1rNBuc4KIJhDhRDhlFBV
xVBzDhIJXFuwGAJTyIqrp8EnciegpANAqZ5SvDa7mYsq9Kq2Ln1lwSYseidRYPyMV7k8dKo5JFJSbd4Nh3...
```

```
1xi4cUaNWC2HqmIscx9SRq3ddluXYjouxYKmuhuq4d34zPH+/rDTpNbqqTPp88JOHtPNi9JELizv
DRAnuqiEFKoip/ZM/4aX5u+RrAElxtSKsRK52VXnMANLMXc/Oo8Y52q9VLdC+M7GXY1sf/dqnTUo
9Rx7ZRLdRimGiCHdJFtt5uHjpCxYiTSIqJVp0bPdmmn77i0o9PufEfm0c2xN6tUST5DhM/XYrOqQ
30PQb3a2Z/qWmhmfuDBMlNEmOrfO4xUK4c1AWkuMCN+ezEKFHy0aO/ds+WQ8AsOpm2WfKPCtkbVR
8Y1CHJvjtseEa2z4Q2PQK4QiLO8NECS1SEtz2sZCzP7wvqBdFAcBqAZokx+F/f7pU026rh8E/VSX
aE/u1rqNndMPizjBRws/b3UdmWizkV3F/ZcFuzb4ye9ZHx1p1fpjpDeJcnrO8yo5HRnWSXlfP2fi
dg6RFW+HH9xiusLgzDOOAqDbgyB/M3XxEs68ESulFecVebU2HiX1dz8hbpiY4ArLqG6ikuEtGQ1p
3FneGYRxYiLwoveMafVZHZcKun7n/tk8r7nXZx6dd28ORiEzv/ugO5bru3ELNAIs7w0QZF7eSFjS
Noj69tbkfOHke6/JOadqMTB8Wl2YZ3b5K3C9pq80R07RBfG3Eaz2d8xX3yGioyMSukAwTZcyePAA
JMcYzXgt5Z3Mf9vpyAMCDwzrg0dFS6T59bVSgVoz1buzxuvGoNfuDm/ug/z+W1vaNteDOQVlYf+A
0JvRqUa9xKrb96hrzizvP3BkmShjbvRnapCUiMdYGIjI0fRCRT/bo95blOrbLq5yLYiiirU9S9tS
4izT76sN6T5rEWBtapyVi/kOD0STZ9cKrEcrMvcqd078JYHFnmCih2i7cLkAqOuoPbxJ9Fkqg1vy
iN8OU6lwmv1h3yLFt003zE108dXiCct1oMMuwuDNMFFBeVYPFO487iaoeZUbtD+0rr3KeHSsBUvp
x6GfnR0vKHdtEwKw7+wEAfntsWL3t7GpiZLNMNCyoss2dYaKAd5fuAwDsOGKcn0VB0U27ELB6kCV
ybd4pNFcV21Dz3LwdTm1KMi+9uI/p3tzlNZJjbRjaKR3508a7HY9b5MseLSlHq1T3aYQjGZ65M0w
UkFd03qN+5Ji5ezZ1nzh9LYa+tlzTtvf4OXyTU6DxXX9ibBdYSCXuOpu7OpkYANyoijz1ZaauZ+1
+yaPn8Tlb/XbOcIVn7gwTBRQWu8+HDtSaZXwxuV/13konk8y9Q9vj3qG1NU71gq2fyT9zZVfMXl/
g/SBcoPjd55307MsukvFp5k5EfyWiHUS0nYhmE1E8EWUR0ToiyiWi/xJR3dmKGIYJOJ6G6ReXVQJ
wv+C46+hZHDzlLJAN422GtnZ349F7zwSqitKY7tGTGdJrcSeilgAeApAthOgOwApgIoBXALwlhOg
A4AyAu/wxUIZhvCfOQw+Tf63IAwCsP3C6zn5j3/nNyRwDAJd3burRdfQz9/okDvMFV0WzzYivNnc
bgAQisgFIBHAUwHAAc+TjswBc7eM1GIbxEX0xDHdUVDu7MXqCka2+fbpznnj9g4Q7Lx5/oUTlZnp
QkzXS8VrchRCHAbwO4BAkUS8BsAFAsRBCcVotBNDS6PVENJmIcogop6ioyKgLwzB+oq4si0Z46yl
oZKtf+sjlTm16M0ywcq3HWC3o0qwBujSr35ddJOKLWSYVwAQAWQBaAEgCMMbT1wshpgUQ2en
p6d4Og2EYN5wtr3LUML2sQxOPXuOuJJ8RnTKSseuY1tXygWHtDfsmx9nw6vU9630Nf7D72Dks2OE
+/XGk44tZZiSAA0KIIiFEFYBvAQwCkCKbaQCgFYDDPo6RYRgfGP76r5izoRDJcTZ8fnd/j17jTYS
qhcjJ5bJxkusnhhuyudBGpUH+HX/hi7gfAjCAiBJJeqYaAWAngGUArpf73A5grm9DZBjGCCEEvs4
pwMp9J+vsd7K0AoBzci4j0pIk57aG8c7l69yhjipViLF5LzGxVgs6NE32+vXhzsZDZ9Dp6Z+dctn
7C19s7usgLZxuBLBNPtd0AI8DmEJEuQAaA/jYD+NkGEbHp6vz8/v/verugE
AMhoaR53WhT53u6/sfXkslkwZ6tdzqjl9vjJg5/aE32WPpJW5dX85e4tPzqRCiOcAPKdrzgPQz5f
zMgzjnlW5p9x3UhHvQWGLOHmmLfxUscMexgm6vt1YiLsHtwvZ9QN9Zzj9AMNEKEt2Ha9X/wQPStI
pPivuTO6eOre4C4YKpUviodOeRe1GKpx+gGEijLyiUgx/41eP+qoXRj2ZuVs8yC1TYxcepydwl6N
m3p8vQ/F5/5pzPCVYvvXu0Kc79hc8c2eYCOOzNQc97quuhhTvwcx9S2ExAOCLtYcMj585X4kXf9z
p8fXdfQk0jI9BmyDP3pWnhYHtGgf1unoOnpKeHL5cZ3yvfYXFnWEijE9X53vcV10NSV+E2gjFlTH
noHP6geNny/Hi/J31ur6/bPf+5L2b+gAIfcGOxkmBTbvFZhmGMQFHii+gRUqCU3tO/hnHtkemFKU
Sk655bd4pTJy+1uXLpMLazu1XXWwYoB5SlEXj6hCKe3WNHVsPlwAAXrq6e0CuwTN3hjEBT3+/3bA
9Ma5+JekcVmid7u05dq7O13XKaIDURGff+GYuCnmEEqVsX3UI66jO/r02333XFoFJhcDizjAmwJW
Inq+oXwIwJceLfk5bVln3eVqlJiA5XjIEKIFQgTY7eIuSgbK6JnQz92dUX8b6PDv+gs0yDBOmCCF
QVlmDpDj3/6blLsS3tKLWE6VJsnuxVWTmgK6YRV1eL49e0Qm3DszEjdPXAriAmZP6wi4E2jUJz+h
Sm1UW9zDxwQ+Uzw7P3BkmTJm9vgDdnluIAg/8sb/dZJzCqVQ1c/dkhugqe2TnjAYuX/PH7NZolBD
jEE0BKW96IwMzTThgk4tkV4ewSPa4HrVFQwI1c2dxZ5gw5ZfdUpDSzqN1F7UGgPuGGmdfVD/+j+v
hugi1wqRLMw3bT5yrcGq7omsGiIBGCZKIK+aOmhDasj3BYZYJ4cxdvYahfCn6GzbLMEyYogTZKCH
8D365EYt3aqNSr+iagUU7jzs8QFyx5onhaJLsPqe7UsBaz5PfbXNqe3R0Z0y/Lduxf7JUytVSEcB
Mh/7AYZYJoc19vyp7ZkyAxJ1n7gwTpijirixm/rj1qEY4+2el4Z2JvRFns7gV1OaNElwKtxpLPRR
BH/GqhPMfOhXeYf0x1tC6QuqrXHnyd/EGFneGCVO2FEh+0I98s8Xw+L9uvQQJsVbE2SyaYCUFb3K
ye1pIGwDidVkmx8rFpxt4kS44mFgd3jKhecKYt/mIZt8WIHFnswzDhCmHiy/UeTwlUfJ+iYuxOs3
cF+44hnv/s6He16zP4p4+Pzw5AqDCwwvFFYrNvSrIM/cau8Abi/bgiO7vGiizDIs7w0Q4kllGO3P
3RtgBwOIimVa7JknI07lHJujFXXbq8+KBIagQEawWCvrC74q9Rfhg+X6n9pj62MLqAZtlGCYCuWd
wlmM7zmZBRZV/hMpVpsQB7bVJtr68p7+zrTg8kix6hM1CQV9QPVtunP0yLobNMgwTtazWVetp2zj
JsV1ZY0dhHSaculJLo/pcZUFV19049L2zoW2Hx7REbnHSzGkU/gXvI+xWoK+oDp7vTb7IxHQqWk
DJMYGRoZZ3BkmArhphraUnk2lwgWnL6Dg9AV8tf4QZqw8gEUPD9H0nVCP5F2ubO6eCGHHjAZY+Nc
hbvuFA1YLBX1BdW2eNtPmDw9ehu4tGwXsemyWYZgIZPuREqe2qd9uQ+6JUnyy6oCmPTsz1ePzuhL
3KpUQPjDMOGAqkoixUtAXVC9qrk0QdqYssDVcWdwZJgxxN6u01bEIpy/8PKiDswnFFa5s7mpXy7H
d3Ue6hjtWC6EmyDb3zhnaXDuB8m9X8OnsRJRCRHOIaDcR7SKigUSURkf+NvzaQPDMACcH+H
1xNYRkWrkkeEprmzu5aoF20CLUjCwWSyoUnnLFJwuQ4+/L0RxAGfT53XJ3Xq1TgnYtQDfZ+7vAFg
ghOgC4GIAuwBMBbBUCNERwFJ5n2GYenDLx+vqPO6ZSX3rdV1ycV61q2VdXyyRQoyVHJWYTpZ
WYPCry3CuvBq9XlgcsGsePKV1JdXHCfgbr/9KRNQIwBAAHwOAEKJSCFEMYAK3WQCu9nWQDBO
t/G10Z69f2yQ5DsO6NK3364yCatQzd1uYFJb2BavKFXL7Yef1i0Cw93ipYzsY99CXr+AsAEUAZhL
RJiKaQURJADKEEEflPscAZBi9mIgmE1EOEeUUFbmv7Mgw7Ut1JLBOAy1ds/V3Xp/6xNGFaa5sRSG6z
LhvFaRzq1zd0cM3eLY5G4MgiJzvSupCsfHx7wa/ryV7IB6APgQyFEbwDnoTPBCmnOGM7brrYAhh
iWwiRnZ4e/n6xDBMsPlV5uyTH23BjvzZOfdR516/sabzA6W2kqM1CGtdHIQR2yylqv7i7PzIahl/
```

pvPoiRagGb0H18teXO7Y7NE0OSvlBX8S9EEChEEIxDs6BJPbHiag5AMi/T/g2RIaJMlR27wZxNhS
eqTvL4p8uN3ZN9LZGqE0X4FNwujZAqj6eN+GMzWpxuEJ6UunKV5SMmZOHtMOSKUMDfj3AB3EXQhw
DUEBEilFwBICdAOYBuF1uux3AXJ9GyDBRwoXKGryxaA+6NKutepQQa8Vv+7TRqd10BZVbpyUans/
bmalNF+AToEJBIcVKQIVsalIvFndt7v9i1WWV1Y5tpSh2MPD1K+vPAL4golgAeQDugPSF8TUR3QX
gIIAbfLwGw5iWLQXF6NmqEYgI/V5egnMV1chuW+s9rLdvz31gELrqxL2hixS7FwzSAHuCzUqwC8l
ObLGQy2RikczGQ8UAJJPTG4v2OtoDURVpc0GxY7thQvDSIfu0MiKE2CzbzXsKIa4WQpwRQpwSQow
QQnQUQowUQtTtsMswUcqy3Scw4f1VmLkqHwBwrkKa4SnZF1umJKBpg3g0k23cDeNtuLh1iqGf+Wv
X93Rqu6V/W6/GpS9Dp8ziW6YkeHW+cKbGLrDjiFTGsElyHKoCENik/nvd78KEFggif9mbYSKUD3+
Vgo0+XZ2vaVe8Zb6aPAAAsGrqcDz3h67Y8twVLs91bZ9WGNAuzbH/xd398eDwDl6NS5mpF5VKdVM
VkX9sjPdumeFKpcr81LRBnCbNgr9Qe8oM7hg85xEWd4YJEesPSA+1Iy4y9kVX6qJaLYQ7BmW5DDB
S+nw1eaBjf+/xc3X2r4s1+08BAJ6S66Yq/uB1pTyIVKpqBPplSl+KHTOSfRL35XtOYH9RqVN7mWw
e65eV5jK9QyDgrJAME2Jc5RX3xZ88yQ9pZIvLqlBRXYMXftwBIDD26FBTeKYM6/OlL9kYq8WrHO8
/bzuKP32x0bGfP2285vjGg2cAAA+P7OjDSOuP+b6KGSbCsLtwSPdF3K/oZhg76BGKd87mgmJ8t/E
wVuVKM/lAlYMLBY9e0QkAMP7dlY626hq729KGRqiFHQAmvL9Ks6/EJHTKaIBgwuLOMCGivVxEo0f
LRoZui/E273OPxPnwWjXqZFdWE5llmjVyXhz+Xi5crc8BU1+2qLxjhBAO81uD+OAaSszz12KYCEM
pcG0XWnc5BV9cEP2VIkConipiTOQSWddTSH3SEWw4aOwMqPi2PzN3O37cKmVj8dcXrqewuDNMiFD
ytZRcqNLkbvEH/lq4U0eq2kyQ6lch1uC9tE6TZvN1xQfMWp2P95flAgAe/moTrvtwjWG/rs8uBCB
9cYcK8/y1GCbCUAT9lQW7wzYZl9pcFOyydIHEKFbglWulWIELla7F/bl5O/Dawj0Aas04rjh9vhL
WEIb3srcMw4SI/UW1tt2T5yr8cs7P7+pvWILPW9Q+2nEBzj8eTGIMvkzjY6X3V1ZVg0U7jmFg+8Z
o4CL6d+7mw05tf7q8PT5UFUrp82LgcsN7Aos7w4QBeo8LI7OBJ1zWsQku6+hbcq+B7RpjTd4p9Gm
TghqVzV2f0yaSMbK5J8rifsfM3wEAY7s3w4e3XOI4rv6i+8tXm5HVJAkH5GjizkH2hPGE8HwWZBi
Tkzl1vmH7tX1aAgit2+GorpIb5cZDxdhaWPsUYIYiHQqFZ5xdHhN0TyZKJkdXr1GEHZD+bh/WUd5
w5ePDvBmmT7C4M0wYcXN/KXd7KBfirs9u5dhemVubkTKY0ZWBRl88A3BeMN5x5KwmYvW1RXtcns/
I20lNq1TjzJ2BhMWdYcKIRDmyNJSzZPUioNot0Nt0BuHI2O7aAifPXNkVyQZRvWcvVDm2u9dhltp
cUIw/e5nLJ1CwzZ1hgoy6jJ6eDk2TMbhjE9wywLuMjv7A0+LbkUyMTfseB7ZrjEaJdafjVdeR1WM
hwpRRnZCWFIvnf9jplzH6Cos7wwSZI3WEuMdYLfjPXf2DOBpnokDbnZKguVrjUC8ov7Vkr2EfALB
YpCebge0bOx0L1WIrm2UYJsioE3DdoLJvhwtGtvV3JvYKwUgChz6uwFWAlqfRqoqAt9DlvB/epak
jdXOwYXFnmCCzUy4OAQAnVP7toRIBPUZmmQm9WoZgJKGnwkNxf/fG3gCkqljqrJC3DmiL1KTYgIz
NHSzuDBNkpny9xbFdolqwawWczOkAABlKSURBVCcnEgs1JnKKqZPr+tQ+NbnKzFkh29nVdVD/cHE
Lp36JLlIs926T4ssQfYLFnWFCyL1DasuuuaqFGmzM5BVTF69c1wP3Dm0HAGjsYnb9+bqDAICc/DO
OtlQ3C69qlORwoYDFnWGCyFXvrdTsN1SlgY03UXh/JGCzWvD46C7Y/vxoJxEeKVfH+nLdIQC1ycR
SEmOw+9g5R7/5D12GNU8MD9KI64fP3jJEZAWQA+CwEOJKIsoC8BWAxgA2ALhVCOHa94thooSFO45
pIj7vGZyFQi+KQzD+w2IhJMc5y2DjpDjN/r3/2QAAmDykHX7cctTR3q1FI8Pz/vyXwYbnDSb+mLn
/BcAu1f4rAN4SQnQAcAbAXX64hmk4ca4c7Z/8CRsPfFyXp6
9J84ZHq+pEdh5VFoQVxck13NR84ZonRb8qFQ1Pok7EbUCMB7ADHmfAAwHMEfuMgvA1b5cw2ys2X8
KNXaBmavyQz0UJsicLHXO/Ni8UTwAYNKlmUEeDWNEwwTJnj6ovXHyNauVcJOcImJ4F+PC5uGCr88
NbwN4DIDipd8YQLEQQllaLgRg6ENFRJMBTAaANm3a+DiMyOHVBVJ+CjMlYWK8Jz7G6lRQmQkdiht
ox4xkR5u6GlWzhvE4I0cYu3CwCRu8nrkT0ZUATgghNnjzeiHEdCFEThEthAiOz093dthRBxKAV4zJWF
iGLOgxJfZhcCVPZujQbwNVTW1Kn5N75a4Y1AWerZqhOsuCb8ANDW+zNwHAbiKiMYBiAfQEMA7AFK
IyCbP3lsBcM5qz4Cl3RwcKynH9sMlGCmnyXXFe7/s0+y/ecPFgRwW4yXqyNWUxBicK6/GvC21FZe
ICC1SEjDvwctCMbx64fXMXQjxhBCilRAiE8BEAL8IIW4GsAzA9XK32wHM9XmUUjqjOFE7uOnjV
M8+orA/65FHd/loPvN7mew5RWVOP1Rdq8JHXl12FCxxNjL8KkSzMxvkcL7DoqLao++s0WN68KTwL
h5/44gClElAvJBv9xAK4R8bC2B48NB09j7Du/4ZNVBwJ2jaW7T7g81v25hU5triIaw5Eb+7UO9RC
CRmpSLP5+VTfE2izYcDCyPdr88gkTQiwHsFzezgPX/nZRh/kCfXKn1p/i4s2nEcX97T32WiKG+
5prdzSDqgzSMTqWQ0jA/1EBgv4AjVEPHV7wUor3JdZZ3xH+r8LevzT+N0mX9i6tReFFaL8b/SM3O
1vu3X9JacxxJiIycalXiFKCKJnGdDE3K0pBxZTcIjWZQZKTxThlmr8/Hv37TmGH+5sL2ztHaRtKy
i2rCP/tH+gWEd0L1lI/wxzD0t1ESrCfHGfq0xe32BYz9csnZ6Cs/cGVMhhMCC7UdRYxd44MtNTsI
OQFMX0xfeXlIr7lUeLtYmxlpx12VZfjcLBZIo1XY8PLKTZj/Sookj5xNmEjIb14Yk7z9RGsKRmJN
5W47gvs834uOVeS5n02q/ZX+xZv8pp7bth0s0+x/dcolTMYdwp01aIv6YHT0Lqmr0aw2RFpvC4h5
kLKoPyN2f5YRwJObkSHE5AOBUaSX2ufjy9MfMvUY3U5+9XsoeWFltR8HpMgDAlf+nzQA5pnszn68
bbJZMGYpmjXhBNRJhcQ8iNXaB4rIqTdvQ15aFaDTm5IJcVEFJ7mTEFW+tqNMv3ROMZuoT3l+FTk/
/jMGvLsP9X2gDt/tnRdYjvUKkzVaZWljcg8hT321zqnx/8FRZiEZjTg7I9/O3fSfr7Pfwfzf7dJ3
Siiqnti0FxY7tn7Ydc2zHWAmz74msxTgF1vbIhcU9iHz1e4H7ToxPKBV1mgfAlFBjF7j9k/XInDo
fpRWeu7H+9thwjTkukoiWqkxmhMWdMQ0LdxzDp6vzAUhupu6oqK7Bh8v3Y8eREsd/+XSkK3ly8B7/
uLQIAvL1ESiegrsPpivQGcW77hBu9Woeu9ifjH1jcGdOgVMsx4s0bLsawztrsoyfOVuCVBbsx/t2
VEEKg89ML8OR325xeu2zPCeQXleFXNB0wUz6bLhBoAmNSTRbNMsWWQJ0dbP6xv35eeGbPJG5r3eVDUVz/jKqjxGR7JxUHlOOrL69mV1GRWGNt/SyWcHmMQwmbCrS1ZqPY4bEv3Nk+y9f1HTp93zpD6GV5nhYvWEUD44ujLXgF6zftmdQcGFd3ob8kHQqyFnF2ocSPELhUgSj6Xe0oKqCaqVhAcmKKzFiycyAcU2SHQDN1YGlmvnHB06qGPPD7TtNQg7SqDjfU4mCunTMh5a6EFeCsLsmkWm0jTZUc/5DzeNZnDf

```
5XMXZnM87TK6tqY5GZphgE9Hi/s6SffjXijw8NmdrqIdSL5qnmDvib3XuSdz68XqsyTuFL9cdQmW
1f3K5eIra0yM1KRbZ9cgJomRtBICzF5zTF8TaLGidVptCwGIh/PbYME2fK3u2wPJHL6/XdZnwZOa
kvgCAi5o3DPFI6k9Ei7vihVDqIodIuDK4o9Zro6wyssavZvvhEnywPNexf+Z8JW6asQ4rc2tt1Rs
PBbbogf7+bVYFE+lJjpNs5E+Pv8jwuPqZ6oetR5yO731prEP0W8p5Yqp17pNWCyGTs32agmFdmmL
F34bh63sjLwgtosX9xn5tAAA9WrpeKAsUB0+dx/A3lqPoXIXh8c/W5GsSRym5vx8aIXlPLHh4sOO
YXhwAwG4XYW9uKq+qwZX/txKvLtiDEjmtQv9/LnXq9+S3gTPNXPXeSnR9VlvpaHyP5i77b39+NPK
njcfdg9vho1suwT+v7YEOTaVK98lxNk12x8Iz2ujhUXKd1GWPXo6EGCt+/LNUR7NtWiJuHdAWyx+
93B9viQkz2jRORAM3XlHhSESLe6vUBNgs5LcUrvVh5qp85BWdxw9bnGd32wpL8OzcHZrEUUqiKcX
e3qVZ3Y95ry7cgy7PLAjrSj5KQA8Ax+zdyASTd/J8wMawtbD2CzQ5zoYHh3VAapL0j6h8kbpiTPd
muLFfG3x6R1+8fE13NG0Qh0OnawVdnyri6l6SySYtKRa7XhyDVDka1mIhvHh1d56tM2FFRIs7EaF
BvC0kZhmlwLXdoPLDH95b6dRWI/czCmipMhDEj37dDwCY8L7zub7dWIgT59xHYAYadTTnz9uPuXz
SaGcgeiVlVTh+1r/v4fU/XoxHR3fGI6M6Y/KQdnhwWAePXtcqNRE392+LvJPnsaWgGOvypGClWJU
nzMrHh2Fcj8jL6shELxEt7oC0wOXNgt0/f96F+VuPen1dAc9ygh+QZ612eYgWg1wd6gXhnUfOYtS
bvzr29bnH756Vgylfb8HIN37FD1uO4C9fbcLh4gv1Hb5fOKIK8T90ugxdnlmgOd4qVbJJKzP3ElV
GzItfWIT+/3A24dSXhio/80YJ0ow9NSkWT467CLEGniye8Pk6KX1vcVkl2qUnYdMzo9AqNZHzrDA
RhdfiTkStiWgZEe0koh1E9Be5PY2IFhPRPvl3qv+G64w34n6hsgb/+jUPD3y50evrKqagd5fu09T
S1PPL7hMA1DN35z5Ld59A5tT5+PPsTRj37m8u85ADwJJdxwEAZ8ur8efZmzB38xHc/4X378MXXvx
xp8tjL1/THYv+Whu+njl1Pi5+YREW7jim6afPi14fNh06g7PltU9t/srh8sOWI8g9cQ5bC0vQOjX
RYX5hmEjCl5l7NYBHhBBdAQwA8AARdQUwFcBSIURHAEvl/YBx8lwl9p5wHZlohHqhzNtFy6/k2op
ny6txw7/WoMYuUHimDJt0niGKAFbLXwY2F4WUARja7wHgmDxD3nPM+H0qqWZ3HT2L95fl1vllo8Z
uFzhb7py61lf6tEnBzf3bIjHWhoQYbZDPvf/ZgH2qSNL2T/7k9XWu0UWV+jMT5Mg3V6DkQpVmXYF
hIgmvc8sIIY4COCpvnyOiXQBaApgA4HK52ywAywE87tMo6+BCVQ22H67fouOot1Y4tmeuykeN3Y4
Hh3ueA0QIofFw+T3/TJ0ideDkeYeJwNN56ieTsrFk1wl8ue4QjpRcQOPkWIx+e4XL/k9+tw1fyua
E1xbuQdvGiVj+6OV1mhJmrMzDP37aLb2Hp0Y6zXzvmLkey/YUIefpkWiSXHtMCIEHv9xUx9j7Orb
7t0vD8j1agVTff285ojNFDe/SFElxvqVK2vzsKPR6YbFP52CYcMEvNnciygTQG8A6ABmy8APAMQA
Z/riGv3ht4W7N/isLduP1RXuxoh4ztKW7Trjto/ajHvb6cry2QLru3M21FYDUgqlm38tjMbxLBq7
sKbn0XfvBanR86mfH8fE9myN/2njkTxvvaFOEXeHgqTKNr7kR6gLP+3R5WYQQjiRb2S8twTLZdFR
SVoVVuacwf5vxekVqYgxSEmvNGE+P71rnGADgvBcL4ur3u+aJ4fjolkvqfQ496nEzTKTjs7gTUTK
A/wF4WAihmUILyT5gOFkloslElENEOUVFgX30ffSbLcicOh8j3/wV7y+TvFD0EWe3fbLeo3P96fM
NHtU+Hd2tmcYk8f1myeQyrHNTR9uSKdqUqlf3aoEhndIRIxvmE2OdZ6JdmzfEWzf0cuwf+Oc4l2N
48ced2FJQjAXbjzmZX+x2gbLKWpNUaUU15m4+jO2HS/C/DYVOM9gX50vmpTcX73HKt6JEaL44oRt
WTR2uOdahaTIGtKs7UvPrnPoVMfly3SHkyusSV3TNQPNGCV4vnurpp4sq3f3iGL+cl2GCjU/PsUQ
UA0nYvxBCfCs3Hyei5kKIo0TUHIDhNFcIMR3AdADIzs72elVtYt/WmLOhsM4+yvFc1ULlTw9dhlF
vrdC0fbYmH7cNzETm1PkAoJkZK/y8vXZB8KNbLkFljR0Pza41USx/9HKcOl+B1mmJWDV1OPq8qBX
JuwfXZglUzxQ/mZSN4V20DzlKdkE13z1wqUbI9GaXdU+OQF7Redz477XYe7wUE95f5Timmemv187
0J9eRCx0A8ookj5dZaw5q2hvG29A6LdHwXim8O7E3Hv/fVk263XE9mmFU1wz89b9b8PwPO3HHIM+
yJ544V67JV/PeTX08ep2npOgSfcXr1gwYJlLwxVuGAHwMYJcQ4k3VoXkAbpe3bcw1/vhuae0ohr
VduEykKnaoD3n6ZEgIiyZMlQjSs/O3YHP19aK13UfahfsCk5rIxb7ZqbiqotbOPZnTuqLzCZJuKS
tNPtL03lZxFjJEf6uRy/sgHGJM30WQgDY8PRIx3ZGw3gMbN/Y8Brqhdanv99u2EfP//50aZ3Hbx3
Y1u05mjaMx8w7+mHzs6MweUg75E8bjw9uvgQTLq7N4zLs9eUejWeGLvujv2bsCskq18rG7CXDRDC
+/GcMAnArgOFEtFn+GQdgGoBRRLRLPh72XEXPqBPPYHWHUYlMD5pJEc83AFIBj3DmkHQFp0fu+XfW69lz5
eWSvuMVb/+53/Qf6ynn7rJVj6yFC/n59hgoXX4i6EWCmEICFETyFEL/nnJyHESEBDgOFEhHU2f3kheXCVNPsLLTpMJPIdiXaaSetVyvuRvYxXjtMxm2Z7Kh3dhgBjSRDETPRrqYb
```

sxc3926KZHws2MAzDhDNEtEEIkW10zBQ2d4uF8MgVxmH0DMMw0YgpzDIMwzCMFhZ3hmEYE8LizjA
MY0JY3BmGYUwIizvDMIwJYXFnGIYxISzuDMMwJoTFnWEYxoSERYQqERUBOOjly5sAOOnH4ZgZvle
ewffJM/g+eUYg71NbIUS60YGwEHdfIKIcV+G3jBa+V57B98kz+D55RqjuE5tlGIZhTAiLO8MwjAk
xg7hPD/UAIgi+V57B98kz+D55RkjuU8Tb3BmGYRhnzDBzZxiGYXSwuDMMw5iQiBZ3IhpDRHuIKJe
IpoZ6PMGGiFoT0TIi2klEO4joL3J7GhEtJqJ98u9UuZ2I6F35fm0loj6qc90u999HRLeH6j0FEiK
yEtEmIvpR3s8ionXy/fgvEcXK7XHyfq58PFN1jifk9j1ENDo07yRwEFEKEc0hot1EtIuIBvLnyRk
i+qv8P7ediGYTUXzYfZ6EEBH5A8AKYD+AdgBiAWwB0DXU4wryPWgOoI+83QDAXgBdAbwKYKrcPhX
AK/L2OAA/AyAAAwCsk9vTAOTJv1Pl7dRQv78A3K8pAL4E8KO8/zWAifL2RwD+JG/fD+AjeXsigP/
K213lz1kcgCz582cN9fvy8z2aBeBueTsWQAp/npzuUUsABwAkqD5Hk8Lt8xTJM/d+AHKFEHlCiEo
AXwGYEOIxBRUhxFEhxEZ5+xyAXZA+eBMg/ZNC/n21vD0BwGdCYi2AFCJqDmA0gMVCiNNCiDMAFgM
YE8S3EnCIqBWA8QBmyPsEYDiAOXIX/X1S7t8cACPk/hMAfCWEqBBCHACQC+lzaAqIqBGAIQA+BgA
hRKUQohj8eTLCBiCBiGwAEgEcRZh9niJZ3FsCKFDtF8ptUYn8qNcbwDoAGUKIo/KhYwAy5G1X9yw
a7uXbAB4DYJf3GwMoFkJUy/vq9+y4H/LxErm/2e9TFoAiADNl89UMIkoCf540CCEOA3gdwCFIol4
CYAPC7PMUyeLOyBBRMoD/AXhYCHFWfUxIz39R7e9KRFcCOCGE2BDqsYQ5NgB9AHwohOgN4DwkM4w
D/jwB8prDBEhfhi0AJCEMn0wiWdwPA2it2m8lt0UVRBQDSdi/EEJ8Kzcflx+PIf8+kzclx+PIf8+Ibe7umdmv5e
DAFxFRPmQzHfDAbwDyYxgk/uo37PjfsjHGwE4BfPfp0IAhUKIdfL+HEhiz58nLSMBHBBCFAkhqgB
8C+kzFlafp0gW998BdJRXqGMhLVTMC/GYgopst/sYwC4hxJuqQ/MAKB4KtwOYq2q/TfZyGACgRH7
cXgjgCiJKlWclV8htpkAI8YQQopUQIhPS5+QXIcTNAJYBuF7upr9Pyv27Xu4v5PaJspeuLSAWPaJsvdDFoCOANY
H6W0EHCHEMQAFRNRZbhoBYCf486TnEIABRJQo/w8q9ym8Pk+hXnn2cdV6HCQPkf0Angr1eELw/gdBej3fCmCz/DMOkn1yKYB9AJYASJP7
E4BX/DMPRbNs6iEHnfEjxO6mIJ/dBMSdoFxr0UPKxJlTUfCjvNqeXsLMomkvzLJ6cgPFYj0rbAQAZKvH/ry
D9Ii8FcBm+WccJHveUgD7ACwBkCb3JwDvy/drG4Bs1bnuhLSgkwvgjlC/twDes8tR6y3TTv5nygX
wDYA4uT1e3s+Vj7dTvf4p+f7tATA21O8nAPenF4Ac+TP1PSRvF/48Od+n5wHsBrAdwH8gebyE1ee
J0w8wDMOYkEg2yzAMwzAuYHFnGIYxISzuDMMwJoTFnWEYxoSwuDMMw5gQFneGYRgTwuLOMAxjQv4
f7I+MqBDbyGQAAAAASUVORK5CYII=\n"

```
          },
          "metadata": {
            "needs_background": "light"
          }
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "len(test_data)"
      ],
      "metadata": {
        "id": "g_YckjXtLJNC",
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "outputId": "d81bb2c6-e49a-42c1-cdb3-b5145a73c130"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "2876"
            ]
          },
          "metadata": {},
          "execution_count": 33
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "x_input=test_data[2866:].reshape(1,-1)\n",
"x_input.shape"
      ],
      "metadata": {
        "id": "RgawVIleLW-v",
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "outputId": "84eceae5-f4e4-4376-fbb8-9714c26300c4"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(1, 10)"
            ]
          },
          "metadata": {},
          "execution_count": 34
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "temp_input=list(x_input) \n",
        "temp_input=temp_input[0].tolist()"
      ],
      "metadata": {
        "id": "1g1Iu241LcWh"
      },
      "execution_count": null,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "temp_input"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "529unKThLeVQ",
        "outputId": "135943b3-ea30-4d6c-fe1d-5beea39a94bc"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "[0.44172960165852215,\n",
              " 0.48111950244335855,\n",
              " 0.49726047682511476,\n",
              " 0.4679401747371539,\n",
              " 0.4729749740855915,\n",
              " 0.47119798608026064,\n",
              " 0.47341922108692425,\n",
" 0.4649785280616022,\n",
              " 0.4703835332444839,\n",
              " 0.47149415074781587]"
            ]
          },
          "metadata": {},
          "execution_count": 36
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "lst_output=[]\n",
        "n_steps=10\n",
        "i=0\n",
        "while(i<10):\n",
        "    if(len(temp_input)>10):\n",
        "#print(temp_input)\n",
        "       x_input=np.array(temp_input[1:]) \n",
        "       print(\"{} day input {}\".format(i,x_input))\n",
        "       x_input=x_input.reshape(1,-1)\n",
        "       x_input = x_input.reshape((1, n_steps, 1))
#print(x_input)\n",
        "       yhat = model.predict(x_input, verbose=0)\n",
        "       print(\"{} day output {}\".format(i,yhat))\n",
        "       temp_input.extend(yhat[0].tolist())\n",
        "       temp_input=temp_input[1:] #print(temp_input)\n",
        "       lst_output.extend(yhat.tolist())\n",
        "       i=i+1\n",
        "    else:\n",
        "       x_input = x_input.reshape((1, n_steps,1))\n",
        "       yhat = model.predict(x_input, verbose=0)\n",
        "       print(yhat[0])\n",
        "       temp_input.extend(yhat[0].tolist()) \n",
        "       print(len(temp_input))\n",
        "       lst_output.extend(yhat.tolist())\n",
        "       i=i+1"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "ULCbP4K1LlBQ",
        "outputId": "6947d757-5897-4056-d132-215b0b9e2843"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "[0.47442466]\n",
            "11\n",
            "1 day input [0.4811195  0.49726048 0.46794017 0.47297497
0.47119799 0.47341922\n",
            " 0.46497853 0.47038353 0.47149415 0.47442466]\n",
            "1 day output [[0.47781762]]\n",
            "2 day input [0.49726048 0.46794017 0.47297497 0.47119799
0.47341922 0.46497853\n",
" 0.47038353 0.47149415 0.47442466 0.47781762]\n",
            "2 day output [[0.47653615]]\n",
            "3 day input [0.46794017 0.47297497 0.47119799 0.47341922
0.46497853 0.47038353\n",
            " 0.47149415 0.47442466 0.47781762 0.47653615]\n",
            "3 day output [[0.47364426]]\n",
            "4 day input [0.47297497 0.47119799 0.47341922 0.46497853
0.47038353 0.47149415\n",
            " 0.47442466 0.47781762 0.47653615 0.47364426]\n",
            "4 day output [[0.47442248]]\n",
            "5 day input [0.47119799 0.47341922 0.46497853 0.47038353
0.47149415 0.47442466\n",
            " 0.47781762 0.47653615 0.47364426 0.47442248]\n",
            "5 day output [[0.47467044]]\n",
            "6 day input [0.47341922 0.46497853 0.47038353 0.47149415
0.47442466 0.47781762\n",
            " 0.47653615 0.47364426 0.47442248 0.47467044]\n",
            "6 day output [[0.47518066]]\n",
            "7 day input [0.46497853 0.47038353 0.47149415 0.47442466
0.47781762 0.47653615\n",
            " 0.47364426 0.47442248 0.47467044 0.47518066]\n",
            "7 day output [[0.47546706]]\n",
            "8 day input [0.47038353 0.47149415 0.47442466 0.47781762
0.47653615 0.47364426\n",
            " 0.47442248 0.47467044 0.47518066 0.47546706]\n",
            "8 day output [[0.4767432]]\n",
            "9 day input [0.47149415 0.47442466 0.47781762 0.47653615
0.47364426 0.47442248\n",
            " 0.47467044 0.47518066 0.47546706 0.47674319]\n",
            "9 day output [[0.47736228]]\n"
          ]
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "day_new=np.arange(1,11) \n",
        "day_pred=np.arange(11,21)\n",
        "len(data_oil)\n",
        "plt.plot(day_new, scaler.inverse_transform(data_oil[8206:])) \n",
        "plt.plot(day_pred, scaler.inverse_transform(lst_output))"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 282
        },
        "id": "0O0jGgIjL67v",
        "outputId": "c2e74268-f353-4dcb-b101-b9421d92ae59"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "[<matplotlib.lines.Line2D at 0x7f2e26c59e50>]"
            ]
          },
"metadata": {},
          "execution_count": 38
        },
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXAAAAD4CAYAAAD1jb0+AAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO3de3ycZZ338c8v52aSHpJMzw3tpLS
UUwuEgihgKUXKKlUUhPXAYbWuK+viruvKsg+PuoeXoKzi6sKiIrjLg4BScRHlpLYQ7QUikgPaek
hGdKk6SRtzofr+eOeaadp0mSaOc/3/XrNK8k99+T+T+dTr99pprroM55xARkcyTl+oCRETk+CjARUQ
ylAJcRCRDKcBFRDKUAlxEJEMVJPNiVVVVbu7cucm8pIhIxlu/fn2nLnU1dUl85I
iIhnPzHYOd1xdKCIiGUoBLiKSoRTgIiIZSgEuIpKhFOAiAihlKAS4ikqEU4CIiGUoBPga7Wzt5+s2
mVJchInIEBfgYfPOpzXz2f9bT2z+Y6lJERA5RgI9icNDx3NYW+gcdu1o7Ul2OiMghCvBRvNHYTmt3
HLwD1zQpwEUkfCvBRrN3cfOj77QpwEUkjCvBRrNvazGmzJuEvL6a++WCqyxEROUQBfgzt3X1s2BX
iggVVBKp8bFeAi0gaUYAfw/PbXhgW4NgEY9SFIiLpQwF+DGu3tFBWXMAZ1Op8fsIdfY
d+kBTRCTVRt3QwcwWAg9FHQoq6GQs65JXGvMEWcc6zb0sy751dSmJ9Hjb8MgPrmg1T
4KlJcnYjIGALcObcZWAJgZvlAA7DGOfftyDlmdgfQlqgU8iqgiU6G+uYOGUBd/tawGgIDfB8D25oOcPVc
BLiKpF+uWasuBeufcoe19zMyAq4CL4llYqb3b4g0fvOBEbxu62VNKKcrP01BCEUkbsfaBX
Y+UCTc27rcA8ws9VmdmdmdmdmdmIx4Pcxp6IUgPw8Y25VqYYSikjaGHOAm1kRcDnwyJC
7ruHoUD/EOXePc67Wcr9x+1qXJa6u4b4KUd+w61viMCVWVqgYtI2oilBb4S2OCcO7Qsn5kVA
w5IecGe+Vt1vp7hvkwgVDAtzvY1drJ30DWtRKRFIvlgAfrqV9MfCWc25P/EpkwJE
fVtb4y8KLWnWmqDIRkcPGFOBm5gNWAI8OuNWu4PvGMt25rM0vnVlBadORnvJGRKPV71Q8uIqk3plE
ozrkOoHKY49fFoyhQG9KMCjvLa
njfbu/mG7TyICVWXs16JWIpIGFOBR1m1pJs/gPfOrRjynZurhNVFERFJJAR5l7ZZmFs+ZzOhH
PCVSFR6KoH1xEUkwBHhbq7OX1PaGjps8PNXvKBArzjfoWtcBFJLUU4GG/39bCo+GC0gvw85lb
61AIXkZRTgIet3dzMvZMpmFLJ49adRzA36fViUkZRTgBPefWdrM/uiUkZRTgBPefWdrM++ZX0VB/uhPScBfxq59WtRKRFJ
LAQ5sbjpAU3vPUasPjiRQ5aN/0LFbi1qJSAopwDk8ff78BSMPH4xWMzyUkdRTgeNPnF0w
rY8akCWM6v+bQUEL1g4tI6uR8gHf29vW7vpYJEoPIpSOR/gL21vpXdgcNThg0P
V+MvYrrHgIpJCOR/ga7c0U1KYx9lzKYx9lzK0Y/OYo3lFAtcBFJnZwP8HVbmlk8aw6a
q1KJWIpIaowa4mS00s41Rt3Yzuyl8341mtnnz11+b2Vtm9pYzuz3x5m3ctd0jDp9fjeMs
XXHDw+Mbm7zG2Cq9e1pxV6ILEpvg0M8m+F1TMfJwVZNgAUqRyZYChcjhmmzJhgoTAM5rU5wsy1Jp
ufdrPPeCUp8MtzibKjNXn3EGVMVYP3aSvBG+6uWRUSBO5XZhRc4B2Q
CmM7dh53TBdDoCqUNgL1GdJxIp+Jgtb3FmCuhhTEPK2gHaHs8ezpmZNUKrz3dR7lGo5Pv0yvjZOj2
ZAqAIaIlfaYm1bksLZ1ZPZmJJ4XH/joDfR9+AY/f+rjhWJiIyNmMKcDPzASuAR6MO3wsEzOwKENa
8S46/lYA+bGtrG1X0CXhcKaE0UEUmNUYcrAjjnOoDKAcd6gY/HvigR2ZGzuHbdgmHZeSGdHWjL+
Atc45F/8S46/lYA+bGtrG1X0CXhcKaE0UEUmNUYcRAjjnOoDKIcd6gY8noqhE+/1W741CrNPnh6r
xe6NX6psPsnzRtHHXJSISi5ycibl2SzMVviJNn67jvHMrm0iAotaiUiKZJzAT446HhuazPnn1h
FXl7swweHClRpf0wRSY2cC/A3g+20HHOXnmIGvJFw9K5urJCSajW5AHGQRTFmtanTfNoQdbjl9Pj
3MmFsHi4Ksk+n6tHn9a0jSLrQW13+V2w1EJPXZ7x7aN3KM5dygDy1qK66MmF1eX9fDoBJdN
IjFXOBfi6Lc2cPGMiL4AuGRKPVqhYtIkuVUgB/s6WWyeQrd7jHn0GvTITh/dqTLEMFKMRk0Z29
roX/QjXv8d7Q5FaUU5Bn1GgsuIkmWUwG+bmszvrZuFSZvoPSifbmszvqL8uC7/WpifR3W1fSrUSkeTTLmQB3zrFrl
kqCC+f+xAVZm6UEQk6XImwN/e18nZ1Ez1sXSNPPfpa1EpGkypkAXxcePhjDDBgO7
YZI8WtRKRJMqZAF+7pZm5laWcDNOtvQqXmxQpS45mdQLZsCgVYXQ1G7ciL0R8B6Ws5hXZbXHhTfhBZp
CfP3b++nqG4jb9Pmhp viKmFJaqKGEIpJUORHgv9/WQkGe9oEM6a4999ZzQBfxn1aoGLSBLlRIDv3Nf
JnIpSfMVjWv78uNT4tSqhiCRXTgR4Y1sxMyfHZ+2TQc7KGtS4taiUhy5ESAB0AB0PdzJg0IaH
XCFRF1kRRP7iIJMeoAW5mC81sY9St2cxuMrOKtBryOwxPHLKLFwrPoGBtl7oJuZkxLfAgeNRBGR5Bm
1U9g5txlYAmBm+UADsAa4HviWc+bnJrauxl0MGNyYlvgJ1R6i1ppLLiIJEusXrSjLgXrn3
EFJMIwbZuAGYmOMAL8/Ooriilfq9a4CKSHLEG+NXADnid3ZiL
NsURa4InuQgFvRubbLZ0MDLqEX0tEJJYWB5YmosDxagx1UV5SQFk
Cx4BHBPy+8KJWnQm/lohILAF+DVHdJ+rBAwDAfcC
jUYdvN7NNZvY6sAz4QgLD7dgWxcztktD/DYfHgmtNFDlAw
SUZwF27pZPGdyUq5Atxw+CSUQfA3yi2Huu2H+Cph/MVSdCGaprlAyWOI7hlO
oq3eA1o7epIxAATAzAlU+6veqBS7Am4/BYzd6IX3ZN6F1B2x7Gp682btNrvbC/MQVMO8CKIr/WvW
```

```
SAgP90L7H+8+7dQfs3+F9XXYLTDs5rpfK6gAPtoWHECapDxy8fvC1W5I/XFLSSH8vPP1/4KW7YdZ
Z8JEfwZQTwnf+G+zfCdue8W6v/QTqfgj5RXDCeYdb5/6Fap2ns74u2P+2F8yt2w+H9P4dENoFg/2
Hz80vhilzoas17mVkeYCHhxAmeB2UaDX+Mn66fg/t3X1MLClM2nUlTezf6XWZNG6Acz4LK74GBUV
HnjPlBDj7L7xbfw/segG2Pu0F+lO3eLdJ1TB/ebh1fiEUl6Xkj5PzBgeg8VXYsQ721YeDejscCB5
5XvEkqJgLMxbDyR+EinlQEYAp86B8BuQlZtmprA7wQ5N4ErwSYbRD26s1d7AkSX3vkibe+iX8/LP
ggI/+Dyz6wOiPKSiGwHu92/v+1Wu9bXsGtj4Dmx6B9T+CvEKvi+Xir8CM0xP4BxAADu6F+t94/6n
W/+Zwy7lsmhfKgWVeQE8Jh3TFPJgwJSXvmLI8wL0W+PQk9YGDty44eKsSKsBzxEAfPPMVeOG7MGM
JXHmf94/6eEyuhtobvFt/L+x+0QuS1x6Ee94L7/48XPgPUJi8d5VZb6AfGuoOvwsKbvSO+/yw4H1
el1bNRVBakdo6h5HVAR5s66KqrJjigvykXbO6wkd+nmkseK4I7YafXg97XoGlq+GSf/Fa1fFQUOS
1vOddAO/5gtev/vtvwZu/gMu/A3PfE5/r5KL2INQ/64X29t9CdxtYPsxZChf9k/dZxPTTE9b1ES9
ZHeCNbd1J7T4BKCrwFrXSmig5YMuTsOYzXgvuyvvglA8l7lqlFbDqe3DalfC/fwP3/RmcdR1c/FW
YoHd6oxrog90vhVvZz0LTJu94+Qyvq2v+xV431oRhl3RKW1kd4MFQ16E+6WTyhhKqBZ61BvrgN/8
Mf7gTpp8GV94PlTXJuXbgvfDZF+B3/wYvfA82/xr+7A5Y9P7kXD+VnIO+Tq+13N0GXaHD33e3QXf
o8Ncj7gvBwWbo74K8Aqh+l/d5wvwVMO2UjB7tk7UB7pyjMdTFu+dXJf3aAb+P57a1MDDoyM/L3Be
HDKOtAX56g9c3fdb1cOnXoTC57/IoKvW6ak65An7xeXjoY7Docm+sefm05NYSL/29cKDRe37bG6B
tT/hrgzem+sA7XigPjrJlYaEPSiZ5twmTYeJMmLoISiu9YZrzLoSSicn5MyVB1gZ4e3c/Hb0DSVs
HJVqNv4ze/kEaQ13MqShN+vUlQbY+A2tWe0P/PvxDOO0jqa1n1pmw+rfw/Hfgd7fBjrVesJ/xifR
pVUZazV0haG/0wni4kD7YhDd8J0rJJJg4GybNgplnet1IJZOgZPKRIR39c35uDd3N2gCPTOKZkeQ
+cDi8qNW25oMK8Gww0O91WTx3B0w9Ba6635sGnw7yC+H8v4NFq+B/Pw+/+Gt4/WH4wJ3x69bp2g/
7tntdET0Hhtzah3wd5j43ePTvLPR5wTxxFpy46HBQT5wFk2Z7XzX2fVTZG+Ch5E/iiYgeC75sYdI
vL/H24n964X3GJ2Dl7V4XRrqpmg/XPg4b7oenb4W7zoNl/wjnfg7yx/jPvLcTWjbD3j/B3jeh6U3
v+wONIz+m0AfF5UfefH4onnjksZKJUD7TC+dJs7xWc7q8S8hgWRvgDSmYxBNR6Sti0oRC7VCfLZZ
+2psKffLlqa7k2PLyoPZ6b+zyL7/oBfmmn8Kq73ozBCMG+qG1Piqkw0Hdup1D3Rj5xd50/nkXeH3
IVQu8LozoUC4qH/t/DpIQWfvsB9u6yM8zppYnP8DNzNteTWPBs0PhhPQP72gTZ8LVD3iLaT3x93D
PMjjjY9DX7YV1yxYY6PXOtTyoqPFGY5x2pbfY0tSTvVmGCue0l7V/Q8FQN9MnlqRsFEigqozntmb
+olYDg45fbgqyvfkgn7/oRPI0qiYzmMEpH4TAhfDUP8GG//b6lacu8mYVTjvlcMtaszozVtYGeGM
SN3IYTsDv42cb9nCgu4/yOCxqlewhif0Dgzz+epD/+M1W6sPvJGr8ZXxg8cyk1SBxMGGKNwHo/Xe
qRZ2F0nue6DgE27qTspHxSGrCI1F2jHNzB+cctz72Bqf+3yf54iOvsX7nfpxL3KbJ/QOD/HT9HlZ
8ax03PbSRgrw8vvvnZ3DS9HLueGozfQPDjCiQ9KfwzkqjBriZLTSzjVG3djO7Ker+vzMzZ2bJnzE
zgsFBRzDUnbSNHIYTWdRqvNurffc32/jxCzs5bfYkfrUpyIfvep73fXsd9/5+B6HO3niUCkDfwCA
Pv7Kbi+5YyxcfeY2Swnzu/viZ/Opvzuf9p8/kS5cu5O19nTz0yu64XVNExmfU/5adc5uBJQBmlg8
0AGvCP88BLgF2JbDGmO3r6KV3YDCpGzkMVV1ZSp6Nb4Pjh1/ZzR1Pb+GKM2Zxx1WL6ewd4H9fa+T
BV3bztcff5Ou/fouVp07n6rOrOTdQgR3HsKze/kF+tmEP3/vtNvbs7+LUWRO55xNnseLkaUf8vmU
Lp3L23Cnc+exWPnzmbCYUJW+BMBEZXqzvq5YD9c65neGfvwV8CXgsrlWN06FJPClsgRcX5HuLWh1
ngP/mrSZuXrOJ80+s4raPnI6Z4Ssu4Oql1Vy9tJo/Bdv5ycu7WPNqA49tbGRelY+Pnj2HD585G3/
56Kvh9fQP8EjdHu76XT0NoS4Wz57E11adwrKFU4f9j8DM+IdLT+Ijd7/AvX/YweeWzT+uP5eIxE+
sAX418CCAma0CGpxzrx2r5Wdmq4HVANXV1cdZZmwi64CnsgUO3ozM4+lC2bg7xOceeJWTZ0zkro+
fRWH+0T1di2ZM5KurTuXmyxbxxKYgP3l5N1//1Vt888nNrDh5Glcvreb8+VVHjRrp7hvg4brd3PW
7eoJt3ZxRPZl//dCpXLjAP2oLvnZuBRcvmsrda+v52DnVTC4tOub5IpJYYw5wMysCLgduNrNS4B/
xuk+OyTl3D3APQG1tbeI+fYsS2YknlS1w8FYl/MO2FgYH3ZiH321vPsgN972Cv7yYe687m7LiY/8
VlRTmc8WZs7nizNls23uQh17Zxc82NPCrN95uhJfPTsOVxVO4fJpYU8+PIu7l5bT1N7D7UnTOH
2j5zOe+ZXxdT18sX3LWTlnc9x1+/qufmyRWN+XCo0hrp4acc+Xtreyqu7Qjx247spKVTj2SPWFr
gK4ENzrkmMzsNmAdEWt+zgQ1mttQ5904C6oxJsK2L4oI8KnypbSHWTC2jp3+QhjEuatV8oIdrf/Q
yAPffsHRMXSHR5k8t45Y/O5kvvm8hT7/ZxIMv7+Lfn97Ct5/ZwsQJQhYQ6+zhnXgXfumoJ76qpPK4
+85OmT+RDZ8zivuff5rp3z03JUgUj2d3ayUs7Wnlx+z5e2rGP3a+Gs3a3ef+QTSwpOq+CUGcf0ycpwCV
7xBLg1xDuPnHObQKmRu4ws7eBWudcS1yrO07eRg4Tjiug4ilQFV4TpaVj1AA/2NPP9fe9TMuBXh5
cfS7zqo5/HfPignzef/pM3n/6THbu6+ChV3azs7WTT5x7AucGKo/79WIFP5akDuf2crXP5y
aPRqdc+xq7fTCensrL+1oPbR8wuTSQpbOreC68gOeVE7JSuNKcDNzAesAD6T2HLiIxh
K7SSeiMiqhPV7D3LhAv+I5/X2D/LZ/1nPn4IH+MEna+O6l+YJlT6+dOlJcft9AHMqSvnYudXc//z
bfOr8APOnJn7VOOcc21s6wmHthfY77d5nHZW+Is4JVLD6ggDnBCpYMLVcM0YlJ4wpwJ1zHcCITTf
n3Nx4FRQPjaHulGzkMFRVWRHlJQXH3F7NOceXf/Y6z21t4faPnM6yk6aOeG46+dyy+d4wx6c2c9f
Hz0rotZ79UxM3P7qJvQd6APCXF3POvArOCVRy7rwK5k8tS/m7LZFUyLrpWf0Dg+w90M2sFKxCOJS
ZUeMvO+ZQwtt+vZlHX23g71Ys4KraOUmsbnyqyor59AUBvv3MVl7bHWJxHN81RHujoY0b/9+rzK3
ycdPFCzg3UMG8Kp8CW4QsnErfdKCHQUdKp9FHC/h9Iw4lvO8POw4NybvxoswbV/2p8wNU+oq47dd
vJWR6/94D3Xz6x3VMKS3k/hvO5s/PqSbgV2tbJCLrAjyYJkMII2r8ZTS193Cwp/+I409sCvLVx9/
kkpOn8bVVp2ZkKJUVF3DjRfN5vn4fv98W38+vu/sGWP3j9YQ6+/j+tbUpWRZYJN1lXYAf3sghPVr
gkTVRdkR1o7y0fR83PbSRM6un8J1rzsjoERJ/fk41s6dM4LZfv8XgYHxa4ZHPBTbuDvGtjy7mlJm
T4vJ7RbJN1gV4sC2ylVp6tNgiI1EiH2RufucAn/pxHXOmTOCH19Zm/MSS4oJ8/nbFAt5oaOeJN4J
x+Z3/+bt6fr6xkS9esoBLT50Rl98pko2yL8BDXZSXFMRlDe54OCG8qFX93oM0hrq49t6XKS3K5/4
blmbNVPRVS2axcFo5dzy1ZdzLzf76jXf4xpObWbVkptZbERlF1gV4Y1s3M9NodmBxQT5zKkp5dXe
```

```
I6370Mh09/dx3/VJmT0nDjXGPU36e8aVLF7KjpYOH645/udk/NrbxhYc2snjOZG778OkZ+bmASDJ
lX4CHupiRBkMIowWqfDy3tYW3Wzr5r0+exaIZE1NdUtxddNJUak+Ywp3PbKWrdyDmx+890M2n769
jcmkh3//EWRnftSSSDFkX4MHwNPp0smBaOQB3XLWY82pSP8EoEcyMf1h5EnsP9PCj53fE9NjuvgE
+89/r2d/Zx/c/WcvUien1H7BIusqqAO/uG6C1ozelO/EM57PvreHRvzov6/eTPHtuBctPmsrdv6u
nrbNvTI9xznHzo5t4dZc34uTUWRpxIjJWWRXgh0egpFcLfHJpEWdWT0l1GUnx95cu5EBPP/+5dtu
Yzr9rbT1rwjNRNeJEJDZZFeCH1gFPsz7wXHLS9Il8aMks7vvD27wT/g91JE/+0RtxcvnimRk5E1U
k1bIywGelWR94rvnCigUMOsedz24Z8Zw3G9v5wkMbOX3WJG7/iEaciByPrArwSBfK9DTrA881cyp
K+dg5J/Bw3Z5h14FpPtDDp+5/hYklhXz/k5k/mUkkVbIswLuoKiuiuECBkGo3XjSfkoI87nhq8xH
HvREndbR29vKDazXiRGQ8sirAG0LdafcBZq6qKivmU+cHeGLTO7y2OwR4I07+8dFNbNgV4t+vWqI
RJyLjlFUBHgx1MVMfYKaNT50/jwpfEbc/+RYAd6/dzqOvNvC3KxZw2WkacSIyXqMGuJktNLONUbd
2M7vJzP7ZzF4PH3vKzFI+yDnYphZ4OikvKeTGZfP5w7Z9/Mvjb3L7k2/xgcUz+WuNOBGJi1ED3Dm
32Tm3xDm3BDgL6ATWAN9wzp0ePv44cGtiSz229u4+Dvb0qwWeZj52bjWzJk/gB7/fwemzJvENjTg
RiZtYu1CWA/XOuZ3Oufao4z4g/luyxCAYSs9JPLmuuCCfr606haVzK7hHI05E4irWPTGvBh6M/GB
m/wp8EmgDlg33ADNbDawGqK6uPr4qx6AxzTZykMOWL5rG8kXTUl2GSNYZcwvczIqAy4FHIsecc7c
45+YADwA3Dvc459w9zrla51yt3+8fb70jamyLBLi6UEQkN8TShbIS2OCcaxrmvgeADyWkKhHEekLCPM
Tn2faO1FEckYsAX4NR3afnBh13yrgrXgVdTwa27qYVl6c0ftLiojEYkx94GbmA1YAn4k6/HUzWwg
MAjuBv4x/eWPXGOpS/7eI5JQxBbhzrgOoHHIspV0mQwXbujl99uRUlyEikjRZMRPTOeftxKNFrEQ
kh2RFgO/r6KW3f5AZCnARySFZEeAaAy4iuShLAtybhakAF5FckhUBHgxP4lEXiojkkiwJ8G6KC/K
o8BWluhQRkaTJigBvCI8B1yp3IpJLsiLAg6EudZ+ISM7JjgDXRg4ikoMyPsD7BwZpau/WKoQiknM
yPsCbDvQw6DSEUERyT8YHeDCkIYQikpsyPsAb2zSJR0RyU8YHuFrgIpKrMj7AG0NdlJcUMnHCmFdGF
SRESSKvMDvK2bmRpCKCI5KOMDPNjWxQwNIRSRHJT5AR7SJB4RyU2jbqkW3vfyoahDErBBbwAaA
XqAeud86FElHkSLr7BtjX0csstcBFJAeN2gJ3zzi1zBzCpzgTgS3CI6gTXA08CpzrkGz
B8BBCtcBFJBfF2oWyHKh3zu10zj3lnOsPH38RmB3f0kZ3aAihWuAikoNiDfCrgQeHOX4d8Cyhz
mq82szszqmpubY63vmA5N4lELXERy0JgD3MyKgMuBR4YcvwXoBx4Y7nHOuXucc7XOuVq/3z+eWo8
S2QtzuibxiEgOGvVDzCgrgQ3OuabIATO7Dng/sNw55+Jc26iCbV1UlRVRUpif7EuLiKRcLAF+DVH
dJ2Z2KfAl4ELnXGe8CxuLRg0hFJEcNqYuFDPzASuAR6MOfxcoB542s41mdncC6jumYJt24hGR3DW
mFrhzrgOoHHJsfkIqikEw1M15NVWpLkNEJCUydiZme3cfB3r6tROPiekEQkt2VsgDe
2eUMI1QIXkVyVsQEeaYFrJx4RyVUZG+CNoS7yp5WqBi0huytwAb+tiWnkx+XmW6lJERFIiYwM
8GOpmhrpPRCSHZW6At3Wp/1tEclpGBrhzLrwXpvq/RSR3ZWSA7+vopbd/UNPoRSSnZWSAH5rEoy4
UEclhGRngkUk8sxTgIpLDMjPAI1upGAG0PeOuBmmsQ
jIrkrIwM82NatMeAikvMyMsC9FrgCXERy26gBbmYLw1umRW7tnZn80s0Ezq01GsQD9A4M
0tXdrGVkRyXmjbqnmnNsMLAEws2ygEfgxuWsVFXCeOzeKARTBJ0LAEBMGeD5zAHBMQpdnIpZMoBhUAECxXxumLTUmlEJQKFEqfjEFWd5IIJxBUhxGBEGAAAgWWlEEYOCswsvZgQJOrAOZN
sDxKD2shBRASIvQ/8auDBWB5gZqvNrM7M6pqbm2O83NEatJGDiAgQQ4CbWRFwOfBILBdwzt3jnKt
1ztX6/f5Y6ztKUJN4RESA2FrgK4ENzrmmRBUzFsG2bsqLCygvKUxlGSIiKRdLgF9DjN0nidAY6mK
G+r9FRMYW4GbmA1YAj0Yd+5CZ7QHeBfzSzJ5MTIlHatRGDiIiwBhHoTjnOoDKIcfW4A0nTKpgqJv
TZk1O9mVFRNJORs3E7O4bYF9Hr3biERehwwL8nTZt5CAiEpFRAR5ZB1yTeEREMi3Awy3wL5u4N1u
SWQEemcQzXX3gIiKZFeCNbUKlIg8AAAbhSURBVN1U+oooKcxPdSkiImXUQEe1BhwEZFDMirAI1u
piYhIhgV4MKSt1EREIjImwA9093Ggp18tcBGRsIwJcK0DLiJyuIwJ8Mg64GqBi4h4MibANYlHRORwGRTg3RTkGVVlRakuRUQkLWRMgDe2dTFtUjF5eZbqUkRE0kLmBLgm8YiIHCZzAlxbqYmIHCYjAnxg0LHngDZyEBGJlhEB3nKwh/5Bp0k8IiJRMiLANYlHRORIGRbg6gMXEYnIrADXKBQRkUMyI8BD3ZQVF1BWXJDqUkRE0sZYNjWuAP4VOAVwwEfxdqv/NjAfuNQ5d9xt0FHTZ8G8dNHrs5cIl9hgR/hlF/qO/E5iIiQ9rwP0KW7wR6L/LY5xLNoxQfAf2d4yJFz+Vu+ZmA7ydsGvUEbPlmkFuKfw2Q+fSqvIinyBQ6i+Z1XQ+TtYTR40TzC5wQXcKyObnrZNKLDnGrChJgi5a8lIiD4ZbEURq3JVsmkPh4+2p8OQ8+KjuenPXZxQFTC9fIHoU4MPLwL0YzahCXpuIO+YSQ==
FupXCX/cOc05Kn0czuw54P/Cx8D/wo4zhtZAQzrkm59yAc24Q+P4I103181cAXAE8NNI5yXr+Rsi
UpLwGcyrAw31mPwT+5Jz79xHOmR4+DzNbivcc7UtSfT4zK498j/dh1xtDTvsF8MnwaJRzgbaot2r
JMmLLJ5XPX5RfAJFP9K8FHhvmnCeBS8xsSriL4JLwsYQzs0uBLwGXO+c6RzhnLK+FRNUX/ZnKh0a
47ivAiWY2L/yO7Gq85z1ZLgbecs7tGe7OZD1/x8iU5LwGE/kJbbrdgPfgvZV5HdgYvl0G/CXwl+F
zbgT+iPep+ovAeUmsLxC+7mvhGm4JH4+uz4Dv4Y0A2ATUJvk59OEF8qSoYyl7/vD+IwkCfXh9iH8
BVALPAluBZ4CK8Lm1wA+iHnsDsC18uz6J9W3D6/uMvAbvDp87E3jiWK+FJNX33+HX1ut4QTRjaH3
hny/DG3VRn8z6wsfvi7zmos5NxfM3UqYk5TWoqfQiIhkqp7pQRESyiQJcRCRDKcBFRDKUAlxEJEM
pwEVEMpQCXEQkQynARUQy1P8HxES/r4Rb57EAAAAASUVORK5CYII=\n"
          },
          "metadata": {
            "needs_background": "light"
          }
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "df3=data_oil.tolist() \n",
        "df3.extend(lst_output) \n",
        "plt.plot(df3[8100:])"
      ],
      "metadata": {
        "id": "SR0uR3ytMD7f",
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 282
        },
        "outputId": "3f10071c-24cf-4b39-9fc4-0a60aefbb4bc"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "[<matplotlib.lines.Line2D at 0x7f2e26bec890>]"
            ]
          },
          "metadata": {},
          "execution_count": 39
        },
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXoAAAD4CAYAAADiry33AAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO3dd3yc1ZX4/8/RqHerWdWWLfcKRoB
pCT0QwCRLCqmkksZ3+WVJNrDJkl2S7G4a28LuhirQQUgkhDiUEGoI3QL3Ltuyrd67Rhr No8f84y
sMpLGqqPReb9eflnzzPPM3NFIZ67OvfdcUVWMMcZErqjZboAxxxxpjpZYHeGGMinAV6Y4yJcBbojTE
mwlmgN8aYCBc92w0YLisrS4uLi2e7GcYYM6e8+eabzaqaPdJ1YW0rdXXzYzjDFmThGR46P
dZ6kbY4yJcCEFehG5SkiQNwe5P6PiUiDiOxw/n599q5j/3A0tmW
e4AqgEtgmIltVdd+wU59Q1evNr1FV7QMeBK4P8fH
fATytqs1OcH8auGpiTTXGGDMRoQT6G6GG2XF6CvJSJSFlDQ0OITTf
GGBOKqRqM/SNQrKob8PfaHzidi1ygi67BvgooGnS70Dk2QFWbVLXUXufl
j4KxQrzXGGDO9Qgn0PKnDlIrJERGKBG4Gtgx8KuUt3Tn9T
```

z7QT9Xt9sN2XWjDvrRlX7ReQW/AHaBdynqntF5C6gTFW3An8rIluAfqAZ+JhzbbOIfAP/hwXAXar
aPA2vwxhjgnrtaBNf+d1u8tMTuGj5/EwNh7QyVlWfAJ4YduzOQV/fAdwxyrX3AfdNoo3GGDNh1a0
9ALT1eGa5JbPHVsYaYyJafYd/+LDT3T/LLZk9FuiNMRGtts0NQIcFemOMiUx17U6g77VAb4wxEan
OSd10uC1Hb4wxEanOSd1Yjt4YYyKQ16c0dAZ69BbojTEm4jR19eL1KQCdlqM3xpjIU9fm782LWI7
eGGMiUmDGTUF6gs26McaYSFTX4Q/0y3KSLUdvjDGRqK7NjQgsyUqyWTfGGBOJ6tp7yUqOY0FiLD0
eL555WsHSAr0xJmLVdbhZmBpHcpy/fmPXPM3TW6A3xkSs2jY3uanxpMT7A/18zdNboDfGRKz6jl5
yLNBboDfGRKbefi/NXX0sTIknJT4GmL9z6S3QG2MiUn27f7FUbtqpHP18XR1rgd4YE5HqnTn0lrq
xQG+MiVC1TvmDhSnxJAcC/Tzt0Ye0Z6wxxsw1gfIHuWnxJMa6AMvRj0lErhKRgyJSLiK3j3HeDSK
iIlLq3I4RkQdEZLeI7BeRoBuIG2PMVKvrcBPrimJBYgxx0VFER8m8XR07bqAXERdwD3A1sAb4gIi
sCXJeCnAr8Pqgw+8F4lR1PXAW8BkRKZ58s40xZmz17b1kp8QhIogIKfHRlqMfwzlAuaoeVdU+4EH
g+iDnfQP4NuAedEyBJBGJBhKAPqB9ck02xpjxtfV4SE+MGbidHB9ts27GUACcHHS70jk2QEQ2AUW
q+viwax8GuoAa4ATwPVVtnnhzjTEmNN19/QO5eYCUuBjL0U+UiEQBdwO3Bbn7HMAL5ANLgNtEZGm
Qx7hZRMpEpKyhoWGyTTLGGHr6vCTEnppvkmypmzFVAUWDbhc6xwJSgHXACyJSAWwGtjoDsh8E/qS
qHlWtB14GSoc/gareq6qlqlqanZ09sVdijDGD9Hi8JMScCnGpFujHtA1YLiJLRCQWuBHYGrhTVdt
UNUtVi1W1GHgN2KKqZfjTNZcCiEgS/g+BA1P8GowxZoTuPi+Jg3v0cVObo3+5vJHSbz7Nb7admPR
j+XzKq0ea2FYxPZntcQO9qvYDtwBPAfuBh1R1r4jcJSJbxrn8HiBZRPbi/8C4X1V3TbbRxhgzHn/
qZlCOPn7qcvSqyneeOkhTVx9f+d1uvvzbnbg93tN+nM7efv796UO87bvP84Efvcb/PF8+Je0bLqQ
FU6r6BPDEsGN3jnLuxYO+7sQ/xdIYY2ZUd5+XxJhTgT4w60ZVEZFJPfaLhxvZebKVb717HXVtbv7
ruXJ6PF5+8MFNIT/GwdoOPveLNznW1MWFy7L48jtWcuWa3Em1azS2MtYYE3F8PqXH4x066yY+Go9
X6e33ET/oA+B0qSr/9exh8tPiee9ZRcRGRyEi/Oezh/nI5ibOXZo56nUnmrvZX9POnqp2fvLSMZL
iovn1pzezeZRrpooFemNMxOnt928ZGD9keuWpwmaTCfSvHmnizeMt3HX9WmKj/dnvz769hN+WneS
ux/ax9ZYLcUUJvf1ejjd1c7ShizePN/P0vjoqmroBEIELl2Xx/fduJCc1fsJtCZUFemNMxOnu8w+
6Dk7dDK5Jn50SN6HHVVX+/ZlD5KTE8b7SU5MRE2JdfOXqVdz64A7+7y9HaO3u41evn6Crz5+3j3E
J55Vk8YkLl3BGUTrLcpKHDBRPNwv0xpiI0+0E2OGzbmD0mvTl9R28drSZD29ePOrj/mlPLdsqWvj
Wu9eN+Ktgy8Z8fv7qcb771EGiBK7dkM9lq3NYmpXM0uwkkuJmL9xaoDfGRJweZwZMwrAcPfhTNzt
OtvL1rXv59g3rWZWbSluPh4//dBsnm3u4el0umckje/y9/V7+5cn9rFyYwvtLi0bcLyJ8970beXR
7Fe/ZVMiizMRpenWnz+rRG2MiTs9Aj37orBvwB/qfvVrBzpOtfOQnb1DR2MVXHt7FyeYeAPZUBy/
Hdf/LFZxs7uFr164m2hU8dC7JSuLvrlgRVkEeLNAbYyJQIHWTMCi9kurk6Ju7+nh6Xx3nl2TS7/V
x3X+/xJ/21vL/Ll0GwJ6qthGPt/1EC/c8V86lq3K4aPncW71vgd4YE3F6PP48/ODUTSBH/+SeGjr
c/Xz6oqU88IlzUODy1Tl88fIVLM5MZG/1qUDv9nj5tycPcMP/vkJKfDR3XjuiQvucYDl6Y0zECTo
Y66RuXipvJCU+mguWZREbHcXLt19Kclw0UVHCuvw0dlW1Dlzzjcf28cvXT3Dj2UX8wzWrB/4qmGu
sR2+MiTjdQXL0Ma4o4mOiUIUr1iwcmAOflhCDK8q/UnZdQRonm3to6/bQ7/XxxO4atmzM599u2DB
ngzxYj94YE4HcQWbdgH8uvdvTyzvX5QW9bl1BKgB7qtuIjhJauj28Y+30lCWYSRbojTERJ9hgLPh
Xx/b0ebloRVbQ69blpwH+AdnGzl5iXVG8feXcG3wdzgK9MSbijBboz1+WSUp8DHHRwUsgLEiKpSA
9gd1VbeypauO8ksyBQdy5bO6/AmOMGaanr5/4mCiiooZWqfzmu9aPe+26glReONhAZ28/n7xoxIZ
4c5INxhpjIs7wTUdOx/qCtIEyCVesXjiVzZo1FuiNMRHHv43gxCpUri3w5+k3FKaRmzb9lSVnggV
6Y0zEGb671OnYUJBGdJRExGybAMvRG2Mijj91M7FAn5kcxxO3XsSSrKQpbtXssUBvjIk4PX0TT90
ArFiYMoWtmX2WujHGRJzh2wjOdxbojTERp7uvf0Z3cAp3IQV6EblKRA6KSLmI3D7GeTeIiIpI6aB
jG0TkVRHZKyK7RSQyhrGNMWGrp887qX1hI824H3ki4gLuAa4AKoFtIrJVVfcNOy8FuBV4fdCxaOA
XwEdUdaeIZAKeKWy/McaM0G2pmyFC6dGfA5Sr6lFV7QMeBK4Pct43gG8D7kHHrgR2qepOAFVtUlX
vJNtsjDFjmsysm0gUSqAvAE4Oul3pHBsgIpuAIlV9fNi1KwAVkadE5C0R+ftgTyAiN4tImYiUNTQ
0nEbzjTFjeeNYM0caOme7GTPK61P6+n0TnkcfiSY9WiEiUcDdwMdGefwLgbOBbuBZEXlTVZ8dfJK
q3gvcC1BaWqqTbZMxBjrcHj74o9fwqnLN+jxuvWw5yyNs2mAwgY3BrUd/Sig9+ipg8Jbnhc6xgBR
gHfCCiFQAm4GtzoBsJfCiqjaqajfwBLBpKhpujBnb9hOt9PuUq9bm8vyBeq77wUscqA2+8XUk6e5
zthG0wdgBoQT6bcByEVkiIrHAjcDWwJ2q2qaqWaparKrFwGvAFlUtA54C1otIojMw+3Zg38inMMZ
MtbKKZlxRwnffu5Fnb7uYlPgYPveLt2h3R/Z8iJ5AiWKbXjlg3ECvqv3ALfiD9n7gIVXdKyJ3ici
Wca5twZ/W2QbsAN4Kksc3xkyDbRUtrMlLJTkumty0eO754CZONHfz5d/uRDVyM6TBthGc70L6yFP
VJ/CnXQYfu3OUcy8edvsX+KdYGmNmiMfrY/vJFm48e9HAsXOWZHDH1av45uP7eXJPLe9cH3w7vbm
uZ5RtBOczWxlrTATaW92O2+Pj7OKMIcc/fsESEmNdvHGseZZaNv0CqZtEy9EPsEBvTAQqq/AH8tL
iBUOOu6KENXmp7Klqm41mzYiBbQStRz/ARiuMmcO8PqWsohl3vw+AjYVppCfGUlbRwqKMRBamjqw
4sq4gjd9sO4nXp7iGbbUXCQKzbixHf4oFemPmqLZuD7f+ZjsvHDy1yDA/LZ6ff+pcyo4387YV2UG
vW1+Qxk9fqeBoQ2dEzqt3e2zWzXCWujEmDNW09fDR+97gr4eDrxQvr+/g+nte4uXyRv7x2jX87nP
n85ObSunt9/Gue16msbOP0sUZQa9dX+jfKm93hKZvui1HP4J95BkTZqpae/jAva9xorkbn0+5aPn
Qnnl3Xz8fu38bbo+XX31685AB14c+ex4f+fHrdLj7OXtYfj5gaVYS8TFR7K5q4282FU7ra5kNlqM
fyQK9MWGkpq2H9//wVdp7PFy5ZiHP7K+jsbOXrOS4gXO+/+dDVLb08JubN4+YVVOSncwjn7+At06
0jJqWiXZFRfSAbE+fFxGIi7aERYB9J4wJIw+XVVLZ0sMvP7WZL16xAp/Ck3tqB+7fcbKV+18+xof
OXcS5SzODPkZuWvy4c+TXF6Sxt7odry/yFk5193lJjHEhEnkDzRNlgd6YMFLd5iYrOZb1hWmsyk2
hJDuJx3ZWA9Db7+X23+0iOyWOr1y9alLPs64gje4+L8caI6+yZY/HawOxw1igNyaM1Lb1DEyJFBG
u3ZDPGxXN1LW7uf13uzlQ28G33rWe1PiYST1PYEB2T1XkFTnr6eu3qZXDWKA3JozUtveSl3Zq7vt
1G/NQhY/dv41Ht1dx2xUruHzNwkk/z7Ls5IEB2Uhjm46MZIHemFny5vEWzvrG09R3nNqUrbath9x

```
BgX5ZTgqrclPYX9POe84q5JZLl03Jc0e7olidlxqRgb7HY/vFDmeB3phZ8uz+Opq6+thX7U+fuD1
eWro95A5bzfqlK1fykc2L+Zd3r5/SAcYlmUlUt/ZM2eOFC+vRj2QjFsbMkrdOtABwsrkbgLp2f88
+Ny1hyHmXr1k4Jema4RYkxdLc1Tfljzvbevq8LEic3BhGpLEevTGzoN/rY+dJf9rkeJM/0Ne0OYE
+SH2a6ZCRFEt3n3egZECksFk3I1mgN2YWHKjtGKibfsLp0dcGAn3azAV6gJbuyOrVd/f1W/mDYSz
QGzMLtjtpm1W5KacCffvMBvoFif5AH2npm+4+r5U/GMYCvTGz4K0TrWSnxHFeSSYnmrtRVWrb3KT
ERZMcNzNph0CPPtICvdtjgX44C/TGzIK3TrSwaVE6izMS6e7z0tTVR82wqZXTLRIDvcfrw+NVS90
MY4HemBnW2NnL8aZuzly0gEWZiYB/QLa2vXdWAn1LBAV6q1wZXEiBXkSuEpGDIlIuIrePcd4NIqI
iUjrs+CIR6RSRL022wcbMddtPtAKwadECFmX4A/3J5m7/YqkZmnEDkJYQg0hk9ejbezwAM5b+miv
G/W6IiAu4B7gCqAS2ichWVd037LwU4Fbg9SAPczfw5OSba8zct/1EC9FRwgan3gzA0YZOGjqGlj+
Ybq4oIT0hhuYImnXz5nH/IPe6grRxzpxfQunRnwOUq+pRVe0DHgSuD3LeN4BvA+7BB0XkXcAxYO8
k22pMRNh+opXVeanEx7iIj3GRmxrPmyda8CksnMFAD/70TUuXZ0afczq9eqSJ1PhoVuelznZTwko
ogb4AODnodqVzbICIbAKKVPXxYceTga8A/zzWE4jIzSJSJiJlDQ3Bt04zJhKoKgdq21lXcCoQLcp
M5K3j/nTOTPbowR/oIyl189qxJs5dmhmRm55PxqQHY0UkCn9q5rYgd/8T8O+qOmbRa1W9V1VLVbU
0Ozv4hsbGRIKGjl5auj2sGLT706KMxIHFU7mpCaNdOi0WJEZOoK9u7eF4UzebR9mQZT4LZcSiCig
adLvQORaQAqwDXnAKLuUCW0VkC3Au8B4R+Q6QDvhExK2qP5iKxhsz1xys6wBgZe6pQL/YGZCFmVs
sFZCZHMv2k60z+pzT5dUjTQCcZ4F+hFAC/TZguYgswR/gbwQ+GLhTVduArMBtEXkB+JKqlgEXDTr
+T0CnBXkznx2sdQL94B69M8UyNjpqxotxLUiMpaWrD1Wd81vvvXa0ifTEGFblBt8rdz4bN3Wjqv3
ALcBTwH7gIVXdKyJ3Ob12Y0yIDtZ2kJUcR+agzb6LnB59Xlr8jAfbjKRY+n1Ku7t/Rp93Orx6tIl
zl2QQZfn5EUKabKqqTwBPDDt25yjnXjzK8X86zbYZE3EO1nWwMjd5yLFA6mbhDM6hDxi8aCotYe6
W9j3Z3E1lSw+fvHDJbDclLNnKWGNmiM+nHKrrYOXCoVP/MpJiSY6LnvEZN+CvSQ/M+bn0rx118vM
llp8PxpaPGTNDTjR34/b4RuSQRYRv37CB4qzEUa6cPhmJkVEGYX9NBwkxLlbkWH4+GAv0xsyQwIy
bFUEGC6/ZkDfTzQFOpW6a5nigb+3pIyMp1vLzo7DUjTEzJDDjZsXC5HHOnDmRUtisvcdD6hweY5h
uFuiNGUVlSzeHnV74VDhY18GijEQSw2ibu8RYF7HRUXM+R9/W4yHdAv2oLNAbE0RVaw/v/p9X+NT
PykK+pry+gwdeqRj1/oO1HUMWSoUDESHDmUs/l7V2e+b0rKHpZoHezGvP7q8b0Wtvd3v4xP3baOj
w142vaesJ6bH+4dE9fH3rXlqD9I57+70ca+waslAqXERCvZu2Hgv0Y7FAb+at3n4vn/5ZGdf94CW
27qwG/OWCP/OzNznS0MntV68C4I1jzeM+1raK5oHzDtWNLO20p6oNr0/DrkcPERToZ3hV8Vxigd7
MW7VtbnwKcdEu/vbX23nnf/6VS7//F7ZVNPOvf7OeT124hKRYF9sqxg/0//N8OYnOrkaHguT1H3j
lOMlx0bx9ZfgV7VuQFEtL99wtVez2eOnt91mPfgwW6M28Vd3q3zrhP248g49fUIwCX37HSl6541L
eW1pEtCuKTYsXsO1YS5Bre3ho20nq2t3sqWrj+YMNfO7tJSTHRY8I9FWtPTy+u4Ybzy4iNT78glF
mUixNnb2j3v9/fzkS0ofdbAnsKmWBfnThM/xvzAyrbvXn3hdnJPL169YGPeec4gy+//QhWrv7SE+
Mpd/r46evVHD304fo7vPiihKyk+NIiYvmo+cX89zB+hGBPjBA+/EwXZ6/IDGWdnc/Hq+PGNfQvl9
vv5fv/OkAV6/L4+zijFlq4dhaLdCPy3r0Zt4KDLLmp49eA/6cJf7gVlbRgs+nfPyn2/jm4/s5d0k
Gv/3seXz6oqUAfO6SEtISYli5MGVIjr7D7eHXr5/gnevzKBjjeWZTRpI/QLYGSd+caOrGp7C3um2
mmxWyNgv047IevZm3qlrdZCTFEh/jGvWcjUXpxLqieKOimdp2N3893Mg/XruGT1xQjIhwdnHGwKA
twPKFKTy47SSNnb1kJcfxUFklHb39fPqi8OzNw6l6Ny3dfWSnxA2570hDFwAVTd10uD2khGHqa3
bAv14rEdv5q2ath7y08cuJBYf42JDYRrP7Kvj3548wPklmQNBPpjA9MlDzirY371ZyRlF6WwoTJ/
axk+hzCR/cG/sGJmnP9p46q+T/TVTt3hsKlmPfnwW6M28Vd3aQ17a+OmUs5dkcLSxC4/Xx7/+zfo
xa8YHyhscquugorGLfTXtXDtLdWxCtdjZ+ORoY9eI+442dJHg/MUTrumbQKBPt+mVo7JAb+atmlZ
3SHnzwNZ0X7xiBYszk8Y8NzsljvTEGA7Vd/LEnhoArl4f3oE+Ly2exFgX5fUj5/8fbehkQ2EaWcl
x7K1un4XWjS8wGBuOaaVwYTl6My+1uz109PaHVAP+ouVZPPSZ8yhdvGDcc0WEFTkpHKrtYFdlK2c
UpYftIGyAiFCSncyRhpGB/lhjF1etyyM+xsWeqvDs0bf3eEiJj8ZllStHZT16My/VOHPox5pxEyA
inHMaW9StyE1mV1Ube6raeef63Em1c6Ysy0nmyLAefUtXHy3dHkqyk1ibn0p5fSe9/d5ZauHorPz
B+CzQm4h2uK6DTz2wbUTRrsAc+vEGYydixcIU+vp9AFy9LrzTNgHLcpKpbnPT1Xtq79jAQOzS7CT
W5qfR71MO1Y7s9c82C/Tjs0BvIlZvv5f/9+vtPLO/nleONA25rzqEOfQTtcKZebOhMG1g4+9wV5L
tH3sYnL4JTK1cmpXM2nz/9ofhOCDrX8xmgX4sIQV6EblKRA6KSLmI3D7GeIeIiIi7IpIqXP7ChF5U0R
2O/9fOlUNN2Y8d//5EAdqOxAZGaBqWt24ooSclKnv0a/KTSEuOorrzyiY8seeLsty/LOFBg/IHm3
oIsYlFC5IYFFGIslx0WE5IGs9+vGNOxgrIi7gHuAKoBLYJiJbVXXfsPNSgFuBuU9VqEVk
HPAXMnZ9+M2e9eqSJe/96lA+du4g3j7eMCFDVrT3kpsZPywBeemIsL3z5YhZOw4fIdFmcmUR0lAz
p0R9t6GRRRiLRTlmENXmpYdmjb+vpt0A/jlB69OcA5ap6VFX7gAeB64Oc9w7cVt6tqtXN
zL5AgInFBrjVmylQ0dnHLr95iSWYSX71mNWvz00YG+raekGbcTFReWsKc2r80xhXFoszEIT36Y41
dLM0+te3hmvxU9td04PXpbDQxKFW1bQRDEEqgLwBODrpdybBeuYhsAopU9fExHucG4C1VHbH8TkR
uFpEyESlraGgIoUnGBNfQ0ctN97pN97+BT5Uf31RKYmw0a/NTaezspb59oA9CdaubvDCf9jjTlmUnDwR
6r0853tTN0uxT6wbW5KfS4/FyvGnkwqqp4PZ4+cFzh8espDnyGh99XitRPJ5JD8aKSBRwN3DbGOe
sxd/b/0yw+1X1XlUtVdXS7Ozwq9dt5oa+fh+f+Ok26tt7ue9jXZw/0RgMDiXuctIPPp9S2uadlxs1
ctiwnmeNN3Xi8Pipbuunz+ijJGtSjz/N/H6ejFIKqcscju/nenw/x5311IV/X2uOfTZWeEDvlbYo
koQT6KqBo0O1C51hACrAOeEKwBA7KFwKPAR1X1X1yFQ02phgdlW2srpo0tblo
TmDFS5U/fNHb10uf1kR9C+YP5ZFlOMv0+5XhT10AFzsE9+mU5ybiihP01Uz8ge9/LFTy63R9WAlN
fQ2F1bkITSqDfBiwXkSUiEgvcCwCCGwN3KmaBA7KFwKPAR1X1yFQ02phgdlW2sruqjbuuX8uZi04tblo
BP68kc8jxlPgYFmcmDuTpT2ex1HxS4vwF9MaxFu56bC9ZybGsdnrx4C/wtjQriQO1UxvoXy5v5F+
e2M871i4kLy2eqtMJ9Fa5MiTjBnpV7QduwT9jZj/wkKruFZG7RGTLOJd/FAiQexE32pg
```

```
gDtV2kBTrClpyYG1+Kntr/KmbQI9xOgdj56ISZ4rl17fuob69lx/fdDZJcUMn5q3OS53S1M3+mnY
++/M3WZqVxPffdwYF6QnWo58GIdW6UdUngCeGHbtzlHMvHvT1N4FvTqJ9xoTsYF0HK3JTglaXXJu
fxhO7a2nr9vC7t6qIcQmLMufGYqaZkhwXTV5aPLXtbv73Q2dyRtHI0sqr8lLYurOatu7Jb8Zd2dL
NTfe9QVJcNA984hyS46LJT09gx8nWkB/DAn1obGWsiQiqysHajoF68MMFBmRvf2QXz+yv4/arV4f
l/q2z7dbLlvO992zkqlFKNwRSOfsnmb5pd3u46b43cHu8/OyT5wyk0fLTE6hp68EX4hTOgUBvK2P
HZNUrTURo7PQX4FoxaqBPA+DJPbVcsWYhn7igeAZbN3fceM6iMe8PzLw5UNPO5qWZY547Gp9P+eK
DOzje1M0vPnXukPesYEECHq/S0NnLwtTxU2ttPR5EICXOQtlYIrpHr6qcaOqe7WaYGXDYGYhdmRs
80GenxJGXFk9BegLffc+GMTcPMaPLSYkjIyl2Unn6/3jmEM8eqOfr160Z8WFR4Ex5DXVAtq3HQ2p
8zJxanDYbIjrQ/35HFW//3vPsPI2cn5mbAjNuli9MHvWcH320lAdv3kx6os25nigRYVVuyoRTN8/
sq+O/nivnfaWFfHjz4hH3B1I4oQ7IWp2b0ER0oP9tWSWq8OC2E7PdFDPNDtV1sCAxhuzk0StsrCu
YO9Ukw9nqvFQO1o5fCkFVKa/vQNV/XlNnL7c/sos1eancdf26oH9VBWZMVbWEFuhbuy3QhyJiA31
tm5tXjzYRHxPFH3fW0N3XP/5FZsp09/Xz7P66gV/y6XawtoMVC4PPuDFTa3VeKr39Po4F2WN2sKf
31XH53S9yxyO78Xh9fO33e2jv6efu928k3tmHdriU+BhS4qMHevQNHb383UM7BgZdwb8w7uP3v0F
1aw9tPR4rURyCiA30f9xZjSr885a1dPb28/iumlHP3V/Tzh92VI16vzl9f9hRzScfKOPBbSfHP3m
SVJVDdZ2j5ufN1Fqd5/8+j7dC9q0TrYjAg9tOctV/vMiTe2r54hUrWJWbOuZ1BekJVDmL2h7fVc0
jb1Xx1N7agft/9foJnj/YwM5u1mgAAB9dSURBVId/8jo1bT1W0CwEERvof7+jio2FabyvtIglWUk
8VDZ6wPmPZw5x64M7eP1o06jnmNMT2Jbum4/t42Tz9A6IV7e56eztH3XGjZlay3KSiXVFDdQOGs2
+mnZW56by/fdu5ERzN5sWpXPz25aO+/j5gxZNvXa0GYC/Hm4E/B/qfz3cyMqFKVS39lDX3mupmxB
ETKB3e7z8yxP7qWjs4nBdB3ur27n+jAJEhPeVFrGtoiXo5scAuyr9P7B3PLIbtyf89sSciyqausl
NjUdE+NJvd4Y8L3oiDtWOPePGTK24aBer81PZcWLsSQ77a9pZnZfKDWcV8txtF/OzT54bUv3/gvQ
Eqp259K8f83e+XjrcgM+nVDR1U9Xaw4c3L+L/PnwWMS4hL4RpmPNdxAT6nSdbuf/lY1zy/Rf4xAP
bcEUJ123MB+CGswpwRQmPvFU54rq6djc1bW4uW5XD0cYufvBc+Uw3PSJVNHWxsSiNO69dw+vHmrn
v5WPT9lyBGTcrcizQz5QzCtPYXdU2MCDr9Sm/ev0EPX3+jlJDRy8NHb0DBeWKnB2qQpGfnkBrt4f
tJ1tp6fZw0fIsWro97K1u56XD/jLmFy3P5uKVOTx328V8OoS/Eua7iAn05y7N5OWvXMotlyyj093
PFasXkp3in4GRkxLPhsI03jjWPOK6wNTLz19Swg2bCvm/vxyhvH7qy7DOJ16ff/1CcWYS7y0t5Io
1C/n2nw5M2zTXow2dZKfE2erIGbSxKJ3uPu9A/frnDtTzD4/u5mGnMxXI3wfy+acjUD460DG77cq
VALx4uIEXDzdSuCCBxU75iqKMxFEHds0pERPoAXJS47ntypVs++rl3POhTUPuO6Mond1VbfR7fUO
O76xsxRUlrMlL40vvWEG/T3nxUONMNjvi1La76fP6WJyZhIjw3fdsICclni/86q0hsyemSk2bm3w
rUDajNjp1cAIf3s8dqAfgeef/fU6gX5M39sBrMIEpln/cWU3hggTOKEpnTV4qLxys57UjTVy0PMt
mV52miAr0AdGuqBG5wDOK0nF7fAN1tgN2VbaxcmEKCbEuclPjyUiK5WCt9egno8KZdlec5e91pSf
G8t8fPJPaNjd3PLJryp+vts0d0nJ5M3WWZCaRGh/NjspWVJUXDvoD/CtHGnF7vOyvaSc/LX5Ci9M
KFvgDfbu7f2Dl7EUrsthW0UJHbz8XLrPNiU5XRAb6YDYWOj2QylPpA59P2XmydaB3IiKsXJgykPM
1oWl3e7jjkd20dvt3+6lwtporzjy1acWmRQv4/MUlPLG7lto2d9DHmajadreVHJ5hUVHCxqJ0dp5
s5UBtBzVtbq5Zn4fb4+PVo03sr2kfyM+frpyUU5u2BwL925b7g7sInF8ysRo789m8CfSLMxNJT4w
ZkieuaOqi3d3PGUVpA8dW5qZwuK5jWmeJRJpXjzTx6zdO8OQe/1zn403dxEVHkTusl/2Odbn+849
OXWqsq7efDnc/Cy3Qz7iNhekcqO3gyd3+NSq3X72KhBgXf9pdy5GGriGblpwOV5QM/OycuyQDgLM
WLyA+Jor1BWksSLISFqdr3gR6EWFjYfqQWteBaZUbCk/V3V6xMIWuPu9p7XIz31U6y9VfOeKfCne
ssYvFmYkjCk2tzk0lPTGGV8pPrVfo6fPS0tU34eeudTb8th79zNtYlI7Xp/z0lQrWFaRSlJHIBcs
yeXR7FV6fTig/H1CwIIGC9ISBkhXxMS7+ectavuQMzJrTM28CPfh/MA/VddDV6y+HsONkKwkxLpb
nnCqEtTLX//UhS9+ELLAg6tUjjaj69xxdPChtExAVJZy3NJNXjjQNlEb40sM7+cCPXpvwcwfSQJa
jn3kbnb+E2939XLLSv3HcxStz6HMmPEy0Rw/w1Xeu5u73bRxy7P1nL+JtKyw/PxHzKtCfUZSGT2F
Plb8nv7OylfUFaUS7Tn0bAqsrLU8fukCPvrGzjwO1HRxv6qZ4lN2bzi/JpKq1h5PNPdS1u/nTnlq
ONHROOFUWCPR5ttH3jMtJiR+YIXPJqpwh/yfFulg0iQJyG4vSOXeC9e7NSPOqWv/gAdnctHj2Vrf
z0WGlUlPiYyhITxhYbWnGV9nSzarcFA7UdvDo9ip6+30UZ43s0QOcV5IFwMtHGmns6MXrU7xAY2c
vORPolQdSN8PHA8zM2LR4Ab393oHfrYL0BFbnpZIaH2014sPIvAr0mclxFGUk8PS+On76cgVJsS4
+eO7IHXVWLEzmgAX6kKgqlS09vOesQno8Xh5+07/IpThI6gagJDuJnJQ4XjrcyI6TrSTFugbGRCY
U6NvcpCXEkBBri2Zmw53XrqHd7RkynflHHz3L5rmHmXmVugF/r35bRQtdfV5+8alzWZo9cqOKFbk
pHG3owjNscZUZqbXbQ2dvP4ULEji/JJNmZ2B18SipGxHh/JJMntxTQ1VrD5+4cAngX/Q0EbXtbuv
Nz6LslDhKhv0OFS5IHEjpmPAQUqAXkatE5KCIlIvI7WOcd4OIqIiUDjp2h3PdQRF5x1Q0ejLesTa
XnJQ4fv7Jcwb2ER1u5cIU+rw+jjeNXW/bnMrPFy5IHEjLxEZHkT9Gzvz8kix8ClnJsdx0fjEQ+o5
Cw9W2ucm1GTfGjGnc1I2IuIB7gCuASmCbiGxV1X3DzksBbgVeH3RsdXAjsBbIB54RkWWqqGmslIq/
bmM+1G/LG/NMyUAXxYG0ny6xQ1pgqW/wzbooyEshJ8QfcRRkjp1YOdl5JJiLw3tIiMpNiSYp1TXg
6a227e1LT+IyZD0Lp0Z8DlKvqUVXtAx4Erg9y3jeAbwOD/wa/HnhQV7jzerBovf1iSnUy
U2MybUJx0An3hgkSyU+JYX5A2buAtykjk4c+ex62XLUdEhtQfPx0er4/Gzl7r0RszjlAGYwuAwbt
2VALnDj5BRDYBRar6uIh8edi1rw27tmDE4jIzcDNAIsWjRWcxMS6Ks5I4OMENkOeTypYeUuO
jBzZ/+PknzxkyXXU0Zy3OGPg6Pz1hQjn6+o5eVLFAb8w4Jj0YKyJRwN3AbRN9DFW9V1VLVbU0Ozs
8FkRsKEijrKLFSiGM42RzN4ULTg28pifGhlx3PCA/PX5CPfraNv81FuiNGVsogb4KKBp0u9A5FpA
CrANeEJEKYDOw1RmQHe/asHXJqhyauvqGFEGbDao6YxtsT0RlSw9FGZObYZGflkBjZ99p7+5V29Y
L2Bx6Y8YTSqDfBiwXkSUiEot/cHVr4E5VbVPVLFUtVtVi/KmaLapa5px3o4jEicgSYCmwJehfwJhp
w9hXZRMmpOtuz5Y5HdvP+eydeImA6BebQD+7RT0S+MxXvdNM3NU6P3urcGDO2cQO9qvYDtwBPAfu
```

```
Bh1R1r4jcJSJbxrl2L/AQsA/4E/CF2ZxxczrSE2MpXZwxq4H+r4cbeHDbSd441kxjZ++stWM0TV1
99Hi8FC2YZI/eCfSnm76pa3cTFx1lm0MbM46QcvSq+oSqrlDVElX9lnPsTlXdGuTci53efOD2t5z
rVqrqk1PX9Ol3yaoc9la3T3n99FC4PV6+9vs9pMb7893BtkGcbYFiZpPv0ft75Kcb6Gva/HXobRW
mMWObdytjT8dlq/0Fmp4/OPO9+h88V87xpm5+8MFNJMW6eO1o0/gXzbDAYqmiSRSvglODqdWtp/e
BWtduO0sZEwoL9GNYnpNM4YIEnt0/s4G+qbOXH754hL85s4C3rcjmrOKMsAz0p+bQTy51ExftIjs
lbqBH/5ttJ/jBc4cHBmffOtHCZd9/gR/+5ciQ6wI9emPM2OZVUbPTJSJcuiqH35ZV4vZ4Z2y3+Rc
PN+DxKh+/wF8HZvPSDL7zp4M0dvaSlRw3I20IRWVLDxlJsSSd5nTKYPLTE6hu66Gps5c7/7CX3n4
fv32zkqvW5vKTl47R71P+eriRz7y9BPBvA1nf3ms7SxkTAuvRj+Nty7Pp8XjZ7dSwnwnPH2ggKzm
Otc6em4F9M8MtT1/R2DXptE1AgTOX/hevnaC338e/vHs9USL88MWjXLIqh8tX53Cs8VTtoZp2N31
eH0WTHB8wZj6wQD+OZc7uUxWNM1PgzOtTXjzc4J/e6dSLWV+QRmIY5ukP1XUO2Z1rMvLSEqhq7eH
nr1VwycpsPnjuIp689SJ++9nzuPcjZ7GhMJ2q1h56+vzpnMNOeYrARjHGmNFZoB9HwYIEXFHC8ab
uGXm+HSdbae32cMmqUyuEY1xRlBZn8PrR8OnRt3T10djZy4qFUxPo89MTcHt8NHb28emLlgL+UhR
nF2cgIizN9te3D/TqD9d1AkzZB40xkcwC/ThiXFEULkjgePPMBPoXDtbjihIuWja0FMTmpRkcrOu
gKUzm0x+udwLtFPWoC5wplmvyUjmvZOQWckuz/AH9aGOn8/wdZCXHsiApdkqe35hIZoE+BIszk2a
sNv3zB+vZtCidtMShi4DOXeIvAvbm8ZYZacd4Dk1x6iSQIvvsxSVB58UvcbYmPNrg9OjrOweuMca
MzQJ9CIozEznW2DXtNWfqO9zsqWrn4pU5I+5bmesfmA30pGfb4boOkuOiyZ+iWS/LclJ4+fZL2bI
xP+j9CbEu8tPiOdrQiapSXtdp+XljQmSBPgSLMhLpcPfT2u2Z1uf5y8EGAC4JEugDQbU8TAL9oTp
/j3oqV6WOt/3c0uxkjjZ2UdfeS0dvv+XnjQmRBfoQBDa6rpjm9M0bx5rJSIpldV7wnuqyhSkcrg+
PzVAO13dM2UBsqJZmJ3GsoWsgbWS7fxkTGlswFYLiLP9c7eNN3Zy5aMG0Pc/uqjY2FKaN2ktenpP
ML19vwufTUbfqq21zc/8rxzje2E1tu5uvXLUq6ODmZDR39dHY2cfyGQ60S7OS6Ojt55Uj/mmmM/1
BY8xcZT36EBQuSERkenv03X39HKrrYENB8A3LwR/o3R7fmPur/vy1Cn74l6Mcru9gb3UbT+2tnfK
2BnrUy2e8R+9/vj/vrSUjKZbMMFolbEw4s0AfgvgYF/lpCdM6l35vdTs+hQ2F6aOeEwisY6VvDtb
6FzE9e9vFrM5L5UjD1Of0Z2uxUmAu/dHGLptxY8xpsEAfosWZidPao9950r+T1YbC0Xv0y7L9gTW
wWCgYf+7cf97SrKSB6YiT9bNXK7j87r/Q2NnL4fpOUuKiZ7ygWH5aAnHR/h9ZG4g1JnQW6EPkn0s
/fT363VVt5KbGkzNG2d20xBhyUuJGnWLZ3dfPiebugUBfkp08pGzAZDy7v57y+k4+/8u32FfdzrK
FUzvjJhRRUTIwn96mVhoTOgv0ISrOTKS5q4+2numZYrmrsm3M3nzA8oXJo06xLK/vRPXUIGUgpx1
YTTpRqsqeqjYWZSTyxrFmyo63sGKWZrwE0jfWozcmdBboQ7TYmWJ5Yhp69W09Ho41drGxaPT8fMD
ynBQnoI9cvHXISemsyHV69Dn+Nh8Jkr75bdlJHnilIqT21bX30tTVxycvXMInL/SXTp7pgdiAEuf
Da5nNuDEmZDa9MkSLM/1TLCuaulgfQs97PF6fUt3aQ1FGIrsr/SWQ148x4yZgWU4ynb391La7yUs
busDoUF0Hsa4oFjulg4szkxCBo0EGZH/x2nHK6zt5/9lF49bZ3+OUaF5XkMqHzl1EcWYi12wIvoJ
1un1k82JKspPJSbE69MaEynr0IQoE+hNTUNzM4/Vxy6/e4qLvPM99Lx1jV9X4A7EBgdkmwQZkD9V
1UJKTTLTL/7bGx7goXJAQtEdf1eqmq8/LXw41jPuce6rbEIHVealEu6L4yHnFZMxSMbGc1HjedWb
BrDy3MXNVSIFeRK4SkYMiUi4itwe5/7MisltEdojISyKyxjkeIyIPOPftF5E7pvoFzJTE2GhyUuI
G5pBPVF+/jy/88i2e3FPL6rxU7npsH/e9VMGijETSE8cPnoHcdLAB2UO1HawcltIoyU4e0aN3e7w
0OlUwn9xdM+5z7qlqpyQ7mcRY+wPQmLlo3EAvIi7gHuBqYA3wgUAgH+RXqrpeVc8AvgPc7Rx/LxC
nquuBs4DPiEjxFLV9xl24PIvnDtTT2z+xWSyqyhd/s4M/76vjn7esZestF3DN+jwaO3tD6s0DZCb
HkZEUS/mwufQdbg/Vbe4RZYOXZiVztKELn+9UTr+2zb8Jd2p8NM/srx/Ym3U0e6vbWOfsdmWMmXt
C6dGfA5Sr6lFV7QMeBK4ffIKqtg+6mQQEoooCSSISDSQAfcDgc+eU688ooMPdzwsHx093BPP7HVU
8vruGv79qJTedX0yMK4r/vPEMvnLVKj7r7IUaitV5KZRVDC1XHBiIXTks0JfkJNHj8VLT7h44Fti
E+4PnLqazt5+XDjeO+lyNnb3UtLlZF8L4gTEmPIUS6AuAk4NuVzrHhhCRL4jIEfw9+Mj8dj8MdAE
1wAnge6o6YpskEblMRZMpEpKyhYWJBdCZcUJJJZlIsW3dUn/a19R1u/mnrPjYtSuczbzsV1KNdUXz
u4pLTCqRXrc3lcH0nB2pPfWaOVh9+YMOOQembQAmF95xVSGp8NE+Mkb7ZW+1/jjXWozdmzpqywVh
VvUdVS4CvAF9zDp8DeIF8YAlwm4gsDXLtvahqUcw0e6odOSuKK7ZkMcz++vocIc+n15V+dq
je+jxePnOezbiGqUgWaiuXp+HK0r4485THziH6jpIcAZfBxuYYjkop1/d6kYEijISuHJtLk/vrxs
1HRWYcbM233r0xsxVoQT6KqBo0O1C59hoHgTe5Xz9QeBPqupR1XrgZaB0Ig0NF9efkU9v9vv48/760
L+Zo3jjXz5311fPHyFVNSoyUrOY7zSzL5486agfn0h+o6WL4weURVy+zkOFLio4fMvKlu7SErOY6
4aBfXrM+jw93PXw8FT9/srfYvlEpLiAl6vzEm/IUS6LcBy0VkiYjEjEAjcCWwefIKqtg+6mQQEoooCSSISDSQAfcDgc+eU688ooMPdzwsHx093BPP7HVU
CuNQ5JwnYDByYbKNn06ZFCyhckMAfdoaevnlqbx2x0VHcdP7iKWvHlo35nGjuZmdlG+X1HWw/0cr
q3JHpFf/G2slDVsdWt/WQ72zyccGyLNITY3hs18jXo6rsqmxjXYGlbYyZy8YN9KraD9wCPAXsBx5
S1b0icp+IXOeYRb0Ja/1cCJwbYryBw8B+FGaL24BXIh5j/nVaaCfoJr2P8G30SHAzcm4b6JsbsR2zKE
yQUnmlE5PvHJtLrGuKB54pYKP/3Qbibgrwz+jD2zGHRsdxdXrcvnzvrohNXF
Ulbse20dlSw8XLQ/fdJoxZnwh5ehV9QlVXaGqJar6LefYnap6qCBi9apMXQ9drKEtqPrvMbZ7rSEo
17lujqt+dvpcyc9bmp+L16Zh14QMO13dysrmHy9csnNI2pCXE8PaV2Ty6vYr69Agh18vYr6915+9NFSipwVscO
tL0ijoaOXmrYeVP0rcvMHraq9bkM+3X1enjtQD/iD/L8+eYD7X67g4xcUc+PZRUEf1xgwzN9jK2Ak
IpD2qQwj0z+z35/IvWzW1gR7gA+cUEetM0Rxr56vAfTtOtNLS7cHt8Q28BoBzl2aSnRI3MLj738+
Vc++LR/nw5kXcee2aGa9SaYyZWrbUcQICveHqVvc4Z8Iz++pYX5BG7jTUbr901UJ2/dOV49aqWZO
XSmx0FNtPtg70+gcHeleUcM36PH71xgl+/Nej3P30If7mzALrLOgrwxEcB69BOQnRJHdJSM26N
v6Ohl+8lWLl899b35gPGCPPjz8OvyU9l+omWgzQXpQ6dhbjkjn75+H998fD9vX5HNt9+zYR0aY0
xc4sF+glwRQkLU+OpaRu7R//8gXpU4bLVOTPUstGdUbSAXZVtA0XZ8tOH/oVxZlE6KxYms7Eonf/
50CZiXPajYUyksNTNBBWkJ4w7GPuXQw3kpsazNgxWlZ65KJ37Xj7GcwfqiYuOGlF9UkR49PMXEB/
```

```
jmvSCLmNMeLFu2wTlpcdT0zZ2oD9Q2876wrSwyHOfuci/qclrR5vIT08I2qakuGgL8sZEIAv0E5S
fnkBtm3tIVcjBPF4fx5u6p2Ql7FQoSE8gOyUOn45M2xhjIpsF+gnKT4vH49WBuu7DHW/qpt+nLMs
Oj0AvIpzpbFWYP2xnKmNMZLNAP0GBbfxGy9MHNvAuCZMePZyaT5+fboHemPnEAv0EBYLlaDNvjjh
lgUuyk2asTeM5w+nRD59aaYyJbBboJyiQ5x5tLv2R+k5yU+NJiQ+fqo/nLMngq+9czVXrc2e7Kca
YGWTTKycoLSGGxFjXqKtjyxs6w2YgNsAVJXz6bSO2AzDGRDjr0U+QiJCXFh+0R6+qHKkPv0BvjJm
fLNBPQn56QtC59DVtbrr6vGGVnzfGzF8W6CchPy2BqiCpm4GBWOvRG2PCgAX6SchPT6Cxs3fEfqu
BqZWWujHGhAML9JOQ58y8qR02xbK8vpPU+Giyk+Nmo1nGGDOEBfpJKEgPXpe+3BmIDYcaN8YYY9M
rJyHP2UzksV3V/OSlo9S0ubnx7CLK6zu5dNXslyY2xhiwQD8pgdWxv3z9BNkpceSkxPGPf9gLWH7
eGBM+Qgr0InIV8J+AC/ixqv7bsPs/C3wB8AKdwM2qus+5bwPwQyAV8AFnq+r4e/DNAfExLu5+30Y
SY11ctnoh0VHCm8dbeGxXDddtzJ/t5hljDACiGrzM7sAJIi7gEHAFUAlsAz4QCOTOOamq2u58vQX
4vKpeJSLRwFvAR1R1p4hkAq2q6h3xRI7S0lItKyub7Osyxph5RUTeVNXSYPeFMhh7DlCuqkdVtQ9
4ELh+8AmBIO9IAgKfHlcCu1R1p3Ne01hB3hhjzNQLJdAXACcH3a5w9g0lEbgaCrrgn5Ap+8Ljl
EnhKRt0Tk74M9gYjcLCJlIlLW0NBweq/AGGPMmKZseqWq3qOqOqJcBXgK85h6OBC4EPOf+/W0QuC3L
tvapaqqql2dnZU9UkY4wxhBboq4CiQbcLnWOjeRB4ODdc2Nhf4l/N1JfCiqjaqajfwHLAh9OYaY4yZ
rCDEh7Xhi0Shcw59KrTp2Yv9aRmVIxvbT78qA6QJdI7XxNaUbnZH72lUVY3EuD0jE53bCOD
QNdDRNYhHvQXjpIwDQ26OHPtfJAJu2FAdUZIRR7UpxmOqanUPfrb5XfgWdkoCVhCnetP8BBtctV
A6R8S8ZMzzUeERWeX6AnSUgJZN49O3tH08BPELi8jXhNdhHp0nMjQthNXLE9mIM1Kd5PvYctP6k
TjnLLR3G7ov1QDjNPP0QCsgQ7eExPSzXNA7eGmq5+3e36Bu2aVgz8KbL9sVQCBRnNncSKdhU4
...
```

```
    "metadata": {
            "id": "fY2QI3mJRQfn",
            "colab": {
              "base_uri": "https://localhost:8080/",
              "height": 282
            },
            "outputId": "6646f50f-420a-4a99-bb86-e47562b84513"
          },
          "execution_count": null,
          "outputs": [
            {
              "output_type": "execute_result",
              "data": {
                "text/plain": [
                  "[<matplotlib.lines.Line2D at 0x7f2e25114490>]"
                ]
              },
              "metadata": {},
              "execution_count": 41
            },
            {
              "output_type": "display_data",
              "data": {
                "text/plain": [
                  "<Figure size 432x288 with 1 Axes>"
                ],
                "image/png":
    "iVBORw0KGgoAAAANSUhEUgAAAXcAAAD4CAYAAAAAXUaZHAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
    zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
    0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO2dd3wUZf7HP9/dTQ+QBELoJHRpAoY
    mAtKkeWI7X9ixoaee56Gn2M92h738bMehiKfiKaeColRBpBt6hxACCTWUBEJI3ef3x8xsZmZns5v
    tO/t9v155ZeaZZ2eenWw++8z3+RYSQoBhGIYxF5ZQD4BhGIbxPyDYPyzuDMMwJoTFnWEWuDMMw5g
    QFneGYRgTYv1AACgSZMmIjMzM9TDYBiGiSg2bNhwUgRbnQsLMQ9MzMTOk5oR4GwzBMREFEB10
    dY7MMwzCMCWFxZxiGMSEs7gzDMCaExZ1hGMaEsLgzDMOYEBZ3hmEYE8LizjAMY0JY3BmGwZwNhdh
    +uCTUw2D8SFgEMTEME1oe/WYLACB/2vgQj4TxFzxzZxiGMSEs7gwT5XA1NnPC4s4wUU5VDYu7GWF
    xZ5gop9puD/UQmADgVtyJ6BMiOiWi0OkFE2w2OPUJEgoiayPtERO8SWiPoEYNMMw/qOmfuZsS
    TmfunAMboG4moNYArABxSNY8F0FH+mQzgQ9+HDBMIKnimbspcSvuQogVAbB8JiYVsH4uBHjp
    M4weqeUHVlNRb3IkoEcCTAJ715cJCiOlCiGwhRHZ6umGVKIZhgsCmguJCguJQD4EJAN5EqLYHkA
    AVgCxEBQCsAG4moH4DDAFqr+raS2xiCVMemr0p1ENgAkC9Z+5+5CiG1CiCsPAPMA3CZ
    7zQwAUCKEOOrfITMMwTMMwzDu8MQVcjaANQA6E1EhEd1VR/efAOQByAUwHcAf/VYDL
    o5nimalsAeMD3YTEMwzC+wBGqDMMwJoTFnWEYxoxSwuGMFneGYRgTwuLOM4ME8X
    Y3BkmiuHUA+aFxZ1hohohibaWFxZ5goxs7qblpY3BkmiqlhcTctLO4ME8X
    UCBZ3s8LizjBRjNCJu36fiVxY3BkmitFX2GMzjXlgcWeYKEYv5r5r/u5ZKXZoHFnWGimJmDmr
    Oh2gkjL9hcWeYKGb+Nm2htCq73Svqmabu1lgcWeYKKZKZ3hKZg3
    DRDEjL2qq2T90ugznK6pDNBrGn7C4M0OwUo/eWmbflCK77cHWIRsP4E7fiTkSfENEJItqu
    T0VYi+o6IUlTHniCiXCLaTAGN0bYsBdC9AS
    wF8ATAEBEXQFMBNBN/swHRGT122gZhvE7f/9pNwRnBzYdbsVdCLECwGld21wqQ7U
    2BABfCSEqhBAHAOQCyPLjeBmGCSTmi4aCkEfBRDKdfoczs
    xVBzDhIJXFuwGAJTyIqrp8EnciegpANUAvqjva4UK7PT0dF5SWs7ibDZu3LyS
    iSQCuBDBC1KaSOwygtapbK7mNYZgwJis9KdRDYDEAVwkhylSH5gFFoCO
    A9b4Pk2GYQNK8UXyoh8D4GbczdyKaDeByAE2IqBDAEbBDAc5C8Y+IALCYiAFgrhLhPCLGDiL4GsBOSueY
    BIURNoAbPMIx/sBAXyjYbbsVdCHGjQfPHdfR/Gc/wuJuPtj/iWEYWC0s7maDxxZ1hGLC
    2mw8Wd4ZhQGyWMR0s7gwTpYSiGPaC7cfw9pK9Qb9uNLizjBRSlVN8MSdEXcSHa8z4hsaBIpKz
    hopRqXUm9G7JbuejpHw6dKnPfifEbLO4ME6UoJfWevbIAKAnpBbi0YBvd7+otKAnp/RwuLOMFHKpoIzI
    zAIClu48DACwR7DKzYm8WF+phhBVe55ZhGCaymTTI8ISSAIA1gR+BLjmW
    J+MAzd4aJcl6+pjsAzTJSjpB4B4IdncsL8+Pv/6DxeGHnkL2XY68tx+FHNvEEhLDGoX+Y/qmKvDxKwTf/kTe//ktfdyxi0IdbL4bE/eWN0O8xZWtGGXelAxjKUlN8XgNpUtsWr/QGPoRW9vbPL3rLAvnuIJ3X
    J1AOdgiDNjbjX2GvF/fVFnkWx6qNsy6sh/+c0CyzuDBOFqUkUkt01AzZ3mJ7C7nrAnOv/ul48/+ulD1ZbiUbOUp/7e7uQdajq9nD2uGYdB+
```

CvWz3CbfnXLP/lGa/3OAao7pmAAA6Nk0OScqFUMHizjBRiNqUoczYbQE0y1TX2PHqgj1199GJ+7b
DJW7Pe7pMa0dXvkBOlVYgc+p8LN55HFsLiwEA+06UYm3e6foMO6JhcWeYKERtmz5wUoocDaRVpqL
avRdLjc6bJsYD38x4m1WzXyyL/a6j5wAA93yWg+NnKxzHSwwWXM0KizvDRCFqs8ypUkkQA2lzr/H
AHKLPdRNjdT+e9AZxmv07P80xPJdCRXX0VP1kcWeYKMSuEtshndIBBDZCVT8rr9Z5tRw6VYZLXlq
iafNk5u5KxF9baGwCqvTgCcIssLgzTBRy+MwFx7Yi6YE0y+jt6RPeX+XY/nZjIYa8tszpNbEeLPC
6Slu848hZw/aEWKthuxlxe/eI6BMiOkFE21VtaUS0mIj2yb9T5XYioneJKJeIthJRn0AOnmEY71i
Ze9KxrchjICNUa3TirhbfN1y4PcbHeCLu0kz863sHatoHd2xi2L9RQozbc5oFT2bunwIYo2ubCmC
pEKIjgKXyPgCMBdBR/pkM4EP/DJNhGH9yRhWtqZhoAjtzr785xJNMAkpKA719fkC7xi7Oya6QDoQ
QKwDo/YcmAJglb88CcLWq/TMhsRZAChE199dgGYbxDzNUyblapyYCAAjBm7nXhTLrrvHgC0GZucd
YLZh0aSYaJcTAbhfYccTYjZLF3T0ZQoij8vYxABnydksA6rjhQrmNYZgwYZXKJAMASXFS5u9AZh/
Q29wB4IPluRBCOLlJTuzbxuVr9FQ5Zu4WxNksqKiuwYg3f8VP245p+j0jFyTx5JxmwecFVSGFfNX
7jhHRZCLKIaKcoiLPEwoxDOM9f/tmC26esc6x/9iYzo7tQOaWsRuI6qsL9iD/VBlOllZo2pPjpS8
bZZb9694iLNl53PC8irnHZiXE2iwor7LjwMnzmj7jezbHoA6NNeeMBrwV9+OKuUX+rcQJHwbQWtW
vldzmhBBiuhAiWwiRnZ6e7uUwGIapD99sKNTsj+teazUNpreMgpEve3Kc5NGi2NNv/2Q97v4sx3H
8cHGtp88LP0jpgWMsFny/2VBq8P5NfRzRt6eiKDOkt+I+D8Dt8vbtAOaq2m+TvWYGAChRmW8Yhgk
z2jZOdGwHNIjJhbgbmdUVM5HRa1bsLcKgab/g/WW5uFBZ4xBrm5VQqHLvVHj1+p4Aat/bM99vjxp
fd09cIWcDWAOgMxEVEtFdAKYBGEVE+wCMlPcB4CcAeQByAfwbwP0BGTXDMD5jtZBG0AOZE9JVEQ2
jyNWkWEncv1h30CldwL4TUqqE1xbuwdUqX3nF5q7nhmzJkFChyhYZLVGqbmuoCiFudHFohEFfAeA
BXwfFMIz/OaMzSegjUgPp564smibFWnG+slZcjWbnSmrgonMVePK7bZpj6qyPe46fc2zHWAnjejT
HtxuNTTMdM5Id24GuOBUucIQqw0QJ58qrNfv6BdRgBDF9PKkvMlWmILsQuKh5Q8f+81d1Q1KcDQ3
ibLihb2vM36q16rpKK2CzWnD/5e01bTdkt3Jsq1MZLNyh9aQxKyzuDBMl2HUmEL1zTCAntMq1bRb
C1b1rvaNr7AItGsU79m+/NBOAlHqgstqOoZ08c7aIt1kQa9WmFujdJtWw75Svt9Rn6BELizvDRAn
6Yhn6knSBnLkr1hciwqzV+Y72GrvAdoOAI8ln3Y6sJkkend9mtTjlokmMojwyRri1uTMMYw5m/HZ
As69PumUJ4FRPmblbSGtnr7ELTb51hbgYKyqr7fh0Qz4ArVePK/TfTYmx0S1vPHNnmChBqUjkikC
mH1CCmCxEaJRYm7xL7S2jRJECQKzVgvMVtWsEB0+Vub1GSqI2KZgnicfMTHS/e4aJIly5IyoEKkC
1xi4cUaNWC2HqmIscx9SRq3ddluXYjouxYKmuhuq4d34zPH+/rDTpNbqqTPp88JOHtPNi9JELizv
DRAnuqiEFKoip/ZM/4aX5u+RrAElxtSKsRK52VXnMANLMXc/Oo8Y52q9VLdC+M7GXY1sf/dqnTUo
9Rx7ZRLdRimGiCHdJFtt5uHjpCxYiTSIqJVp0bPdmmn77i0o9PufEfm0c2xN6tUST5DhM/XYrOqQ
30PQb3a2Z/qWmhmfuDBMlNEmOrfO4xUK4c1AWkuMCN+ezEKFHy0aO/ds+WQ8AsOpm2WfKPCtkbVR
8Y1CHJvjtseEa2z4Q2PQK4QiLO8NECS1SEtz2sZCzP7wvqBdFAcBqAZokx+F/f7pU026rh8E/VSX
aE/u1rqNndMPizjBRws/b3UdmWizkV3F/ZcFuzb4ye9ZHx1p1fpjpDeJcnrO8yo5HRnWSXlfP2fi
dg6RFW+HH9xiusLgzDOOAqDbgyB/M3XxEs68ESulFecVebU2HiX1dz8hbpiY4ArLqG6ikuEtGQ1p
3FneGYRxYiLwoveMafVZHZcKun7n/tk8r7nXZx6dd28ORiEzv/ugO5bru3ELNAIs7w0QZF7eSFjS
Noj69tbkfOHke6/JOadqMTB8Wl2YZ3b5K3C9pq80R07RBfG3Eaz2d8xX3yGioyMSukAwTZcyePAA
JMcYzXgt5Z3Mf9vpyAMCDwzrg0dFS6T59bVSgVoz1buzxuvGoNfuDm/ug/z+W1vaNteDOQVlYf+A
0JvRqUa9xKrb96hrzizvP3BkmShjbvRnapCUiMdYGIjI0fRCRT/bo95blOrbLq5yLYiiirU9S9tS
4izT76sN6T5rEWBtapyVi/kOD0STZ9cKrEcrMvcqd078JYHFnmCih2i7cLkAqOuoPbxJ9Fkqg1vy
iN8OU6lwmv1h3yLFt003zE108dXiCct1oMMuwuDNMFFBeVYPFO487iaoeZUbtD+0rr3KeHSsBUvp
x6GfnR0vKHdtEwKw7+wEAfntsWL3t7GpiZLNMNCyoss2dYaKAd5fuAwDsOGKcn0VB0U27ELB6kCV
ybd4pNFcV21Dz3LwdTm1KMi+9uI/p3tzlNZJjbRjaKR3508a7HY9b5MseLSlHq1T3aYQjGZ65M0w
UkFd03qN+5Ji5ezZ1nzh9LYa+tlzTtvf4OXyTU6DxXX9ibBdYSCXuOpu7OpkYANyoijz1ZaauZ+1
+yaPn8Tlb/XbOcIVn7gwTBRQWu8+HDtSaZXwxuV/13konk8y9Q9vj3qG1NU71gq2fyT9zZVfMXl/
g/SBcoPjd55307MsukvFp5k5EfyWiHUS0nYhmE1E8EWUR0ToiyiWi/xJR3dmKGIYJOJ6G6ReXVQJ
wv+C46+hZHDzlLJAN422GtnZ349F7zwSqitKY7tGTGdJrcSeilgAeApAthOgOwApgIoBXALwlhlhOg
A4AyAu/wxUIZhvCfOQw+Tf63IAwCsP3C6zn5j3/nNyRwDAJd3burRdfQz9/okDvMFV0WzzYivNnc
bgAQisgFIBHAUwHAAc+TjswBc7eM1GIbxEX0xDHdUVDu7MXqCka2+fbpznnj9g4Q7Lx5/oUTlZnp
QkzXS8VrchRCHAbwO4BAkUS8BsAFAsRBCcVotBNDS6PVENJmIcogop6ioyL
oZKtf+sjlTm16M0ywcq3HWC3o0qwBujSr35ddJOKLWSYVwAQAWQBaAEgCMMbT1wshpgUQ2en
p6d4Og2EYN5wtr3LUML2sQxOPXuOuJJ8RnTKSseuY1tXygWHtDfsmx9nw6vU9630Nf7D72Dks2OE
+/XGk44tZZiSAA0KIIiFEFYBvAQwCkCKbaQCgFYDDPo6RYRgfGP76r5izoRDJcTZ8fnd/j17jTYS
qhcjJ5bJxkusnhhuyudBGpUH+HX/hi7gfAjCAiBJJeqYaAWAngGUArpf73A2rm9DZBjGCCEEvs4
pwMp9J+vsd7K0AoBzci4j0pIk57aG8c7l69yhjipViLF5LzGxVgs6NE32+vXhzsZDZ9Dp6Z+dctn
7C19s7usgLZxuBLBNPtd0AI8DmEJEuQAaA/jYD+NkGMYDiksr8fiXWzDtt314Y/FeLN9VpHk+FQ/ljqcvpsz+tD7wVAR3xsXgsjyrq1tK4Nq5F0CuQxdAIOzB0dsB0
-
m3p8vQ/F5/5pzPCVYvvXu0Kc79hc8c2eYCOOzNQc97quuhhTvwcx9S2ExAOCLtYcMj585X4kXf9z
p8fXdfQk0jI9BmyDP3pWnhYHtGgf1unoOnpKeHL5cZ3yvfYXFnWEijE9X53vcV10NSV+E2gjFlTH
noHP6geNny/Hi/J31ur6/bPf+5L2b+gAIfcGOxkmBTbvFZhmGMQFHii+gRUqCU3tO/hnHtkemFKU
Sk655bd4pTJy+1uXLpMLazu1XXWwYoB5SlEXj6hCKe3WNHVsPlwAAXrq6e0CuwTN3hjEBT3+/3bA

```
9Ma5+JekcVmid7u05dq7O13XKaIDURGff+GYuCnmEEqVsX3UI66jO/r02333XFoFJhcDizjAmwJW
Inq+oXwIwJceLfk5bVln3eVqlJiA5XjIEKIFQgTY7eIuSgbK6JnQz92dUX8b6PDv+gs0yDBOmCCF
QVlmDpDj3/6blLsS3tKLWE6VJsnuxVWTmgK6YRV1eL49e0Qm3DszEjdPXAriAmZP6wi4E2jUJz+h
Sm1UW9zDxwQ+Uzw7P3BkmTJm9vgDdnluIAg/8sb/dZJzCqVQ1c/dkhugqe2TnjAYuX/PH7NZolBD
jEE0BKW96IwMzTThgk4tkV4ewSPa4HrVFQwI1c2dxZ5gw5ZfdUpDSzqN1F7UGgPuGGmdfVD/+j+v
hugi1wqRLMw3bT5yrcGq7omsGiIBGCZKIK+aOmhDasj3BYZYJ4cxdvYahfCn6GzbLMEyYogTZKCH
8D365EYt3aqNSr+iagUU7jzs8QFyx5onhaJLsPqe7UsBaz5PfbXNqe3R0Z0y/Lduxf7JUytVSEcB
Mh/7AYZYJoc19vyp7ZkyAxJ1n7gwTpijirixm/rj1qEY4+2el4Z2JvRFns7gV1OaNElwKtxpLPRR
BH/GqhPMfOhXeYf0x1tC6QuqrXHnyd/EGFneGCVO2FEh+0I98s8Xw+L9uvQQJsVbE2SyaYCUFb3K
ye1pIGwDidVkmx8rFpxt4kS44mFgd3jKhecKYt/mIZt8WIHFnswzDhCmHiy/UeTwlUfJ+iYuxOs3
cF+44hnv/s6He16zP4p4+Pzw5AqDCwwvFFYrNvSrIM/cau8Abi/bgiO7vGiizDIs7w0Q4kllGO3P
3RtgBwOIimVa7JknI07lHJujFXXbq8+KBIagQEawWCvrC74q9Rfhg+X6n9pj62MLqAZtlGCYCuWd
wlmM7zmZBRZV/hMpVpsQB7bVJtr68p7+zrTg8kix6hM1CQV9QPVtunP0yLobNMgwTtazWVetp2zj
JsV1ZY0dhHSaculJLo/pcZUFV19049L2zoW2Hx7REbnHSzGkU/gXvI+xWoK+oDp7vTb7IxHQqWk
DJMYGRoZZ3BkmArhphraUnk2lwgWnL6Dg9AV8tf4QZqw8gEUPD9H0nVCP5F2ubO6eCGHHjAZY+Nc
hbvuFA1YLBX1BdW2eNtPmDw9ehu4tGwXsemyWYZgIZPuREqe2qd9uQ+6JUnyy6oCmPTsz1ePzuhL
3KpUQPjDMOGAqkoixUtAXVC9qrk0QdqYssDVcWdwZJgxxN6u01bEIpy/8PKiDswnFFa5s7mpXy7H
d3Ue6hjtWC6EmyDb3zhnaXDuB8m9X8OnsRJRCRHOIaDcR7SKigUSURkSLiWif/NvzaQPDMACcH+H
1xNYRkWrkkeEprmzu5aoF20CLUjCwWSyoUnnLFJwuQ4+/L0RxAGfT53XJ3Xq1TgnYtQDfZ+7vAFg
ghOgC4GIAuwBMBbBUCNERwFJ5n2GYenDLx+vqPO6pP/rMSX3rdV1ycV61q2VdXyyRQoyVHJWYTpZ
WYPCry3CuvBq9XlgcsGsePKV1JdXHCfgbr/9KRNQIwBAAHwOAEKJSCFEMYAKAWXK3WQCu9nWQDBO
t/G10Z69f2yQ5DsO6NK3364yCatQzd1uYFJb2BavKFXL7Yef1i0Cw93ipYzsY99CXr+AsAEUAZhL
RJiKaQURJADKEEEflPscAZBi9mIgmE1EOEeUUFbmv7cgw0YLaZt7DhTfFXZdlGbar8SR/uxE392+
LhvFaRzq1zd0cM3eLY5G4MgiJzvSupCsfHx7wa/ryV7IB6APgQyFEbwDnoTPBCCm5heGqhRBiuhA
iWwiRnZ4e/n6xDBMsPlV5uyTH23BjvzZOfdR516/sabzA6W2kqM1CGtdHIQR2yylqv7i7PzIahl/
pvPoiRagGb0H18teXO7Y7NE0OSvlBX8S9EEChEEIxDs6BJPbHiag5AMi/T/g2RIaJMlR27wZxNhS
eqTvL4p8uN3ZN9LZGqE0X4FNwujZAqj6eN+GMzWpxuEJ6UunKV5SMmZOHtMOSKUMDfj3AB3EXQhw
DUEBEilFwBICdAOYBuF1uux3AXJ9GyDBRwoXKGryxaA+6NKutepQQa8Vv+7TRqd10BZVbpyUans/
bmalNF+AToEJBIcVKQIVsalIvFndt7v9i1WWV1Y5tpSh2MPD1K+vPAL4golgAeQDugPSF8TUR3QX
gIIAbfLwGw5iWLQXF6NmqEYgI/V5egnMV1chuW+s9rLdvz31gELrqxL2hixS7FwzSAHuCzUqwC8l
ObLGQy2RikczGQ8UAJJPTG4v2OtoDURVpc0GxY7thQvDSIfu0MiKE2CzbzXsKIa4WQpwRQpwSQow
QQnQUQowUQtTtsMswUcqy3Scw4f1VmLkqHwBwrkKa4SnZF1umJKBpg3g0k23cDeNtuLh1iqGf+Wv
X93Rqu6V/W6/GpS9Dp8ziW6YkeHW+cKbGLrDjiFTGsElyHKoCENik/nvd78KEFggif9mbYSKUD3+
Vgo0+XZ2vaVe8Zb6aPAAAsGrqcDz3h67Y8twVLs91bZ9WGNAuzbH/xd398eDwDl6NS5mpF5VKdVM
VkX9sjPdumeFKpcr81LRBnCbNgr9Qe8oM7hg85xEWd4YJEesPSA+1Iy4y9kVX6qJaLYQ7BmW5DDB
S+nw1eaBjf+/xc3X2r4s1+08BAJ6S66Yq/uB1pTyIVKpqBPplSl+KHTOSfRL35XtOYH9RqVN7mWw
e65eV5jK9QyDgrJAME2Jc5RX3xZ88yQ9pZIvLqlBRXYMXftwBIDD26FBTeKYM6/OlL9kYq8WrHO8
/bzuKP32x0bGfP2285vjGg2cAAA+P7OjDSOuP+b6KGSbCsLtwSPdF3K/oZhg76BGKd87mgmJ8t/E
wVuVKM/lAlYMLBY9e0QkAMP7dlY626hq729KGRqiFHQAmvL9Ks6/EJHTKaIBgwuLOMCGivVxEo0f
LRoZui/E273OPxPnwWjXqZFdWE5llmjVyXhz+Xi5crc8BU1+2qLxjhBAO81uD+OAaSszz12KYCEM
pcG0XWnc5BV9cEP2VIkConipiTOQSWddTSH3SEWw4aOwMqPi2PzN3O37cKmVj8dcXrqewuDNMiFD
ytZRcqNLkbvEH/lq4U0eq2kyQ6lch1uC9tE6TZvN1xQfMWp2P95flAgAe/moTrvtwjWG/rs8uBCB
9cYcK8/y1GCbCUAT9lQW7wzYZl9pcFOyydIHEKFbglWulWIELla7F/bl5O/Dawj0Aas04rjh9vhL
WEIb3srcMw4SI/UW1tt2T5yr8cs7P7+pvWILPW9Q+2nEBzj8eTGIMvkzjY6X3V1ZVg0U7jmFg+8Z
o4CL6d+7mw05tf7q8PT5UFUrp82LgcsN7Aos7w4QBeo8LI7OBJ1zWsQku6+hbcq+B7RpjTd4p9Gm
TghqVzV2f0yaSMbK5J8rifsfM3wEAY7s3w4e3XOI4rv6i+8tXm5HVJAkH5GjizkH2hPGE8HwWZBi
Tkzl1vmH7tX1aAgit2+GorpIb5cZDxdhaWPsUYIYiHQqFZ5xdHhN0TyZKJkdXr1GEHZD+bh/WUd5
w5ePDvBmmT7C4M0wYcXN/KXd7KBfirs9u5dhemVubkTKY0ZWBRl88A3BeMN5x5KwmYvW1RXtcns/
I20lNq1TjzJ2BhMWdYcKIRDmyNJSzZPUioNot0Nt0BuHI2O7aAifPXNkVyQZRvWcvVDm2u9dhltp
cUIw/e5nLJ1CwzZ1hgoy6jJ6eDk2TMbhjE9wywLuMjv7A0+LbkUyMTfseB7ZrjEaJdafjVdeR1WM
hwpRRnZCWFIvnf9jplzH6Cos7wwSZI3WEuMdYLfjPXf2DOBpnokDbnZKguVrjUC8ov7Vkr2EfALB
YpCebge0bOx0L1WIrm2UYJsioE3DdoLJvhwtGtvV3JvYKwUgChz6uwFWAlqfRqoqAt9DlvB/epak
jdXOwYXFnmCCzUy4OAQAnVP7toRIBPUZmmQm9WoZgJKGnwkNxf/fG3gCkqljqrJC3DmiL1KTYgIz
NHSzuDBNkpny9xbFdolqwawWczOkAABlKSURBVCcnEgs1JnKKqZPr+tQ+NbnKzFkh29nVdVD/cHE
Lp36JLlIs926T4ssQfYLFnWFCyL1DasuuuaqFGmzM5BVTF69c1wP3Dm0HAGjsYnb9+bqDAICc/DO
OtlQ3C69qlORwoYDFnWGCyFXvrdTsN1SlgY03UXh/JGCzWvD46C7Y/vxoJxEeKVfH+nLdIQC1ycR
SEmOw+9g5R7/5D12GNU8MD9KI64fP3jJEZAWQA+CwEOJKIsoC8BWAxgA2ALhVCOHa94thooSFO45
pIj7vGZyFQi+KQzD+w2IhJMc5y2DjpDjN/r3/2QAAmDykHX7cctTR3q1FI8Pz/vyXwYbnDSb+mLn
/BcAu1f4rAN4SQnQAcAbAXX64hmfmTEVT323Xbs/viv6Zqa56fEs6M2EgsFyXp6
9J84ZHq+pEdh5VFoQVxck13NR84ZonRb8qFQ1Pok7EbUCMB7ADHmfAAwHMEfuMgvA1b5cg2ys2X8
```

KNXaBmavyQz0UJsicLHXO/Ni8UTwAYNKlmUEeDWNEwwTJnj6ovXHyNauVcJOcImJ4F+PC5uGCr88
NbwN4DIDipd8YQLEQQllaLgRg6ENFRJMBTAaANm3a+DiMyOHVBVJ+CjMlYWK8Jz7G6lRQmQkdiht
ox4xkR5u6GlWzhvE4I0cYu3CwCRu8nrkT0ZUATgghNnjzeiHEdCFEthAiOz093dthRBxKAV4zJWF
iGLOgxJfZhcCVPZujQbwNVTW1Kn5N75a4Y1AWerZqhOsuCb8ANDW+zNwHAbiKiMYBiAfQEMA7AFK
IyCbP3lsBcM5qz4Cl3RwcKynH9sMlGCmnyXXFe7/s0+y/ecPFgRwW4yXqyNWUxBicK6/GvC21FZe
ICC1SEjDvwctCMbx64fXMXQjxhBCilRAiE8BEAL8IIW4GsAzA9XK32wHM9XmUJiQakjOFE7uOnjV
M8+orA/65FHd/loPvN7mew5RWVOP1Rdq8JHXll2FCxxNjL8KkSzMxvkcL7DoqLao++s0WN68KTwL
h5/44gClElAvJBv9xAK4R8bC2B48NB09j7Du/4ZNVBwJ2jaW7T7g81v25hU5triIaw5Eb+7UO9RC
CRmpSLP5+VTfE2izYcDCyPdr88gkTQiwHsFzezgPQzx/nZRh/kCfXKn1p/i4s2nEcX97T32WiKG+
5prdzSDqgzSMTqWQ0jA/1EBgv4AjVEPHV7wUor3JdZZ3xH+r8LevzT+N0mX9i6tReFFaL8b/SM3O
1vu3X9JacxxJiIycalXiFKCKJnGdDE3K0pBxZTcIjWZQZKTxThlmr8/Hv37TmGH+5sL2ztHaRtKy
i2rCP/tH+gWEd0L1lI/wxzD0t1ESrCfHGfq0xe32BYz9csnZ6Cs/cGVMhhMCC7UdRYxd44MtNTsI
OQFMX0xfeXlIr7lUeLtYmxlpx12VZfjcLBZIo1XY8PLKTZj/Sookj5xNmEjIb14Yk7z9RGsKRmJN
5W47gvs834uOVeS5n02q/ZX+xZv8pp7bth0s0+x/dcolTMYdwp01aIv6YHT0Lqmr0aw2RFpvC4h5
kLKoPyN2f5YRwJObkSHE5AOBUaSX2ufjy9MfMvUY3U5+9XsoeWFltR8HpMgDAlf+nzQA5pnszn68
bbJZMGYpmjXhBNRJhcQ8iNXaB4rIqTdvQ15aFaDTm5IJcVEFJ7mTEFW+tqNMv3ROMZuoT3l+FTk/
/jMGvLsP9X2gDt/tnRdYjvUKkzVaZWljcg8hT321zqnx/8FRZiEZjTg7I9/O3fSfr7Pfwfzf7dJ3
Siiqnti0FxY7tn7Ydc2zHWAmz74msxTgF1vbIhcU9iHz1e4H7ToxPKBV1mgfAlFBjF7j9k/XInDo
fpRWeu7H+9thwjTkukoiWqkxmhMWdMQ0LdxzDp6vzAUhupu6oqK7Bh8v3Y8eREsd+XSkK3ly8B7/
uLQIAvL1ESiegrsPpivQGcW77hBu9Woeu9ifjH1jcGdOgVMsx4s0bLsawztrsoyfOVuCVBbsx/t2
VEEKg89ML8OR325xeu2zPCeQVlWK1ys5eeEbKDXPnZZluxxWJduv/3NUPCx8eEuphMD7AQUyMaei
c0QB7jhtX0OmXlYaUxBgs21PkaBv8au1i9ibZXv7V7wWYdl1PzWvvmPk7ACA+xnku5K6o9cMjO3o
2+DCjQXwMOjcLj4LdjHfwzJ0xBS/P3+lS2AGgQVwMujY3rncJANd+sNqwfaVqYba8ytmFskG8dn7
09b0DcXEr6TqxVgvuuDSrznEzTKBgcQ8B/7imR6iHYDqMIlHVJMVZ6+2vXV5Vg1s+XldnnzibNkd
Mv6w0R0qJf1zbA40SefbLhIaIFveyymr886ddEZeAKxpK7AkhcOP0tYY27GCy6ZlRmHPfQK/C/T3
5XMXZnM87TK6tqY5GZphgE9Hi/s6SffjXijw8NmdrqIdSL5qnmDvib3XuSdz68XqsyTuFL9cdQmW
1f3K5eIra0yM1KRbZ9cgJomRtBICzF5zTF8TaLGidVptCwGIh/PbYME2fK3u2wPJHL6/XdZnwZOa
kvgCAi5o3DPFI6k9Ei7vihVDqIodIuDK4o9Zro6wyssavZvvhEnywPNexf+Z8JW6asQ4rc2tt1Rs
PBbbogf7+bVYFE+lJjpNs5E+Pv8jwuPqZ6oetR5yO731prEP0W8p5Yqp17pNWCyGTs32agmFdmmL
F34bh63sjLwgtosX9xn5tAAA9WrpeKAsUB0+dx/A3lqPoXIXh8c/W5GsSRym5vx8aIXlPLHh4sOO
YXhwAwG4XYW9uKq+qwZX/txKvLtiDEjmtQv9/LnXq9+S3gTPNXPXeSnR9VlvpaHyP5i77b39+NPK
njcfdg9vho1suwT+v7YEOTaVK98lxNk12x8Iz2ujhUXKd1GWPXo6EGCt+/LNUR7NtWiJuHdAWyx+
93B9viQkz2jRORAM3XlHhSESLe6vUBNgs5LcUrvVh5qp85BWdxw9bnGd32wpL8OzcHZrEUUqiKcX
e3qVZ3Y95ry7cgy7PLAjrSj5KQA8Ax+zdyASTd/J8wMawtbD2CzQ5zoYHh3VAapL0j6h8kbpiTPd
muLFfG3x6R1+8fE13NG0Qh0OnawVdnyri6l6SySYtKRa7XhyDVDka1mIhvHh1d56tM2FFRIs7EaF
BvC0kZhmlwLXdoPLDH95b6dRWI/czCmipMhDEj37dDwCY8L7zub7dWIgT59xHYAYadTTnz9uPuXz
SaGcgeiVlVTh+1r/v4fU/XoxHR3fGI6M6Y/KQdnhwWAePXtcqNRE392+LvJPnsaWgGOvypGClWJU
nzMrHh2Fcj8jL6shELxEt7oC0wOXNgt0/f96F+VuPen1dAc9ygh+QZ612eYgWg1wd6gXhnUfOYtS
bvzr29bnH756Vgylfb8HIN37FD1uO4C9fbcLh4gv1Hb5fOKIK8T90ugxdnlmgOd4qVbJJKzP3ElV
GzItfWIT+/3A24dSXhio/80YJ0ow9NSkWT467CLEGniye8Pk6KX1vcVkl2qUnYdMzo9AqNZHzrDA
RhdfiTkStiWgZEe0koh1E9Be5PY2IFhPRPvl3qv+G64w34n6hsgb/+jUPD3y50evrKqagd5fu09T
S1PPL7hMA1DN35z5Ld59A5tT5+PPsTRj37m8u85ADwJJdxwEAZ8ur8efZmzB38xHc/4X378MXXvx
xp8tjL1/THYv+Whu+njl1Pi5+YREW7jim6afPi14fNh06g7PltU9t/srh8sOWI8g9cQ5bC0vQOjX
RYX5hmEjCl5l7NYBHhBBdAQwA8AARdQUwFcBSIURHAEvl/YBx8lwl9p5wHZlohHqhzNtFy6/k2op
ny6txw7/WoMYuUHimDJt0niGKAFbLXwY2F4WUARja7wHgmDxD3nPM+H0qqWZ3HT2L95fl1vllo8Z
uFzhb7py61lf6tEnBzf3bIjHWhoQYbZDPvf/ZgH2qSNL2T/7k9XWu0UWV+jMT5Mg3V6DkQpVmXYF
hIgmvc8sIIY4COCpvnyOiXQBaApgA4HK52ywAywE87tMo6+BCVQ22H67fouOot1Y4tmeuykeN3Y4
Hh3ueA0QIofFw+T3/TJ0ideDkeYeJwNN56ieTsrFk1wl8ue4QjpRcQOPkWIx+e4XL/k9+tw1fyua
E1xbuQdvGiVj+6OV1mhJmrMzDP37aLb2Hp0Y6zXzvmLkey/YUIefpkWiSXHtMCIEHv9xUx9j7Orb
7t0vD8j1agVTff285ojNFDe/SFElxvqVK2vzsKPR6YbFP52CYcMEvNnciygTQG8A6ABmy8APAMQA
Z/riGv3ht4W7N/isLduP1RXuxoh4ztKW7Trjto/ajHvb6cry2QLru3M21FYDUgqlm38tjMbxLBq7
sKbn0XfvBanR86mfH8fE9myN/2njkTxvvaFOEXeHgqTKNr7kR6gLP+3R5WYQQjiRb2S8twTLZdFR
SVoVVuacwf5vxekVqYgxSEmvNGE+P71rnGADgvBcL4ur3u+aJ4fjolkvqfQ496nEzTKTjs7gTUTK
A/wF4WAihmUILyT5gOFkloslElENEOUVFgX30ffSbLcicOh8j3/wV7y+TvFD0EWe3fbLeo3P96fM
NHtU+Hd2tmcYk8f1myeQyrHNTR9uSKdqUqlf3aoEhndIRIxvmE2OdZ6JdmzfEWzf0cuwf+Oc4l2N
48ced2FJQjAXbjzmZX+x2gbLKWpNUaUU15m4+jO2HS/C/DYVOM9gX50vmpTcX73HKt6JEaL44oRt
WTR2uOdahaTIGtKs7UvPrnPoVMfly3SHkyusSV3TNQPNGCV4vnurpp4sq3f3iGL+cl2GCjU/PsUQ
UA0nYvxBCfCs3Hyei5kKIo0TUHIDhNFcIMR3AdADIzs72elVtYt/WmUjxR+cXlSfVtvp+Oc2lOSx+IBc
vrdC0fbYmH7cNzETm1PkAoJkZK/y8vXZB8KNbLkFljR0Pza41USx/9HKcOl+B1mmJWDV1OPq8qBX
JuwfXZglUzxQ/mZSN4V20DzlKdkE13z1wqUbI9GaXdU+OQF7Redz477XYe7wUE95f5Timmemv187

```
0J9eRCx0A8ookj5dZaw5q2hvG29A6LdHwXim8O7E3Hv/fVk263XE9mmFU1wz89b9b8PwPO3HHIM+
yJ544V67JV/PeTX08ep2npOgSfcXr1gwYJlLwxVuGAHwMYJcQ4k3VoXkAbpe3bwcw1/vhuae0ohr
VduEykKnaoD3n6ZEgIiyZMlQjSs/O3YHP19aK13UfahfsCk5rIxb7ZqbiqotbOPZnTuqLzCZJuKS
tNPtL03lZxFjJEf6uRy/sgHGJM30WQgDY8PRIx3ZGw3gMbN/Y8Brqhdanv99u2EfP//50aZ3Hbx3
Y1u05mjaMx8w7+mHzs6MweUg75E8bjw9uvgQTLq7N4zLs9eUejWeGLvujv2bsCskq18rG7CXDRDC
+/GcMAnArgOFEtFn+GQdgGoBRRLQPwEh5P2D8KPuqr8s7jQH/WIohqgIMgHO5tdVThzvZujc+M8q
xrRa9DQe1ni83/nutZr+xfJ5Zd/bDyIuaOrIBuqKqRjgJtrvUCdufH433buqNvS+NdTk7bpwc52S
DN7JBj1D5zyu4CvQZ2ikd39w3EN1aGEfSKnlVerf23NM1JVHyP1ewWAj3DmkHQFp0fu+XfW69lz5
eWSvuMVb/+53/Qf6ynn7rJVj6yFC/n59hgoXX4i6EWCmEICFETyFEL/nnJyHEKSHECCFERyHESCH
EaX8O2BU1QuDY2XIcOl2G8qoa/PGj1fg9/zR+0i38tUhJcHptYqzrR+/MqfMdM3altBoA/KL6xx/
aKR0zbu/r9FoAuPuyus0N395/aZ123eQ4G67s2aLeM9Qx3ZthYt/Wmra8ovPYfeysI7AKAB4d3dn
w9bPu7Ie+mWmIj7Fi5EXSl9bf5L7f3DcQSx8ZiiVThmJkV9/Wy9Vfvq8v2ouezy9yRL4eOHken67
SztTVfvGByIs/rHNT5E8bjyu6NeMFViaiMU2ZvQuqxcG/zdmK3/PP4I8frXG0TRnVycnnWsEoJ/f
0Wy9x2KEHv7oMcx8Y5Di296WxHovt01d2xQx5tpnR0Nk7JsZqQaDMutOu64nnJ3SD3Q5c9KwUPTr
m7d8cx2/Iloo7508bj6JzFUhvEIfcE+ecSsepv7geUM30lYRbvvDq9T0xT+XfX1ltx4/bjuKqi1s
4TDXX9G6FRokxGrPS8C5NMa6OBGEME+1EfPqBKaM6AQAOna6djRoFA/15eAfcI5sA9BjZtkfpZqT
qhcn6zqKVL4YXJ3Sv1+v8QZzNigQXTyb3DW3v2FZ83Ds0bYCmDYOXbz4+xorBHZto2h6avQlvLt7
r2F++V1qT/z1fMpOlJsbgk0l9ffZrZxgzE/H/HRN6tcCbi/c6gnFc4S4vyIzbstEuPQlpSbFIjLW
57K8XIk+4uHVKnd4kwaBJcixOlmqzHLZtHB5ZDP9zV3+Hh5LCu0trffD/8tVmTOjV0uGCyWUKGcY
9ET9z92T25omwjuyagXbpyUhJjHXMzL++dyD+/gdtEE7rtMgsnfb53f2d2owyVIYrby7agy7NGgC
Az3Z+hokGIn7mXpe72t9Gd0ZHH+zC/bLS0C8rDd9tPuLI36IsLkYaXZo1xPqnRsBmscjZDn23l/u
TSZdm4tPV+S6Pv/uLlC+eCI4gL4ZhXBPx/yV1mVseGNYBV3TzPQf3dX0kf+wpozoZ+qNHCk0bxCM
tKTbshB0A/n5VN8NKRrPv0ZY38zAnGsNEPRE/cw8GN/Vrg6Gd0sPGRm1WMpsk4Y+XtMI3ckRx/rT
xOFlqXMaQYZi6ifiZOyD5igcSm9XCwh4kXr6mB0Z3y8D250cDAFIStG6Zj17RKRTDYpiIwxTirtQ
l7Z+V5ogqVPulM5FDrM2Cf92a7UjTYFPZ1/e+NLZeqZkZJpohTws7BJLs7GyRk+M+06Ir7HaBt5b
sxc3926KZHws2MAzDhDNEtEEIkW10zBQ2d4uF8MgVxmH0DMMw0YgpzDIMwzCMFhZ3hmEYE8LizjA
MY0JY3BmGYUwIizvDMIwJYXFnGIYxISzuDMMwJoTFnWEYxoSERYQqERUBOOjly5sAOOnH4ZgZvle
ewffJM/g+eUYg71NbIUS60YGwEHdfIKIcV+G3jBa+V57B98kz+D55RqjuE5tlGIZhTAiLO8MwjAk
xg7hPD/UAIgi+V57B98kz+D55RkjuU8Tb3BmGYRhnzDBzZxiGYXSwuDMMw5iQiBZ3IhpDRHuIKJe
IpoZ6PMGGiFoT0TIi2klEO4joL3J7GhEtJqJ98u9UuZ2I6F35fm0loj6qc90u999HRLeH6j0FEiK
yEtEmIvpR3s8ionXy/fgvEcXK7XHyfq58PFN1jifk9j1ENDo07yRwEFEKEc0hot1EtIuIBvLnyRk
i+qv8P7ediGYTUXzYfZ6EEBH5A8AKYD+AdgBiAWwB0DXU4wryPWgOoI+83QDAXgBdAbwKYKrcPhX
AK/L2OAA/AyAAAwCsk9vTAOTJv1Pl7dRQv78A3K8pAL4E8KO8/zWAifL2RwD+JG/fD+AjeXsigP/
K213lz1kcgCz582cN9fvy8z2aBeBueTsWQAp/npzuUUsABwAkqD5Hk8Lt8xTJM/d+AHKFEHlCiEo
AXwGYEOIxBRUhxFEhxEhxEZ5+xyAXZA+eBMg/ZNC/n21vD0BwGdCYi2AFCJqDmA0gMVCiNNCiDMAFgM
YE8S3EnCIqBWA8QBmyPsEYDiAOXIX/X1S7t8cACPk/hMAfCWEqBBCHACQC+lzaAqIqBGAIQA+BgA
hRKUQohj8eTLCBiCBiGwAEgEcRZh9niJZ3FsCKFDtF8ptUYn8qNcbwDoAGUKIo/KhYwAy5G1X9yw
a7uXbAB4DYJf3GwMoFkJUy/vq9+y4H/LxErm/2e9TFoAiADNl89UMIkoCf540CCEOA3gdwCFIol4
CYAPC7PMUyeLOyBBRMoD/AXhYCHFWfUxIz39NgB9AHwohOgN4DwkM4w
D/jwB8prDBEhfhi0AJCEMn0wiWdwPA2it2m8lt0UVRBQDSdi/EEJ8Kzcflx+e
DAFxFRPmQzHfDAbwDyYxgk/uo37PjfsjHGwE4BfPfp0IAhUKIdfL+HEhiz58nLSMBHBBCFAkhqgB
8C+kzFlafp0gW998BdJRXqGMhLVTMC/GYgopst/sYwC4hxL+Ct+ylTOC7ZMMwGl6IkZKKiqHUEsBKfE8A4e9qg9lXtaiEnslv
2bjfZOsmzq4lEwY4Vwkn7cXRY3QMMFD/Yg2LbQbsTAJZbMbJ5EP8mPefjpKTdyMhm2AMXu1+Ct8NHl3YzWxhi+qmMXkVFehNtSq7uNY6pXDVUdKaX7OB60tdrhwrlIK6kQwZ4wNJYqo+Mi/9hiUXZHDBcSo
```

SPLITING DATA INTO TRAIN AND TEST:

```
{
```

```
"nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 2,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().any()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "RAUyZB0-Jp0t",
        "outputId": "7b396aa2-0bdc-4947-c627-34d1260cf12a"
      },
      "execution_count": 3,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             False\n",
              "Closing Value     True\n",
              "dtype: bool"
            ]
          },
          "metadata": {},
      "execution_count": 3
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "data.isnull().sum()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "yorF39lCJsyV",
    "outputId": "94c4ab77-12d0-49ea-9fc7-9a40dc1532d0"
  },
  "execution_count": 5,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "Date             0\n",
          "Closing Value    7\n",
          "dtype: int64"
        ]
      },
      "metadata": {},
      "execution_count": 5
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "data.dropna(axis=0,inplace=True)"
  ],
  "metadata": {
    "id": "g3FWuWVPJwms"
  },
  "execution_count": 6,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "data.isnull().sum()"
  ],
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "vIHbyOs4JzIf",
    "outputId": "2a7a4f4d-a27c-482f-d791-a2687f620cc9"
  },
  "execution_count": 7,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "Date             0\n",
          "Closing Value    0\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 7
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil=data.reset_index()['Closing Value']\n",
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "IoxUNwrvJ2b6",
        "outputId": "c8566900-c503-4f97-d55b-637fefb05fa6"
      },
      "execution_count": 8,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "0        25.56\n",
              "1        26.00\n",
              "2        26.53\n",
              "3        25.85\n",
              "4        25.87\n",
              "         ...  \n",
              "8211     73.89\n",
              "8212     74.19\n",
              "8213     73.05\n",
              "8214     73.78\n",
              "8215     73.93\n",
              "Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "metadata": {},
          "execution_count": 8
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from sklearn.preprocessing import MinMaxScaler\n",
        "scaler=MinMaxScaler(feature_range=(0,1))\n",
        "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
      ],
      "metadata": {
        "id": "5m-DUFI9J_WN"
      },
      "execution_count": 9,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil"
      ],
      "metadata": {
        "id": "fyMwOo1jKLqL",
        "outputId": "cbe459af-a1ce-41ff-9dd0-1bc24769a668",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "metadata": {},
          "execution_count": 10
        }
      ]
    }
  ]
}
```

TAKING CARE OF MISSING DATA:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,

"metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 3,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().any()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "RAUyZB0-Jp0t",
        "outputId": "3a675598-661a-4f55-9da8-78bc6eb43832"
      },
      "execution_count": 4,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             False\n",
              "Closing Value     True\n",
              "dtype: bool"
            ]
          },

"metadata": {},

 "execution_count": 4
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "yorF39lCJsyV",
        "outputId": "2aff2084-ea41-42fd-a3f0-92dbd1d6f03a"
      },
      "execution_count": 5,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date                 0\n",
                "Closing Value    7\n",
                "dtype: int64"
              ]
            },
            "metadata": {},
            "execution_count": 5
          }
        ]
      },
      {
        "cell_type": "code",
        "source": [
          "data.dropna(axis=0,inplace=True)"
        ],
        "metadata": {
          "id": "g3FWuWVPJwms"
        },
        "execution_count": 6,
        "outputs": []
      },
      {
        "cell_type": "code",
        "source": [
          "data.isnull().sum()"
        ],
        "metadata": {
          "colab": {
            "base_uri": "https://localhost:8080/"
          },
          "id": "vIHbyOs4JzIf",
          "outputId": "2d751d32-31b0-451b-8e0a-51c6e518497a"
        },
        "execution_count": 7,
        "outputs": [
          {
            "output_type": "execute_result",

  "data": {
              "text/plain": [
                "Date             0\n",
                "Closing Value    0\n",
                "dtype: int64"
              ]
            },
            "metadata": {},
            "execution_count": 7
          }
        ]
      },
      {
        "cell_type": "code",
        "source": [
          "data_oil=data.reset_index()['Closing Value']\n",
          "data_oil"
        ],
        "metadata": {
          "id": "IoxUNwrvJ2b6",
          "outputId": "fbfcd0c7-34fa-4574-9227-791bac289e27",
          "colab": {
            "base_uri": "https://localhost:8080/"
          }
      },
      "execution_count": 9,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "0       25.56\n",
              "1       26.00\n",
              "2       26.53\n",
              "3       25.85\n",
              "4       25.87\n",
              "        ...  \n",
              "8211    73.89\n",
              "8212    74.19\n",
              "8213    73.05\n",
              "8214    73.78\n",
              "8215    73.93\n",
              "Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "metadata": {},
          "execution_count": 9
        }
      ]
    }
  ]
}
```

DATA PREPROCESSING:

```
{
  "cells": [
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "outputs": [],
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 2,
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "outputs": [],
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 3,
        "metadata": {
          "colab": {
            "base_uri": "https://localhost:8080/"
          },
          "id": "RAUyZB0-Jp0t",
          "outputId": "9f1ec869-c35f-4764-8989-0d5d82646e18"
        },
        "outputs": [
          {
            "data": {
              "text/plain": [
                "Date             False\n",
                "Closing Value     True\n",
                "dtype: bool"
              ]
            },
            "execution_count": 3,
            "metadata": {},
            "output_type": "execute_result"
          }
        ],
        "source": [
          "data.isnull().any()"
        ]
      },
      {
        "cell_type": "code",
        "execution_count": 4,
        "metadata": {
          "colab": {
            "base_uri": "https://localhost:8080/"
          },
          "id": "yorF39lCJsyV",
          "outputId": "4001984b-54cc-4b54-f73b-926ff03ac00f"
        },

  "outputs": [
          {
            "data": {
              "text/plain": [
                "Date             0\n",
                "Closing Value    7\n",
                "dtype: int64"
              ]
            },
            "execution_count": 4,
            "metadata": {},
            "output_type": "execute_result"
          }
        ],
        "source": [
          "data.isnull().sum()"
        ]
      },
      {
        "cell_type": "code",
        "execution_count": 5,
        "metadata": {
          "id": "g3FWuWVPJwms"
        },
        "outputs": [],
      "source": [
        "data.dropna(axis=0,inplace=True)"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 6,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "vIHbyOs4JzIf",
        "outputId": "47f79e73-6c7b-4d04-a0bd-09ae39d18bcf"
      },
      "outputs": [
        {
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    0\n",
              "dtype: int64"
            ]
          },
          "execution_count": 6,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "data.isnull().sum()"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 7,

 "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "IoxUNwrvJ2b6",
        "outputId": "2011717f-466c-4af2-973c-3980b6229a4d"
      },
      "outputs": [
        {
          "data": {
            "text/plain": [
              "0       25.56\n",
              "1       26.00\n",
              "2       26.53\n",
              "3       25.85\n",
              "4       25.87\n",
              "        ...  \n",
              "8211    73.89\n",
              "8212    74.19\n",
              "8213    73.05\n",
              "8214    73.78\n",
              "8215    73.93\n",
              "Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "execution_count": 7,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "data_oil=data.reset_index()['Closing Value']\n",
        "data_oil"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 8,
      "metadata": {
        "id": "5m-DUFI9J_WN"
      },
      "outputs": [],
      "source": [
        "from sklearn.preprocessing import MinMaxScaler\n",
        "scaler=MinMaxScaler(feature_range=(0,1))\n",
        "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 9,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "fyMwOo1jKLqL",
        "outputId": "6e2bdf01-77aa-4e93-f117-6ab963e21c0a"
      },
      "outputs": [
        {

  "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "execution_count": 9,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "data_oil"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 10,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 282
        },
        "id": "GdNJartuKUfk",
        "outputId": "0640d4b9-abe5-4e76-a910-871af84bc19c"
      },
      "outputs": [
        {
          "data": {
            "text/plain": [
              "[<matplotlib.lines.Line2D at 0x7f5a0c7b5850>]"
            ]
          },
          "execution_count": 10,
          "metadata": {},
          "output_type": "execute_result"
        },
        {
          "data": {
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXQAAAD4CAYAAAD8Zh1EAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO2dd3gU1frHv282DUIIJQklARJ6lRa
KgHSRooId7OXKtXtt94d6sV1UbFdsV1GvyvXasKOgKEV6C70JBAgQakJJJSCD9/P7Ymd2Z2dndyWZ
ny+z7eR4fZ86cnTk7bL5z5j1vISEEGIZhmPAnKtgDYBiGYfwDCzrDMIxFYEfnGIaxCCzoDMMwFoE
FnWEYxiJEB+vCycnJIiMjI1iXZXiGCUvWr19fIIRI0TsWNEHPyMhAdnZ2sC7PMAxjEdj2sC7PMAwTlhhDRAXfH2OT
CMAxjEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWwaugE9FHRSCiLa5f9OU5E9CYR5DRFiLq5f9
hMgzDMN4wMkP/BMBoD/D8fHAGgn/TcZwLu1HxbDMAxTU7wKuhhBiKYBTHfrQMB/BfYWc1gHpE1MRBtr2ONh
DYfxIQBdFhRDvCyGyhBBZKSm6kasMwwSAc+VwR4CYwL+EPTDAFoo9tOlNoZhQpRqrlRmSfwh6HM
A3Cx5u/QHUCiEOiEOqH8zIMYxJV1SzoVsRrci4i+gLAFWMNRM0DWMUYAlOqSMb6RDJSFFEM3k2AN6Da
zBsswjH/gGbo18SroQohJXo4LAHf7bUQMwzBMleBIUYZhGIvAgs4wDGMRWNAZhmEsAgs6wzCMRWBBZxiG
MYxhsKKBQOhCodBmGYhhSwGCCzjA8ZmQ68lUYUW6oSWbMMbBIozCbmsZ9lkzbP1nQ0LsRuH87n8i3e6nmJUQLb0ejG0GHRI07A9M6t
IYcsghBKTBKUkjqBDhfXZ2Gs6H8jTLGrJNkgULbQ9F83cdjWpTDkBGL0ezXzr8iLxYSWuRm03LUBhYgBxTrD8cRV5iHaZyXnEhDlmDEQ+nHfu0vl6SdCGhb7NfWNeE9jVOkh
KvhKhI+jCTwmUaLUnmzk4EdKtQwvMUVyrfOBlsUwl+gEByCd+1EmFZTGrf1kh3W4/XQTqxlPl3N0uvWI4j9OAYlU7GCBM1G6bCkq9B02yn4wrxv6nTB7zP2WdkCWa/v5fXpo3oBLOYT/2jRrjtjgSZOYq5Rm4Xw
Jq6gZ6tYf5XzgXlvuE2C4gXy9a0pnHCjpXkKbJXNhUWWH5cxEjg6+HKcQYdvgHjJCuoq6ZD9ktxFvpOt0ah6IixGohWGn65bMcTeYXHMaEs1RykIQSpEnUB14FSAfXUPxIpTTNY1OPKuyxS3ZNxjRQh5x2Mn84FaHZ1XIwOc5ehQJWgeYZV2YrdEUa4sPnMjZT2r9wggDY/K2SDgvzCqQFufoyHgC7V2tV6cArm2r12fFJGA2rxDnlzkAMLiGZQPAmUsv/81Cqt9dSJe5EcPcpRRe3PiRSlh3hiD8/U0Fss47dRRnEz1y5rrVnkFEBiz4LJVKt2G/zBgN+9O/g2fD8lnUyx2z+W2dS+pa5S14/i04ypjmXpkOJ6jCuWN0xGS4JbC4e+7vUncH5jZ9ZBJe5gZhpwO/w6/QjYi2WhH33/dl1IX9BFagtJMn9PcmwIlNwoGhmY7P26iTSZDNz8bAh9eVfuevcRMe+IKRg+4Z3STJvnB6Dvu20rB0BMMAoQVxjN1UvDXnHA8/wkxxDzNJDoiN1/r2dLCMPsJ/60F+/LsO6s2Wlqfu+LUuJZn8hw+AHJ
```

```
xbcBL0Ge0Xi2As76olnrx9uyPVZL4/rjpMNbpLOICTtGPthFAQM6JYpc+Dwxv63hw8QydYRjLsGa
/Whib1HcG5phpQ6/JYmS9OJv9M5I55cEvN+Ga9+xeMNXVAkcUJfJemb8LABATFaV681Dy8KgOsEl
BSiVl7lMDWA0WdIaJYMx0W6xyY+rQm7nXjZVn6K7H/rN8PwZMX4Qlu/N1M0Zq+fi2PgCAXGkReNI
Hq2s28DCGS9AxTAShzX6odS30J+78xat0gofOldtn0Ut256NrWpLq2NI99pzqt3y0FiM6pnq97rA
O9j6nz1XUaLxWgGfoDGNhtMFE2shQM7Pnugu515uF1421m1xOlZTjopcXq44dVphbFmrS4daPdz8
nvWNQpuGxWgUWdIaxMNpsh9p0tmZ6ucgml4WPDFG1a8P7Z1zXA82T7Mm7+rVu7HIeTxkbX7+uh2p
/6qWdHdtyegPAnmYgEmBBZxgLoxVPraCbaUOXJ+IJsdHo0CTR0V5VLVT7E3qmIVbKy17TzIhaO3r
7JvV0++mlGbAiLOgMY2FOnVP7Zmtn5GbO0OWHSRQBuxSl4KqqhWofgEPQyyqr0CYlwfA1tNWLZNN
NpMKCzjAW5uVfd6n2dyiKPwDm2tDlGbrWNVKviIUtihBjI5RXVmOvZGK5qF2y12toy8/J3jKRiiF
BJ6LRRLSLiHKIaIqbPtcS0Q4i2k5En/t3mAzDmIGZfuiye6L2oaH0C5+hsIHH2qKQf7bMsb/x4Bm
cLdX3VGlSPw4AMKCNWvR5hu4FIrIBeAfAGACdAUwios6aPu0APA5goBCiC4C/mTBWhmH8jFkz9NK
KKuSdttfztEURruyZ5jimfIZMULTHxdjw9XpnRsbiskq8Ol/9hiEzsU9LAEDTJHX1Iu0MPdGDF4w
VMTJD7wsgRwixTwhRDuBLAOM1fe4E8I4Q4jQACCEiq9Q2w4QpZs3QO0791VFwgoiQqohOlRc+J/V
tofpMrE4eglmr9HO/DFKYYx67pINjWztDv0LxwIgEjAh6GoBDiv08qU1JewDtiWgFEa0motF6JyK
iyUSUTUTZ+fn5vo2YYRi/cWUv+59yamKcadeIIkDA6W0jC3rfzEaqfkYKRMv0yXB+9t5hbfHaNd3
RoUmii6A/Oa4TAHO9eUIJfy2KRgNoB2AogEkAPiCiBtpOQoj3hRBZQoislJQU7WGGYQJM3dhojOr
cBI0SYk27RhQRLruguWP/5o/WAgBsUb7Jz60DMlzaruqdjvkPDXZ544iLtgu8gcp2lsDIHT0MQPl
ulC61KckDMEcIUSGE2A9gN+wCzzBMEGlYN8ZrnygiQ7U8jaJMpAXYbehd05LwxkR1EFB0DQz4Q9o
7J4BDOvBk0B1GBH0dgHZElElEsQAmApij6fMD7LNzEFEy7CaYfX4cJ8MwPmAkn0lUlLqAdG257n1
1EI88adYGNWn3PdGyUV0MbGuPIq1pYethHVIQFx0ZHtpev6UQohLAfQDmA9gJYLYQYjsRPUdEl0v
d5gM4SUQ7ACwG8JgQ4qRZg2YYxn+Qn2foh06pZ+hy8JJWiA+dOqfa97SAmZIYh/NSAq+auiamJMa
ZalIKJQz59Agh5gGYp2l7SrEtADws/ccwTBgRRQT4Sc8Listc2mRB19q3d2uiRT1NvG++sBXmbz8
GwGkXN4otKspt5kerERnvIQwT4ciz39FdmrociyLXnC9G2Ha4ENsOF6razuiYeNxZVlxMMApFf2C
EegkuMT7GYRaq6VpqjI3c5ma3GizoDBMBvH5dD+x/cSzevbGXyzH7omjNz3npW8tx6VvL8XW206t
Zr36nO1937UxbmVfm4Yvbq47ZogjPX9EVWa0aom2qfgIud9iiyFEJyeqwoDOMhemaVt+xmEhEuuJ
KPs7QZV77bbdjuybZEm8fqM5XXuFlFt2rZUN8c/eAGptcYmxRXs9tFVjQGcbCVFUDdWI8L5VFEfn
NT9tdUQs94mLU8vPdBq03tH+wRVHEFIpmQWcYi1JQXIadR4sQY/Ps5uerDV1GGeFZWmG8ILM358M
nxnYEAPz5T93Ac8PERBEqqgREBEQXsaAzjEW57/MNAICjhZ5D6msaWLToz+OqrIhKbvt4ndvPaa0
9yfU8pxuYPLgNcqePQ3xM7TIolkkpdkvKjT9swpXISkXGMBHE6n32smubDp3x2I9qsChaWVWN2z/
Jdmlfl3sKRwtLUak40aS+LbF6n/twFG21IbOYucQe4zhrZS7uHdY2INcMFizoDBPhRBEMmyMqdZS
/Xlw0rnnPtcTbi1d2q9E4fr5/EC59a3mNPlMTXpm/y/KCziYXholwDpw8h4Licq/91h84jcOaPC0
AcImOb7svaHOb+wvZyycS4Bk6w0Q4y3MKANhrfXrKr3LVuyt12+XqQbXFrPqmY7o2w4qcyMhEwjN
0hmEA+O7pomd/H9vNddbuTa7NMqlf1t2eundCj+ZeeoY/LOgMY1G0FYG84aug69nf/31Db5c2b94
qZlVPSqoTg3px0V69aqwACzrDWJCTxWXIO223d+sVhNDDUzClu8CcurE2F0+Wl6+6QLfvRe2S8Ds
lnXWPAebN0AF7fdIPl+837wIhAgs6w1iQ3tMWYNmeAnRsmohnLu9i6DOeZugVVfpqbyPC5jx1gq4
GbopqEBFuH5SpeyySqEl6hJrCgs4wYURFVTU+WbEfW/I8+5bL1CQox5PBRc9dsUvz+jhbVunSHuN
jMYnEePuDYLyFbd1fZx9C+3/8goMnz3nv7APs5cIwYcST32/F7Ow8AEDu9HFe+9cxIOh/G9kOMxb
s8TxD15lVupu1x/hYKxQw9p1qQ3llNWKDWL1o3tajAICc/LNo2biu38/PM3SGCSN+2XrMax/lK31
8jPc/cXlm7GlNVC9bobush95yxwQTbVGNQGN2NhkWdIYJI/RMHFqUVYPqGCjXJsuvp2hRvYo/7ma
6vppcAoE/S+2FImxyYZgwYMnufNzy0VpDfZUeKfEGcofL3oKetK5Sx7wSa9MX7mgv7irZ/xgZtII
TZgUvGUW+x2Z9/9B9lDIM4+Cxrzcb7ltW6cwqGG9ghv7VOnvFoewDp3WP5xaUYMaCPS7t7mbo5CW
EKLlenGlh/t7wFAkbCJbszgcATP50vSnnZ0FnmDDghJt0tXoUlzkF/YhO7hUtfx6z25VzThSr2oU
QOF5Uips+WoPvN7oWn3DnfidMtxTXnHdvsJfeC5VSdN7eYnyFBZ1hwhR3Xibfb8hzbB8vMv4g0Ar
xp6sPoN8LC3HolP5Dwd0su11qouFrBgrZfVOv5mmgKCqtQGZyAgDgu3sGmHINFnSGCVPW5Z7Sbe/
UrL5ju44BLxcZrQ191V7PCa2S6ugHEBlZiA000ZLnjZ4/faB4ZPZm7C8oAWBeZkkWdIYJU9wtSp5
XlIFrUDfW5/NvPOg5eCkl0ZkbJatVQwDAaD+l0vU30ZJvfDBNLr/vOO7YNmtxlgWdYUKI6mqB8wZ
LpZ0q0c9hXlzqdG2syWx51spc9Xk8uEj+c0JXTB7c2rE/86be+PzOfpgxsYfh6wUS5ww9eCYXJWY
tzbKgM0wI8dzPO9DpqV8NVamftSpXt7243CnENgMzQTmfuXbh9ZqsdLefual/K8Qo3hCICAPaJNe
6/qdZyIuQwZyhK01UPENnmAjgm/X2Bc3C8xVe+17VS19w5RqaAHBhG+/VeqZfqZ8dcW9+iUvblb3
S0EzH/hu6saF2HCaXINrQlf+m0SZF03JgEcOEELKftOzBMur1JS75WG4fmImPVuz3Ootf/n/DkNa
gjtdrJsbry8BSyWdaybQJXVE31rV/kON1vOIwubjxDDIbbem+GDfrH7WFZ+gME0LIgl4qLWzuPl6
sSk/78MXtcdcQu+26zEsa1vSGdQ0VjYiqgU+0u8jTIE58DRETZC+XQ6fU2RVZ0BkmApAXOoe88of
u8QdGtHNEf5ZWuC6eylGi7lwK9aiJPVcr/g3ryom9QlvRbQ6TS3Bm6O/+sVe1b1bEKptcGCZEqXY
zm4yTQu61M/Q3F+7Bv37fDcCYDV6mNtIivwGEtpw7F0X1koyZybnySsxYsAcr9xYE5HqGZuhENJq
IdhFRDhFN8dDvKiISRJTlvyEyTGSgneW6M6nI/ufa47KY1/i6Pn3KjjNTYy1OEgBkG7oR7yF/MvW
H7Xh/6b6APUi8CjoR2QC8A2AMgM4AJhGRS2FAIkoE8CCANf4eJMNEAlr7rtak8vLVdm8UIkJcdBT
KdEwuvuDOXDK4fYpq/4d7B7r0CfXFUBlnYFFgTS47jhYF9HpGZuh9AeQIIfYJIcoBfAlgvE6/fwJ
4CUCpH8fHMBGDNtnSVxkPq7IepisjMsspqjwm7Hh3V3vB13c0dtSafHi0auPR5/boe6N2qIRol+B6
RGgiCtSi6U0fQL+tuXok9Izb0NACHFPt5APopOxBRLwAthBBziegxdycioskAJgYaz5ahrE
wWhPK7Z9kq/a16Wq/33gYTZPisevYWXx0ax/VsT4ZjQxf1525RLmA6M798aJ2KbioXYrusVDCFgK
BRQCw7dlLUC/OvKXLWp+ZiKIA/AvArd76CiHeB/A+AGRlZYW41Y1hAou3avDROrU6Ze8JrZ9zeiP
```

```
j9SrdmVyUdt9pE7oaPl8oIrsJ6pXSC+w4zLVRGTG5HAbQQrGfLrXJJALoCuAPIsoF0B/AHF4YZZi
aoSxMUVNW5qi9KIwEFMm4s0Iobfg9W7qaW8IJeYZeFeAZupy0TKY2BbSNYOTs6wC0I6JMIooFMBH
AHPmgEKJQCJEshMgQQmQAWA3gciFEtv7pGIbRY8HOEx6Pe6qH+dg3W3y+buuUBN12paCbFQgTKBx
ui4qn15p9JzHopUVe34xqQ4km0VpNgrh8weu/khCiEsB9AOYD2AlgthBiOxE9R0SXmzo6hokg/vn
zDo/HjbrcfV/D4gnJ9eJ025U2fXfl5sIFIkJ0FKFKMrlsO1yI695fjbzT53HZW8tNu+6e42dNO7c
ehmzoQoh5AOZp2p5y03do7YfFMJHLwxe31/UpN+KhcedFmejZsqHXfkYorXAKulkl0wKJLYoci6K
v/rbL0b7LRNFV/ptdbqJ3i0x4P3YZxiIsUSTCuvSCZrp9+mV691zx1S1vZKcm6KyodARA5eduJCd
MqBNji3Is9BpJK1xbtD7v06/qZvo1WdAZJgS4c5ZzyalefLSqjJyMkVzj7tIFeCM6ilRuiiVllTh
bVolmSfH49m5z6l8GGpvC5JJa35kCeICBFMO+0PbJXxzbozo30c1S6W9Y0BkmxEiMi9ENSFFy99A
2uu2+ztCjbaT67JuL9gAAjhaWoncr/5hwgk2MjRyLoj1aJDnaE0z0CweA6Vd2w/s3B8bpjwWdYYL
IiaJSvL1oDyb1dXoGx+sUdr5HI+CD3QTz+JqrJFphXwaA1sn6ni/hDBGhTFoXUC74avPN+4Ojhc6
4gPIAphvgbIsME0CqqwW2HSnEBel2v+6+LywEAHRPd84Ytfbqb+++EL00C53uKhGd9zG/S7QtSvU
waF4DP/ZwIf9sGb7dkIfXru2Op37c7mg3I6XudxucoTrKlA1mwzN0hgkgL8/fhcvfXoHs3FOqdrm
Y8/X91Ckxpl7aGb1bNdJdlLy4cxOXttFdmvo0rugoclRJAgCS8ii6K3NnFdqm1jM9E6I2yZmZsKA
zTAB5b4k9VH/ToTOq9tX77AL/vBRiv+WZUXj5qgtwx6BMt+d69Zru6JrmXDxd++QIjOmm7yHjjfI
qe7IvOcimSgpi0j5grEbdWJvqQeYvlAvYgVgMlWFBZ5gg0KFpom4OFXkmXj8+Btf2aeFyXElSnRj
8fP9Fjv3YWkRzyiaCr9YdBOB0ubOC/7ke9eKiER1Fkiuj74L+85YjOFHkmmBWPueDI9r5fG5fYBs
6wwQBGxHOlfsnn7mMP3zFK6oEzpwrx7M/2aNWzapOH0yOFp5HcVklmtSPs3u++GByeWX+n3hnsbO
sXO70carjcoHt2z28YZkBz9AZJggQEUrKK/16zprUEXVH7skSPPfzDhyUihqHew4XJVf0TAMAXPj
iIgDA8aIyFJ6vxNr9pzx9TBelmAPAfZ9vUO13aW43hfnj36QmWOdfi2HCCCEEzvt5hu4PThaXo6C
43LFvVjHjYNCqsWtKYdnfv7ZFrn/ectSxXVFVjY0HzwTUu0WGTS4MEwQqqwW+XZ8X7GG4UC0EtuQ
5F2zNTvcaSDy9bZRWVDs8jTxRVS1cFrSd56hCfIwN/V5YiFMl5bp9zMY6/1oME0acr6iCTSOWwzo
Ev/JPtRCqQhpWsqF7WjT25L//6vxdmJ1tL9rWaeqvuOrdlbr9Lnj2NwDOwtnBgAWdYYLAcz/tQMv
G6uCdULFXK4dhJUHXqxb08lX2wtueBP3txTn4u5Rv3lPUZ3llNU4UlQa8bqmS0PgFMUwEoLTTHj5
zHieK1EWefZWBNyb2cAhTbakfr17ESwigD7XZxOjkdI+XzCznyioxZ/MRlwyJSnILSlzahmreqvq
+sBCF5ytqOVLfsc6/FsOEOPnFagF/8Zc/Vfu+rsuN75Hm65BcyMpoiKV7nKl8zU5cFUi0b0A9WzZ
AXSkA6OLXlwIAjozpiLuGOPPm7FeI+KxVuarPT+rbAlvyCt1er09G4JOa8QydYQLAufJK9H1+oe6
xIVJoeG09LWrDoLbJAID/+3Zr0Bb0zGbxn+oSfzFRUSgqVc+m88+qH7q7jjmLX3y8Ild1bFDbFGw
/4j4r5oc39/FxpL7Dgs4wAaDgrHuRvKl/KwCea4aazeNjOzq2mybFe+gZvhwtVEd0RtsIJWXqWID
/LN+v2n/gy41uz3fqnOcHX1LdwPqgAyzoDBN0ZF/v4Mk5VJ4th06d99AzfHlgRFvV/qOXdEC7Jok
eP3OJh2RnJ4pKVbl0QgEWdIYJAH8ec/9q3jujIXq2bBDwvB9KLBQ/5Batm2jb1Hro39pztSI9zxi
ZKCJ8d/dAXN3bNSNli0bBST9snRUPhglhlLZYLfXjY/D9PQMDOBpXrFAz1Bv149Vy5y1o6mxphSq
vuZYoIsRGRyFOx3umX6Y5Ze28wTN0hgkAyhqhE3qYX/29puiF+H98W+AX9cykp6ZIiLvZt7w4XVr
hOQtj43qxAIDbBmao2v8+ugOmSWmQAw0LOsMEgN92HHNsRyvc59Y+OSIYwzHEsA6pwR6CqbjLUyM
HD3mrZHR9X3uu+LapidjyzChH+w39Whkq6G0GLOgMEwBmZzvztihzjKcmhoZHSTA9bAKJsuC1OzO
TXG/0yBnn4nByPddEW1GKf0dlXVKtaSeQsKAzTIDp3qJBsIfgQoToOT6/sx9GdPT85vHb9uMAgFk
rDzja9OzkSpQP6WCuR/CiKMOYSEVVNYa9+oeq7Viha4Wb4BMZih4XbcPMm3rr5mQZ2iEFf+zKx6N
fb1Z5rlzTOx1fKzJjLnlsqEtZOVnEg5EyVwkLOsOYyLUzVyHvtPPV/cmxnfDF2oNBHJE+QcwnFXC
ibVGqdQwZbd6aOZuPALBXHZIFvWtafbRqnKB73m/vHoCWjVxzrgcSNrkEgO1HCpH5+FyVTY6JDDY
eVOfOvnNwa1zZy3+5V/xFJPihe2PuVmeRCmWdUOX6wv+N7gh39G7VEClBnqGzoAeA/60+ACGAxbt
OeO/MWJ7LutvdFl++2j8ZEv1Bm5R6wR5C0EiW3A8v7+50J1WaZIQAmta3L14HuqRcTWFBDwBfrLU
nx7dqBXWmZrRqnIDc6eNwbVaLYA/FgXYhz0ql57whf/cnxnZytCkXievG2hwz71BfPDYk6EQ0moh
2EVEOEU3ROf4wEe0goi1EtJCIWvl/qOFPtIXKeTE1R86qGA7UCZIfdTCwkZxLR6BbWhK6ptVHhWK
G3jqlHqZN6Iq+mY3Qoann3C/BxqvCEJENwDsAxgDoDGASEXXWdNsIIEsIcQGAbwC87O+BWoFImvV
Ymb35xVi196THPkIIPPvTdkVacVUAABb7SURBVFXb/cPbuunNBJP4GLsMVgsgMT4a2w4XYeaSfao
+3Vs0wOy/Xhi0gCGjGJky9gWQI4TYJ4QoB/AlgPHKDkKIxUKIc9LuagCu2WoYFvQAIoTADg+5qmv
DiNeWYNIHqz26H+46ftYlf3aoi4GSYC/uBZKPb+uLe4e1QfOkeKyUHtRfSTVEww0jgp4GQPnt8qQ
2d9wB4Be9A0Q0mYiyiSg7Pz9fr4ulYRt64Pjf6gMY++YyrMwpMO0aRwvdey2NnrHMpS2cHuhTL+3
kvZNFyExOwGOXdLREgjK/GnWJ6EYAWQBe0TsuhHhfCJElhMhKSQkfe6K/iAqjP+hwZ9Mhe2mw6z9
cg3s/2+C385ZVOosJN07Qn8X+4cabKdQX1JRoA2eY8MCIoB8GoFyOT5faVBDRSABPArhcCFGmPc4
Af/10fbCHEDEo63cq/Ytri7I8W4Wb5E23frxOtT+wrT2VajjN0MNnpIwSI4/hdQDaEVEm7EI+EcD
1yg5E1BPATACjhRDsbM0Eja15hVi6Jx9Ld5tj0rvvc2dJsnNlVR56Onl7Ui/M23Y05D0klFjB/OA
LKYlxqrqiix4ZEsTR1ByvM3QhRCWA+wDMB7ATwGwhxHYi6oe6e6vQKgHoCviWgTEc0xbcQMo6G
8shrzt9vT01729nK8Mn+XSx9/FWBef+C0Y9vdDF1Lw4RY3NAvvDx5I1TP8dEtzhzwzZPi0TrMAq4
MGcqEEPMAzNO0PaXYHunncVmWE0WlSK0fGilTrcJLv/6J/yzfj68m93fbp7JaeCwn5gu7jp1FL03
RhO825Kn2P72jr1+vGQi6NK+Prs2Tgj2MoNAt3fm9K8MwwQ1HugSYiR+sDvYQLEfOiWIAwLly9ya
QCp3sejVl48HTqv3Hv9sKACgpq3S4MD48e7Oqz0Xtwm/xf+4DF6FObPi4WJpFUWlFsIdQY3gp22T
KK9VCsi+/BLd+vBaf3BZ+M7dQRV6oPF7k3i+881Pz8e3dF6J3q0Y+X+ftRTkubQNeXIgjkpiP15S
W++vg1j5fiwk+3krQhSI8QzeZ4a/94dL2x67I88E3k62H7S6K7y7Z67HfVe+uqtV19N4AjiiCi37
cdMSxPbFPC0wZ4z4zH8OYAQu6yShzYTP+RwjhyIA3umtTv5+/tKIKfZ9fgIwpc5GZop8HW4/nr+g
WsZ4iTPBgQWfCmn/9vhuF5+22Tm3+DT2qqwVe/GWnIzd9aUWVRw+Yez7bgBOSG9uhU/bsFtdmec9
```

```
sEU4+5zIcyRz+sKCbSGmFMT9lxnfe0rFry/z3dtd1ii2HCzFzyT4MmL4IJ4pK0XHqry45VwDgp81
HcOJsKRb96QyrWLanAETAyE5NPI5JTvYUbqx+YgT+eHRosIfB1ILw/OWFCSVllcEeQkQzqG0yerV
UF2Se8M4Kx/amQ/ZqQj9uPqLqU1Ragfu/2IhbPlJHfAJA3RgboryYUj68uY/H46FKcr04ZCQbNyt
ZnVAvZqEHC7qJhJ8Xa3hx/xcbPR6PiiJc46GIxGQpFUOxxj1tjrS4ufOoa7ZGIoI2rf2yvw9zbHd
uVh99M333pGFCh8T48HMCZEE3EaVpNtxCiMOBnzQza8A1wnFS35Zez3PmnFPQ1x84jX/8sM1t3+K
ySpAm00mLRnUdJcpmTOyB2Gj+s2KCQ9j98o4XleK133ahOgyiuJSLbZFQraiiqhoDpy/Ch8u8L07
Wlko3gUJCACumDMfP9w8yfK5LFN4xRee9B5P0zmjo0nZ9P/uDIzWC8ogzoUfYqcztn6zDW4tyHLk
7QhnlIychztqRdz9tPoJuz8zH4TPnMW3uTtOvV+IhKjStQR10TTMeut4k0ZmKQS86cGSnVMd2Vqu
GqB8fgw9uzlL1+euQ1lj292FoUDfW8HWZ0ORxKX6gf+vGQR5JzQk7QZcXGs+GwYJjleItonE968z
cVuQU4PM1Bx37+/KLcf8XG1WRddoIWX9SXS1wsrhmGZqT6sRgdBd9P/UqRZKt2TqVaj5UJGwa260
ZAOCYJio1LtqGFo3q1mhMTGjy1yFtsODhwXj+iq7BHkqNCTtBf/ryLgDslbgDzdr9pzDuzWW67oh
CCLy5cA/yTp9ztMmC/vLVFwAAXr+uu8fzV1WLkHd1PFlchhs+XIMnvt/qMCkNf22JS783F+4xbQy
tn5ine01A32Uwd/o4bH56FN67qTdevLIb3ruxt+p4heLBW6xJifvGxB4AgLkPDEKT+nG4ob/dtHJ
1r3Rc1Ssdm566uFbfhQlN2qYmIi46/N6qw07QOzWtDwAoOh/4GfpTP27D9iNF2Jdf4nJs1spc/Ov
33fjLrGxHmyzocsDGFT09B6Tc/NEadJz6K054yEkSbN5e7PT7XrP/lKF+ZtIoIRYvXNENHaVc419
OvtBj/0l9W2J016b45cGL8NaknrBFEQ4ronlPlahn/p2a2X9vXZonYc0TIx1/5HVibXjt2u5sYmF
CirATdNmVqLgs8JnQZP/jap3Iwmd+2gEA+PPYWUdbldRPL2pQLzpxRY69QO2/ft+taq+uFvh09QG
c92A3DhTKMazMKXBr+riql+vD68TZUlXFH3/w39v74vp+LfHujb3x0Mj26J5uzHbeqVl9XNa9Oaq
qBeZsPuJ4iCZLprEm9eOw/P+GoX2T8ClKwTBhJ+iyS1hNbbRV1QIPz96E9Qfczyq9YdSvRvbAkD1
x9AJRPly237G9au9J9Prn7479L9ep7bitn5iHqT9sw6PfbMaslbl47OvNjnD3QPOrYjH6zUU56D1
tger4FT3t9cO/3ZAHIYRqnH2fX6j6nv6gYYJ9hpyZnIAHR7bzOX+KXO39zLkKXNi6MRY8PATpDdk
mzoQXYSfo0VEEopoL+qZDZ/DdhsO1yrgn59T+dNUBVbt25rxHys/taYb+/LydyJgyF8/M2Y5JH6w
2NHOdu+Uonp6zHV+vz8M7ATJpaFH6bCvpk9EQL1zRDa9I6wUAkPn4PHR/9jccPHlO9zO+MGOB+u2
lqZ+Khfztq03ILSjB/oIStE2th8T48IsSZJiwE3QighDAwVM1E4kz55yC6Ws5MrmQwlfZh/D0j/b
gkz3Hz2Jz3hlVvzFvLAMAVFapbeh6fLIyV7ddnt0v2HFc97jsjbFyr93jxOh3qqyq9jklgadrdGm
ehOv7tUS0zfUn9cjXm5BzwmmKypgy16frA8CMBerFVn8mwRr66h8AgBV7C/x2ToYJJGEn6DI/bHK
NEvTEHYrFyt92HFe53RlB630ya9UBPPvTdlz8+lJMfN9ehahdqrP+YHFZJQok+7LRyMGf7x/kWNw
rPF+Bo4Xn8Zf/Zuv2PXOuAu8szsH1H9g9TjIfn4e/fek5FB4AnpqzHV2eno+MKXN1BbrDP35BxpS
5LhV+zpdXOb6nHp5yf6/LPY2R/1rqdWzeWKtZhL1naJvan/PJES5teoveDBMOhK2g14Sb/rNGtf/
XT9fjie+3OlKoGuHh2Ztc2rRZ+l5SmBsGv7wY93y2AQCQW+BdIHKnj0PXtCRHJr+e//wdF764yHH
8oZHtkTt9HHKnj3O0aYsh/7DpiFcPGeWDrKhUPVMvq6xCmWTKum7mKsxamYtuT8+HEAKvzN/l1qu
lb2YjxMc4XbxeuqqbxzEAvr0lTf7U+XBb++QIPDKqQ43PoSU10dVkM8aEvOoMEwgsK+jj3lyGEa/
9gYwpc7Fsj/0VelRnddrTAdMX6X3UhdEzlmLeVu+Rqd3TnZn9TpWUOyrc9FEka1r7hHpGeEF6Em4
dkOHYj9ExWdxyYSs8MKKtY3/3tDFux/D6gt1Yve8kftt+DGWV6rcKra3/WGEpPl19AAdPnsObC/e
gwz9+dRzbcPAMnp6zHWfLKrE8pwAfrdiv+uwP9w4EAPzvjn743x39VMeu6+M9f8rxIuOBQVXVAh8
u2+cwQ029tDNSE+NNyzn+7xt6mXJehjGb8EsnBuDC1o1x/Kz7magQAtuPqDPldU9Pwrs39kabJ+a
p2vflFyOjcQJaS+3KGbCM0hXx5/sHYW9+MR780jlj/+PRoSguq4QtivDNXRfi6vfUC69dFBXUUxW
LePP/Nhgdmqrd4u4a2hqvaxb+nh2vjljTmnC2PjMK367PwzM/7cAXaw/hi7VOLxnl93lQY5K5ZIY
xM8hN/1mr2h/SPgU9WjTQvVcyvz80GK/+tgvzt9vXAKKjCGO6NYON7G8S/V9c6PHzSlbkFKjSCdw
+MMPQ53yFKw0x4UpYztCPFJ7HvvwSt6/th3VMKd/fMxC2KHIxWwx/bQk+W+P0WnlhnjoPidbNsWt
aEsb3SHPsr5gyHBnJCY7cIVqBbu0hv7S2LwCX6LTLuzd36QMAqx4f7thOjI/BrQMz3V4HsHsF/eZ
mgVXL30d7NmVM6us+Ja1MuyaJmHlTFhY9MgRTxnREzgtj8daknnj6si6OPn+Z5ZpvXI9//rxDte9
vwW2e5HzIDmqb7NdzM0wgCUtBPyC5wVVUCWRMmYu7pLzWMloPmM1Pj0KU5vVcNhkAwNQftzu231+
qzhSodHNspvjDnzahK24dkIG0BnVU/evFqV969unYz2Ojo5Bcz32E4donRuDj2/pgz/Nj8Oaknrp
9miXVcXk4ZbVyzQIoZz48rfDycXfOR0e1x7wHLlI9sPTo0cL1Ou5onVIPdw1xLl42TIh1iOaCnSd
UphR3yG6gAHBlL89j84XJg1sDAH68dyBm3tTbS2+GCV3CUtBlSiUb8a/bj6GguAyXvbUcuQUlLn7
iepVHtMKrJGPKXBTruPbNf2iwY/vG/q3wzOVdXPoQES69oJnHcW99ZhRWTnH1rpBJrR+PYR1Sde3
pnvj6rgsxvGOqqm3a3J3YfOiMakHT3az/vuHt0Ll5faQ1qIM4yawjR3xunHoxtj4zCsv+PgxNk2r
n+708x+kWOG3uTjykWHDecPA0ftx02LGvTZP70Mj2tbq2HrcOzETu9HHo3qIBEjz8Lhgm1AnrX++
BAudMPEuKWJR9iQF7Miw3abN1k3vdNjDD4bnS9en5mPuAM6e2UXsvALx9fS/8vMXua31jf9cFQrO
S/hARPrq1j8PFsuNU+yLneEXZtXeuty/47X9xLAqKy5GSGIeteYVo2VgdFblLsfD62rXOpGL+CLi
Z+8AgjHtzuWP/x01HMPXSzkiuF4cr/70SgP2hQ0Q4qQi4Gt+jOZpr3ogYhnESljP0we1TAABr9p/
02O+Knum4urd+QizZzU4p7PcNa6vqoxSdmjLjOnuWvpv6Z/h8Dl+Jj7Gp3AiVXCx5+hARUqRiDN3
SkwJaP7GzlPBKSda0BSqXSjly9ou19rYreqbhjYk9TfNsYRgrEJYz9Kt7p2Pp7nyPhRSaeTELNEq
IxYzremBA28aIIkLDurFwpxWPXVJzf+cJPdMwoaf/7b21JRTKoxER9r4w1sXj6Invtzq2e09bgNz
p4xyRof8Y1ymgY2SYcCT4f90+YGSOtupx9zZqmQk905CaGI/kenGwRRGICDNv6o2pl3ZW9evS3HV
GGQ68eKU6wEe7gBtMjMy0v84+hATpDcpKBUIYxizCcoZ+iZvKM/Xjo3H30Lbo2bKB7vGanFvpKte
qsXvXw1BmUt+WGNExFVVCoFqElqADdtPZ0t35bo8/9s0WAECblPC8/wwTaMJyhu7ObNAsqQ7uHtr
GL7UA5ejNT27rg0wPvuShTmr9eDRLqhNyYg7Yc5lrizmnNaiDhy9We7Ls5dwqDGMIQ4JORKOJaBc
R5RDRFJ3jcUT0lXR8DRFl+HugRth1/Kz3TgaZMqYjVkwZjqEdUr13ZnxGWcz5+n4tsWLKcHRv4fs
bFsNEMl4FnYhsAN4BMAZAZwCTiKizptsdAE4LIdoCeB3AS/4eqJZHR7n6I/vz1Tw+xhaSs1orsvm
```

```
pUbikSxNMk1IcaIOuONiHYYxhZIbeF0COEGKfEKIcwJcAxmv6jAcwS9r+BsAIMjkhRp1Yu/n//uF
t8fmd9uRQ/9UkiWLCg6S6MZh5U5YjmldZ9i13+ji3ayYMw6gxsiiaBkBZEy0PgFY5HX2EEJVEVAi
gMQBVpQAimgxgMgC0bOk9I58nbujXEseLSnHXkDZIiIuuUeAPE9rE2KL435NhfCCgi6JCiPeFEFl
CiKyUlJRanSs+xoYnxnbiUG2GYRgJI4J+GIAyvV661Kbbh4iiASQB8BzGyTAMw/gVI4K+DkA7Iso
kolgAEwHM0fSZA+AWaftqAIuEr4U7GYZhGJ/waq+QbOL3AZgPwAbgIyHEdiJ6DkC2EGIOgP8A+JS
IcgCcgl30GYZhmABiyAAthJgHYJ6m7SnFdimAa/w7NIZhGKYmhGWkKMMwDOMKCzrDMIxFYEFnGIa
xCCzoDMMwFoGC5V1IRPkADnjtqE8yNFGojC58n4zB98k4fK+MYeZ9aiWE0I3MDJqg1wYiyhZCZAV
7HKEO3ydj8H0yDt8rYwTrPrHJhWEYxiKwoDMMw1iEcBX094M9gDCB75Mx+D4Zh++VMYJyn8LShs4
wDMO4Eq4zdIZhGEYDCzrDMIxFCDtB91aw2uoQUQsiWkxEO4hoOxE9KLU3IqLfiWiP9P+GUjsR0Zv
S/dpCRL0U57pF6r+HiG5xd81whYhsRLSRiH6W9jlcat9FRJcE55uYCxE
1IKJviOhPItpJRBfy78kVInpI+rvbRkRfEFGMFX9PYSHojsLVIwHsBBFCVEr7yu+rKnIOQC5ybvV7BNhnifkAPpbMUx8SUQL496RCCHEIdu
6SduJAHbDXrz7ZQBTpPYpAF6StscC+AUAAegPYI3U3gjAPun/DaXthsH+fn6+Vw8D+BzAz9L+bAA
Tpe33ANwtbd8D4D1peyKAr6TtztJvLA5ApvTbswX7e5lwn2YB+Iu0HQugAf+eXO5RGoD9AOoofku
3htpvKtxm6EYKVlsaIcRRIcQGafssgJ2w/9iUhbpnAZggbY8H8F9hZzWABkTUDMAlAH4XQpwSQpw
G8DuA0QH8KqZCROkAxgH4UNonAMMAfCsdor2P8v39FsBw6fjxAL4UQpwTQuwHkAP736FlIKIGAAYD
iXAhxBvx70iMaQB2pKltdAEcRYr+pcBN0vYLVaUEaS9CRXuN6AlgDoIkQ4qh06BiAJtK2u3tm9Xs
5A8DfAVRL+40BnBFCVEr7yu+rKnIOQC5ybvV7BNhnifkAPpbMUx8SUQL496RCCHEYwKsADsIu5IU
A1iPEflPhJuiMBBHVA/AtgL8JIYqUx4T93S5i/VGJ6FIAJ4QQ64M9ljAgGkAvAO8KIXoCKIHdxOI
g0n9PACCtIYyH/QHYHEACQvANJNwE3UjBastDRDGwi/lnQojvpObj0qsvpP+fkNrd3TMr38uBAC4
nolzYzXLDAbwBu3lArtKl/L7uipxb+R7J5AHIE0Kskfa/gV3g+fekZiSA/UKIfCFEBYDvDvYP+dhdR
vKtwE3UjBaksj2eH+A2CnEOJfikPKQt23APhR0X6z5J3QH0Ch9Co9H8AoImozT5GSW1hjxDicSF
EuhAiA/bfyCIhxA0AFsNexBzAe7nAJgoeSxkAmgHYG2AvkZAEEIcA3CIiDpITSMA7AD/nrQcBNCfiOKllF7bX0WpO2LL9J4Efw2Y1Bw3+mzSfeoiTbpU4BhNy5i2gQFdmqGCx0E1BFEG3a
cBNCfiOpKf4PyfQqt31SwV499WG0eC7tnx14ATwZ7PEH4/oNgf/3dAmCT9N9Y2O1zCwHsAbAAQCO
pPwF4R7pfWwFkKc51O+yLMjkAbgv2dzPpfg2F08ultfTHkwPgawBxUnu8tJ8jHW+t+PyT0r3bBWB
MsL+PSfeoB4Bs6Tf1A+xeKvx7cr1PzwL4E8A2AJ/C7qkSUr8pDv1nGIaxCOFmcmEYhmHcwILOMAx
jEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWgQWdYRjGIvw/IlbURK54GdgAAAAASUVORK5CYII
=",
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
            ]
          },

 "metadata": {
            "needs_background": "light"
          },
          "output_type": "display_data"
        }
      ],
      "source": [
        "plt.plot(data_oil)"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 11,
      "metadata": {
        "id": "S9THgFu3KaKH"
      },
      "outputs": [],
      "source": [
        "training_size=int(len(data_oil)*0.65)\n",
        "test_size=len(data_oil)-training_size\n",

"train_data,test_data=data_oil[0:training_size,:],data_oil[training_size:len
(data_oil),:1]"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 12,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "mbvhl2HCKmSG",
        "outputId": "5142d0f7-21d1-46bb-e69e-bd71c69ef729"
      },
      "outputs": [
        {
          "data": {
            "text/plain": [
              "(5340, 2876)"
            ]
          },
          "execution_count": 12,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "training_size,test_size"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 13,
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "M14MiYVzKrJg",
        "outputId": "b23a9806-135a-424d-f65b-6029116ad975"

},
      "outputs": [
        {
          "data": {
            "text/plain": [
              "(5340, 1)"
            ]
          },
          "execution_count": 13,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "train_data.shape"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 14,
      "metadata": {
        "id": "vrH4kcM5K9S8"
      },
      "outputs": [],
      "source": [
        "def create_dataset(dataset,time_step=1):\n",
        "  dataX,dataY=[],[]\n",
        "  for i in range(len(dataset)-time_step-1):\n",
        "    a=dataset[i:(i+time_step),0]\n",
          "    dataX.append(a)\n",
          "    dataY.append(dataset[i+time_step,0])\n",
          "  return np.array(dataX),np.array(dataY)"
        ]
      },
      {
        "cell_type": "code",
        "execution_count": 15,
        "metadata": {
          "id": "dv2OIjslLA3v"
        },
        "outputs": [],
        "source": [
          "time_step=10\n",
          "x_train,y_train=create_dataset(train_data,time_step)\n",
          "x_test,y_test=create_dataset(test_data,time_step)"
        ]
      },
      {
        "cell_type": "code",
        "execution_count": 16,
        "metadata": {
          "colab": {
            "base_uri": "https://localhost:8080/"
          },
          "id": "ApfZGkVhLD2p",
          "outputId": "979643ec-00c5-4f15-93f0-8529e83335a0"
        },
        "outputs": [
          {
            "name": "stdout",

  "output_type": "stream",
            "text": [
              "(5329, 10)\n",
              "(5329,)\n"
            ]
          },
          {
            "data": {
              "text/plain": [
                "(None, None)"
              ]
            },
            "execution_count": 16,
            "metadata": {},
            "output_type": "execute_result"
          }
        ],
        "source": [
          "print(x_train.shape),print(y_train.shape)"
        ]
      },
      {
        "cell_type": "code",
        "execution_count": 17,
        "metadata": {
          "colab": {
            "base_uri": "https://localhost:8080/"
          },
          "id": "iLQiXr21LE35",
        "outputId": "3dca6c25-95ed-4b85-bb1a-d2e218179c5d"
      },
      "outputs": [
        {
          "name": "stdout",
          "output_type": "stream",
          "text": [
            "(2865, 10)\n",
            "(2865,)\n"
          ]
        },
        {
          "data": {
            "text/plain": [
              "(None, None)"
            ]
          },
          "execution_count": 17,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "print(x_test.shape),print(y_test.shape)"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 18,
      "metadata": {
        "colab": {

"base_uri": "https://localhost:8080/"
        },
        "id": "w95tBQ4gLJWj",
        "outputId": "f1567ef1-fb46-4909-857b-c3c34c6f1dd1"
      },
      "outputs": [
        {
          "data": {
            "text/plain": [
              "array([[0.11335703, 0.11661484, 0.12053902, ..., 0.10980305,
0.1089886 ,\n",
              "        0.11054346],\n",
              "       [0.11661484, 0.12053902, 0.11550422, ..., 0.1089886 ,
0.11054346,\n",
              "        0.10165852],\n",
              "       [0.12053902, 0.11550422, 0.1156523 , ..., 0.11054346,
0.10165852,\n",
              "        0.09906708],\n",
              "       ...,\n",
              "       [0.36731823, 0.35176958, 0.36080261, ..., 0.36391234,
0.37042796,\n",
              "        0.37042796],\n",
              "       [0.35176958, 0.36080261, 0.35354657, ..., 0.37042796,
0.37042796,\n",
              "        0.37879461],\n",
              "       [0.36080261, 0.35354657, 0.35295424, ..., 0.37042796,
0.37879461,\n",
              "        0.37916482]])"
            ]
          },
          "execution_count": 18,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "x_train"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 19,
      "metadata": {
        "id": "59VQDX1LLL3D"
      },
      "outputs": [],
      "source": [
        "x_train=x_train.reshape(x_train.shape[0],x_train.shape[1],1)\n",
        "x_test=x_test.reshape(x_test.shape[0],x_test.shape[1],1)"
      ]
    }
  ],
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "display_name": "Python 3",
      "name": "python3"
    },

"language_info": {
      "name": "python"
    }
  },
  "nbformat": 4,
  "nbformat_minor": 0
}
```

SPRINT 3:

ADDING LSTM:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "source": [
        "import pandas as pd\n",
        "import numpy as np\n",
        "import matplotlib.pyplot as plt"
      ],
      "metadata": {
        "id": "GiRQ27X4JRcH"
      },
      "execution_count": 1,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data=pd.read_excel(\"/content/Crude Oil Prices Daily.xlsx\")"
      ],
      "metadata": {
        "id": "dbaHUfMiJW8I"
      },
      "execution_count": 2,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().any()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "RAUyZB0-Jp0t",
        "outputId": "9f1ec869-c35f-4764-8989-0d5d82646e18"
      },
      "execution_count": 3,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             False\n",
              "Closing Value     True\n",
              "dtype: bool"
            ]
          },
          "metadata": {},
          "execution_count": 3
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
 "id": "yorF39lCJsyV",
        "outputId": "4001984b-54cc-4b54-f73b-926ff03ac00f"
      },
      "execution_count": 4,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    7\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 4
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "data.dropna(axis=0,inplace=True)"
      ],
      "metadata": {
        "id": "g3FWuWVPJwms"
      },
      "execution_count": 5,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data.isnull().sum()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "vIHbyOs4JzIf",
        "outputId": "47f79e73-6c7b-4d04-a0bd-09ae39d18bcf"
      },
      "execution_count": 6,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "Date             0\n",
              "Closing Value    0\n",
              "dtype: int64"
            ]
          },
          "metadata": {},
          "execution_count": 6
        }
      ]
    },
    {
      "cell_type": "code",
"source": [
 "data_oil=data.reset_index()['Closing Value']\n",
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "IoxUNwrvJ2b6",
        "outputId": "2011717f-466c-4af2-973c-3980b6229a4d"
      },
      "execution_count": 7,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "0        25.56\n",
              "1        26.00\n",
              "2        26.53\n",
              "3        25.85\n",
              "4        25.87\n",
              "         ...  \n",
              "8211     73.89\n",
              "8212     74.19\n",
              "8213     73.05\n",
              "8214     73.78\n",
              "8215     73.93\n",
              "Name: Closing Value, Length: 8216, dtype: float64"
            ]
          },
          "metadata": {},
          "execution_count": 7
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "from sklearn.preprocessing import MinMaxScaler\n",
        "scaler=MinMaxScaler(feature_range=(0,1))\n",
        "data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))"
      ],
      "metadata": {
        "id": "5m-DUFI9J_WN"
      },
      "execution_count": 8,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "data_oil"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "fyMwOo1jKLqL",
 "outputId": "6e2bdf01-77aa-4e93-f117-6ab963e21c0a"
      },
      "execution_count": 9,

"outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703],\n",
              "       [0.11661484],\n",
              "       [0.12053902],\n",
              "       ...,\n",
              "       [0.46497853],\n",
              "       [0.47038353],\n",
              "       [0.47149415]])"
            ]
          },
          "metadata": {},
          "execution_count": 9
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "plt.plot(data_oil)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 282
        },
        "id": "GdNJartuKUfk",
        "outputId": "0640d4b9-abe5-4e76-a910-871af84bc19c"
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "[<matplotlib.lines.Line2D at 0x7f5a0c7b5850>]"
            ]
          },
          "metadata": {},
          "execution_count": 10
        },
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 432x288 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAXQAAAD4CAYAAAD8Zh1EAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO2dd3gU1frHv282DUIIJQklARJ6lRa
KgHSRooId7OXKtXtt94d6ZGdsV1GvyvXasKOgKEV6C70JBAgQakJJSCD9/P7Ymd2Z2dndyWZ
ny+z7eR4fZ86cnTk7bL5z5j1vISEEGIZhmPAnKtgDYBiGYfwDCzrDMIxFYEFnGIaxCCzoDMMwFoE
```

```
FnWEYxiJEB+vCycnJIiMjI1iXZxiGCUvWr19fIIRI0TsWNEHPyMhAdnZ2sC7PMAwTlhDRAXfH2OT
CMAxjEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWwaugE9FHRHSCiLa5OU5E9CYR5RDRFiLq5f9
hMgzDMN4wMkP/BMBoD8fHAGgn/TcZwLu1HxbDMAxTU7wKuhBiKYBTHrqMB/BfYWc1gAZE1MxfA2Q
Yxhw+XLYPeafPBXsYjB/xhw09DcAhxX6e1OYCEU0momwiys7Pz/fDpRmG8YV9+cWYNncnBr20ONh
DYfxIQBdFhRDvCyGyhBBZKSm6kasMwwSAc+VVwR4CYwL+EPTDAFoo9tOlNoZhQpRqrlRmSfwh6HM
A3Cx5u/QHUCiEOOqH8zIMYxJV1SzoVsRrci4i+gLAUADJRJQH4GkAMQAghHgPwDwAYwHkADgH4Da
zBsswjH/gGbo18SroQohJXo4LAPf6bUQMw5hOVXWwR8CYAUeKMkwEwiYXa8KCzjARCAu6NWFBZ5g
IpIpt6JaEBZ1hIpCVOQXBHgJjAizoDBOBzFy6L9hDYEyABZ1hGMYisKAzDMNYBBZ0hmEYi8CCzjA
MYxFY0BmGYSwCCzrDMIxFYEFnmAhDKIKKWicnBHEkjL9hQWeYCGP+9uOObVsUBXEkjL9hQWeYCON
saYVj+3wFVy6yEizoDBNhxNicf/Z5p88HcSSMv2FBZ5gIg9jKYllY0BmGYSwCCzrDRBjEU3TLwoL
OMBEGy7l1YUFnmAgjSjFDT2tQJ4gjYfwNCzrDRBhKi8vhM+zlYiVY0BkmwjlRVBrsITB+ggWdYSK
MqT9sU+0Xnq9w05MJN1jQGSbCOFlSrtrngtHWgQWdYSKcqmoWdKvAgs4wEQ5P0K0DCzrDRBh3XpS
p2t9fUILyyuogjYbxJyzoDBNhaC0s93+xEY9/tzU4g2H8Cgs6w0QYss28Z8sGjrblOfnBGg7jR1j
QGSbCqBYCDerG4L5hbR1txAkBLAELOsNEGFXVAjYirlZkQQwJOhGNJqJdRJRDRFN0jrckosVEtJG
IthDRWP8PlWEYf/DZmoM4WVKOTYfOBHsojJ/xKuhEZAPwDoAxADoDmERENTXd/gFgthCiJ4CJAP7
t74EyDONf9pwoDvYQGD9jZIbeF0COEGKfEKIcwJcAxmv6CAD1pe0kAEf8N0SGYcxgQJvGjm1OkW4
NjAh6GoBDiv08qU3JMwBuJKI8APMA3K93IiKaTETZRJSdn8+r6gwTaIQiqhPRqMgjoQxA38tik4
C8IkQIh3AWACfEpHLuYYUQ7wshMw
GACHEKgDxAJL9MUCGYfxHpUrQWcathhFBXwegHRFlElEs7IueczR9DgIYCsOEGOV
VzhD/M5w213J4FXQhRCWA+wDMB7ATdm+W7UT0nBQZwBcAbhWCU/4wTKhRocjZopy
fc+FoaxBtpJMQYh7si53KtqcU2zsADPTv0BiG8TdKk0uMjeMKrQb/izJMBKHMqhioSflnaw7g09U
HAnOxCMfQDJ1hGGtQobChN0uqE5BrPvm9veTdTf1bBeR6kQzP0BkmgpBNLqmJcWiUBNLqmJcWiUEIvkenGmXm/
JbvaNCCQs6AwTQcgml2kTugIA+mQ0BAAcPnPelOutzCkw5byMPizoDBNBfLshDwCwN78EQHj7ov+
46TC+XZ8X7GGEFGxDZ5gI4uMVuQCAg6ckQQ/jFLoPfrkJAHBV7/QgjyR04Bk6w0QI1QqXxRv62Rc
ozdZzb8EopRVVuOCZ+fh9x3FzBxIhsKAzTISgjBJNiLO/nNtMNrnsOX7W4/G80+dQVFqJO/+bXaP
zctyiPizoDBMhKF0Wo6WpudkRoot3GfdyqdJWr3aDEAKZj8/z2GfjwdO49K1lKr/7SIAFnWEiBKW
4ybbzGFuwbejO6x8+bczTptKA8F/x75XYdrgIu728IVgNFnSGiRDKdWboZi+Ktmpc13Dv
vuLRSta9889Cy6M8Thq9vBVjQGSZCUM7Q5QLRcdHmSoD3QtTO2fYnK3Nx8OQ5r+d85OvNqv3Siiq
3fQ8YOJ+VYEFnmAhBKejyDD3axBn6scJS7JP83d2hNZ85Lgr79SCEypszFzqNFjmP
fbsjDuXL1jN7KsKAzTISgNLkQzF8U/fOYU1hbNNLPG1NZpRZnIzb9tAbxqv1zZXZB/2XrMQDAmDe
WqY5H0sIoCzrDRAgqLxJJN810clE+LNx5GWo9W4yk9B3UVl2+UjbBlFXqm17KWNAZhrEaSu1MqhM
DwFw/dOWZSyuqVYFNALB2/ymMf2eqaqUXdIK8CCzjARQnbuKZc
2c2fozu2C4jI89s0Wx/7rv+/GtTNXuXzGiEtiRVXNBJoFnWEYy/HZmoMubWYm5yKozy0nBgOANxb
u0f2MEfGV3RS/nNzf0DiMPCSsAgs6w0QI+wtcPU7MXBT15dRGBL1SEvQ2KfVU7Zd0aeLzOa0CCzr
DRDDBjhOVuSA9CYDRGbq9T4yNMLJTKro0r4/qauGwpWthQWcYxlLM3XLUsZ3VqqFj20yTi56p47M
1+rVF/zq4jdvPaJFNLjG2KMRF21BWWY3WT8xDQXG5qt9dQ+RzspcLwzAW4bqZq3Dv5xsc++/c0Mu
xbcBL0Ge0Xi2As76olnrx9uyPVZL4/rjpMNbpLOICTtGPthFAQM6JYpc+Dwxv63hw8QydYRjLsGa
/Whib1HcG5phpQ6/JYmS9OJv9M5I55cEvN+Ga9+xeMNXVAkcUJfJemb8LABATFaV681Dy8KgOsEl
BSiVl7lMDWA0WdIaJYMx0W6xyY+rQm7nXjZVn6K7H/rN8PwZMX4Qlu/N1M0Zq+fi2PgCAXGkReNI
Hq2s28DCGS9AxTAShzX6odS30J+78xat0gofOldtn0Ut256NrWpLq2NI99pzqt3y0FiM6pnq97rA
O9j6nz1XUaLxWgGfoDGNhtMFE2shQM7Pnugu515uF1421m1xOlZTjopcXq44dVphbFmrS4daPdz8
nvWNQpuGxWgUWdIaxMNpsh9p0tmZ6ucgml4WPDFG1a8P7Z1zXA82T7Mm7+rVu7HIeTxkbX7+uh2p
/6qWdHdtyegPAnmYgEmBBZxgLoxVPraCbaUOXJ+IJsdHo0CTR0V5VLVT7E3qmIVbKy17TzIhaO3r
7JvV0++mlGbAiLOgMY2FOnVP7Zmtn5GbO0OWHSRQBuxSl4KqqhWofgEPQyyqr0CYlwfA1tNWLZNN
NpMKCzjAW5uVfd6n2dyiKPwDm2tDlGbrWNVKviIUtihBjI5RXVmOvZGK5qF2y12toy8/J3jKRiiF
BJ6LRRLSLiHKIaIqbPtcS0Q4i2k5En/t3mAzDmIGZfuiye6L2oaH0C5+hsIHH2qKQf7bMsb/x4Bm
cLdX3VGlSPw4AMKCNWvR5hu4FIrIBeAfAGACdAUwios6aPu0APA5goBCiC4C/mTBWhmH8jFkz9NK
KKuSdttfztEURruyZ5jimfIZMULTHxdjw9XpnRsbiskq8Ol/9hiEzsU9LAEDTJHX1Iu0MPdGDF4w
VMTJD7wsgRwixTwhRDuBLAOM1fe4E8I4Q4jQACCEiq9Q2w4QpZs3QO0791VFwgoiQqohOlRc+J/V
tofpMrE4eglmr9HO/DFKYYx67pINjWztDv0LxwIgEjAh6GoBDiv08qU1JewDtiWgFEa0motF6JyKi
iyUSUTUTZ+fn5vo2YYRi/cWUv+59yamKcadeIIkDA6W0jC3rfzEaqfkYKRMv0yXB+9t5hbfHaNd3
RoUmii6A/Oa4TAHO9eUIJfy2KRgNoB2AogEkAPiCiBtpOQoj3hRJQU7WGGYQJM3dhojOr
cBI0SYk27RhQRLruguWP/5o/WAgBsUb7Jz60DMlzaruqdjvkPDXZ544iLtgu8gcp2lsDIHT0MQPl
ulC61KckDMEcIUSGE2A9gN+wCzzBMEGlYN8ZrnygiQ7U8jaJMpAXYbehd05LwxkR1EFB0DQz4Q9o
7J4BDOvBk0B1GBH0dgHZElElEsQAmApij6fMAkn0lUqAdG257n1
1EI88adYGNWn3PdGyUV0MbGuPIq1pYethHVIQFx0ZHtpev6UQohLAfQDmA9gJYLYQYjsRPUdEl0v
d5gM4SUQ7ACwG8JgQ4qRZg2YYxn+Qn2foWiA+dOqfa97SAmZIYh/NSAq+auiamJMa
ZalIKJQz59Agh5gGYp2l7SrEtADws/ccwTBgRRQT4Sc8Listc2mRB19q3d2uiRT1NvG++sBXmbz8
GwGkXN4otKspt5kerERnvIQwT4ciz39FdmrociyLXnC9G2Ha4ENsOF6razuiYeNxZVlxMMApFf2C
EegkuMT7GYRaq6VpqjI3c5ma3GizoDBMBvH5dD+x/cSzevbGXyzH7omjNz3npW8tx6VvL8XW206t
```

Zr36nO1937UxbmVfm4Yvbq47ZogjPX9EVWa0aom2qfgIud9iiyFEJyeqwoDOMhemaVt+xmEhEuuJ
KPs7QZV77bbdjuybZEm8fqM5XXuFlFt2rZUN8c/eAGptcYmxRXs9tFVjQGcbCVFUDdWI8L5VFEfn
NT9tdUQs94mLU8vPdBq03tH+wRVHEFIpmQWcYi1JQXIadR4sQY/Ps5uerDV1GGeFZWmG8ILM358M
nxnYEAPz5T93Ac8PERBEqqgREBEQXsaAzjEW57/MNAICjhZ5D6msaWLToz+OqrIhKbvt4ndvPaa0
9yfU8pxuYPLgNcqePQ3xM7TIolkkpdkvKjT9swpXISkXGMBHE6n32smubDp3x2I9qsChaWVWN2z/
Jdmlfl3sKRwtLUak40aS+LbF6n/twFG21IbOYucQe4zhrZS7uHdY2INcMFizoDBPhRBEMmyMqdZS
/Xlw0rnnPtcTbi1d2q9E4fr5/EC59a3mNPlMTXpm/y/KCziYXholwDpw8h4Licq/91h84jcOaPC0
AcImOb7svaHOb+wvZyycS4Bk6w0Q4y3MKANhrfXrKr3LVuyt12+XqQbXFrPqmY7o2w4qcyMhEwjN
0hmEA+O7pomd/H9vNddbuTa7NMqlf1t2eundCj+ZeeoY/LOgMY1G0FYG84aug69nf/31Db5c2b94
qZlVPSqoTg3px0V69aqwACzrDWJCTxWXIO223d+sVhNDDUzClu8CcurE2F0+Wl6+6QLfvRe2S8dS
lnXWPAebN0AF7fdIPl+837wIhAgs6w1iQ3tMWYNmeAnRsmohnLu9i6DOeZugVVfpqbyPC5jx1gq4
GbopqEBFuH5SpeyySqEl6hJrCgs4wYURFVTU+WbEfW/I8+5bL1CQox5PBRc9dsUvz+jhbVunSHuN
jMYnEePuDYLyFbd1fZx9C+3/8goMnz3nv7APs5cIwYcST32/F7Ow8AEDu9HFe+9cxIOh/G9kOMxb
s8TxD15lVupu1x/hYKxQw9p1qQ3llNWKDWL1o3tajAICc/LNo2biu38/PM3SGCSN+2XrMax/lK31
8jPc/cXlm7GlNVC9bobush95yxwQTbVGNQGN2NhkWdIYJI/RMHFqUVYPqGCjXJsuvp2hRvYo/7ma
6vppcAoE/S+2FImxyYZgwYMnufNzy0VpDfZUeKfEGcofL3oKetK5Sx7wSa9MX7mgv7irZ/xgZtII
TZgUvGUW+x2Z9/9B9lDIM4+Cxrzcb7ltW6cwqGG9ghv7VOnvFoewDp3WP5xaUYMaCPS7t7mbo5CW
EKLlenGlh/t7wFAkbCJbszgcATP50vSnnZ0FnmDDghJt0tXoUlzkF/YhO7hUtfx6z25VzThSr2oU
QOF5Uips+WoPvN7oWn3DnfidMtxTXnHdvsJfeC5VSdN7eYnyFBZ1hwhR3Xibfb8hzbB8vMv4g0Ar
xp6sPoN8LC3HolP5Dwd0su11qouFrBgrZfVOv5mmgKCqtQGZyAgDgu3sGmHINFnSGCVPW5Z7Sbe/
UrL5ju44BLxcZrQ191V7PCa2S6ugHEBlZiA000ZLnjZ4/faB4ZPZm7C8oAWBeZkkWdIYJU9wtSp5
XlIFrUDfW5/NvPOg5eCkl0ZkbJatVQwDAaD+l0vU30ZJvfDBNLr/vOO7YNmtxlgWdYUKI6mqB8wZ
LpZ0q0c9hXlzqdG2syWx51spc9Xk8uEj+c0JXTB7c2rE/86be+PzOfpgxsYfh6wUS5ww9eCYXJWY
tzbKgM0wI8dzPO9DpqV8NVamftSpXt7243CnENgMzQTmfuXbh9ZqsdLefual/K8Qo3hCICAPaJNe
6/qdZyIuQwZyhK01UPENnmAjgm/X2Bc3C8xVe+17VS19w5RqaAHBhG+/VeqZfqZ8dcW9+iUvblb3
S0EzH/hu6saF2HCaXINrQlf+m0SZF03JgEcOEELKftOzBMur1JS75WG4fmImPVuz3Ootf/n/DkNa
gjtdrJsbry8BSyWdaybQJXVE31rV/kON1vOIwubjxDDIbbem+GDfrH7WFZ+gME0LIgl4qLWzuPl6
sSk/78MXtcdcQu+26zEsa1vSGdQ0VjYiqgU+0u8jTIE58DRETZC+XQ6fU2RVZ0BkmApAXOoe88of
u8QdGtHNEf5ZWuC6eylGi7lwK9aiJPVcr/g3ryom9QlvRbQ6TS3Bm6O/+sVe1b1bEKptcGCZEqXY
zm4yTQu61M/Q3F+7Bv37fDcCYDV6mNtIivwGEtpw7F0X1koyZybnySsxYsAcr9xYE5HqGZuhENJq
IdhFRDhFN8dDvKiISRJTlvyEyTGSgneW6M6nI/ufa47KY1/i6Pn3KjjNTYy1OEgBkG7oR7yF/MvW
H7Xh/6b6APUi8CjoR2QC8A2AMgM4AJhGRS2FAIkoE8CCANf4eJMNEAlr7rtak8vLVdm8UIkJcdBT
KdEwuvuDOXDK4fYpq/4d7B7r0CfXFUBlnYFFgTS47jhYF9HpGZuh9AeQIIfYJIcoBfAlgvE6/fwJ
4CUCpH8fHMBGDNtnVxkPq7IepisjMsspqjwm7Hh3V3vB13c0dtSafHi0auPR5/boe6N2qIRol+B6
RGgiCtSi6U0fQL+tuXok9Izb0NACHFPt5APopOxBRLwAthBBziegxdycioskAJgNAy5Ytaz5ahrE
wWhPK7Z9kq/a16Wq/33gYTZPisevYWXx0ax/VsT4ZjQxf1525RLmA6M798aJ2KbioXYrusVDCFgK
BRQCw7dlLUC/OvKXLWp+ZiKIA/AvArd76CiHeB/A+AGRlZYW41Y1hAou3avDROrU6Ze8JrZ9zeiP
j9SrdmVyUdt9pE7oaPl8oIrsJ6pXSC+w4zLVRGTG5HAbQQrGfLrXJJALoCuAPIsoF0B/AHF4YZZi
aoSxMUVNW5qi9KIwEFMm4s0Iobfg9W7qaW8IJeYZeFeAZupy0TKY2BbSNYOTs6wC0I6JMIooFMBH
AHPmgEKJQCJEshMgQQmQAWA3gciFEtv7pGIbRY8HOEx6Pe6qH+dg3W3y+buuUBN12paCbFQgTKBx
ui4qn15p9JzHopUVe34xqQ4km0VpNgrh8weu/khCiEsB9AOYD2AlgthBiOxE9SXmzo6hokg/vn
zDo/HjbrcfV/D4gnJ9eJ025U2fXfl5sIFIkJ0FKFKMrlsO1yI695fjbzT53HZW8tNu+6e42dNO7c
ehmzoQoh5AOZp2p5y03do7YfFMJHLwxe31/UpN+KhcedFmejZsqHXfkYorXAKulkl0wKJLYoci6K
v/rbL0b7LRNFV/ptdbqJ3i0x4P3YZxiIsUSTCuvSCZrp9+mV691zx1S1vZKcm6KyodARA5eduJCd
MqBNji3Is9BpJK1xbtD7v06/qZvo1WdAZJgS4c5ZzyalefLSqjJyMkVzj7tIFeCM6ilRuiiVllTh
bVolmSfH49m5z6l8GGpvC5JJa35kCeICBFMO+0PbJXxzbozo30c1S6W9Y0BkmxEiMi9ENSFFy99A
2uu2+ztCjbaT67JuL9gAAjhaWoncr/5hwgk2MjRyLoj1aJDnaE0z0CweA6Vd2w/s3B8bpjwWdYL
IiaJSvL1oDyb1dXoGx+sUdr5HI+CD3QTz+JqrJFphXwaA1sn6ni/hDBGhTFoXUC74avPN+4Ojhc6
4gPIAphvgbIsME0CqqwW2HSnEBel2v+6+LywEAHRPd84Ytfbqb+++EL00C53uKhGd9zG/S7QtSvU
waF4DP/ZwIf9sGb7dkIfXru2Op37c7mg3I6XudxucoTrKlA1mwzN0hgkgL8/fhcvfXoHs3FOqdrm
Y8/X91Ckxpl7aGb1bNdJdlLy4cxOXttFdmvo0rugoclRJAgCS8ii6K3NnFdqm1jM9E6I2yZmZsKA
zTAB5b4k9VH/ToTOq9tX77AL/vBRiv+WZUXj5qgtwx6BMt+d69Zru6JrmXDxd++QIjOmm7yHjjfI
qe7IvOcimSgpi0j5grEbdWJvqQeYvlAvYgVgMlWFBZ5gg0KFpom4OFXkmXj8+Btf2aeFyXElSnRj
8fP9Fjv3YWkRzyiaCr9YdBOB0ubOC/7ke9eKiER1Fkiuj74L+85YjOFHkmmBWPueDI9r5fG5fYBs
6wwQBGxHOlfsnn7mMP3zFK6oEzpwrx7M/2aNWzapOH0yOFp5HcVklmtSPs3u++GByeWX+n3hnsbO
sXO70carjcoHt2z28YZkBz9AZJggQEUrKK/16zprUEXVH7skSPPfzDhyUihqHew4XJVf0TAMAXPj
iIgDA8aIyFJ6vxNr9pzx9TBelmAPAfZ9vUO13aW43hfnj36QmWOdfi2HCCCEEzvt5hu4PCLFrn/ectSxWjYNHzwTUu0WGEwQqqwW+XZ8X7GG4cUV9FZv0TAuzq
Ev/JPtRCqQhpWsqF7WjT25L//6vxdmJ1tL9rWaeqvuOrdlbr9Lnj2NwDOwtnBgAWdYYLAcz/tQtQMv

```
G6uCdULFXK4dhJUHXqxb08lX2wtueBP3txTn4u5Rv3lPUZ3llNU4UlQa8bqmS0PgFMUwEoLTTHj5
zHieK1EWefZWBNyb2cAhTbakfr17ESwigD7XZxOjkdI+XzCznyioxZ/MRlwyJSnILSlzahmreqvq
+sBCF5ytqOVLfsc6/FsOEOPnFagF/8Zc/Vfu+rsuN75Hm65BcyMpoiKV7nKl8zU5cFUi0b0A9WzZ
AXSkA6OLXlwIAjozpiLuGOPPm7FeI+KxVuarPT+rbAlvyCt1er09G4JOa8QydYQLAufJK9H1+oe6
xIVJoeG09LWrDoLbJAID/+3Zr0Bb0zGbxn+oSfzFRUSgqVc+m88+qH7q7jjmLX3y8Ild1bFDbFGw
/4j4r5oc39/FxpL7Dgs4wAaDgrHuRvKl/KwCea4aazeNjOzq2mybFe+gZvhwtVEd0RtsIJWXqWID
/LN+v2n/gy41uz3fqnOcHX1LdwPqgAyzoDBN0ZF/v4Mk5VJ4th06d99AzfHlgRFvV/qOXdEC7Jok
eP3OJh2RnJ4pKVbl0QgEWdIYJAH8ec/9q3jujIXq2bBDwvB9KLBQ/5Batm2jb1Hro39pztSI9zxi
ZKCJ8d/dAXN3bNSNli0bBST9snRUPhglhlLZYLfXjY/D9PQMDOBpXrFAz1Bv149Vy5y1o6mxphSq
vuZYoIsRGRyFOx3umX6Y5Ze28wTN0hgkAyhqhE3qYX/29puiF+H98W+AX9cykp6ZIiLvZt7w4XVr
hOQtj43qxAIDbBmao2v8+ugOmSWmQAw0LOsMEgN92HHNsRyvc59Y+OSIYwzHEsA6pwR6CqbjLUyM
HD3mrZHR9X3uu+LapidjyzChH+w39Whkq6G0GLOgMEwBmZzvztihzjKcmhoZHSTA9bAKJsuC1OzO
TXG/0yBnn4nByPddEW1GKf0dlXVKtaSeQsKAzTIDp3qJBsIfgQoToOT6/sx9GdPT85vHb9uMAgFk
rDzja9OzkSpQP6WCuR/CiKMOYSEVVNYa9+oeq7Viha4Wb4BMZih4XbcPMm3rr5mQZ2iEFf+zKx6N
fb1Z5rlzTOx1fKzJjLnlsqEtZOVnEg5EyVwkLOsOYyLUzVyHvtPPV/cmxnfDF2oNBHJE+QcwnFXC
ibVGqdQwZbd6aOZuPALBXHZIFvWtafbRqnKB73m/vHoCWjVxzrgcSNrkEgO1HCpH5+FyVTY6JDDY
eVOfOvnNwa1zZy3+5V/xFJPihe2PuVmeRCmWdUOX6wv+N7gh39G7VEClBnqGzoAeA/60+ACGAxbt
OeO/MWJ7LutvdFl++2j8ZEv1Bm5R6wR5C0EiW3A8v7+50J1WaZIQAmta3L14HuqRcTWFBDwBfrLU
nx7dqBXWmZrRqnIDc6eNwbVaLYA/FgXYhz0ql57whf/cnxnZytCkXievG2hwz71BfPDYk6EQ0moh
2EVEOEU3ROf4wEe0goi1EtJCIWvl/qOFPtIXKeTE1R86qGA7UCZIfdTCwkZxLR6BbWhK6ptVHhWK
G3jqlHqZN6Iq+mY3Qoann3C/BxqvCEJENwDsAxgDoDGASEXXWdNsIIEsIcQGAbwC87O+BWoFImvV
Ymb35xVi196THPkIIPPvTdkVacVUAABb7SURBVFXb/cPbuunNBJP4GLsMVgsgMT4a2w4XYeaSfao
+3Vs0wOy/Xhi0gCGjGJky9gWQI4TYJ4QoB/AlgPHKDkKIxUKIc9LuagCu2WoYFvQAIoTADg+5qmv
DiNeWYNIHqz26H+46ftYlf3aoi4GSYC/uBZKPb+uLe4e1QfOkeKyUHtRfSTVEww0jgp4GQPnt8qQ
2d9wB4Be9A0Q0mYiyiSg7Pz9fr4ulYRt64Pjf6gMY++YyrMwpMO0aRwvdey2NnrHMpS2cHuhTL+3
kvZNFyExOwGOXdLREgjK/GnWJ6EYAWQBe0TsuhHhfCJElhMhKSQkfe6K/iAqjP+hwZ9Mhe2mw6z9
cg3s/2+C385ZVOosJN07Qn8X+4cabKdQX1JRoA2eY8MCIoB8GoFyOT5faVBDRSABPArhcCFGmPc4
Af/10fbCHEDEo63cq/Ytri7I8W4Wb5E23frxOtT+wrT2VajjN0MNnpIwSI4/hdQDaEVEm7EI+EcD
1yg5E1BPATACjhRDsbM0Eja15hVi6Jx9Ld5tj0rvvc2dJsnNlVR56Onl7Ui/M23Y05D0klFjB/OA
LKYlxqrqiix4ZEsTR1ByvM3QhRCWA+wDMB7ATwGwhxHYieo6Lpe6vQKgHoCviWgTgEEc0xbcQMo6G
8shrzt9vT01729nK8Mn+XSx9/FWBef+C0Y9vdDF1Lw4RY3NAvvDx5I1TP8dEtzhzwzZPi0TrMAq4
MGcqEEPMAzNO0PaXYHunncVmWE0WlSK0fGilTrcJLv/6J/yzfj68m93fbp7JaeCwn5gu7jp1FL03
RhO825Kn2P72jr1+vGQi6NK+Prs2Tgj2MoNAt3fm9K8MwwQ1HugSYiR+sDvYQLEfOiWIAwLly9ya
QCp3sejVl48HTqv3Hv9sKACgpq3S4MD48e7Oqz0Xtwm/xf+4DF6FObPi4WJpFUWlFsIdQY3gp22T
KK9VCsi+/BLd+vBaf3BZ+M7dQRV6oPF7k3i+881Pz8e3dF6J3q0Y+X+ftRTkubQNeXIgjkpiP15S
W++vg1j5fiwk+3krQhSI8QzeZ4a/94dL2x67I88E3k62H7S6K7y7Z67HfVe+uqtV19N4AjiiCi37
cdMSxPbFPC0wZ4z4zH8OYAQu6yShzYTP+RwjhyIA3umtTv5+/tKIKfZ9fgIwpc5GZop8HW4/nr+g
WsZ4iTPBgQWfCmn/9vhuF5+22Tm3+DT2qqwVe/GWnIzd9aUWVRw+Yez7bgBOSG9uhU/bsFtdmec9
sEU4+5zIcyRz+sKCbSGmFMT9lxnfe0rFry/z3dtd1ii2HCzFzyT4MmL4IJ4pK0XHqry45VwDgp81
HcOJsKRb96QyrWLanAETAyE5NPI5JTvYUbqx+YgT+eHRosIfB1ILw/OWFCSVllcEeQkQzqG0yerV
UF2Se8M4Kx/amQ/ZqQj9uPqLqU1Ragfu/2IhbPlJHfAJA3RgboryYUj68uY/H46FKcr04ZCQbNyt
ZnVAvZqEHC7qJhJ8Xa3hx/xcbPR6PiiJc46GIxGQpFUOxxj1tjrS4ufOoa7ZGIoI2rf2yvw9zbHd
uVh99M333pGFCh8T48HMCZEE3EaVpNtxCiMOBnzQza8A1wnFS35Zez3PmnFPQ1x84jX/8sM1t3+K
ySpAm00mLRnUdJcpmTOyB2Gj+s2KCQ9j98o4XleK133ahOgyiuJSLbZFQraiiqhoDpy/Ch8u8L07
Wlko3gUJCACumDMfP9w8yfK5LFN4xRee9B5P0zmjo0nZ9P/uDIzWC8ogzoUfYqcztn6zDW4tyHLk
7QhnlIychztqRdz9tPoJuz8zH4TPnMW3uTtOvV+IhKjStQR10TTMeut4k0ZmKQS86cGSnVMd2Vqu
GqB8fgw9uzlL1+euQ1lj292FoUDfW8HWZ0ORxKX6gf+vGQR5JzQk7QZcXGs+GwYJjleItonE968z
cVuQU4PM1Bx37+/KLcf8XG1WRddoIWX9SXS1wsrhmGZqT6sRgdBd9P/UqRZKt2TqVaj5UJGwa260
ZAOCYJio1LtqGFo3q1mhMTGjy1yFtsODhwXj+iq7BHkqNCTtBf/ryLgDslbgDzdr9pzDuzWW67oh
CCLy5cA/yTp9ztMmC/vLVFwAAXr+uu8fzV1WLkHd1PFlchhs+XIMnvt/qMCkNf22JS783F+4xbQy
tn5ine01A32Uwd/o4bH56FN67qTdevLIb3ruxt+p4heLBW6xJifvGxB4AgLkPDEKT+nG4ob/dtHJ
1r3Rc1Ssdm566uFbfhQlN2qYmIi46/N6qw07QOzWtDwAoOh/4GfpTP27D9iNF2Jdf4nJs1spc/Ov
33fjLrGxHmyzocsDGFT09B6Tc/NEadJz6K054yEkSbN5e7PT7XrP/lKF+ZtIoIRYvXNENHaVc419
OvtBj/0l9W2J016b45cGL8NaknrBFEQ4ronlPlahn/p2a2X9vXZonYc0TIx1/5HVibXjt2u5sYmF
CirATdNmVqLgs8JnQZP/jap3Iwmd+2gEA+PPYWUdbldRPL2pQLzpxRY69QO2/ft+taq+uFvh09QG
c92A3DhTKMazMKXBr+riql+vD68TZUlXFH3/w39v74vp+LfHujb3x0Mj26J5uzHbeqVl9XNa9Oaq
qBeZsPuJ4iCZLprEm9eOw/P+GoX2T8ClKwTBhJ+iyS1hNbbRV1QIPz96E9Qfczyq9YdSvRvbAkD1
x9AJRPly237G9au9J9Prn7479L9ep7bitn5iHqT9sw6PfbMaslbl47OvNjnD3QPOrYjH6zUU56D1
tger4FT3t9cO/3ZAHIYRqnH2fX6j6nv6gYYJ9hpyZnIAHR7bzOX+KXO39zLkKXNi6MRY8PATpDdk
mzoQXYSfo0VEEopoL+qZDZ/DdhsO1yrgn59T+dNUBVbt25rxHys/taYb+/LydyJgyF8/M2Y5JH6w
```

2NHOdu+Uonp6zHV+vz8M7ATJpaFH6bCvpk9EQL1zRDa9I6wUAkPn4PHR/9jccPHlO9zO+MGOB+u2
lqZ+Khfztq03ILSjB/oIStE2th8T48IsSZJiwE3QighDAwVM1E4kz55yC6Ws5MrmQwlfZh/D0j/b
gkz3Hz2Jz3hlVvzFvLAMAVFapbeh6fLIyV7ddnt0v2HFc97jsjbFyr93jxOh3qqyq9jklgadrdGm
ehOv7tUS0zfUn9cjXm5BzwmmKypgy16frA8CMBerFVn8mwRr66h8AgBV7C/x2ToYJJGEn6DI/bHK
NEvTEHYrFyt92HFe53RlB630ya9UBPPvTdlz8+lJMfN9ehahdqrP+YHFZJQok+7LRyMGf7x/kWNw
rPF+Bo4Xn8Zf/Zuv2PXOuAu8szsH1H9g9TjIfn4e/fek5FB4AnpqzHV2eno+MKXN1BbrDP35BxpS
5LhV+zpdXOb6nHp5yf6/LPY2R/1rqdWzeWKtZhL1naJvan/PJES5teoveDBMOhK2g14Sb/rNGtf/
XT9fjie+3OlKoGuHh2Ztc2rRZ+l5SmBsGv7wY93y2AQCQW+BdIHKnj0PXtCRHJr+e//wdF764Yhh
8oZHtkTt9HHKnj3O0aYsh/7DpiFcPGeWDrKhUPVMvq6xCmWTKum7mKsxamYtuT8+HEAKvzN/l1qu
lb2YjxMc4XbxeuqqbxzEAvr0lTf7U+XBb++QIPDKqQ43PoSU10dVkM8aEvOoMEwgsK+jj3lyGEa/
9gYwpc7Fsj/0VelRnddrTAdMX6X3UhdEzlmLeVu+Rqd3TnZn9TpWUOyrc9FEka1r7hHpGeEF6Em4
dkOHYj9ExWdxyYSs8MKKtY3/3tDFux/D6gt1Yve8kftt+DGWV6rcKra3/WGEpPl19AAdPnsObC/e
gwz9+dRzbcPAMnp6zHWfLKrE8pwAfrdiv+uwP9w4EAPzvjn743x39VMeu6+M9f8rxIuOBQVXVAh8
u2+cwQ029tDNSE+NNyzn+7xt6mXJehjGb8EsnBuDC1o1x/Kz7magQAtuPqDPldU9Pwrs39kabJ+a
p2vflFyOjcQJaS+3KGbCM0hXx5/sHYW9+MR780jlj/+PRoSguq4QtivDNXRfi6vfUC69dFBXUUxW
LePP/Nhgdmqrd4u4a2hqvaxb+nh2vjljTmnC2PjMK367PwzM/7cAXaw/hi7VOLxnl93lQY5K5ZIY
xM8hN/1mr2h/SPgU9WjTQvVcyvz80GK/+tgvzt9vXAKKjCGO6NYON7G8S/V9c6PHzSlbkFKjSCdw
+MMPQ53yFKw0x4UpYztCPFJ7HvvwSt6/th3VMKd/fMxC2KHIxWwx/bQk+W+P0WnlhnjoPidbNsWt
aEsb3SHPsr5gyHBnJCY7cIVqBbu0hv7S2LwCX6LTLuzd36QMAqx4f7thOjI/BrQMz3V4HsHsF/eZ
mgVXL30d7NmVM6us+Ja1MuyaJmHlTFhY9MgRTxnREzgtj8daknnj6si6OPn+Z5ZpvXI9//rxDte9
vwW2e5HzIDmqb7NdzM0wgCUtBPyC5wVVUCWRMmYu7pLzWMloPmM1Pj0KU5vVcNhkAwNQftzu231+
qzhSodHNspvjDnzahK24dkIG0BnVU/evFqV969unYz2Ojo5Bcz32E4donRuDj2/pgz/Nj8Oaknrp
9miXVcXk4ZbVyzQIoZz48rfDycXfOR0e1x7wHLlI9sPTo0cL1Ou5onVIPdw1xLl42TIh1iOaCnSd
UphR3yG6gAHBlL89j84XJg1sDAH68dyBm3tTbS2+GCV3CUtBlSiUb8a/bj6GguAyXvbUcuQUlLn7
iepVHtMKrJGPKXBTruPbNf2iwY/vG/q3wzOVdXPoQES69oJnHcW99ZhRWTnH1rpBJrR+PYR1Sde3
pnvj6rgsxvGOqqm3a3J3YfOiMakHT3az/vuHt0Ll5faQ1qIM4yawjR3xunHoxtj4zCsv+PgxNk2r
n+708x+kWOG3uTjykWHDecPA0ftx02LGvTZP70Mj2tbq2HrcOzETu9HHo3qIBEjz8Lhgm1AnrX++
BAudMPEuKWJR9iQF7Miw3abN1k3vdNjDD4bnS9en5mPuAM6e2UXsvALx9fS/8vMXua31jf9cFQrO
S/hARPrq1j8PFsuNU+yLneEXZtXeuty/47X9xLAqKy5GSGIeteYVo2VgdFblLsfD62rXOpGL+CLi
Z+8AgjHtzuWP/x01HMPXSzkiuF4cr/70SgP2hQ0Q4qQi4Gt+jOZpr3ogYhnESljP0we1TAABr9p/
02O+Knum4urd+QizZzU4p7PcNa6vqoxSdmjLjOnuWvpv6Z/h8Dl+Jj7Gp3AiVXCx5+hARUqRiDN3
SkwJaP7GzlPBKSda0BSqXSjly9ou19rYreqbhjYk9TfNsYRgrEJYz9Kt7p2Pp7nyPhRSaeTELNEq
IxYzremBA28aIIkLDurFwpxWPXVJzf+cJPdMwoaf/7b21JRTKoxER9r4w1sXj6Invtzq2e09bgNz
p4xyRof8Y1ymgY2SYcCT4f90+YGSOtupx9zZqmQk905CaGI/kenGwRRGICDNv6o2pl3ZW9evS3HV
GGQ68eKU6wEe7gBtMjMy0v84+hATpDcpKBUIYxizCcoZ+iZvKM/Xjo3H30Lbo2bKB7vGanFvpKte
qsXvXw1BmUt+WGNExFVVCoFqElqADdtPZ0t35bo8/9s0WAECblPC8/wwTaMJyhu7ObNAsqQ7uHtr
GL7UA5ejNT27rg0wPvuShTmr9eDRLqhNyYg7Yc5lrizmnNaiDhy9We7Ls5dwqDGMIQ4JORKOJaBc
R5RDRFJ3jcUT0lXR8DRFl+HugRth1/Kz3TgaZMqYjVkwZjqEdUr13ZnxGWcz5+n4tsWLKcHRv4fs
bFsNEMl4FnYhsAN4BMAZAZwCTiKizptsdAE4LIdoCeB3AS/4eqJZHR7n6I/vz1Tw+xhaSs1orsvm
pUbikSxNMk1IcaIOuONiHYYxhZIbeF0COEGKfEKIcwJcAxmv6jAcwS9r+BsAIMjkhRp1Yu/n//uF
t8fmd9uRQ/9UkiWLCg6S6MZh5U5YjmldZ9i13+ji3ayYMw6gxsiiaBkBZEy0PgFY5HX2EEJVEVAi
gMQBVpQAimgxgMgC0bOk9I58nbujXEseLSnHXkDZIiIuuUeAPE9rE2KL435NhfCCgi6JCiPeFEFl
CiKyUlJRanSs+xoYnxnbiUG2GYRgJI4J+GIAyvV661Kbbh4iiASQB8BzGyTAMw/gVIw7cAM/gDw/waq+QbOL3AZgPwAbgIyHEdiJ6DkC2EGC2EGIOgP8A+JS
IcgCcglDMMwFoCMKCzrDMIxFYEFnGoDMmOLj2MjcIkgCMMwDOMKCzrDMIxFCDtB91aw2uoQUQsiWkxEO4hoOxE94kbnkUuA+YiP9P+GUjsR0Zv
S/dpCRL0U57pF6r+HiG5xd81whYhwbZ2FzInpcat9FRJcE55uYCxE
1IKJviOhPITpJRBfy78kBbBAZgCdgz2uAN+DZgB
6SduJAHbDXrz7ZQBTpPYpAF6StgcD+AyAIeg/PYP
Tpe33ANwtbd8D4D1peyKAr6TtztJvLA5ApvTbtswX7e5lwn2YB+Iu0HQugAf+eXO5RGoD9AOoofku
3htpvKtxm6EYKVlsaIcRRIcQGafssgJ2w/9H2w/9iUhbpnAZggbY8H8F9hZzWABkTUDMAlAH4XQpwSQpw
G8DuA0QH8KqZCROkAjgL2IOuN4jvSLn4wF8KYQoE0LsB5AD
iXAhxBvx70iMaQB2pKltdAEcRYr+pcBN0vYLVaUEaS9CRXuN6AlgDoIkQ4qh06BiAJtK2u9/Ef
5A8DfAVRL+40BnBFCVErH7yu/UQL496RCCHEYwsADsIu5IU
A1iPEflPhJuiMBBHVA/AtgL8JIYqUx4UQAkBYFU0yEIRXFMwT2bE5nuCH1MgrAGnJhqO3MK1BXTq4O
g0n9PACCtIYyH/QHYHEACQvANJNwE3UjBastDRDGwi/lnQojvpObj0qsvxu2bAC4
nolzYzXLDAbwBu3lArtKl/L7uipxb+R7J5AHIE0Kskfa/gV3g+fekZiSA/UKIfCFEBYDvYP+dhdR
vKtwE3UjBaksj2eH+A2CnEOJfikPzANwibd8C4EdcifSF

```
EuhAiA/bfyCIhxA0AFsNexBxwvUd6Rc7nAJgoeSxkAmgHYG2AvkZAEEIcA3CIiDpITSMA7AD/nrQ
cBNCfiOpKf4PyfQqt31SwV499WG0eC7tnx14ATwZ7PEH4/oNgf/3dAmCT9N9Y2O1zCwHsAbAAQCO
pPwF4R7pfWwFkKc51O+yLMjkAbgv2dzPpfg2F08ultfTHkwPgawBxUnu8tJ8jHW+t+PyT0r3bBWB
MsL+PSfeoB4Bs6Tf1A+xeKvx7cr1PzwL4E8A2AJ/C7qkSUr8pDv1nGIaxCOFmcmEYhmHcwILOMAx
jEVjQGYZhLAILOsMwjEVgQWcYhrEILOgMwzAWgQWdYRjGIvw/IlbURK54GdgAAAAASUVORK5CYII
=\n"
          },
          "metadata": {
            "needs_background": "light"

}
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "training_size=int(len(data_oil)*0.65)\n",
        "test_size=len(data_oil)-training_size\n",

"train_data,test_data=data_oil[0:training_size,:],data_oil[training_size:len
(data_oil),:1]"
      ],
      "metadata": {
        "id": "S9THgFu3KaKH"
      },
      "execution_count": 11,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "training_size,test_size"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "mbvhl2HCKmSG",
        "outputId": "5142d0f7-21d1-46bb-e69e-bd71c69ef729"
      },
      "execution_count": 12,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(5340, 2876)"
            ]
          },
          "metadata": {},
          "execution_count": 12
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "train_data.shape"
      ],
      "
```

```
metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "M14MiYVzKrJg",
        "outputId": "b23a9806-135a-424d-f65b-6029116ad975"
      },
      "execution_count": 13,
      "outputs": [
        {
          "output_type": "execute_result",

"data": {
            "text/plain": [
              "(5340, 1)"
            ]
          },
          "metadata": {},
          "execution_count": 13
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "def create_dataset(dataset,time_step=1):\n",
        "  dataX,dataY=[],[]\n",
        "  for i in range(len(dataset)-time_step-1):\n",
        "    a=dataset[i:(i+time_step),0]\n",
        "    dataX.append(a)\n",
        "    dataY.append(dataset[i+time_step,0])\n",
        "  return np.array(dataX),np.array(dataY)"
      ],
      "metadata": {
        "id": "vrH4kcM5K9S8"
      },
      "execution_count": 14,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "time_step=10\n",
        "x_train,y_train=create_dataset(train_data,time_step)\n",
        "x_test,y_test=create_dataset(test_data,time_step)"
      ],
      "metadata": {
        "id": "dv2OIjslLA3v"
      },
      "execution_count": 15,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "print(x_train.shape),print(y_train.shape)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
```

```
 },
        "id": "ApfZGkVhLD2p",
        "outputId": "979643ec-00c5-4f15-93f0-8529e83335a0"
      },
      "execution_count": 16,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "(5329, 10)\n",
            "(5329,)\n"
          ]

},
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(None, None)"
            ]
          },
          "metadata": {},
          "execution_count": 16
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "print(x_test.shape),print(y_test.shape)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "iLQiXr21LE35",
        "outputId": "3dca6c25-95ed-4b85-bb1a-d2e218179c5d"
      },
      "execution_count": 17,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "(2865, 10)\n",
            "(2865,)\n"
          ]
        },
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "(None, None)"
            ]
          },
          "metadata": {},
          "execution_count": 17
        }
      ]
 },
    {
      "cell_type": "code",
      "source": [
        "x_train"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "w95tBQ4gLJWj",
        "outputId": "f1567ef1-fb46-4909-857b-c3c34c6f1dd1"
      },
      "execution_count": 18,
      "outputs": [

 {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0.11335703, 0.11661484, 0.12053902, ..., 0.10980305,
0.1089886 ,\n",
              "        0.11054346],\n",
              "       [0.11661484, 0.12053902, 0.11550422, ..., 0.1089886 ,
0.11054346,\n",
              "        0.10165852],\n",
              "       [0.12053902, 0.11550422, 0.1156523 , ..., 0.11054346,
0.10165852,\n",
              "        0.09906708],\n",
              "       ...,\n",
              "       [0.36731823, 0.35176958, 0.36080261, ..., 0.36391234,
0.37042796,\n",
              "        0.37042796],\n",
              "       [0.35176958, 0.36080261, 0.35354657, ..., 0.37042796,
0.37042796,\n",
              "        0.37879461],\n",
              "       [0.36080261, 0.35354657, 0.35295424, ..., 0.37042796,
0.37879461,\n",
              "        0.37916482]])"
            ]
          },
          "metadata": {},
          "execution_count": 18
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "x_train=x_train.reshape(x_train.shape[0],x_train.shape[1],1)\n",
        "x_test=x_test.reshape(x_test.shape[0],x_test.shape[1],1)"
      ],
      "metadata": {
        "id": "59VQDX1LLL3D"
      },
      "execution_count": 19,
      "outputs": []
    },
    {
      "cell_type": "code",
"source": [
        "from tensorflow.keras.models import Sequential\n",
        "from tensorflow.keras.layers import Dense\n",
        "from tensorflow.keras.layers import LSTM"
      ],
      "metadata": {
        "id": "UFWasZVxMc3z"
      },
      "execution_count": 20,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model=Sequential()"
      ],
      "metadata": {

"id": "7TNg_dAdMl0e"
      },
      "execution_count": 22,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "model.add(LSTM(50,return_sequences=True,input_shape=(10,1)))\n",
        "model.add(LSTM(50,return_sequences=True))\n",
        "model.add(LSTM(50))"
      ],
      "metadata": {
        "id": "anZVlUc2M19G"
      },
      "execution_count": 23,
      "outputs": []
    }
  ]
}
```

PROJECT.NPYL

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": [],
      "collapsed_sections": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "markdown",
      "source": [
"**IMPORTING LIBRARIES**"
      ],
      "metadata": {
        "id": "KuaQGcgOP_dj"
      }
    },
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {
        "id": "Xpc4nrINCnEP"
      },
      "outputs": [],
      "source": [
        "import numpy as np # linear algebra\n",
        "import pandas as pd # data processing, CSV file I/O (e.g.
pd.read_csv)\n",
        "import datetime\n",
        "from pylab import rcParams\n",
        "import matplotlib.pyplot as plt\n",
        "import warnings\n",
        "import itertools\n",
        "import statsmodels.api as sm\n",
        "from keras.models import Sequential\n",
        "from keras.layers import Dense\n",
        "from keras.layers import LSTM\n",
        "from keras.layers import Dropout\n",
        "from sklearn.metrics import mean_squared_error\n",
        "from keras.callbacks import ReduceLROnPlateau, EarlyStopping,
ModelCheckpoint\n",
        "from sklearn.metrics import mean_squared_error\n",
        "from sklearn.metrics import mean_absolute_error\n",
        "import seaborn as sns\n",
        "sns.set_context(\"paper\", font_scale=1.3)\n",
        "sns.set_style('white')\n",
        "import math\n",
        "from sklearn.preprocessing import MinMaxScaler\n",
        "# Input data files are available in the \"../input/\"
directory.\n",
        "# For example, running this (by clicking run or pressing
Shift+Enter) will list all files under the input directory\n",
        "warnings.filterwarnings(\"ignore\")\n",
        "plt.style.use('fivethirtyeight')\n",
        "import os\n",
        "for dirname, _, filenames in os.walk('/kaggle/input'):\n",
        "    for filename in filenames:\n",
        "        print(os.path.join(dirname, filename))"
      ]
    },
    {
      "cell_type": "markdown",
      "source": [
        "**IMPORTING DATA**"
      ],
      "metadata": {
        "id": "rZ8wvz_NQLwh"
      }
    },
    {
      "cell_type": "code",
 "source": [
        "dateparse = lambda x: pd.datetime.strptime(x, '%b %d, %Y')\n",
        "#Read csv file\n",
        "from google.colab import files\n",
        "uploaded = files.upload()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 74
        },
        "id": "KRqOakawLGQS",
        "outputId": "501f7bac-45b2-4cc8-a55c-2883833482a1"
      },
      "execution_count": 2,
      "outputs": [
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<IPython.core.display.HTML object>"
            ],
            "text/html": [
              "\n",
              "     <input type=\"file\" id=\"files-2ced4f3e-b49c-4ad5-9829-
7295fcec4ced\" name=\"files[]\" multiple disabled\n",
              "        style=\"border:none\" />\n",
              "     <output id=\"result-2ced4f3e-b49c-4ad5-9829-
7295fcec4ced\">\n",
              "      Upload widget is only available when the cell has been
executed in the\n",
              "      current browser session. Please rerun this cell to
enable.\n",
              "      </output>\n",
              "      <script>// Copyright 2017 Google LLC\n",
              "//\n",
              "// Licensed under the Apache License, Version 2.0 (the
\"License\");\n",
              "// you may not use this file except in compliance with the
License.\n",
              "// You may obtain a copy of the License at\n",
              "//\n",
              "//      http://www.apache.org/licenses/LICENSE-2.0\n",
              "//\n",
              "// Unless required by applicable law or agreed to in writing,
software\n",
              "// distributed under the License is distributed on an \"AS
IS\" BASIS,\n",
              "// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied.\n",
              "// See the License for the specific language governing
permissions and\n",
              "// limitations under the License.\n",
              "\n",
              "/**\n",
              " * @fileoverview Helpers for google.colab Python module.\n",
              " */\n",
              "(function(scope) {\n",
              "function span(text, styleAttributes = {}) {\n",
              "  const element = document.createElement('span');\n",
"  element.textContent = text;\n",
              "  for (const key of Object.keys(styleAttributes)) {\n",
              "    element.style[key] = styleAttributes[key];\n",
              "  }\n",
              "  return element;\n",
              "}\n",
              "\n",
              "// Max number of bytes which will be uploaded at a time.\n",
              "const MAX_PAYLOAD_SIZE = 100 * 1024;\n",
              "\n",
              "function _uploadFiles(inputId, outputId) {\n",
              "  const steps = uploadFilesStep(inputId, outputId);\n",
              "  const outputElement =
document.getElementById(outputId);\n",
              "  // Cache steps on the outputElement to make it available
for the next call\n",
              "  // to uploadFilesContinue from Python.\n",
              "  outputElement.steps = steps;\n",
              "\n",
              "  return _uploadFilesContinue(outputId);\n",
              "}\n",
              "\n",
              "// This is roughly an async generator (not supported in the
browser yet),\n",
              "// where there are multiple asynchronous steps and the Python
side is going\n",
              "// to poll for completion of each step.\n",
              "// This uses a Promise to block the python side on completion
of each step,\n",
              "// then passes the result of the previous step as the input
to the next step.\n",
              "function _uploadFilesContinue(outputId) {\n",
              "  const outputElement =
document.getElementById(outputId);\n",
              "  const steps = outputElement.steps;\n",
              "\n",
              "  const next =
steps.next(outputElement.lastPromiseValue);\n",
              "  return Promise.resolve(next.value.promise).then((value) =>
{\n",
              "    // Cache the last promise value to make it available to
the next\n",
              "    // step of the generator.\n",
              "    outputElement.lastPromiseValue = value;\n",
              "    return next.value.response;\n",
              "  });\n",
              "}\n",
              "\n",
              "/**\n",
              " * Generator function which is called between each async step
of the upload\n",
              " * process.\n",
              " * @param {string} inputId Element ID of the input file
picker element.\n",
              " * @param {string} outputId Element ID of the output
display.\n",
              " * @return {!Iterable<!Object>} Iterable of next steps.\n",
              " */\n",
              "function* uploadFilesStep(inputId, outputId) {\n",
              "  const inputElement = document.getElementById(inputId);\n",
 "  inputElement.disabled = false;\n",
              "\n",
              "  const outputElement =
document.getElementById(outputId);\n",
              "  outputElement.innerHTML = '';\n",
              "\n",
              "  const pickedPromise = new Promise((resolve) => {\n",
              "    inputElement.addEventListener('change', (e) => {\n",
              "      resolve(e.target.files);\n",
              "    });\n",
              "  });\n",
              "\n",
              "  const cancel = document.createElement('button');\n",
              "  inputElement.parentElement.appendChild(cancel);\n",
              "  cancel.textContent = 'Cancel upload';\n",
              "  const cancelPromise = new Promise((resolve) => {\n",
              "    cancel.onclick = () => {\n",
              "      resolve(null);\n",
              "    };\n",
              "  });\n",
              "\n",
              "  // Wait for the user to pick the files.\n",
              "  const files = yield {\n",
              "    promise: Promise.race([pickedPromise,
cancelPromise]),\n",
              "    response: {\n",
              "      action: 'starting',\n",
              "    }\n",
              "  };\n",
              "\n",
              "  cancel.remove();\n",
              "\n",
              "  // Disable the input element since further picks are not
allowed.\n",
              "  inputElement.disabled = true;\n",
              "\n",
              "  if (!files) {\n",
              "    return {\n",
              "      response: {\n",
              "        action: 'complete',\n",
              "      }\n",
              "    };\n",
              "  }\n",
              "\n",
              "  for (const file of files) {\n",
              "    const li = document.createElement('li');\n",
              "    li.append(span(file.name, {fontWeight: 'bold'}));\n",
              "    li.append(span(\n",
              "        `(${file.type || 'n/a'}) - ${file.size} bytes, `
+\n",
              "        `last modified: ${\n",
              "            file.lastModifiedDate ?
file.lastModifiedDate.toLocaleDateString() :\n",
              "                                      'n/a'} - `));\n",
              "    const percent = span('0% done');\n",
              "    li.appendChild(percent);\n",
              "\n",
              "    outputElement.appendChild(li);\n",
              "\n",
              "    const fileDataPromise = new Promise((resolve) => {\n",
 "      const reader = new FileReader();\n",
              "      reader.onload = (e) => {\n",
              "        resolve(e.target.result);\n",
              "      };\n",
              "      reader.readAsArrayBuffer(file);\n",
              "    });\n",
              "    // Wait for the data to be ready.\n",
              "    let fileData = yield {\n",
              "      promise: fileDataPromise,\n",
              "      response: {\n",
              "        action: 'continue',\n",
              "      }\n",
              "    };\n",
              "\n",
              "    // Use a chunked sending to avoid message size limits.
See b/62115660.\n",
              "    let position = 0;\n",
              "    do {\n",
              "      const length = Math.min(fileData.byteLength - position,
MAX_PAYLOAD_SIZE);\n",
              "      const chunk = new Uint8Array(fileData, position,
length);\n",
              "      position += length;\n",
              "\n",
              "      const base64 = btoa(String.fromCharCode.apply(null,
chunk));\n",
              "      yield {\n",
              "        response: {\n",
              "          action: 'append',\n",
              "          file: file.name,\n",
              "          data: base64,\n",
              "        },\n",
              "      };\n",
              "\n",
              "      let percentDone = fileData.byteLength === 0 ?\n",
              "          100 :\n",
              "          Math.round((position / fileData.byteLength) *
100);\n",
              "      percent.textContent = `${percentDone}% done`;\n",
              "\n",
              "    } while (position < fileData.byteLength);\n",
              "  }\n",
              "\n",
              "  // All done.\n",
              "  yield {\n",
              "    response: {\n",
              "      action: 'complete',\n",
              "    }\n",
              "  };\n",
              "}\n",
              "\n",
              "scope.google = scope.google || {};\n",
              "scope.google.colab = scope.google.colab || {};\n",
              "scope.google.colab._files = {\n",
              "  _uploadFiles,\n",
              "  _uploadFilesContinue,\n",
              "};\n",
              "})(self);\n",
              "</script> "
            ]
```

```
   },
          "metadata": {}
        },
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "Saving Crude Oil Prices Daily.xlsx to Crude Oil Prices
Daily.xlsx\n"
          ]
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "import io\n",
        "df = pd.read_excel(io.BytesIO(uploaded['Crude Oil Prices
Daily.xlsx']))\n",
        "df.head()\n",
        "df[:10]\n"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 363
        },
        "id": "dvTAuKK2LRE-",
        "outputId": "e7855c35-13d9-465e-e0ac-8570c978480b"
      },
      "execution_count": 8,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "        Date  Closing Value\n",
              "0 1986-01-02          25.56\n",
              "1 1986-01-03          26.00\n",
              "2 1986-01-06          26.53\n",
              "3 1986-01-07          25.85\n",
              "4 1986-01-08          25.87\n",
              "5 1986-01-09          26.03\n",
              "6 1986-01-10          25.65\n",
              "7 1986-01-13          25.08\n",
              "8 1986-01-14          24.97\n",
              "9 1986-01-15          25.18"
            ],
            "text/html": [
              "\n",
              "  <div id=\"df-2367b016-b67f-445c-8b15-4defa2e543d3\">\n",
              "    <div class=\"colab-df-container\">\n",
              "      <div>\n",
              "<style scoped>\n",
              "    .dataframe tbody tr th:only-of-type {\n",
              "        vertical-align: middle;\n",
              "    }\n",
              "\n",
              "    .dataframe tbody tr th {\n",
              "        vertical-align: top;\n",
              "
}\n",
           "\n",
           "    .dataframe thead th {\n",
           "        text-align: right;\n",
           "    }\n",
           "</style>\n",
           "<table border=\"1\" class=\"dataframe\">\n",
           "  <thead>\n",
           "    <tr style=\"text-align: right;\">\n",
           "      <th></th>\n",
           "      <th>Date</th>\n",
           "      <th>Closing Value</th>\n",
           "    </tr>\n",
           "  </thead>\n",
           "  <tbody>\n",
           "    <tr>\n",
           "      <th>0</th>\n",
           "      <td>1986-01-02</td>\n",
           "      <td>25.56</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>1</th>\n",
           "      <td>1986-01-03</td>\n",
           "      <td>26.00</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>2</th>\n",
           "      <td>1986-01-06</td>\n",
           "      <td>26.53</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>3</th>\n",
           "      <td>1986-01-07</td>\n",
           "      <td>25.85</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>4</th>\n",
           "      <td>1986-01-08</td>\n",
           "      <td>25.87</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>5</th>\n",
           "      <td>1986-01-09</td>\n",
           "      <td>26.03</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>6</th>\n",
           "      <td>1986-01-10</td>\n",
           "      <td>25.65</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>7</th>\n",
           "      <td>1986-01-13</td>\n",
           "      <td>25.08</td>\n",
           "    </tr>\n",
           "    <tr>\n",
           "      <th>8</th>\n",
           "      <td>1986-01-14</td>\n",
           "      <td>24.97</td>\n",
           "    </tr>\n",
           "
```

```
<tr>\n",
              "      <th>9</th>\n",
              "      <td>1986-01-15</td>\n",
              "      <td>25.18</td>\n",
              "    </tr>\n",
              "  </tbody>\n",
              "</table>\n",
              "</div>\n",
              "      <button class=\"colab-df-convert\"
onclick=\"convertToInteractive('df-2367b016-b67f-445c-8b15-
4defa2e543d3')\"\n",
              "              title=\"Convert this dataframe to an
interactive table.\"\n",
              "              style=\"display:none;\">\n",
              "        \n",
              "  <svg xmlns=\"http://www.w3.org/2000/svg\"
height=\"24px\"viewBox=\"0 0 24 24\"\n",
              "       width=\"24px\">\n",
              "    <path d=\"M0 0h24v24H0V0z\" fill=\"none\"/>\n",
              "    <path d=\"M18.56 5.44l.94 2.06.94-2.06 2.06-.94-2.06-.94-
.94-2.06-.94 2.06-2.06.94zm-11 1L8.5 8.5l.94-2.06 2.06-.94-2.06-.94L8.5
2.5l-.94 2.06-2.06.94zm10 10l.94 2.06.94-2.06 2.06-.94-2.06-.94-.94-2.06-.94
2.06-2.06.94z\"/><path d=\"M17.41 7.96l-1.37-1.37c-.4-.4-.92-.59-1.43-.59-
.52 0-1.04.2-1.43.59L10.3 9.45l-7.72 7.72c-.78.78-.78 2.05 0 2.83L4
21.41c.39.39.9.59 1.41.59.51 0 1.02-.2 1.41-.59l7.78-7.78 2.81-2.81c.8-
.78.8-2.07 0-2.86zM5.41 20L4 18.59l7.72-7.72 1.47 1.35L5.41 20z\"/>\n",
              "  </svg>\n",
              "      </button>\n",
              "      \n",
              "  <style>\n",
              "    .colab-df-container {\n",
              "      display:flex;\n",
              "      flex-wrap:wrap;\n",
              "      gap: 12px;\n",
              "    }\n",
              "\n",
              "    .colab-df-convert {\n",
              "      background-color: #E8F0FE;\n",
              "      border: none;\n",
              "      border-radius: 50%;\n",
              "      cursor: pointer;\n",
              "      display: none;\n",
              "      fill: #1967D2;\n",
              "      height: 32px;\n",
              "      padding: 0 0 0 0;\n",
              "      width: 32px;\n",
              "    }\n",
              "\n",
              "    .colab-df-convert:hover {\n",
              "      background-color: #E2EBFA;\n",
              "      box-shadow: 0px 1px 2px rgba(60, 64, 67, 0.3), 0px 1px
3px 1px rgba(60, 64, 67, 0.15);\n",
              "      fill: #174EA6;\n",
              "    }\n",
              "\n",
              "    [theme=dark] .colab-df-convert {\n",
              "      background-color: #3B4455;\n",
              "      fill: #D2E3FC;\n",
              "    }\n",
              "\n",
              "
```

```
[theme=dark] .colab-df-convert:hover {\n",
              "      background-color: #434B5C;\n",
              "      box-shadow: 0px 1px 3px 1px rgba(0, 0, 0, 0.15);\n",
              "      filter: drop-shadow(0px 1px 2px rgba(0, 0, 0,
0.3));\n",
              "      fill: #FFFFFF;\n",
              "    }\n",
              "  </style>\n",
              "\n",
              "      <script>\n",
              "        const buttonEl =\n",
              "          document.querySelector('#df-2367b016-b67f-445c-
8b15-4defa2e543d3 button.colab-df-convert');\n",
              "        buttonEl.style.display =\n",
              "          google.colab.kernel.accessAllowed ? 'block' :
'none';\n",
              "\n",
              "        async function convertToInteractive(key) {\n",
              "          const element = document.querySelector('#df-
2367b016-b67f-445c-8b15-4defa2e543d3');\n",
              "          const dataTable =\n",
              "            await
google.colab.kernel.invokeFunction('convertToInteractive',\n",
              "                                                     [key],
{});\n",
              "          if (!dataTable) return;\n",
              "\n",
              "          const docLinkHtml = 'Like what you see? Visit the '
+\n",
              "            '<a target=\"_blank\"
href=https://colab.research.google.com/notebooks/data_table.ipynb>data table
notebook</a>'\n",
              "            + ' to learn more about interactive tables.';\n",
              "          element.innerHTML = '';\n",
              "          dataTable['output_type'] = 'display_data';\n",
              "          await google.colab.output.renderOutput(dataTable,
element);\n",
              "          const docLink = document.createElement('div');\n",
              "          docLink.innerHTML = docLinkHtml;\n",
              "          element.appendChild(docLink);\n",
              "        }\n",
              "      </script>\n",
              "    </div>\n",
              "  </div>\n",
              "  "
            ]
          },
          "metadata": {},
          "execution_count": 8
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "#Sort dataset by column Date\n",
        "df = df.sort_values('Date')\n",
        "df = df.groupby('Date')['Closing Value'].sum().reset_index()\n",
        "df.set_index('Date', inplace=True)\n",
        "df=df.loc[datetime.date(year=2000,month=1,day=1):]"
    ],
      "metadata": {
        "id": "SKu5lBkmMYIG"
      },
      "execution_count": 9,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "df.head()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 237
        },
        "id": "3puldFJ2NBYi",
        "outputId": "1833c97a-153e-4ffc-8bc3-9a0053b45b50"
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "            Closing Value\n",
              "Date                     \n",
              "2000-01-04          25.56\n",
              "2000-01-05          24.65\n",
              "2000-01-06          24.79\n",
              "2000-01-07          24.79\n",
              "2000-01-10          24.71"
            ],
            "text/html": [
              "\n",
              "  <div id=\"df-6af25580-39d5-4204-8adf-55ea4e83ce2e\">\n",
              "    <div class=\"colab-df-container\">\n",
              "      <div>\n",
              "<style scoped>\n",
              "    .dataframe tbody tr th:only-of-type {\n",
              "        vertical-align: middle;\n",
              "    }\n",
              "\n",
              "    .dataframe tbody tr th {\n",
              "        vertical-align: top;\n",
              "    }\n",
              "\n",
              "    .dataframe thead th {\n",
              "        text-align: right;\n",
              "    }\n",
              "</style>\n",
              "<table border=\"1\" class=\"dataframe\">\n",
              "  <thead>\n",
              "    <tr style=\"text-align: right;\">\n",
              "      <th></th>\n",
              "      <th>Closing Value</th>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>Date</th>\n",
              "
    <th></th>\n",
              "    </tr>\n",
              "  </thead>\n",
              "  <tbody>\n",
              "    <tr>\n",
              "      <th>2000-01-04</th>\n",
              "      <td>25.56</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>2000-01-05</th>\n",
              "      <td>24.65</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>2000-01-06</th>\n",
              "      <td>24.79</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>2000-01-07</th>\n",
              "      <td>24.79</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>2000-01-10</th>\n",
              "      <td>24.71</td>\n",
              "    </tr>\n",
              "  </tbody>\n",
              "</table>\n",
              "</div>\n",
              "       <button class=\"colab-df-convert\"
onclick=\"convertToInteractive('df-6af25580-39d5-4204-8adf-
55ea4e83ce2e')\"\n",
              "              title=\"Convert this dataframe to an
interactive table.\"\n",
              "              style=\"display:none;\">\n",
              "        \n",
              "  <svg xmlns=\"http://www.w3.org/2000/svg\"
height=\"24px\"viewBox=\"0 0 24 24\"\n",
              "       width=\"24px\">\n",
              "    <path d=\"M0 0h24v24H0V0z\" fill=\"none\"/>\n",
              "    <path d=\"M18.56 5.44l.94 2.06.94-2.06 2.06-.94-2.06-.94-
.94-2.06-.94 2.06-2.06.94zm-11 1L8.5 8.5l.94-2.06 2.06-.94-2.06-.94L8.5
2.5l-.94 2.06-2.06.94zm10 10l.94 2.06.94-2.06 2.06-.94-2.06-.94-.94-2.06-.94
2.06-2.06.94z\"/><path d=\"M17.41 7.96l-1.37-1.37c-.4-.4-.92-.59-1.43-.59-
.52 0-1.04.2-1.43.59L10.3 9.45l-7.72 7.72c-.78.78-.78 2.05 0 2.83L4
21.41c.39.39.9.59 1.41.59.51 0 1.02-.2 1.41-.59l7.78-7.78 2.81-2.81c.8-
.78.8-2.07 0-2.86zM5.41 20L4 18.59l7.72-7.72 1.47 1.35L5.41 20z\"/>\n",
              "  </svg>\n",
              "      </button>\n",
              "      \n",
              "  <style>\n",
              "    .colab-df-container {\n",
              "      display:flex;\n",
              "      flex-wrap:wrap;\n",
              "      gap: 12px;\n",
              "    }\n",
              "\n",
              "    .colab-df-convert {\n",
              "      background-color: #E8F0FE;\n",
              "      border: none;\n",
              "      border-radius: 50%;\n",
              "      cursor: pointer;\n",
              "
    display: none;\n",
              "      fill: #1967D2;\n",
              "      height: 32px;\n",
              "      padding: 0 0 0 0;\n",
              "      width: 32px;\n",
              "    }\n",
              "\n",
              "    .colab-df-convert:hover {\n",
              "      background-color: #E2EBFA;\n",
              "      box-shadow: 0px 1px 2px rgba(60, 64, 67, 0.3), 0px 1px
3px 1px rgba(60, 64, 67, 0.15);\n",
              "      fill: #174EA6;\n",
              "    }\n",
              "\n",
              "    [theme=dark] .colab-df-convert {\n",
              "      background-color: #3B4455;\n",
              "      fill: #D2E3FC;\n",
              "    }\n",
              "\n",
              "    [theme=dark] .colab-df-convert:hover {\n",
              "      background-color: #434B5C;\n",
              "      box-shadow: 0px 1px 3px 1px rgba(0, 0, 0, 0.15);\n",
              "      filter: drop-shadow(0px 1px 2px rgba(0, 0, 0,
0.3));\n",
              "      fill: #FFFFFF;\n",
              "    }\n",
              "  </style>\n",
              "\n",
              "      <script>\n",
              "        const buttonEl =\n",
              "          document.querySelector('#df-6af25580-39d5-4204-
8adf-55ea4e83ce2e button.colab-df-convert');\n",
              "        buttonEl.style.display =\n",
              "          google.colab.kernel.accessAllowed ? 'block' :
'none';\n",
              "\n",
              "        async function convertToInteractive(key) {\n",
              "          const element = document.querySelector('#df-
6af25580-39d5-4204-8adf-55ea4e83ce2e');\n",
              "          const dataTable =\n",
              "            await
google.colab.kernel.invokeFunction('convertToInteractive',\n",
              "                                                     [key],
{});\n",
              "          if (!dataTable) return;\n",
              "\n",
              "          const docLinkHtml = 'Like what you see? Visit the '
+\n",
              "            '<a target=\"_blank\"
href=https://colab.research.google.com/notebooks/data_table.ipynb>data table
notebook</a>'\n",
              "            + ' to learn more about interactive tables.';\n",
              "          element.innerHTML = '';\n",
              "          dataTable['output_type'] = 'display_data';\n",
              "          await google.colab.output.renderOutput(dataTable,
element);\n",
              "          const docLink = document.createElement('div');\n",
              "          docLink.innerHTML = docLinkHtml;\n",
              "          element.appendChild(docLink);\n",
              "        }\n",
              "
```

```
   </script>\n",
              "    </div>\n",
              "  </div>\n",
              "  "
            ]
          },
          "metadata": {},
          "execution_count": 10
        }
      ]
    },
    {
      "cell_type": "markdown",
      "source": [
        "**DATA PRE-PROCESSING**"
      ],
      "metadata": {
        "id": "H7NROSsvQTWW"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "def DfInfo(df_initial):\n",
        "    # gives some infos on columns types and numer of null 
values\n",
        "    tab_info = pd.DataFrame(df_initial.dtypes).T.rename(index={0: 
'column type'})\n",
        "    tab_info = 
tab_info.append(pd.DataFrame(df_initial.isnull().sum()).T.rename(index={0: 
'null values (nb)'}))\n",
        "    tab_info = 
tab_info.append(pd.DataFrame(df_initial.isnull().sum() / df_initial.shape[0] 
* 100).T.\n",
        "                               rename(index={0: 'null values 
(%)'}))\n",
        "    return tab_info"
      ],
      "metadata": {
        "id": "s2xUK3XnNIV5"
      },
      "execution_count": 11,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "DfInfo(df)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 143
        },
        "id": "-OWXz3WXNISY",
        "outputId": "f11baec3-52a7-4ef9-8937-b0e2765c86c7"
      },
      "execution_count": 12,
      "outputs": [
        {
```

```
 "output_type": "execute_result",
          "data": {
            "text/plain": [
              "                 Closing Value\n",
              "column type            float64\n",
              "null values (nb)             0\n",
              "null values (%)            0.0"
            ],
            "text/html": [
              "\n",
              "  <div id=\"df-4e32bf9f-80e7-48c2-a7c6-abf9974fa83d\">\n",
              "    <div class=\"colab-df-container\">\n",
              "      <div>\n",
              "<style scoped>\n",
              "    .dataframe tbody tr th:only-of-type {\n",
              "        vertical-align: middle;\n",
              "    }\n",
              "\n",
              "    .dataframe tbody tr th {\n",
              "        vertical-align: top;\n",
              "    }\n",
              "\n",
              "    .dataframe thead th {\n",
              "        text-align: right;\n",
              "    }\n",
              "</style>\n",
              "<table border=\"1\" class=\"dataframe\">\n",
              "  <thead>\n",
              "    <tr style=\"text-align: right;\">\n",
              "      <th></th>\n",
              "      <th>Closing Value</th>\n",
              "    </tr>\n",
              "  </thead>\n",
              "  <tbody>\n",
              "    <tr>\n",
              "      <th>column type</th>\n",
              "      <td>float64</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>null values (nb)</th>\n",
              "      <td>0</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>null values (%)</th>\n",
              "      <td>0.0</td>\n",
              "    </tr>\n",
              "  </tbody>\n",
              "</table>\n",
              "</div>\n",
              "      <button class=\"colab-df-convert\"
onclick=\"convertToInteractive('df-4e32bf9f-80e7-48c2-a7c6-
abf9974fa83d')\"\n",
              "              title=\"Convert this dataframe to an
interactive table.\"\n",
              "              style=\"display:none;\">\n",
              "        \n",
              "  <svg xmlns=\"http://www.w3.org/2000/svg\"
height=\"24px\"viewBox=\"0 0 24 24\"\n",
              "       width=\"24px\">\n",
              "    <path d=\"M0 0h24v24H0V0z\" fill=\"none\"/>\n",
              "    <path d=\"M18.56 5.44l.94 2.06.94-2.06 2.06-.94-2.06-.94-
.94-2.06-.94 2.06-2.06.94zm-11 1L8.5 8.5l.94-2.06 2.06-.94-2.06-.94L8.5
2.5l-.94 2.06-2.06.94zm10 10l.94 2.06.94-2.06 2.06-.94-2.06-.94-.94-2.06-.94
2.06-2.06.94z\"/><path d=\"M17.41 7.96l-1.37-1.37c-.4-.4-.92-.59-1.43-.59-
.52 0-1.04.2-1.43.59L10.3 9.45l-7.72 7.72c-.78.78-.78 2.05 0 2.83L4
21.41c.39.39.9.59 1.41.59.51 0 1.02-.2 1.41-.59l7.78-7.78 2.81-2.81c.8-
.78.8-2.07 0-2.86zM5.41 20L4 18.59l7.72-7.72 1.47 1.35L5.41 20z\"/>\n",
              "  </svg>\n",
              "      </button>\n",
              "      \n",
              "  <style>\n",
              "    .colab-df-container {\n",
              "      display:flex;\n",
              "      flex-wrap:wrap;\n",
              "      gap: 12px;\n",
              "    }\n",
              "\n",
              "    .colab-df-convert {\n",
              "      background-color: #E8F0FE;\n",
              "      border: none;\n",
              "      border-radius: 50%;\n",
              "      cursor: pointer;\n",
              "      display: none;\n",
              "      fill: #1967D2;\n",
              "      height: 32px;\n",
              "      padding: 0 0 0 0;\n",
              "      width: 32px;\n",
              "    }\n",
              "\n",
              "    .colab-df-convert:hover {\n",
              "      background-color: #E2EBFA;\n",
              "      box-shadow: 0px 1px 2px rgba(60, 64, 67, 0.3), 0px 1px
3px 1px rgba(60, 64, 67, 0.15);\n",
              "      fill: #174EA6;\n",
              "    }\n",
              "\n",
              "    [theme=dark] .colab-df-convert {\n",
              "      background-color: #3B4455;\n",
              "      fill: #D2E3FC;\n",
              "    }\n",
              "\n",
              "    [theme=dark] .colab-df-convert:hover {\n",
              "      background-color: #434B5C;\n",
              "      box-shadow: 0px 1px 3px 1px rgba(0, 0, 0, 0.15);\n",
              "      filter: drop-shadow(0px 1px 2px rgba(0, 0, 0,
0.3));\n",
              "      fill: #FFFFFF;\n",
              "    }\n",
              "  </style>\n",
              "\n",
              "      <script>\n",
              "        const buttonEl =\n",
              "          document.querySelector('#df-4e32bf9f-80e7-48c2-
a7c6-abf9974fa83d button.colab-df-convert');\n",
              "        buttonEl.style.display =\n",
              "          google.colab.kernel.accessAllowed ? 'block' :
'none';\n",
              "\n",
              "        async function convertToInteractive(key) {\n",
              "          const element = document.querySelector('#df-
4e32bf9f-80e7-48c2-a7c6-abf9974fa83d');\n",
              "          const dataTable =\n",
              "            await
google.colab.kernel.invokeFunction('convertToInteractive',\n",
              "                                                     [key],
{});\n",
              "          if (!dataTable) return;\n",
              "\n",
              "          const docLinkHtml = 'Like what you see? Visit the ' 
+\n",
              "            '<a target=\"_blank\"
href=https://colab.research.google.com/notebooks/data_table.ipynb>data table
notebook</a>'\n",
              "            + ' to learn more about interactive tables.';\n",
              "          element.innerHTML = '';\n",
              "          dataTable['output_type'] = 'display_data';\n",
              "          await google.colab.output.renderOutput(dataTable,
element);\n",
              "          const docLink = document.createElement('div');\n",
              "          docLink.innerHTML = docLinkHtml;\n",
              "          element.appendChild(docLink);\n",
              "        }\n",
              "      </script>\n",
              "    </div>\n",
              "  </div>\n",
              "  "
            ]
          },
          "metadata": {},
          "execution_count": 12
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "df.index"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "za-gykqnNIPS",
        "outputId": "8bf3abdb-be6f-467b-eab3-9fd33c2e569a"
      },
      "execution_count": 13,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "DatetimeIndex(['2000-01-04', '2000-01-05', '2000-01-06',
'2000-01-07',\n",
              "               '2000-01-10', '2000-01-11', '2000-01-12',
'2000-01-13',\n",
              "               '2000-01-14', '2000-01-18',\n",
              "               ...\n",
              "               '2018-06-26', '2018-06-27', '2018-06-28',
'2018-06-29',\n",
              "               '2018-07-02', '2018-07-03', '2018-07-04',
'2018-07-05',\n",
              "
'2018-07-06', '2018-07-09'],\n",
              "              dtype='datetime64[ns]', name='Date',
length=4673, freq=None)"
            ]
          },
          "metadata": {},
          "execution_count": 13
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "y = df['Closing Value'].resample('MS').mean()"
      ],
      "metadata": {
        "id": "mtd2hfeCNIKp"
      },
      "execution_count": 14,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "y.plot(figsize=(15, 6))\n",
        "plt.show()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 416
        },
        "id": "bU9TeHtgNH4R",
        "outputId": "5968d51a-c472-49ba-d7a0-1c28b1070668"
      },
      "execution_count": 15,
      "outputs": [
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 1080x432 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAA9MAAAGPCAYAAABf3oPgAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nOzdd2Ac9Z028Ge2r3qXbFnucu/GgAs
9xhhIAGMuF4ghl0be5F7nWi7H8YYQyOVIciGJcwe5kEvOoQVwTEgAAwkEm2IbV9xtuUu2et2mbTP
vH7K0O79ZSbvSrjS783z+8jZZp5Vmt9jvfJnV0dCggIiIiIioriZRvsJEBERERERERERERERERERaUbBtNERER
ERERECWIwTURERERERJQgBtNEREREREREREREREREREECWIwTURERERERJQgBtNEREREREREREREREREREECWIwTURERERERJQ
gBtNJVFNTM9pPgWLgcdEfHhN94nHRJx4X/eEx0SceF/3hMdEnHpfkYTBNRERERERERERERERERERElCAG00RERER
EREQJYjBNRERERERERElCAG00RERERERERERERERQJYjBNRERERERERElCA
G00RERERERERQJYjBNRERERERElCAG00RERERERERERQJYjBNRERERERElCDLaD8BIiIiPTvrCuGnB12
QIGHlODuur3TAbpZG+2kRERHRKGMwTURE1A9ZUXD3n1txpCMEAPjVcQ/BJuHu/Amk1dFkl
iYE1ERGRELPMmIiLqx+muUF8g3asrqOC3p3z4qz+34kfpmRERE dFoYzBNRETUj8PtoQFvf/G
UD3Xuge9DRERERMmYnBNBERUT8OtwdVl2MVdJ93h0fmyRAREZGuMJgmIiLqx+E2dTD90+UFuG6sXXV
dc7c8kk+JiIiIdILBNBERUT/EzPS8Iq/Ig3Xx5Ba/GIqxaUmqzZHgbGM1R+ZQgfdHZ9lh
R6lQH040+ZqaJiIiMiME0ERFRDEeFrPTPUPAucFgnlTvWfzmZmpomIiAyJwTQREVEMh9vuvUU1o
BQJOZbmJmmoiIyJAYTBMREcUg9kvPLuoJpsuEzqT4jdaXmGZzmbpvNxaWXfp7ZW/aZmsqHdTRd
EZEgMpomIiASKomjzWl4VkQFnycNxRRTBKqsnsy0jkWCU6
```

```
z1HebPwzVfYmIiMgYGEwTEREJYmWlJakngJYkSdM3zYneREREExsNgmoiISHC4PfYk715iMM1d00R
ERMbDYJqIiEigyUwLwbS4HquZwTQREZHhMJgmIiISaNdiWVSXy7kei4iIyPAYTBMREUXxhRSc7FK
Xec8cJDPdxMw0ERGR4TCYJiIiinK8Iwg5ajj3xFwzcq3qP5dlDiEz3c3MNBERkdEwmCYiIopySCz
xFrLSAFDGzDQREZHhMZgmIiKKEmstlkic5s2eaSIiIuNhME1ERBRlsLVYADPTREREExGCaiIioj6I
oOCRkpufEDKbVfz6bu8NQFEVzPyIiIspcDKaJiIguafTJaPNHssxZFgkTc82a++VYTciySH2X/WG
gM8BgmoiIyEgYTBMREV1yRBg+NrPAArNJinnfWNlpIiIiMg4G00RERJecd6sD4qn5ln7vW+Zg3zQ
REZGRMZgmIiK6pNatHj5WldN/MF3Kid5ERESGxmCaiIjoklqPOiAen6Ptl+5VzoneREREhsZgmoi
I6JJaocx7XHb/wbSYmW5mME1ERGQoDKaJiIguqRMy01UDZKbFAWSNLPMmIiIyFAbTREREAEKygot
CMF05QGa6TCzz7mZmmoiIyEgYTBMREQFo8IYRjloVXeIwIcvS/5/JModY5s3MNBERkZEwmCYiIoK
2xHugfmkgRmaaPdNERESGwmCaiIgIiQ0fA2KvxlIUpZ97ExERUaZhME1ERITEho8BQI7VhGyL1Hc
5IAOdAQbTREREREsFgmoiICDEy0zmWQR8TKztNREREExsBgmoiICECdJ6S6XDVImTcAlHOiNxERkWE
Netr9iSeewNtvv426ujo8+uijuPHGG/uu37p1K8zmng8S99xzD2655Za+xz377LPYtGkTAGDBggV
44IEHYLPZUvEzEBERDZuYmR6szBsASoWJ3k1eZqaJiIiMYtDM9IoVK7BhwwYsXLhQdf38+fOxceN
GPPfcc/jRj36EH//4xzhz5gwA4MCBA9i0aRM2btyIzZs3w+124/nnn0/NT0BERDRMiqIMKZjmrmk
iIiLjGjSYnjdvHiorKXXL1++HA6HAwZswYlJSUoKGhAQDw5ptvYvXq1cjLy4MkSbjrrrvwxht
vJPmpExERJUdnQIE7FBke5jADxfbBO6HKnNw1TUREZFSDT1eJw44dO9O9DZ2Yl58+YBAaaGjB9+vS
+2ysrK/sC7f7U1NQk46mMukz5OTINj4v+8Jjok1GPywm3BMDZd7nMJuPkyZODP9BlARBpYapp7EB
NTVPSn59Rj4ue8ZjoE4+L/vCY6BOPS3yqq6sHvH3YwfThw4fx6KOP4rHHHkN2dvaQv85tTzQd1NT
UZMTPkWl4XPSHx0SfjHxcTp73AWjruzyl0Inq6qpBHzfb5gNORR7nt2ajunpCUp+bkY+LXvGY6BO
Pi/7wmOgTj0vyDGua98GDB/HNb34Tjz76Kb6nKObPn993fUVhSoTfeHCBVRUVAznWxEREaWMZi1WHJO
8AaBcXI3FnmkiIiLDGHIw/fHHH+OBBx7/7XSatEh126pVQmxZ0dXVBURS89NJLfVPAiYi
I9KbOk/jwMSDGADIvg2kiIiKjGLTMe8OGDXrrrbfQ3NyMz28cK3rbfQ3t6OkydPYsNeeeQYr
77xe/+EVcd911mDdvHtasWYP77rsPiqqJg/vz5u4wFEWBJE1Je25
EREEkT4MG0+vXr8f69etj1t+3bt3AD/uVzfbPZYwMVr+Dzb9aFTM6aZNgZmDdXmsnHkV3odbNLvk2WH1wGeVyEUk6c
4QO6ZDF5gZzBNRESU6YbVM01ERJQhrJjulep0DfdyPVYREREhsBgmoiIDOu8EPSOS3CSy/
fTJeLfdM+9k0TEREZAYNpbHqvOpNckWWCZxGMFVzIytHqvOpNckWWCzRx/mbaYmW5iZpqIiMgQGEwTEZGhnR/i8
m+qKHwTQREZERMJgmIiJDq3MPbfhY5P7YNBRHhxEHBNRESGVusZXZZ/mcmPxGRodUNY5J3z/3VmexaIdNNRERERmYnBNBERGdqwTQRERmaNjOdWM90qdM
Ee1Q83RVQ0BngeiwiIqJMx2CaiIgMS1EU1HnUZdmJZqZNkqR5DPumiYiIMh+DaSIiMqyWbhndUXUF
vrlVCvi3+HdO92DdNRERkPAymiYjIsOqE/uaqbDMkaQjBNDPTREREhsNgmoiIDOu8EPSOS3CSdy/
NeiwOISMiIsp4DKaJiMiwNJnpBIePgyH4NpIiIyrJqOoOpyosPHemky0+yZJiIiyng
MpomIyLD2tKiD6TlF/iF9He6aJiIiMh4G00RERJEjekIwj7epgenHJ0ILpsdlmmKLmljX5ZPhCynC
eHhEREekcg2kiIjKk/S1BhKPi3Um5ZhQ5hlbmbTVJGJulfqy4v5qIiIgyC4NpiIypD0tAdXly0p
tw/p62r5plnoTERFlMgbTRERkSHuahRLv4QbT3DVNRERkKAymiYjIkY9MzNNIyMkK1AU9tATEWU
aBtNERGQ4jd6wase01QTMKRza8LFe44Vd0+fZM00AnjrqRvVv67Hs90041BYc/AFERJQ2GEwTEZH
hiP3Sc4uscFikfu4dH2amSXTRE8aDH3Wi3a/gaEcID+/uHO2nRERESScRgmoiIDGePUOI93H5pgME
0ab3f4EdAjlze3hiAzHJvIqKMwWCaiIgMZ7cwfGy4/dIAMC5bXeZd7w0jJDNwMrKPmtQnbTwhBed
5koWIKGMwmCYiIkORFQX7hDLvxSXD65cGAKdFQkj8mc1rAAXvQycjGyHEEwDYN80EVEGYTBNRES
GUtMZQlcwkjYtMTZ4BHxY6k39eoKyDjSrg2cY11HRETpicE0EREZirgSa3GJEDxHYv6jIHMj1MSTvR0lJKcujY
MU689zQHEqvI/0s4p70REmYLBNBERGcreFnmVmMBnDx3pVCX3T590MnIxqZ4wSb4CZaSKiTJKcujY
iIqI0oclMlyQxmGZmmi4Rh4/1OtUVQndIGfYqNhpdnqCMh3d3YV9rAOOyLZhbZMWcIivmFlkxNts
8+BcgoozAYJqIiAzDF1JwuE3TA9/+FgvTTDtYTBtRGFZwa7m2MF0WAGOdwYxvzh5J3FSQVEUPF3
jxUdNAdw5yYnrKh2j/ZR0Q1YUfH5rO96s7QbQsx3g92d9fbdfPcaOZ28oQq6VBaBEmY6/5UREZBg
HWgMIRfWxTsw1o8SRvCxSVY76HHUty7wN6WhHCK5g/2vRxBM6evRMjRfrP+jAMzVe3PWnVk1Fh5E
9tt/VF0jHsq3ejycOu0fwGQ3sgwY/1n/Qjl8edUPhnnOipGIwTUREhrG7Jfn7paVdNVCeWddZ4wZH5
4NZydTf4Bb9f7EDJFUfCfhyLBYEgBvrWrk4EYgFfP+fCD/a5B7/fWAMH2SKrpDOLOt1rwmxNe/NO
OTvzb3sGfOxHFj8E0EREZxt4U9ksDQIHdxDpXBfWHwaafXJSvwfpnzh8bG6RupVA70PIdjcHcbx
THfBvbwzgrbqBTxJkuuMdQXxlW7vquhKHCd+/Ih+fm5alun5faxAd/tH/3X/ysAfdUd0mjx90YWe
jsY8jUTIxmCYiIsM4LAQxi0qS7tg1vZ6M6mP789GzVZb0H00/XeGJe/53dnQjH2vdlAJ0
BGfe83QZ3VJ+IRQI2XleE+2fl4CfLCzGrINLmISvA+w2jG7R2BmS8cMqruk5WgPvfa4c7OPqBPlE
mYDBNRESGEJYVnO5SfCJK+G0GbEUu+CNGyXeX93IE6GoLk0H0K1q1b+7yjqNYdvAL0ZPl+r+GJ6S3p2tTcf0PVpBZOGQbWqxcvCY/Sa1QZ8m0aTLsFT8PH1ILOeZGsn+xzFV2H1IcrpMtnWPUfVeqeaVwkU7wXXaMHZK
edqQjhBf7uS3TfWd3F04K7x/fuzwfyyvsfZevGWtX3b714ugG08+f9MIT0p78OOsK41u7OkfhGRF
lHgbTRERkCLWeMAJRYpbzDHJHhbFUpkkaMTsEeq+5JDPtjrrndoekfSlkfsrTkD3a6NHGq3y21+6
udA9weyZSFAW/O6PO8N49NQtfmqmuaSCPp5EIHDzBKJgcnKI0abRnAuBnijPGxBQKTXIUb5DbGqYhnrfqdxBw0xUH5TXE1HgSfQ6wWwRYBKz+mQ05o2nq+WkvxM74Abe0EUPdhN
EhlAj9IBOzU/NdkjumjY2sV/68rKevvxZherXm9hyoBfP1KiDxjsnORG94anOE8b/HO8/SMtEF70
yOgOREwg5FgmPLy2AJKl3hS+rsCF6fXhNZwgXRqnN492LflUm3SL1bC+I9n8/aEerTiskiNIFg2k
iIjIETTCdl5pguiJL/YG1TQdDiGjkfCRM8r6irKf0d3ah/oeQ1XQGsV3o935gYa6m5/tHH7vQGTD
O6/qocKxmFFrgsEia++VaTZoNAVsvjk729xdCVvr2SU78zzVFMEc97UafjId3d43wMyPKLAymiYj
IEE4KwXR1ijLTuVb1h+wuAwUdRucLKdjfqg68rujLTOs/mH5WyEovLbdhar4V31iQfqdt/ll/K+
BstNHO9THauYAsxbEvunRKPU+5wpp9mB/aUY2Fpfa8I/zc1XXv3LWx/V9RMPAYJqIiAxBHB6Uqj
```

```
vPJv6T6sryA+qRrG/NYDoIckTcsx9lQpiZvpoe0hXQUxIVvD8SXUwva66Z91TicOMr83OUd32oZD
BzmRHhf72mYX9B9PXxhhCNtL7uX91zIPo7zi3yNrXbvCN+bkotEdOjHQFFZzo1Gf/PlE6YDBNRES
GcLJTnV1KVZm3JjPNFTSGcahN/RrrDWAAYEyWCQW2yGvDE1JwXkf99G/VdaMxaid6rlXCbROdfZd
vHq8erqXHzHqqiJlpsf892mWlNuRElYA3+mQc60htsFrrDuGFU178/Igb/76vCxtPqKsGvjQzu6+
/22qScFmJuhR9V5NxTowQJRuDaSIiynjuoIyL3kigYJaAibnMTFNyXRSGTU2JOmEjSZKm1Ptwmz4
C0qCs4CcH3Krr1kxyIjtq8tj0Aquq37bWHTZEC4OsKDgmZqYHKPO2miQsq1AHq++maEWWrCh4ZE8
n5m9qxP3b2vEvOzvx/f0udEQNSyuwSVg72al63GKhr3tPM4NpoqFiME1ERBlP7JeemGuGzawdIJQ
MYmbaZYCAg3pc8KqD6bHCmjSx1FsvE70f2tWJj4SA6rPV6qFjdrOkqeYQB3NlonOuMHzhSHBaZDe
hzDnwx+drhBVZW1PQN+0OyvjsO214/IAb8gDn69ZNy0aWRf18xSFpu1sy/zgSpQqDaSIiyninNP3
S/WeWhivHqs1Mj3TPJI2OeiEzPVaY7K4dQjb6vaovnfLiySPqsuAbKu24rFT7O6LH559qYjn7jAK
LZiWW6Nox6r7pDxr8CA4U8SbovDuEVa814/XzA08Kr3Ca8LdCrzugzUwfaQ/CzXYUoiFJTY0bERG
RjozUWiygJ4NnNwP+S3FVWAG8IQXZ1tRkwkk/6r3qgGSMJphWv+5Gu+/4UFsQ6z/oUF03LtuM/76
6MGbAOKvQgpfPRi6P9vMfCUeFfmfxhEIsswotKHWY0Nzd83pwBRXsbQ7ginL7II/U+uM5H5447Ea
zT4ZJ6mlRueANoyugDs7LnCZ8aoITBXYTCu0mlDtN+ESlAwV2bd6s0G7C1DxL31BGWQH2twaxoiL
x50dkdIN+mnjiiSfw9ttvo66uDo8++ihuvPFGAEAgEMBjjz2Gffv2AQDWrl2Le+65p+9xzz77LDZ
t2gQAWLBgAR544AHYbDbtNyAiIkoxcZJ3qtZi9cqzmtAcjgRWrqCC7NQlw0kHFEXBxUHKvMUp0Ce
7QugKyJo++5HQ4Zfx2XdaVSXMdjPw9PVFKHGYYz5G0/NtgGD6mLgWa4DhY70kScI1Y+3YdNrXd92
79f6Eg+nz7hC+tLUN3YPMqZtbZMVzNxShKif+97XFpVbV++Ke5gCDaaIhGPTde8WKFdiwYQMWLly
ouv65556D1+vF5s2bsXHjRrz44os4ePAgAODAgQPYtGkTNm7ciM2bN8PtduP5559PzU9AREQ0CLF
nOlVrsXpp+qZZQpnxOgMKvKFIYOowQzW9G+gZThd9IkdWgDdqBy7VTYWgrOALW9tw1qWO0v7jygI
sLOk/8TG7SLsrO9NbGMTs+0DDx6JdI5R6bxtC3/Sbtd2DBtKfnODAGzeXJBRIA8ASsW+aQ8iIhmT
QYHrevHmorKzUXP/mm29i7dq1kCQJeXl5WL16Nd54442+21avXXo28vDxIkoS77rqr7zYiIqKRpCi
KNphOYZk3AOSKE70DmR1wEFAvZqWzzDFLpW8VVky9ctanuU8qKYqCv/uwA29fUAd3fzM9C+umZff
zqB7jc8zIjlr71BFQNKXtmSQoK5oWkYF2TEe7Wgim9zQHEu6b/qCh/wDXLAH/vCAXG68rUk1dj5d
mCBmDaaIhGfKniYaGBowZM6bv8tixY7F169a+26ZPn953W2VlJRoaGgb8ejU1NUN9KrqSK9HpuF
x0R8eE33KxOPS7JfgDkVWw2SbFXTVnYYrhS3MlpAdQKRU9ujZWuR2DD3oyMTjku7EY7Kr3QQgEig
XSIGYx22hWQIQeT3+qdaH/UdrkD1CU2x+fs6KZ2vVAeGc3DC+WNyCmpqWQR8/yWnHIVfktf2nw2e
xrFA/AXUyf1dOeyUE5cixKrHJaDl/CoP/LwGKApTaHGgO9AS63WHgtf2nMTs3vv8rRQG2XXACiLx
RPTbDjwlOGbIClNgUFNm8OHWyMZEfqY9dBuwmJ/xyz9ev98p4/9BJlNuTf+KP71/6xOMSn+rq6gF
v180AssGeaDqoqanJiJ8j0/C46A+PiT5l6nGpr/cDUR9/pxXaMG3auJR+z7JzrUBnpHw3r2wsqic
4B3hE/zL1uKSzWMdkxwkPgMgwryklOaiuHq957FRFwfiTjTjv7slkBxQJp+2VWDM5K6XPGQB+fcy
D/6lVDxybkGPG726uQHlW7D5p0aKmdhxyefsudzrLUF2dm9TnOVTJ/l05dMYLoL3v8twSJ6qrq+J
+/PILbfh9VOXBRXs5bq/WTteO5URHEG3Bpr7L2RYJX7hiEqym5J0FXHCyGTubIhnp1pxKrJg4tPe
p/vD9S594XJJnyBMvysvLUV9f33f54sWLqKioAABUVFSoMtEXLLzou42IiGgknRLKNKtTXOINALl
Cr2wXd01nvFhl3rFIkoTbhIDllXOpL/V+/bwP/7hDHUgX2U343Y3FcQfSgLGGkImrv+IZPhbt8jJ
1KfXOpvj7psUS7yvKbEkNpAGWehMlw5CD6VWrVmHTpk1QFAVdXV3YsmULVq1a1Xfbli1b0NnZCVmW
R8NJLL/VNASciIhpJNV3qD/tTUjx8DAyY+yapswmBtNjsvsPUG8Xguk/1fnhSeGQulfP+fA377Y
humXXaZbwwieKE965bqRd05pJ3nEOH+t1pRBMf9QUiHtg2weN6sB7eQombYu7xBlMEyVu0GB6w4Y
NuPXWW3Hw4EH84Ac/wK233orTp0/j7rvvhsPhwJo1a3Dvvvfdi7dq1mDdvHoCeoWVr1qzBfffdhzv
uuANZWVm4++67U/7DEBEBERicThY6ORmXYxM53xLnqEYHqAbO+iEivGRQXb3pCCP19IfNpzPJ6p8eD
ev7T17T0HAJME/OraQiwpS3xl6WwhO3uiM4hQgoO10sXR9sR3TEebW2yF0xx5L6j3yqj1DDKeGz1
D4j5oEIPp5K+XXSxkpve3ZO6xJEqVQT9RrF+/HuvXr49528MPP+MPP9zv9v9atW4d169YhUxt+QAoxb4/ZRG9XVLlcxW4oadi1ezi
NN9VrsQAgl5lpw7koTLWuHCAzLUkSPjXRgScOe/que+WsXEDbhrsZtABPH+nHlAc9ALHD+3u0lz/+NI
CrB4/tN7YYocZ5U4TGn09P68/DJzqCmF6gllbvfOFFJx2qd87phck9t5hNUlYWGLFh42RjO/OxgD
GX1pj1egN4wtb23CqK4S/nZOLr83u6ac+6wqrpqQ7zRIWDbCybKiqstXH0hdWcLg9iPnFyf9eRJl
qyGXeREREehcIKzjnVmeCpoxEZpp7pg1HU+Y9SB/ybcJAujdru+ELKTjWEcRtb7Rg7NMXsfatFnS
HhnYi5uHdnZpA2iQBG5YX4HRM4QVVFeWa0u9e31jRwfebwi
g3ivjwY868f6lbPT7QlZ6SZlt2CdWYpEkSZOd3tOceceSKJUYTBMRUcY64wqPhBXZpmhUeK
e6S7umc5o/rCClu7ICROTBJQ7B36dLSmzYUxW5D6ekIKvvteOa/7QhK31foQ4n/HimxjPAV4n
tvXo/fnLQrbrOZgI2XleEewfZJR0P7RCyzOubFku8490vLRKHkO24FExf8ITx2vlu1W3f29s1YiX
evTiEjGh4GEwTEVHGGo0SbwDIY2baUMSdJdnDNkoRPCtnpl8/6VV8RamYeVaXHEhWzzHbJMONc
Bc60SNt1Yovl+QzVL6JvOxMz0UeFnmpVgiXevy4Vg9XB7EK6gjKdPeBAWzrF92BjAtvoAPmhUB7S
pGD7Wi8E00fAwmCYiooylGT42QmkfMjVvOL3zMgFzZVvTH8mqu3mb3mtYzgOvpbaBrnZZEuiGfqB
ySm3Pa4IiuWM12JZ33FXt/Hlxbg6jHJC8h
mi5nptgwMpsVJ3kPMTBc5zJgW9b4jKz2l3k+f8Ma8/zd2dKA2qjXFZgIWp6BfutfCEiuiT/uc6Ax
phukRUf8YTBMRUcY6KQQiI9EvDWh7prlnOrPVJzDJO9qVZbZBy8HPuIYQTAuv+7nFyR0ONr3Aiuj
E+zl3OOOqL452JKfMG9CWev/b3i5c8MYOWE8IJwAXl9rgtCS/X7pXrtWEuUXqn+2/j7j7uTcRiRh
MExFRxhqtzHQep3kbJ2Okt13Scq7YieNdXok+FOIFD1hxXURQX3EoCJOcl93Ts
tEibnqr/mMR31Tdd0BvHUUTeerfEgnOCqJ0VR8PgBlyo7bJKGt1LvCiGY3iuU7g8UKqeyxLvXfdP
Vr8FfH/fwBCBRnBhMExFR2lAUBYoS34dGsJ2E9zSGo0LiIuUSXs8bacqHi64
Hk1cV4qWVxRifo378WVf8Zbfn3cLQvWwzHCnIbOqtb7rZF8ZvL1pw3R+bsGRzE76xoxNfe78DX9j
```

aHvd7RndIwf3b2vHIHvUU9BkFlmH9H4rBtOhHSwv6vW1FCoeP9bp7ajaK7ZGQoCuoYOOJxAffERk
Rg2kiIkoLde4QPvFqM6qeqcfDuzsH/YD8/X0udEZN0c61SqhKIMgZDnHPdFcg/pMAlH7qPeqTJfG
WeQOA3SzhkSX5eHFlCT4zNQuSJGGykAUVy7YHcrpLHXhPyk3Na1470XtowXSzL4y/XOjGqc6hZ7b
/bW8XZr7QgB+dtmGfkPX9/VkfficMZIul0RvGrW8048XT6vuaJeBfFuQN+bkBPRUxhfbYwfiCYis
+PyMbN1U5NLdZJGBJaeqDaadFwpdnqae8P3nYjYA4IY2INBhMExFRWnj8gBt7WoJwhxT85KBbs1Y
m2sG2IP7zsLrvb920LJgHmbCcLHazBHtUDBNWAB8/mGasiwnumB7MJKGEOpG+aTHwFgPzZBnqrun
ukIJXzvrwje0dWPpyI6p/24A73mrFkpcb8crZwYNe0YHWAH74sQsDreP+5o5OtHb3n91v98u48bV
m7BZ2LBfYJGy+sRifimNQ3EAkScLlZbHLtT8/oyeI/ZcFuZrbFpXYRmSVHwB8cUY2nFH9BRe9clw
nIYiMjsE0ERGlhR2N6t2rP9jvipntDcsKvv5Bu2rtTGWWGQ8sHF52KVFidtrFXdMZSwymK7OH9/F
qkhAAJzLRW5zknbpgOvHseYdfxtLfN+K+v7ThqWMe1ZAvWQG+v69rgEfHtvWiX3OdBKj6zlv9Mv7
1o85+v8bDuztxzq0+htX5FlKvK8kAACAASURBVLx9axmuGavNGA9FrFLvXKuENZN6AvUFJTbcMl7
9vVaMSX1Wulexw6zp3//ZwdjvsUQUwWCaiIh0LygrqBE+rB9oC+LNOm12+pfHPJoBPz9cmq8JblN
NnOjNvunMJCsKGpKemVY//nQCPdNi4C1muZOlKseiGpxV75UHLQt+usaDMwP8LEc6QjiX4PTyj4S
9yHdPzcLBu8o1md4XTvnw5xjvFx81+bFRWFN1Q6Udf7qlFFOSOGMhVjD96SlZyIl6X3rksnyUOHo
u59skfGlmTtK+fzy+NidHNaX9SEcIf76gPVlBRBEMpomISPdOdYUQKxYVs9N17hAeFYYHfXKCAze
PH16Z5lBoMtOc6J2RWrpl1WszzyYNuzRXzCbrsczbbpZQkRX5ORUAFwbZT/x+/eCB2Ru1/bdviBR
Fwa4mdTD9pZnZGJdjwdfn5mqy53+/vUM1GT0kK/iH7eqMdXW+Bc/dUIwCe3I/Ii8ssUJ8WXxuurp
PeUq+BR/cVobffqIIH6+tGPZJmURNzLXgtgnq98oNB10j+hyI0g2DaSIi0r2j/fRj7m0J9mVOgrK
Cf9jeAXdU82SeVcL3r+h/Um4qiRO9u1jmnZEuCgFkZRICoAnCKqs6dxj+OHrug7KiKVdO1QAyABg
vPM/z7v6D/rCsYLsQ+D6xogDfFDLIWxIIpus8YTT4IsGx3aRgzqWdyTazhJ8tL1RlWmvdYfzT9g7
4Lr1H/OKoB4fa1O8t/3FlAezm5M9WyLKY8Pmo4PmvJjv7nmu08iwzbqpyJj2Yj9f6ueps+HsNAc3
/ERFFMJgmIiLdi+6tFH1/fxfa/TLWvNmCt+rUma+HFucltKYomcRd010s885I4lqsZGQTnRZJFZQ
rGDhQ7VXrDqtmBVQ4TSkdYFUlrPASA/loh9qDqhNKRXYT/npqFu6arM6Evl/vR2ecO453CyXeM3N
kWKOi58WlNnxFmFL921M+LP19I56r8eB7e9VVLHdNduKasanb6/y9y/Px/A1FeOb6ImxYXpiy7zM
cC0tsmnVcHzSw1JuoPwymiYhI9/rLTAPA7uYgrny5Ee81qD9YLym19k3KHQ2aXdNxBgiUXuq9wlq
sJJ28mZgn9E13Dd43LZZ4i4PMkk3ch107QDD9gfD7ubTcBpMkYWq+FVOjnmdIAd65EF92+iMh0z0
3V/s79uDCPEyIsbf7q+8LVSw2Cd9dkh/X9x0qs0nC6vFO3DrBmZLd38mycpx6ENpo7xAn0jMG00R
EpHvHhMy0+OG40Sdrbn/qmiKYpNH7wCpmptkznZnEMu9k9blOHsJ6rJHql+6VSJn3h0J2c3lFJAO
8WphivWWAtXfRxMz0nBjBdLbVhN9cX4Qy58Afeb+1KA/lI9yjrFdDXXtGZEQMpomISNe6QwpOCUH
Cz1b0XyK5tNyGtz9ZiokpmmIcL+00bwbTmUizFitJAZmYVY5n9ZRmLVaKfwfEMu/+MtOyouDDRnX
gu6w8Ukq8ukodTL9V142QPPDviz+s4ONWdZA3Ly/2959fbMNHd5TjyzOzEWvV/IJiq6qf2ehmFqh
fN0fbQ1yRRdQPBtNERKRrNV0hRH+ursox4+oxdqys1PY2/vUUJ36/qgQljtHPMOXaxD3TLPPORJq
e6WHumO4lBsJn48hMi2uxJuel9vdALPM+308wfbwjhDZ/5PWfZ5MwN2r41uVlNhTaI1FuR0DBDqG
EW3SgNYjoX6mqHDNKBljLXGA34QdXFuDdT5aq1lQ5zMCPlxXAHCvKNqjKbDPyotpU3CGl32NLZHQ
MpomISNfEfulZl7ImD12Wj+yovsNvL87Dk1cVpmQS71CImWkOIMtMqSrznjSUnmmXOMk7tZnpcdn
qr3/RG0YwRkZZHGC1tMymCl4tJgk3jkus1FvcL72kdIBIOsq8Yhu23FyCp68vwj/Oy8Hrq0uxcKA
o3IAkScJslnoTxYXBNBER6dqxDvWHuBkFPR/y5hZZ8adbS/G9y/Px3m1l+Pt5uZBGsUdalCdmpln
mnZHEzHRlkgaQiYHwOXcI4QFKn8Oyoslep3oAmdMiqXqRZUV7cgGAtsS7QltVsrpKPdV7y3nfgKX
Fu5uGFkwDgEmS8MkJTnxrcT4WJfA4I9H2Tce/65zISBhMExGRrokf4mZGfcibVWjFV2fnqEpG9UL
TM80y74zjCsroijpJYjMBxUnaD5xnM6HEEflaQblnr3J/6jxhRBc/lDhMyLel/mNeVfbApd6Komg
y08tjBNPXV9oRPbPvtCuMms7+A7hdYma6jEFxMs0qVJ+IYWaaKDYG00REpGvazPToDhaLV65mzzQ
z05mmXghuK7LMSa2OmJQrrnTqP7gUp32nevhYL3Gid60w0ft0V1g1bT/bImF+sfbkV57NhKuEIHt
LbexS74uesOrEgt0MzNPhCbV0NrNA/f850HpCIiNLj08kRERkSJ6gjLNRfaASgGlpE0xzmne6C4Q
VvHOxG2e6wgjJCkIKEJIVBDotuHdMKGUl3r0m5VmwqzkSxJzuCuOasbHvK/ZUi3uqU2WwIWQfNKq
z0peX2WDtZ9jXTVUOvHMxcv8t57vx9bm5mvuJWen5RTbYdDIrIVOIZd4nOkMIhBX+PxMJ0uMTCRE
RGdIJocxzUq4ZWZb0KKrS9EyzzDutKIqCL2xtwx/PxcqO2vAfpxs1meNkDR/rJfZND7RrWrNjeoQ
y05r1WEK2Xizxjl6JJbppvAP/vLOz7/KOpgD2Ngc0fc3ifmmWeCdfgd2EyiwzLlw6YRRSgJrOEGa
zAoBIJT0+kRARkSGJpYUzCtPngxyneae3I+2hfgLpHgq007OTHUxPTmDXtGbHdIqHj/USy7zPC8/
jgwZl4BurXzr6a10hBMbf29elud+uYQwfo/ixb5pocAymiYhIt452qD+YzypIn2Bam5lWBpxOTPr
y+nlfwo8Zk5Xcj1Vi5nugzLR2x/ToZKajy7zPuUKa3uZFg6yh+pcF6rLuP1/wY3tUqXggrGBfKzP
TI2GmcPLyaAeDaSIRg2kiItKtY5rMdPp0J9nNEqLj6ZACdA++Kph04jVhz/HqKge+PicHfzc3BzN
zYh9Isc90uMSA+KwrHPOEjKwoMQaQjUzPtBhMX/CE+1Z4iSuxLiu1wWEZuOf22rF2TSn4v+2NZKc
PtQXhj/rvH5tlSnqvOvUQX8+HuR6LSIPBNBER6ZaYmRYnzOqdONHbxVLvtFDnDmF/a+REjgTgJ8s
K8J0l+Xj4snz8ZoEf2z5Vii/NzEa+rSc4vHW8A9eO7b+EeSiK7SbkRbULeEIKmnza11C9V1adqMm
3SShM0oquweRYTap1YCElsns7npVYIkmS8OCiPNV17zcEsPWiH+1+GQ/u6lTdxqx06rDMm2hw6XO
Kn4iIDKUrIKtKRM0SMDU/vf5s5doktEbFE66AgjLn6D0fis/rQlb68jIbyoV+6HnFNvyw2IbHLs+
HK6igIAXBqyRJmJhrwYG2qInerpDmuWiGj+VZkrqiazBVOWa0+iNBfq0njMpsM969KATTAwwfU92
vwo7rxtrxl6jHf2tXJ3xhRbN7WlynRckzLd8KswSELxVD1LrD6ArImhYWIiPjbwMREemSuF96ap4
F9jRby8LMdHp6XdhvfPN4R7/3NZuklATSvcRSb7E3Ghi9HdO9Yq3HqulU90s7zRKuKIs/8BWz0wf
agppAemaBBZ+tzh7CM6Z4OCwSpgivP+6bJlJjME1ERLp0TCjxTqd+6V55NnXw3xngADK96/DLeL9

enVG9ZYBgOtXEIWTiBHFAm5meNELDx3pVxZjo/RchK72sYvB+6WiXldqwqqr///el5Ta8fnNpQl+
TEif2TYutN0RGx2CaiIh0SezPS7d+aYCZ6XT0Vl03QlHnPKbnWzA1f/Ree2JgvPG4B7865kFQVhC
UFfz6mAfP1nhV9xmp4WO9xMx0rSeMd4Rg+roh9JP/68LcmNffOcmJl28sGbG+cCMT+6YPMzNNpJJ
+p/mJiMgQxMy0uKYlHeQJu6ZdQWam9e41YSXWLRNGLysNANXCnIDmbhn/sL0D/3XYBUWJnake6dk
C4kTvk50hfNyqDrqur0z8/3F+sQ1rJjmx+UzkmPz93Bx8a3EeTCPYE25kYmaaQ8iI1BhMExGR7ii
KgsNtYmY6/f5k5Wp2TTMzrWfdIQVv16kzqjePH92JcZeX2jC3yIqDwu/Dqa7Y67nmFFmxeJBdzsk
2Xijz3tEUgBx13qjCaRry7++G5QXItkg44wrhCzOyccekrOE8VUpQrGBaUZQRHXBHpGfp98mEiIg
y3vHOEJq7I4FntkXSDGJKB7nMTKeVbfV+uKNqvCucJiwqGd2KCLNJwmurS/CTgy48edgDXzj2a8h
hBr48Mwf/OD8XZtPIBjpiZloWnuK1Y+1DDr5yrCb8bEXhUJ8aDdPEXDOyLBK8l34v2v0KGn0yKrK
425sIYM80ERHp0DZxeFG5DZYRDhCSQeyZ7mJmWtfEEu+bxzt1UU6cZzPhocX52Lu2HJ+bloXoofZ
mCfjctCzsvbMCjyzJR/4orC3Kt5n69m3HMpQSb9IHkyRhRgH3TRP1J/1O8xMRUcbbJkxTvnpMeu6
SFad5MzMdn8NtQfz9hx0IyAq+e3k+VozALmFZUbBFWIk12v3SojFZZvxkeSG+NicHvznhhQTg3ml
Zozogrdf4HIumFL3XtUMYPkb6MavQir0tkWN7uD3IEyRElzCYJiIiXQnLCt5rEILpNP0wzmneifO
HFXz6z619O4q/9l479t5ZnvLS5Tdru9HkixyfXKs0IkH8UFTnW/HokvzRfhoqVTnmmMH0nCIrypw
sCU5n4vDHI+1cj0XUi2XeRESkKwfbgqp9zAU2CXOLRj/zNhRiz3QXM9ODeqbG0xdIA8A5d1h1ORV
cQRnf2NGpum7lOAfs5tEv8U4X4nqsXten6YkwiphdyDJvov4wmCYiIl0RS7yvGmPXRd/qUHCad2I
CYQU/PuDWXH+6K7WZsEd2d6kCdpMErJ+Tk9LvmWmqcmIXO15fyWA63YkTvY93BBEWp8wRGRSDaSI
i0pWtGdIvDXDPdKKeP+mNmYU+lcJgekejH7885lFd99VZOVgwwuul0l2szLTDDFxZlr6/v9SjzGl
GiSMSMnSHgTMulnoTAQymiYhIRwJhBdsbA6rrrknnYNrGad7xCsoKfnTAFfO2VAXT3SEF6z/oQPQ
pjom5ZvzrotyUfL9MFiuYXlZuh8OSnlUlpCZmpw+zb5oIAINpIiLSkT0tgb59pkDPnt/q/PSdlck
90/H77Ukvzrtj90anqsz7Pw64cKJT/bV/uqwQWRZ+PErU+Bhl3texxDtjzOR6LKKY+NeCiIh0I9Z
KLClN+6WB2NO8FYUBtSg0QFYaAE67kj+A7FhHED8Rvue66ixcw4FZQ1JgkzRtDdeP5fqkTDFbGAJ
5lME0EQAG00REpCNbL2qHj6UzuxmIjqeDMuBP7WDqtPTSaR/ORgXMYmXwWVcIoSQPPNp43IOoIgh
UOE26WzeVTiRJwl9Pzeq7fPUYO2YVpm9VCamJZd5cj0XUg+9yRESkC96QjF3NQr90mmcJJUlCrtW
ENn+kV9oVlOGwcO9ur7Cs4D8+7lJdd091Fl47342W7p7/t6AM1HnCmJibvI8tYm/+ty/LR4GdOYb
heOSyfMwpssIbUnDXZGdaV5WQ2gyhzPu0KwRfSIGTPfFkcPyrQUREurCzMYBg1HyuibnmmH2Y6YZ
90wP7sDGAU13qrPQ/zMvFlDz1sU/mEDJ3UMbBNnWZ6spx6X3iRg8cFgn3TsvGV2bloNjBE0aZJMd
qwoSoIXOy0rMii8johv0pZc+ePdiwYQNCoRCCwSA+9alP4bOf/SwCgQAee+wx7Nu3DwCwdu1a3HP
PPcN+wkRElJkyaSVWtJ6J3pFgkRO91V4951Ndvn2SExNyLZicZ8HOpkj2+FRnCDdUJud77m0JIhx
1TmNKnhklDP6IBjSr0IpzUUMCD7cHuUKODG/YwfS3v/1tPPTQQ7j88svR3t6OO++8E8uXL8fWrVv
h9XqxefNmuFwurFu3DvPmzcPcuXOT8byJiCjDxBo+lgnEzHQXM9N9FEXBa+e7VdfdNtEJACnNTO9
sVL/WruAuZKJBzS60Ykt5Pf1KPumiYZf5i1JElyunmmYXq8XNpsNeXl5ePPNN7F27VpIkoS8vDy
sXr0ab7zxxrCfMBERZZ7OgIz9reqSwasqMiPAyRV2TbuYme7zcWsQdZ5IpstplnDDpXVKU/LUmeJ
krseKzngDwBVlzK4RDUYcKMf1WERJyEx/73vfwwMPPICf/vSnaG9vx4MPPoji4mI0NDRgzJGzgXx
bO3Ystm7d2u/XGPSRrcN0X9uE1wGPSRrcN0X9
uT9bVoyaQ2EjvTP19+c05K4DIlOAr8oO4cOYUAMDqlgA4+2471upLyv+DrAA7GpwAIhUDFd31qKl
JrGIgU49JuuNxSZ0sj/p38kBLfL+TPCb6xOMSn+rq6gFvH1YwHQ6H8dRTT+HBBx/E0qVLUVdXh69
85SuYPHlywl9rsCeaDmpqajLi58s58g0PC76w2OiT6N5XF494AKiQudl43JQXV01Ks8l2SqbO4BmT9/
lrKIyVFfnxP34TP59+fBQI4BIxvnTs0tRfWm9UkVQBvbX99120W/CxClTYTUNb3rwkfYg3OGmvss
FNgkr502BKYHJ05l8TNIZj0tqTZQVWD++2DcosiVgQnHVZBQNMG+Ax0SfeFySZ1hl3idOnMCZM2e
wdOlSAMC4ceMwZ84c7Nu3D+Xl5aivj/ojePEiKioqhvdsiYgoI+1vVZfdLii29nPP9KPpmQ4Yr2e
61h3CW7XdcEWNaz/ZGcSxjkggbZaAm6oi1Qm5VhPKnZGPKWEFOO8a/pLuj4QS78vLbAkF0kRGZTV
JmJYvlHp3sG+ajG1YwXR5eTk6Ojpw/PhxAEBbWxsOHTqE6upqrFq1Cps2bUWPdnBmbFvzQPaUAAy
qpDxpIiLKLPta1L13CzNoQmye2DMdzKye6e6Qgj+c9WFbvR+Koj1RsL3Rj4WbGvFXf27FtX9oQqO
3JyAWB4+tqLCjUNjzPDkFQ8h2cPgY0ZDNLlSf6DzSxr5pMrZhlXkXFRXhoYcewne+8x0APWXfd99
9NxYtWoTZs2fj3//937937FmzRooioK1a9di3rx5SXnSRESUOdq6q6xqv/nTMRj+wmzKgjXWVu6bj/2+4DaT1W3eisijF4aCFfRDsA+sR9ZO
9NxYtWoTZs2fj3//937FmzRooioK1a9di3rx5SXnSRESUOdq6wzjvVu8ZFj+wpbNM3jMdlBWs/VM
L3m/oyfb+v0V5+Kf5uar7PLKnC6FLP/KprjA+924b/nBTiWYl1q0THBBNybNge2PUeqwkBNOa4WP
lmXPihijVZhZaAUR+dzmEjIxu2APIVq5ciZUrV2qut9vtePjhh6eDR4Rh/mTPwKTe0q4tjZAKtDxz8
mT/P+yQFXXyANAI8fcOGLM7JRcCnDfKIjqAqGAWB7YwBfe68du5rVx/2W8U6IxPVYw53o3eQL44x
LfeJmUUnmnLghSrVZYmaa67HI4Ia9GouIiGg4NCXeGdQvDWTunulDbUH84GOX6jpvSMHTyLD1n5
zwhvzsS+eVmelF5dYMTZbO8Qo2WXeYlZ6XrEVWRZ+FCKKl7ge62hHMGZ7B5FR8C8IERGNKnH4WCb
1SwM9g7SiZULPdFBW8NX32hHrR/nFMQ9CsgJ/WMHzJ2MH06JbJ2iz0oA2Mx1PMC0rCk51hvDKWR+
+v78LPz/ihvvSE93ZqB0+RkTxG5dtRp4tcoLQFVRQ6xn+YECidDXsMm8iIqLhEDPTmTTJGwDybZk
3zfvxAy4c6GfwUK07jNfPdyMkK2j1R6LtXKuEgKzAH+Nzd6x+aQCYnKfOVtd6wgiEFdjM6v9TRVH
w0mkffnXMgwNtQXhD6v/j35zwYNPKEs0k7ys5fIwoIZIkYVaBFTuifpeOtAcxPochBRkTM9NERDR
qWrrDqIvKalhNwOyizAqoMxcx0Z5r3TB9oDeCH+9Xl3eKa2Z8fcWOjUOL911Oz8MMrCzRfb3q+BdX
5sY95lsWEsVmR/z9ZAc661Nnpc64Q7nyrFV/e1o4dTQFNIA309HXe+FqzpgqCw8eIEjdTKPVm3zQ
ZGYNpIiIaNfuFrPTMAivs5swZPgYAxQ71n9p2f/oG0yFZwVff70B0vFrqMOGllSWq+33YGMDWevU
Kqs9Ny8a907Jx37Qs1fW3T4pd4t2rv77psKzgycNuLPt9E9656I/1UJU6TxjR5zGqcswYk6Xt0ya
igWmHkHGiNxkXg2kiIho14iTvhRk4WTnPI6/IAr2NNPnI7equvGIaG8+/FlBbhqjB1Xj+m/ZPq

yUmtfxcEPrizAZ6ZmIcsi4ROVdvzfOTkDfs9YfdPtfhm3vtGCBz7qhCdGJrrQLuGqChuWDpB5vpL
90kRDIgbTxzuYmSbjYoMDERGNmn0tmT18DABMkoQiuwnN3ZG0aGu3HHN6td5tE7LNt0904pOXhof
dPzNbc3uve6dl9/3bbpbw5FWFePKqwri+pxhMH2gNYvOZFuxt0WbDrqqw4YdLCzA93wJJkhCSFXx
jRwd+fVw7CI3Dx4iGZkKO+r2r0ccBZGRczEwTEenYWVcI/3vcg2MdmVlGJ5Z5Z9rwsV5iqXdrmpZ
6f9igPvmxdnKkRPumKofmQzYA5FgkrBmklHsgYpn3i6d9mkA6zyZhw/IC/OGmEswosEKSekoBLCY
Jjy8twIMLczVfd3kFh48RDUWpU/173tItIyynZ7UN0XAxmCYi0qmzrhBu+GMz/u7DDlz5chN+dsg
1+IN0SlYU/OaEB3//YTs+bOjJXjb5wrjgjWQ0bCZgZmFmBtNFdvWf27bu9MvkdAZkHBRKvKPLqM0
mCV+epS3ZXjvZiRzr0D9uiJlp0dJyG3beUY57p2X3BdHRJEnCNxbk4b9WFCDH0nP7vdOyNKWqRBQ
fu1lCQdSWAllJ3xOERMPFMm8iIp368QGX6gPKt3Z1ocEr49EleTDFCBr07LcnvVj/QQcA4H+Pe/H
M9UWwmtQ/w6zCzBs+1qtEzEx3p98Hz52NAUTnnmYWWFAsjPG+Z2oWvre3S9XHfN/0bAzHpFwLJAC
x8l5LSq14cWWxZmJ6LPdUZ+OW8U50BWWu8SEapnKnGR2BSK90o09GmTP9WleIhouZaSIiHWrtDuO
FU9o+z/867Mb929oRSLMBVi+d9vX9WwHwpW3teLrGo7pPJg4f6yWWebekYTC9vVHdD70sRpl0gd2
Ehxbn9V2+a7Jz2KX7DouEyhj95XOLrHhpZUlcgXT082MgTTR8ZU71710T+6bJoPgXhYhIh3593Iv
+KoFfOu1Dc7eMZ68vQvYwymdHSlhWsLtZ3WvrDSn447lu1XWZOHysV7FdHQymY0nkh43qY7isn0n
ZX56ZjavG2NEZkHF5qS1m6XWiZhRYVPvIp+db8PKqYhTY9f/6J8pEYha6yZd+72lEycC/QkREOhM
IK/jlUbfqOrH6+d2Lfvz4gPo+enW4PQhXcPBM+vwMHT4GAEUOsWc6vT54ekMy9gqT15eWxx7gJUk
SZhVasbTcDrMpOWX7X5qZg0vtzpiWb8HvbypBiYMlpUSjhZlpoh4MpomIdOblsz40RJ3lz7ZIeOe
TpZiSpw4eXjqtLQPXo4+aAoPex24GZhZkbjCd7tO8dzcHEYx6yhNzzSO62mtVlQMf3l6GFz9RjPd
uK8OYLAbSRKOp3Mn1WEQAg2kiIl1RFAVPHFZnnO+pzsL8YhveuLkU0Ym+8+4wPEH9B2U7hWC60K7
NVs4utMKWocPHgPQfQNY7gb3Xsn6y0qk0rcCKG6scGTukjiidlAqZ6WaWeZNBMZgmItKRDxsD+Lg
1sn5IAvCVS+uGSp1mTIza46sAqOkMQe92CMH0r68twpJSdRZ6SWnm9ksDQLFdHECWXlkcTb90RWY
fLyIamDYzzWCajInBNBGRjjwpZKVvqnJgctSe3elCKfSxDn0H0xc9YdS61bukryyz4/lPFPdN767
MMuP/zNbuJ84kmp5pHZd5B8KKquIhEFawSzghsnwUMtNEpB9iz3Qzy7zJoDjNm4holPjDCl4+48O
HjX60+2W0+2VNBvCrQpA5o8CCLbWRy8c7gtAzsV96QbENDosEh8WMP91SirOuEMZmm5Flyexzu2J
murVbhqIoSZl0nUw7Gv1Y904bWrpl/J/Z2fjuknx83BqEL2oV25gsEybmsmeZyMjEad7MTJNRMZg
mIhoFHX4Zn/5zq6afONqcIitWCOW06ZaZ3tGk7rW9ImqdksUkYWp+5g4di5ZlkeAwo2/dWUAG3CE
FuVZ9BdPf2dOF5kv93E8c9iAkA2OFYV9Ly+26OwlARCOrxGGChJ52I6Cn2iYoK7AmaYI/UbpgME1
ENMLqvWHc+WYLjgwSCH91VrYmaJlRoH7b1ntmWjxZcEWZMXttJUlCicOs2pXc2i0jV0d7wn0hbTn
3L456kGdTvwb72y9NRMZhMUkodpjQEjVMsdknj+iUfyI90M9fcSIiAzjdFcKq15oHDaTXTHLi01O
yNNdX56uD6bPuMHyhwXc4jwZPUMaBVnWwb9RgGgCKYpR6j4azrhACYe1r5uPWAGK9lLoC6iuXVbB
fmoi4a5rS36bTXtz9dit+dtAFWRnaZylmpomIRsjBtiDWvNnSV0bba2m5DV+emY1CuwmFdhMqnGa
U97NHN9tqwoQcM85dGuolK0BNZxDzivUXpO5pCSI6Zpuca0ap07hZC82u6REOphVFwefebcMrZ7u
RZ5Pw1i2lmBHVNiBmpWMptEua6ggiMqZypxlH2iMnhpvYN01p5P0GP764tR0A8Pr5bjR1y3h0SX7
CX4eZaSKiERCSFXzx3TZNIH1TlQObbyzBHZOycO1YB+YX2/oNpHtpS7312TctDh+7wuAToDXB9Ah
P9N7VHMArZ7sB9GSbcNwVBgAAIABJREFUv72rU3X7R82DB9NLy+0wsV+aiKDdNd3IzDSlkZ8ecKk
u/+yQG6+c9SX8dRhMExGNgI9bgzgu7IT+zNQsPHN9EZyWxIITcQiZXoPpnY3q4WNXGrjEG4hV5j2
yHzzFkvtt9QF0X6rrVhRtv/RDi/MgtnSzX5qIeom7ppmZpnRR0xnEny74Ndf/7fvtqOlMbBYNg2k
iohEgDuK6odKO/1pRAMsQJp9OFzLTx3Q4hExWFE2m8wqDB2Ilo1zmfapLfdLFF1aw/dIJjzpPGA1
RH4QdZuBvZ+fgl9cUwXbpaRfZTfjMVG0fPxEZE3umKV394ogn5vWuoIJ177TBHYz/7zODaSKiESC
WPN9Q6RhyuewMMTPdqb/M9LGOEDqjBlcV2CRMyzd2r+1ol3mf7tK+Tt6+dGZ+t3DiY2GJDTazhNs
mOrH99nL88ppCfLSmDMUO4/a8E5GauGuamWlKBx1+Gc+d9PZ7+7GOEL7+QQeUOAeSMZgmIhoBYgn
t5cMoeZ4mZKZPd4XgjzGdeTTtbNT+vEbvtS22qz94jnRm+rRLmzV650JPD7V4suey0sjrc0q+BWs
nZ6GEgTQRRSlnzzSloadrPPBEra6ocJqwdrJTdZ/fnfHh6Zr+A+5oDKaJiFKszh3CBW/kQ4bdDMw
rsg7wiIHlWk0YF7XLM6wAJ3WWnd7ZpO5FuqLM2MPHAKBIyEy3jWBmOiQrOOvSvkaOdIRw0RPWZKa
XlBq7JJ+IBlcqnGATB2wS6U1YVvDUUXWJ9xdmZOOnywowq1CdqPj5YXdcX5PBNBFRiolZv0WXSmi
HQzvRWz990yFZwbsXhWDa4P3SAFA8inum6zxh9NcC9kZtNz4WhpMtMfiwOCIaXHkWM9OUXl6v7cZ
5tzq58TczspFtNeHp64oRPcbmSEcIbXEMCmUwTUSUYuIgrmRk/cSJ3sd0lJn+y0W/aphVjkXC4hI
GZ+IAspYRDKbF4WPRfnbIhUDUUxmXbcaYQdazEREV2U2IPi/cFVD6NgQQ6dHPj6izzXdFtTBNybd
gdqH6s9WOpsFXRjKYJiJKMTEzPZx+6V7iRG89ZaafFfqMbp/kTHj9VyYSy7zb/TLC8sh88Iw1fKz
XGaGXOhmvTyLKfCZJQqnwvtY0wiv/iOJ1oDWADxrUn8e+MitHdXmpUEW3vZHBNBHRqPKFFM1+32Q
EK9oyb31kptv9Ml4/71Ndd0811ykBgNUkIc8WOamgAOgIjEx2eqDMtOgy9ksTUZw40ZvSxdMn1Cf
6r6qwYY4wv2ZZuXq+S+/6yIEwmCYiQ+nwy/jvI25c/8cmzH6hQVPyEy9vSMaGgy58c0fHgFnhfS0
BRFe9Tcw1az58DMW0fPUfgJOdIQR0MNF702mvqmR4cq4ZVzLT2We0+qbFzHTOAJUCzEwTUbzEid7
cNU169V6DOjD+4swczX3EzPT+lsGr/oy99JOIDGNnox+/Ou7BK2d9iK5C+5ednVhSasPiBLNx/77
PhZ8d6gnEXz3XjQ/vKEO+TXt+MhUl3gBQYDdhTJYJ9d6eYCykAKddIc0O6pEmlnjfXZ0NyeArsaI
VO0yqsuqR2jUtZqY/MzULTx3zaO433EnzRGQspcxMUxpo6w7jWFQFnwTg2rHaLSPlWWZMzjX3rZK
MZwQAM9NElPF+9LELq15vwQun1IF0rw2HXAl9PVlRVEHjBW8Ym07H3ke4Uwimr0hi1k8MnEe71Pt
wWxD7o0raJQB/PcXZ/wMMqFhYJTMSQ8hCsoJzQl/0F2Zmx7zv/KLhT5onIuPgrmlKB+IgsTlF1pg
JEABYWpHYKk8G00SU0fxhBT86MHCw/Mdz3TiTQE/p0faQZkfwxuPaYFpRFOxK4f5ecQjZsVEeQvb
cSfX/wbVj7RiXwwKoaGKZd9sIBNO17rDq7Hqpw4QZBVZN3z3AlVhElBgxM93MzDTpkDhITCznjve

```
2WBhME1FG29UcgFeo06nKMWNcduQDgKwA/3U4/t7p9xu0AykOtAWxv0X9Zn3GFVZlHnMsEmYVJq+
EdjQz03XuEH5xxI1Xz/ngCykIygpePKUOpjl4TKtYmHw7EmXeYon3lLyeIPr6Su3Zd/ZLE1EimJm
mdCAOEhswmC5jZpqIqM979eo30NsmOvDx2nL8v0V5quufqfGgJc6VHrGCaQDYeELdgyqWeC8utcF
iSl4JrSYz3T4ymemW7jCuf7UZ/7yzE599pw3Vz9fjzrda0Rx14iDPJuGW8SzxFo3GADIxmJ58KZi
+odKhuS8neRNRIsSBmsxMk954grJmkNjS8v4D5sl5ZpQ54w+RGUwTUUYTg+mV4xwwSRLunOxUZae
7w8BTR7UDmUSyomj2FPbadNoHdzDyQWKXEEwnu4R2er46mK7pCqE7nmkZw/SLox7VkBl3SME24f9
57aQs7paOQdw13ToCO1nFSd69mell5XYU2iPHaEaBBZXZw580T0TGIQYdzEyT3uxuDqpanSblmlG
R1f/fOkmSEir1ZjBNRBnLF1KwW+hZvurSYAmrScJXZqmHMD111ANvaOCz6kdi9Ev3cgUVvHwmsmN
5Z5M6wEzm8DEAKHKoy9WDMrC1fvCdiMPhDyv4dYwp0KK7WeIdU4kmmE59FkcMpifn9bxmnBYJT6w
oxMRcM2YUWPCTZQUpfy5ElFnKOc2bdE5b4j14GXc89+nFYJqIMtZHTX7VzuPxOWZMyI1kc++bno0
8WyQz1+aX8VxN7KncvcQSbzH3+ptLpd6dARlH2tVBTDKHj/W6cZy6VPf1875+7pkcm8/4VOXcscw
qtGBxCdcrxaIp8x6FnuneMm8AWD3eiX13lmP77WW4MoEPD0REAJBvkxA9FNkTUlQVWkSjTZzkHU/
WmZlpIiIA79Wr30CvHqMOFnKtJnx+ujo7/Z+H3QjL/ZdKvy9kfr8wQ/34Xc1BvHjKi9WvNyP6q0z
Pt6DAnvy33FsmqIPpLbXdkJXUlHorioL/PqIe1Hb/zGy8uroEn5+ejUm5ZswvtuLnVxVyt3Q/NAP
IUpyZDsoKzrnVZZfRwTTQU9LG40VEQyFJEvumSbdCsqJpuVsWx4njOYVW5Fnj+7vIYJqIMtZ7Qhb
5qjHaN9D7Z+XAGvVOeNYVxqvnu2N+PVlR8IFQLvTZ6ixcI3zdL29r12Sllye4tzBeKyrsyI16w2/
yyZrSdtHvTnux8tUmfHlbG9oS6Nn9qCmg2SP9lVk5WFFhx+PLCrBvbQW2fqoM84o5xKo/4p7p/lo
GkqXWHUY46txKudOEXCv/9BNR8rBvmvTqQGsQnqiG6TKnqa/VaSBmkxT3dgv+RSWijOQOytjTT79
0tDFZZnx6irq/9zcnYvcEH24Pod0feVPOs0mYW2TFfdMG7g+uyjHj63Nz4n3qCbGbJawUSr1fOxf
7ZAAA7GsJ4Itb2y9l0H14bP/AO7ij/fyI+v9lVZUDk/K4RzoR+TYJ5qiT3a6gAn84dUPjBirxJiJ
KBnHXNPumSS8+FBIgV5bZ4q7EirdvmsE0EWWknU0B1fTGKXlmjO1nUvGXZ6pLtd+54EetW7uzWSz
xXlZuh9kk4ZYJThT1U8L96SlOvPepMlWvdrLdPF7om67tP5j+zp4uVfn5W3X93zfaBU8Yfzin7sc
WB7jR4EySpHmtpLLUm8E0EaWauGu6iZlp0ontjWK/dPxVgvH2TTOYJqKMtO2iUOI9QJn1vGIb5hd
HBmYpAJ47qR1EJg4f6y0bt5slrBOmVxfZTdh4XRH+++qilPRKR1s5zqEqVa/pDOFEh3bn9NaL3Xh
X+H856wqjPY5S4/855laVC88osGjK2yk+mr7pFJZ6i8H0FAbTRJRkYs900whsKSAajKIo2KEJpuN
vQ1tUYsOcosGHqQ77E14oFMKGDRtw55134jOf+QzuvfdeAEAgEMAjjzyCO+64A3fccQeeffbZ4X4
rIqK4xdMvHU0Mhp+t8aoGefXsl1Z/zRUVkTflf16Qi1XjevqX75zkxIe3l+G2ic6hPv2E5NtMWCG
cLHhd6PtWFAWP7OmK+fj9LQP3WPtCCv73uPrkwv0zczi0aojEzHQifeuJOsNgmohSrEw4QdjkZWa
aRt+JzpDqZHWutac1L14Oi4T3bysb9H7D/qv65JNPoqWlBS+88AIsFgtaWloAAM899xy8Xi82b94
Ml8uFdevWYd68eZg7d+5wvyUR0YA6A7JqUBYATbApWjs5C/9vVyd645rz7jC21ftx7dieEupDbUF
0BCLBdb5NwpzCyJtyttWEF1aWQFGUUQkybxnvwF+iss6vnffh7+bl9l1+9Xw39rRos9UAsLcliOs
qHTFva/fLuH9bm2pQVr5Nwl9NGZkTBZloJCd6i5npSbmDD14hIkpEeZb6faWRPdOkA2JW+vIyG8y
m5H8+G1Zmuru7Gy+++CLWr18Pi6UnLi8pKQEAvPnmm1i7di0kSUJeXh5Wr16NN954Y/jPmIhoENs
b/YjebjU936L5Yy8qsJvwqQnqAPHpE5Fs7PsN2tUKsd6URytbu3q8+rnvbg6i4VJ2ICwr+G4/WWm
gZyhZLHubA7j6D014q06dkb93WjayORF6yEqEYLolRcF0UFZwfpC1WEREwyVO825OQbVNV0DG0yc
8+OSWZsx64f+zd9+BTZXrH8C/J7srekKKiiKIoIgiCKinq9K97nFV9HeRe8c5c/D4RmPQtoG
CLzhAFAEZDkRB9t6UvbrbNGnSphqT9fv6vivd5FhRpNSjYwc2qZN2qT9fv6iaZKe8LbnnOd9nvd5c/
k8JdjdeLLo9KOn/WS/ujUVfV06dPQ6PRAEuGjjvEGjBs3m5UIrZrm
0Be7oFoUvj1pxqNSzodoF8iy+KIr48KAFT20uhV0W56VFKvBg7+B0Jm8tErXSiZ1grZk+aXZI1rm
nRigQzUkQIgqwZF3wMtN/5NrwySELlp2sQLnbCe39AxZckqbF+HaskqIaZXYX3tpbhrf2lsHqkE6
2+LNe2h+NCqadTifMZjNUKhU++eQT5OXl4Wdkq4kJmZqbf75Wdnd2YQwkZLeVztDQcl9ATzDH56YQ
O7oU3nVGM7OyCOl+XIgIZOh3OVlS91uYE3txwEqUOAStOq1C1s3KVdpW5yM4OrVnxIZEq7EDNxeL
1HcVYdrgQW4xKuB/7mCQHfilUwi5WPXbG4sTGfdlI1FSNy/e5Srx4xHOyoE+ME//qXg7jmWMMwBv3
TtFxOswpwG6djecXIzs6r9TUN+XtZX6wAUFO+n6a281wYQPy/DE0cl6ZncQBATd+RfKsDhw9n40K
hVkPHxNe16ILlB/PQrdL78iWqW0v6WxFF4Id8Jd46oUGR3bNCMF4tIrb0FLLrvxtotS5dutT6/UY
F06mpqRAEAePGjQMApKSkYDSOdi7S3ic7Q0HJf
QE8wxybM6cXhdruSxG/q3Q6KufmtF/1xuxkvba0qiXz+ugTxkTtAqcE2/TlCEWAOu2w12vHcqv/r
rE+UKnCiXZiI1CuA/l2XgT2uKq7PYAFCqb4PEitPo3LkzPt2ZB0BapndPzyjMHBQLjTK6Pn
wAsdLqr926vTo0iXB5/Mb+vfy074yAKXVX/dKiUGXLm39fh/yxOtKaOK4NA9RFBG5Nac6E2zCUh
q1wnxWkWDx0QURSzY7XktcpeDaHTpYmjoYbdqLelv5UipHf/YUIrfZNuXXtAlVoV3L4lH76TgZKY
bVe8VFxeHiy++GOvAxwAmM1m7N69G127N1+8dsWYMWE5aCIiX/5vh3Q
f5V7xqnoH0gBwS+dIuC+F9pZ7fnaAPuQCaQDoEadCl7Xrs9ZrLGvNTKFNMVlZT34vhB2jwEBqT5U8ur
sNQrg08sS8PKQOAbSAZLYRPtMH0gFOreu5hUkL0gdIHKQMcg1Ejo6dIQ2lu7Lnf1sHTVNnp5bb/a6TSp
YnCJm7zTh4/NfNMmqTmTkRZqoe9WEUIBQmVerYMTqUAHWr1EnRwAtSPSJAkVtEZ/mphrEv+y
vPZuKIAi4o5bPOyxFgyf76wEA/Q3S7SF2nF83veykdEut0Zk6XNeBa9ICybMBWXC2kTko22u8ayy
DaSIKjrbR8mC6cee1ZSfLJV+PStdi7w2p0LidPvPKXUHdWpBCV7lDxHWrCjFrhxmVsvlorRJ4OCs
a26ek4O4e0VAHoYO3u0ZfWVNSUvD22297PK7VHavH888839u2FJiOpFFEU8G/
rGuXRwXpsGx3eHhHnV5a7OTzQOxoxGgH7l1IjlCgbbQWNUaBetRJvomlN+f4N0lnZnYSXEtlV
```

N19xdy+YuAZcgC6aLg9SA7LBRmhnqHlf//TWJiPzRNloFoOa6edLcuMz0slPSid3r2kdAoxTQJVa
FfSU1733A6MDw1NC+LlPgfXCgDBvyPHciGZWuxZxhcU26cwWnqYmoRfjuRLnHifVfg2MbNCM5rq0
O17bTYenJCujVAp4fFIs/d4tstm2v/KFSCLire93dtrvHqRChFKq7o+aVu7ClVIGDbgGYSgCubuN
9/2lqOG9l3oHen7y4wokCt/JxrRJoxz2miShI5OeXxmSmT5U5sMttlwkBwNi2VdeiHvFqaTBdYsf
w1NDaXYOCq7TShdf3SDuJJekU+NfgWEzpGNHk92oMpoko7JU7RDy7RbqP8lWZWozObFggqBAEfH5
5Ik6aHUiJUEKnCv0g2l8qhYCsRDU25ddMQMw9Ic1cjkjTIl7LrZQCLVIlQKcELlQnVroAs12EXhO
43zP5Vmid9CqoglzqRkStVyDLvJfLlhsNTdEgOaLq/XvEqQHUVFAdMDYuA07hZ+6+MpTYauoQ9Wo
B669Lrv4daWq8SyKisPfWXjPOWGou3CoB+L/BsY1+33YxqhYZSF/QL1EaPO8rk16Irm3HrHQwCII
Ag2y5QGGAm5AdYok3ETWhttHS/NypRpR5y5cbjWtbcy3qES/9OfvZhKxVKa5wYu6+Mslj9/WKbrZ
AGmAwTURhLsfqxOt7pCfWv/SMQpdYBg91ka+bdicAGNeW66WDJUXWlTavPLBNdA6x+RgRNSFvZd6
i6G0/jNoVVjg9lmyNd+vd0TNeem0/UGJv0M+h8PTGnjKY7TXjnaBV4L5edS9tCyYG00QU1hYds1b
vbQlUrUd9rK++GY8ofAww+J5wGJysQWok19gGi3wWPb+cmWkiCl+JWgUi3Sq5yhwiShrQXHHFqQq
43GLj3glqtI+pmQxsG62U/BxjpYi8AJ8/KTTlWp344IBF8tiMPtHQa5o3nGUwTURhTT6D/VCfaMR
xnW+9dI5VIdpHGfv4tizxDiaPzLQ10JlpaTDdLY6ZaSIKHkEQPNZNn2zAuml5F2/5tUghCOguO59
xv+nWYc5uc3XTVKDqOnp3j+bfqpR3nEQUtkRRxEZZMH1ZOrt61pdCENDXR3Z6PLfECqrkyOBlpk2
VLpx1C86VQlUDMiKiYGrXyCZkZrsLv56TBtPjvFyL5JU2+9mErMXbVVSJTw9Js9KP9I1BpKr5Q9n
mPwIiogbKLnVI9uiNUQvoFc9yVn8M8LJuuneCGh0YfAVVMNdMZ8s6eXfUq6BRttxGekQUGhrbhOz
nMzbY3E6F7aKV6B3veS2SNyFjZrrlEkUR7+0vw5XLCmB3m3NuE63E7V2bPysNcGssIgpBh4x2JGg
VSKqjO+PGfGlWenCyBkpu/+OX/omekw/s4h18nmumAxdMs/kYETUHr2XeflxO5F28x7fzvmewtyZ
k1PIUVjhx/+8lWHXG5vG9x/vFQBsik8S8whJRSHn4DyM+PmSBUgDu7xWNfw7U+9wfV17iPSTZd3d
q8s5bR+/x7OIddJ6Z6cCVeXs2H+OlnoiCr22MLDNd5gAM9XutwyVi9WnZemkfE7s9ZGXeB40OuEQ
RCi+BN4WnfcV2TF5d6PXaeHf3KEzrHNkMR+Udy7yJKGQcLXXg4/NrYpwi8ObeMly/ughFFd6zdhv
zpLOVQ1O4Xtpf7WOU6J1Qc2MywKBGTy9ldRRYQc1Ml8qbj3HpAxEFX2PWTO8rscMk2/JocJL3CfK
0SAViNTWBs8Uh4nQDmp1RlWMmB3YVVYbMFmOVThF3/lrsEUjr1QI+vjQerwyL81qx0FwYTBNRg/1
ytgK3/VKEWTtMcLgafxL+LcezlGdtjg2Xfl+AnYXSLHSe1Ylj5pqLp0oABtay1RN5JwgC/jsyHhP
b63B1kgMfXZoQUheplipZlpnOL3fBFaAbGZZ5E1FzkJd5V+01Xb/XbpJVmg1N8b1sSxAEz1JvI0u
9/XXYaMeNPxZiwDd5uPT7Avz9D2NzHxIA4J19ZR6TwkOSNfh9YjImdwydjPQFDKaJqEHOWpy49Zd
iLD1Zgdk7zXhtt7nR7/l7rmcwDQBnLE6M+aEAy07WrKeSr5fum6hGlJqntIboEa/GvFGJeLFbJRu
PNZFIlQJ6dc2NokOEpJleQ5U7RMl2NAKArizzJqImEK9VSLZbtDpE1Hc58ybZNX1oHcu25KXeB0r
Y0bu+iiuc+M9RNYZ9l4/VbuuRPz1sxb7i5p2UOFXmwL93Su8np3SMwPKxBrSLCc1rGe88iahBVp+
ugNVRM+U877C1USVCoihinY9gGgBsTuCvv5Wg4Hw57KZ8lnhTeEuRbY+VZ218MH3E5IB7kUibaGV
IbB1CRC2fIAhoGyM9r+XY6nf+kQfTdfVA4V7T/imzu7D8ZDlmrC9B/2/y8HWOGk4vt2yfHrZ4Pti
EHt9YKtlLOkGrwL+HxPrsnRMKeIUlogbZLiu7PmNxYldRwy9mh0sdkr12o1QC/tpDuu2BxSHi3f1
lADybjw1NYfMxCi+epd6NX/N3WFbqyOZjRNSU5Ntjna2oOwg6a3HijKXm/KdRAP28NMd01yOee03
Xx+85Nly/uhAdF+Rg2i/F+PSwFaWVvhMfXx21wuoIXENMf/xwqhwrZE3onh+kR4Ku9p1dmhuDaSJ
qEHkwDQDLTlZ4eWb9/C5bLz0sRYPZQ+PwdP8YyeP/PWDBWS+BOzt5U7hJkTUhC0RH74NGNh8jouY
jXzedY6s7mN4kayba36Cpc9sj+V7Th432gPRuaUmOmxyYtKoQP5+1odLH5aVdtBLx2pr/a1OliMX
Hy70/OYgsdhce31QqeWxIsga3dgm9NdJyDKaJyG8Wu8vjph3w3CPSH+typcH5JWlVZdv39YqGQVd
zqjLbRdzzW7GkPKmTXunRHZko1AUlM13K5mNE1Hzk61pz6pGZlvdAqc/kuEGnlJxDK13AcTOz0+6
WnCiHw8f8QrRSxPMD9dg0KQV/6iqtAvz0UNOXev9nl1nSkV0pAHOGxYXFdmcMponIb7uL7V7X2hw
0OnCk1P9Sb2/rpUekVgXTUWoF7u8VLfne77nyEm+ul6bwE4zMtOce08xME1HTkWemz1bUHWr4u17
6AnkTsv1sQibxhyzj3yZaiek9orDoykSsHFKOGVkx0KkEj2B6a4Ede843IhNFEV8esWLaz0V4b39
ZULbPWn6yHG/sKZM89teeUZJtO0MZg2ki8tv2Qt8Bc0NKvQ8YHSisqAkkYtQC+ibWnETv6h6FOI3
v2cm6un4ShaJAZ6btLhFHZNuJsJM3ETWl2sq8KxwithVUwmKvud6X2V3YK+sgPaSePVDkpd5sQlb
D6RI9esssvDIR/xkah9GZOmjdLj8d9Cpcli5NSsw7ZIFLFPHEplLc83sJlp+qwBObSvHOPmnQ21h
7i+34y28lcA/R0yIVeLK/PqA/J5gYTBOR33Z4WS99QUNKvdfJ1ktfnKKRdG7UaxS4p2e0/GXVhrH
5GIWhQGemj5sckpK+tEgFYjW8zBNR02kna0CWaxMgiiLyy5246Ns8XLGsAIMX5+NUWdXE37YCaaV
bZ70Khno2nJLvNb2fwXS1vSV2mOw1/7GJWgW61bLs58/dpNnpr49acf86I94/IC35nrXDjLOWxi9
JAoCCcidu/rkIFrcLl0oA3h+ZgJgw2uo0fI6UiELG9gLfwfTWAjvO+Xmile8vfaHE291fe0YjRu2
ZnTboFOjEvZEpDAU6M83mY0TU3OK0CujdKslsLgF55S78Z2fNmtizVieeOt9sSr7NZX2z0oBnML2
nmfdIbmouUcTPZyswP9sCo006GfuHbDncsBQNhFrWH1/TVie5JpnsIr44YvV4nsUh4unNpR6P+8v
mFHHbL8WSddIA8O+hcRiZFl5L9xhME5FfjDYXjpmlTSJ6ykqtlvuRnXaJItb7aD7mLl6rwN3dozw
eH5Jc+wWCKFR5ZqYbF0wflpd4s/kYETUD+fZYO4sq8b9saWC27FQF/si1NXi9NFB17+G+/fBxsxM
mX22rW6B39pbh+tVFuH+dETf8WAi7Wzdz+Xrpi70kKdypFQKmda5f5+zvTpTjl7MN371FFEX8/Q+
jR+O56d2jcKeX+7xQx2CaiPwiL/HuHqfCDR2lJ2B/1k3vL3Gg2G1GVa8R0MdH04n7ekUjQrZdBve
XpnBl0CkkN4IlNhE2b5396sHpEj1ubth8jIiaQzvZuumZ20wo93Jue2ZLKbY0IpiOVCnQRVaZJl9
/3VI5XCJec2vataXAjp/OVF0DRFH0yExfXI97pT918x7IZkYpPZImj240Nvh69fJOMxbIst6Xpmn
xryGxDXq/5sZgmoj8Im8+NsCgwfh2Oslj63JtKLHVb3ZYvr/0xSlaKBXeM81JEUr8pUfNyV6tAMa
1jajXzyEKNUqgyYMoAAAgAElEQVSFINn2DahaQ+YvURTx2KZS/CFrNtMrnplpImp68iZkvrpsby+
UruuN1wro4mdFTZ/E8C31/i3Hhs8OW1Bc4f95f31upSQRAaA6QD1c6kCRTdrU1VeSwl37GBUulzU

```
i6xijxA/XGPD28Hi435kdNTnx5h6z38f96SELZu+Uvq5jjBKfjkqA2se9X6hjME1Eftkuy0wPMGj
QJVYtaWzhFIGVp+uXnZZvieWtxNvdswP1+HufaFyRocVHlyagI9dLUxiTl3rnN6AJ2ZzdZfjooLR
JzACDGoPZ5Z6ImoG8zLu+Bidr/d5XOEsWJO4Ok2D6td1mTFhZiAfXG3H1D4WwOvw79y876bmcbuX
pChRXOD2y0kOTNT6TFHKzhsQi9vya98FJGqy4Jglto1UYkKTBHd2kVYhzdptxwo+9vX84VY6HNxg
lj8VrBXw5OhHx2vANScP3yImoWciD6f6GqguZPDvt7UQvV7VeWt58rPYAQKUQ8NygWHxzlQET2jM
rTeEtRdaELNfPzPT/si14abtJ8lhmlBL/uzyRvQSIqFm0i/HdjXtcWx2UPk5NDdnmUp5x3VMU+sH
0mrMVmLmt5rx9uNSBxcf96zWz1Ms9lt0FLDpW7vd6aXfd4tQ4ODUNmyYlY9U4A1Iia8bynwNjkeg
W9FY4IfkctdmUZ8OdvxbDbVk3dErgyysS0TXMlyQxmCaiesuxOpFjrZk91SqBXgkXgmlpYLvmnA0
VjtrX03x9tBzGyprnxNWyXpqoJUpuRGb6l7MVeGi9dJY/ViNg4ZWJSI+q39YyRESB5iszrRSAl4f
E4k4fa3P9WS99gbzM+4DRjsoGruVtCucsTkyX7asMAJ8d8uyc7cuW/Erk+rhWzD9ibdB6aXcRKgH
d4tQeE7LxWgWeHyTd//m7E+U4WUd2eltBJW76uQju1ewKAfj4sgQMSQmvzt3eMJgmonqTb4nVJ0F
dvcalX6Ia6ZE1pxSrQ8QG2ezoBaIIvL7bjHt+L5E8fnGq/yVeROFMnpn2p6P3k5tKJfuzapXAF1c
kokc8J6SIqPnI10xfMLlDBNpEq/B4/xiPrS7VCqC/wf9g2qBTSu497C7gUGn9S4+bkt0l4s5fi1F
Y4RkIby6oxIF67pO9tJYmr7uK7DhrrbmO6JQN+3/1ZVqXSPSIq5kscYnA+wfKfD7/xzMVuHZlIUp
s0umD14bF4ZoW0vOGwTQR1dsOWfMx9xO0IAi4IkNa6r36jOcJ3+4S8a8jGjzvpTTovl7RATpSovD
Q0Mx0QblTcsMoAPhgZIJf5XxERMGg1ygQr/WcGL///DXeoFNiRp8Yyff6JaoRoWrYZHqfRGmwuLu
o0sczm9fMbSaP7aDczTts8fm9C0RRxPeyEm/5xIS7QUkaaHzV1TeAQhA87tU+P2xFqZctyeZnW3D
TT0WwyqoUH+8X47NzeDhiME1E9eat+Zi7KzOlwfRPZ6WZaZtTxE0/FeG7PGkJmEoA3hkRhxEMBKi
V8chMW+uXmZZ3x+2bqMZE9hAgohAhL/UemaZFP7d7hnt7RWHg+Z4rAoCHZMG1PzzWTYdgE7IfTpX
jrb3SDK57Rh0AvjpqrXN53K4iO06V1VwntEpg5iDfW0oFY4L1xk6RSHa7dpntIj5zmwgQRRGv7DL
j/nVGyCvuH86KxhP9Gj7WoYjBNBHViyiK2FEkD6alF7DL0rVwn1jOLnVIOj2+sceMn2UBtl4j4Ju
rDJjWpeXMUhLVV3JkwzLTB4zSm0WWdhNRKJHfH/yttzSbGalSYMnVBiy6MhEbJyV79F3xh0dH7zq
akJ21OPHz2QqU2f3fPaEhXKKIJzaVSh7LiFTip/HJkgx+iU3EslO1NyJbJivxvjxdh6mdI3xmp4f
7uV66PrRKAXd3l96zvb/fArtLhCiKeHaLyaMxpgDgP0Nj8c+BsS2uOSaDaSKqlxNmp2TNS4zacz9
IvUaBIbIT94/nS70rnaLH9j1to5X4cVwSLk1nRppap4aumd4vW1vXM45bxBFR6HioT0xV6bZCxMN
Z0Rid4Xmdj1YrMDpTh26N7OacJWtCtrfYDlH0zPCeszjx4PoS9FmYi+tXF+GSJfkw2oIfUO8rcUi
yySoB+PiyeKRHKTG1k3S7qXmHai/1lpd4T2gfgUiVApM6eE5GqISqMu9guKt7FHRuc8FnLE4sOVG
OJzeX4ul90gy8Vgl8OioB03u0zKV8DKaJqE5Ol4hPZCf4volqr83CrpKXep8PppefKkeeW9YtRi3
gx3FJjb6IEoUzb2umvd0Eyskb1TAzTUShpH2MCmuuTcKvw8qDno1sF62E3i0za7KLOOkWvBptLry
wtRQDvsnFZ4et1dszHTc7MXe/7+ZZgfLrWWk2eXSmrrqL9Z+6SjO8v+dW4pjJewO1g0Y7Drv1ylA
JwNg2VfdcN3eO9Hh+f4MaUerghHqJOqXHz3xwvRHv7ZfeK8ZqBHx7laFFL0NiME1EtTpmcmDcikK
8KVvrI18vfcFoWROy33IqUeEQ8aEsK31Tp0jJ/oVErZFeLUhm98udIsz22oNpURRxQLZmmsE0EYU
aQRCgaIKKXkEQ0NtHqff2gkoM/CYPr+0pk2zNdMGHByywOoKbnV5zTrq8bZRbNV6PeDUGy7LHn/l
oRPb9CWlWemSaFnHn930emqxBB9n+3hcHedspeSMyeaOxBK0CS682tPjGmAymicgrURTx4YEyjFi
S79F9UgAwvp3O6+t6xqskTTXKnSI+OmTBetm+h3d25xppIkEQvGSnay/1Pm1xosztpkWvETwa2RA
RtSbyUu/dxXZYHS7cvqYYRbWUchfbXPjiSP33ePZXhUPEH3m+g2kAuL2bNMP72WErZu0wYX62Bat
OV+D/tptwxdJ8zNphljxvglu2VxAE3C0ro54Q5Gxwl1g1xrTxfi9o0FUF0lmJwSkzDyVcZEVEXn2
ebcUjG0s9Ho/TCHj94ngMTvY+0ygIAq7M1GHe4ZqL08xt0vfpr3cyk0Z0XmqEUrKeLq/chc6+m7N
6ZKV7xqlbXEMXIiJ/eOvo/druMpyxSCcnUyIU6B6nxtqcmgD3nb1luKNrFJRBSKNvyrdJMuIZkUq
PfjOT2kfgqU2lMJ2vSiq2uTB7pzRwlhMAjGsrDWTv7RmFEpsLG/NsuLFTJAYGab20uwd6RWPVaWk
Ze3KEAkvGGFrNfR6nsonIg9Ml4uUdnvtAj8nUYsOkFFznpdGFu9GyddM2WaJtSpr39UBErVGyrAl
ZXZlpz/XSnBcnotZNngHdlGfDm3ulAelNnSKw/foUvHtJPNyXEh8zO/GDLCAMFI8S7wytx+RnlFq
BKR091zzXZmL7CCTJqpoUgoBnBuixbGwSbu/aNNV/I1I1GJFa83+fGqHAsqtbTyANMDNNRF78nmv
DOWtNWZRWCbwyNA63domsVwbs0jQt1ArA264TyREKjEqsX8diotZA3jsgr47tsfbLt8ViEz8iauW
6xaok9x3GSun63ZQIBf49NA5RagWi1MCUjpGS8u539pbh2kZsz+VLbeul3T3aLwYrTpcjx1r7+b9
3ghrj2+pwf+/Q6IwtCAL+d3ki3tprRqULuLdnNNKjWlc/HAbTRORhgWz90KT2EbjNj1lOvUaBock
a/C5bJw0At3eJgloR/O6ZROHC/8w0m48REbnTKAX0iFNjd7H3PaafHxQLvabmXPtAr2hJML0xvxK
b820+l7A1RGGFE7tke1772go0LVKJHdenYl2uDSfLHDhlduK0xYlcqxMZUUpcnqHDqHQtUkOwcWu
cVoFnB9ayNqmFYzBNRBJmuwvLTkrLnW7u7H+50JWZOo9gWiEAf+oWiYqcRh0iUYuSIivVy60lM+F
wiThcyjJvIiK5Poneg+khyRpM7STNOvdKUOOKDC1+PluTOX57bxk+uzxwwfRaWVY6K0ENg853MKx
TCR7L5Cj0cc00EUl8f6Jcsr1BZpQSl6T538TiSi8XhKvb6NAmmjf+RO78yUwfNzskPQiSIxS13pw
REbUWWQmeVToCgNlDYqHwskTtb7JS6aUnK3Dcxx7PDSEv8b48o2VvEdVaMZgmIgn5FhFTO0V4vQj
VpXucCpmydTN3czssIg/yzHRta6b3y0u8uV6aiAiAZ0dvALijWyT6GbwnBC5N00r2pxYBfHLI+x7
P/hJFEb/Wc700hTcG00RU7VSZA+tkpdk3dfavw+QFgiDgyf4x1V+PaaPDZbyQEHnwJzPNTt5ERN7
1SVQjSlUz+R+vFfDsAL3P5wuCgPt6Sif5fzobmK7eR0wOybZcOiUwJIDrsSl08CpMRNW+kmWlByW
p0SW24ZmvaV2iMMCgQX65C0NTNA3KcBO1dMmyzHRBhQtldheWn6rA9oJKdIYSXbpUfe+ArJN3TzY
fIyICAMSoFZg1JBZPbiqFRgl8elkCEupYBjO2bQQUghGu86vb9pc4kGt1NrrR15qz0qz0xSla6FS
8B2qJGEwTEYCqkqQvj0qD6Zs6NSwr7a5HvBo94hv9NkQtllYpIE4jVG/l4hKB7l/moqy6d4EWhhQ
rJnWI9OzkzTJvIqJqt3eNwk2dIqFWoF5becZrFRhgUGNrQc1E5ZpzNtzcwKo89/dwxxLvlotl3kQ
EANhSUImjppqSJLUCuL5j44NpIqqbPAtS5pDukfrU5lIUVzhxVNYcp1sc58SJiNxplEK9AukLRqV
```

```
LG6auaWSpt90lYl2uLJjOYJfulorBNBEBAL48Ui75+uo2OsRreYogagryUm+5HKsL964zwukWY7e
JVkr2TSUiIv/Ju2yvOWeDSxR9PLtuK05VwGyveX1yhAK92N+ixQrYVXjDhg0YMmQI5s2bBwCorKz
EzJkzMWnSJEyaNAnz588P1I8iogZYc7YCj2004odT5R7fKyh3YuExaYl3Y0uciKj++iV6lmtrZfH
1qtPSbElPZqWJiBptUJIGMeqaTHZBhQt7vexXXR+iKOKNPWbJY2MydX5lyim8BCSYLikpwdy5czF
ixIjqxxYsWACr1YrFixdj3rx5+Prrr7Fnz55A/Dgi8tOmPBuu/7EIHxyw4Jafi/G1bG307J1mySy
qQafAaJYkETWZ+3pF49I0LZIjFLi6jQ7zL0/AoalpSI3wfZnuweZjRESNplYIuCTNMzvdEL/nVmJ
boTQQv69XtI9nU0sQkGD6xRdfxP333w+9vqb9/KpVqzBlyhQIggC9Xo+xY8di5cqVgfhxROSnObv
N1Z0qAeCRjUacPb9lw2Gj3WNfxX9kxUCj5CwqUVNJjVRiydUGHL4pDV+OTsS4dhGI0yrw3KBYn69
hME1EFBiXyxqE/XK2YcH067ulWemxbXQ8V7dwja4R+/rrr5Geno6hQ4di1apV1Y/n5uYiLS2t+uv
09HSsXbvW5/tkZ2d39CQkv5HC1Nax6XU+UCVp+JkDxmqhRx1+qzeLOXDY8e0MAp1pwKMnUuXKr
KQbD/y1rzmIQyjktoGSACvaK12FfmuaY6ynQO2dkNX9dHjcO/ldDEcQk94TAmHe0CgJp7pT9yK7D
nYDbq2FlL4mCZgF/OSe+3ro83Iju7OEBHGVjhMC6hoMuFvSl9aFQwffToUSxduhT//e9/G/M2AOo
+0HCQnZ3dIj5HS9Pax+W/G40ALB6PbzQq8Ua+AWuLpSXf/zfMgJ7tIzyeH0itfUxCFcclND1SdhR
/3iW9o1MIwOg+nbhvaTPh30po4riEnnAZk86iiHaH8nCyrKpqzy4KyItug9GZ9V/y9q81xQBq+tI
MS9FgysCMQB9qQITLuISDRgXTO3bsQEFBAaZOnQoAMBqNUKlUyM/PR0pKCnJycpCRUfVLdO7cOaS
mpjb+iImo3kyVLizItvr8/nzZ9wYnaTChHddKE4WS3jEu3Nw5El8cqfl77aRXMZAmIgoQQRBweYY
WnxyqOc+uOWerdzB9zOTAkpPSBq9/7xMT0GOk0NSoNdNTpkzBypUrsWTJEixZsgSXX345br/9djz
66KMYM2YMFi1aBFEUYTKZsGLFCowZMzQx1E9TA/2yrZr9agUyCqlhvwFy/Ss+MkUQh6bqAeCW5
b1U3uENzqESKi1qYx+02/tVfam6ZnvApXZmp9v4BajKDtq3HLLbdg1qxZmDx5MkRRxJQpU5CVlRW
sH0dEMi5RxAcHyiSP3d09CikRSvx9g9Hj+RPa6TAkhSd+olCUGqnEqnEGzDtkRXQEdtkRXqUEnd1j2ruQyI
ialFGpmmhEFAdFO83OpBjdSItsvaF03lWJxYckVb6zegTw+REKxHQYPq5556r/rdWq8Xzzz8fyLc
nIj+sPlOB42Zn9dcaBXBn9ygk6RRYerIcv7ht+6ASgOdr6RpMRM2vS6waLw3m3ykRUTDEaRUYZNB
gc0Fl9WNrzlbgli61T14uPl4OW83tFtpGK1k91IoEZGssIgo97++XNh2b3CECyRFKCIKAN4fHwaC
r+fOf0ScGHfVBK1QhIiIiCnmjMvzfb3qrW/ANAHd2i4JKwax0a8G7Z6IW4ITZgWe3lOKoyQFRrCp
ROlTqkDznnp7R1f/OjFbhtwnJmHfYgnbRStzcObKpD5mIiIgopFyersXsnTV7Ra85Z4NLFKGopWR
7e6E0mB6aogna8VHoYTBNFOZKK124blUhTriVdMsNTdagn0F6ck+PUuLJ/vpgHx4RERFRWBiYpIF
eLcBkr1o4XVjhwp5iO/omeg+QS2wuyZI6pQBkJaqb5FgpNLDMm1qV4yYHDpTYIYpi3U8OA6IoYsZ
6Y62BNAD8tSebFRERERHVRqUQMDJNVup91nep9w5ZVrp7nAqRKoZXrQlHm1qNt/aa0f+bPAz7Lh9
PbS5t7sMJiHmHrfj2RHmtz7mxYwQmtmcjDCIiIqK6XJ4h3SLrl1rWTW8vtEu+HmBgiXdrwzJvahV
OmB2Yuc1U/fW7+y24JE2La9qGb5C5r9iOJzZJt7jqnaDGe5fEQykACgGI1yqQHFH7lg5EREREVOV
yWROyjXk2WOwuY9dIMplsfZqapVXh5hwl2l/Sxh/8wwmhzeX9BiLPYXfjzr8WocKvjr8+VlI3n8e6
J+PSyePROUKNHvBrd4tQMpImIiIj80D5GhQ4xNfdPlS7g7jxXC4dGvDYLoenC4Rv+f
YcLrMUfeTKeQcsQj46qhnKXRuuQtPbwmfcu8cqxOLjlnx9z9KMGJJPg7LunW/enEcOsfyJE5ERET
UGB6l3mcrPJ5zzuJEjrUmKaNVAr0SeB/W2rDMuw4nzA7c9FMRDhqrApeFVybiykxdHa+iUPLeSTV
8tRubn23F5A4RuCIjdMe00iniL7+V4Lta1kZP6xKJqZ24vRUrRY41K1+Kjg5bqr73tNy0v8e6
ToIaa+0u3Oq0+M222u/DX34pxyZJ8vLTdhBK3st9NeTZcsbSgOpAGgP/bbvL2NhSituRXYm2xdM4
oXis90T203ghTZeiWe/9rh6nWQLpbrAr/HhLbhEdERERE1HJdkqaF0u128aDRgbxSx8kv4DIGZWNzlyKP8GgE6tNbfN
unVp9MP2PP4z46mg59hTb8couM/ouzMWsHSbMz7ZgwqpCFMnWWTg3r9IwnosOgSMGUX+MKgLWpt8uzxwaZMQIm3WwjDQHC
GwisNcJ84PGNx4vmtoTlJ8keuDW/sKfP6PaUAjEjV4MvRiV6bYhARERGR/2I1CgxyW2oOuIS1gxKKkgbHa85JS73
ZyZuAVh5Mb8yz4etj0oyfyS5i9k4z7l9nhM3L1r0igHW5vlvkU+j49Zwv3/OTAWg5I0uL9
XtOTxjw9ZkF0qPSk2N1OlC3/9vURSop6kU2BGn2gsujIRJ6alYdnYJHTQc7UGERERUSCNSve937q37Q
oil4y01wv3Rq12mDa6RLx+KaGNZ/6PYfBdKgTRREvbJNmm0elazEyrerE+FR/PTrppZ2ul5/0bC7
RnB7fVIrTZdIZnQ8vjcfzg2IxOlOHGGajiYiIiIJCvkXWmnM2uMSqFMdxsxPGypph0h14toHMskxu
tUau9G59/xIpdRdJMZKzGs2mAWgFM7STdi/h3ZqZD3q/nbNgpG99/DtRX/ztCJeCentLs9I9eOjU
2lyUnyvHFEavksXt7RuHS9NBtlEZERETUUgwwaKB3iw8iw2bs7sPn9vvKW8+1jdRYtbEZ7sPn9vKW8+1jdRDYXA5mOtUYsMpss
dIhwuX/2bgdJKF2bKspaTO0Rgzw2peGaAvjqoTtIp8M1VBsweEgf3P4/9JQ4UVnipAaeQ8e5+6Tr
j8W116C9byyLvyr4przIkGpHtKbZjxh8lksd6xKnw3EA2GSMiIiJqCiqFgEvTpNpNnpX8539ZYH01w
v3Xq1qGC60ini1p+LkPb5OYxaWoATZu/7Qv97pxmFFTVBU4RSwMxBeug1CjzSNwYHpqbi12uTsOe
GVIxM0yJOq0BWonOc75u3U/M7UmrH6jPS6oEH+0R7PK99jAqd3dYbO8SqjHZj7S6qxGeHLT5
//3yxu0T8e6cJo77PR4mtZjJIrQA+uDQBOhVnPImIiIiayuWyivRr5I6I1oXc9lIIOpwDQYpNiI

9A2VYdqqqVHdPsR1/WlOMSqc0Q33YaMf7sqzljKxoZEbXBFWRKgX6GTSS4GWkbGaKpd6h6/39Fsn
XvaKduMjHSW50pnRcf2pEqfdBox03/1SEkd8X4MH1Roz8Ph/HTPULqPcV2zF6WQH+tcMMh6yo4pk
BevRJYFMLIiIioqY0SrZuemN+Jab9UuyxVLQ/m4+1Wi0mmLbYXXhTtoXQriI7XnLbF/pCd2T3YCU
zSom/9fbMWspdkioLptmELCQZbS7Ml601vjnDAcHHOhZ5qfdPZ6pmHOvD5hRRUO7EvmI7HlhXgou
/y8eK0zXBuKlSxOydtW+5ddbixCMbjLhsab7HiRkAbusSiQd61f37SURERESB1T5GhSxZQuOHUxW
wugUTBp0CbaKU8pdSK9Fi2s59fMiCggrP9a5v7i3D5RlaDErS4IYfizzKMl66KBaRqrrnFIalaqA
UgAuJ7sOlDuRanUiN5B9PKPn8sEVygkuLVOCKRN/r24enaBGhFFB+fmDPWV3YX+JALx+Z4PW5Njy
+qRTZpXavW6fJLTpWjqcHONA2WvqnlmN14rXdZnx6yAJvy7QNOgVeHRaHCe0jPL9JRERERE3izeF
xuG5VoaR7t7sBBrXPpA21fC0iM+0tK+3unt9KcOOPRdiUL13nPDpDi4nt69cdOUat8Cjh4H7TocX
hEvH+AWmJ9/Qe0ahtrkSnEnBJmrQE3Fepd2GFE7f8XIS9xfULpIGqyZd39kp/N/+XbUH/Rbn44ID
```

3QHpS+whsnJTMQJqIiIiomfUzaLB2QrLPfaTlDW6pdQnZYLrcIeKHU+V4cpMRs3eaUFpLl2V5Vjp
CKUi6b+eWu/BHnjSQHpykwSejEvyaSZKXev9WR6l3fcuFyTujzQV7LV3Z5ZafqsAZS02Uq1MCd3S
NrPN1ozOkEyo/nvEeTL+w1YRSH7OSF3SJVXn8zM8OW1F0vvv7bzk2PLjeCG/N4NtGKzFvVAI+GZU
Ag44VD0REREShoF2MCiuuScL0HlEe37s4hcF0axZSZd6VThHfHC/H8pPl+OWcTVKue7DEgU9GJXi
8xurwzErf2ysKAoA5u71nq/sb1Fh4VSJi1P7NJYxM0+I1t591Yd30vmI7XtxuwrpzEXBuOAeHKML
hArRK4Lr2EXhjeDy0SpZ/1FelU8QzW0rxySELVIKAO7pF4tG+MUioI8B8d590vG/qFIkEnRJFdfy
8KzN1wKbS6q83nt8iS6+p+f3YVlCJz7Ola7FVAqDXKKDXCEiPVGJqp0hM6xIJpwisOlOBHGvVBE+
5U8QHByy4s1sU7l5bDPn8QGaUEo/2jcHNnSOh4e8JERERUcjRKgX8Z2gchiVr8PimUhRUuHBd+wi
PJsXUuoRMMG20uTBpdaHHmuYLvjtRjpcsTmTIFvh/fFCalY5WCbi/VzT0GgV+PWfDNtn79UlQY/F
VBsRq/E/KD0nRQK0A7Od/3HGzE3N2mfHKLvP5NbcCgJpIqcIJfHm0HGmRSjw3iHsE10eJzYXbfin
CutyqSgI7RLy734L52VbMyIrBPT2jvK5x315QiY2yMv576tm4q4NehU56JY6aqtLFDhFYm2PDte2
qyqydLhGPbDRKXtM1VoV1E5O9Br8qAPf1isazW2qaj31woAzrc23IL6/5XRUAzBoSizu7RTGIJiI
iIgoDkztGYmL7COSVu5DOxmOtXsiUec/4w+gzkAaqQtSvj0ozg1aHC2/IstJ/6RmFRJ0SaoWADy9
NQIy6JkjpEafCd2MSEa9t2MeOVCkwSLbF0ovbTdXNq3x5c28ZdhXVvS91YYUTB432Vlsenl1qxxV
L86sDaXcmu4iZ20wY9E2e17Xqr+w2S76+PF2L7nH136ZAXur9k1up9/+yrR6/m/8eGltrAHxHtyj
Eamq+X2ITPT7XI31jcE/PaAbSRERERGFEqRAYSBOAEAqmvztR7vFYtEoaZHxxxCoJNH1lpS/ooFd
h5TVJuKFjBO7tGYXlYw1IbORa1BGp/pdyOEXgwfVGOHys/7W7RDy/tRTdvszF0G/zceXyAhw2+p5
YaInWnqvA6GUFOGauvbPXOasLU38swumymv2b9xTb8cMp6TrnB+qx3Zk7zy2ybBBFESU2F17YJt3
eamJ7HS5Lr71xXYxagendfR/DiFQNnugX49cxEhERERFR6AiZYNpdnwQ1fr02CTtvSIF7PH241IH
t5zOEtWWl3fVKUOO/lyZg1pC4Otfc1oevdRERSgH/7GLD2VvTkHd7OhZcIV3fvavIjnf2ea7hPlP
mwPgVhXh9T1n1tltbC+wY+X0+3t5rhtOPBlzh6tvjVly/usijudewFA2eG6iXZHgBwOIQ8YTbGud
XdkmD3YEGNUal+zfpMTxVC/dfj7NWJzosyEH/RbkottVM2ESqBPzfRfUr2f9rzyh4+5VL0inw4aU
JUCqYkVBFLAIAABc9SURBVCYiIiIiClchF0xHqgR8fFk8+hk0MOiUuKqNNAP45ZGqUm95VjpKlpU
OlouSNIiSZcy7xarwy7VJuDbFiSi1AlqlgGvaRmByB+nWRrN2mHC0tCajuvJ0OS75Pt9jyy6gar3
1M1tMGL+yEMdNDo/vtxSfHbbgrrUlcMjmDG7pHInvxhjw96wY7JySiju7SbsnLj9VgZWny3GgxI4
lJ6RZ6Uf7xfi931+ESvDo1m6sFD32FHykbwwyo+vXaiApQolbu0iPWwDw30vjuT85EREREVGYC7l
gevaQWHSJrVnrenNn6TZDi45bYbR5yUr38MxKB4NOJeC5gXoohKrA6ObOkfj52iT0iPdcnzt7SCz
itTVBXYUTuOXnIoxbUYDOX+Tgpp+KUWKrPfO8Ia8SI7/PxxIvZfCrTlfg9l+K8Pc/SnDSHH4B9zv
7yvDgeqNHd+sXBunxzoi46g7o8VoFXhkWi4uSpP/Hj24sxUvbpVnprAQ1xmTWb+9wuWvr2Ne5k17
p94TNg32ikei2Rv+J/jF1logTEREREVHoC5lu3gAwqX0Ebu0iDZ6vytQhXitUB50lNhG3/VLkkZX
2d41sY/ylZzQmnA+8asswJkUo8a/Bcbj395Lqxw6VOnCo1PvzR2doMb5dBJ7bWiopeTbbRfxpTTH
u6xWFFwbFIr/chSc2GbH0ZE1G9tvj5Xh/ZALGtGneQE0URSw+Xo4vj1ihUAh4doAevRPUHs95eac
Zs3dKm4YpBWDuJfGY2slzb2iFIGDOsDhctrSgOvg+XebE6TLpGuuGZKUvmNY5ElsLKvH1UavHPtD
xWgH/HZng9xZnbaNVWH6NActPVqB7nArj2tUesBMREREREUXgImWC6TbQSr10c5xEIaZUCpnSIxH8
PWqof+13WFbmpstLu6lume1OnCCw8asUv5zw7UF+gFIBnB+jxYJ9oKAQBV2Xq8ND6Evx4Vvqaufs
s+D2nEsdNDpTJ6qKNlSKm/lSER7Ji8GT/mGZZj7slvxJPbTZiS0FN87QNuTYsHmOo7oIuiiKe2ly
Kd/dbJK/VKIBPLkuoNdjMStTgrz2iPF57Qc94Fca1bfhkglIh4M3h8Xjj4jhUOIFyhwsWh4hKJ9A
2pqpDfEN0j1P71VmciIiIiIhCX8iUeX84Mh5xPraskpd6u2vqrLS/BEHAaxfHIcHLZ9MqgeGpGiw
fa8CMrBgozk8kpEcp8fWViXhlaCzUspftKbZ7BNLuXtltxqTVRSiUp1aD6JzFielri3Hl8gJJIA1
UbWk1eVUhthZUwukS8bf1Ro9gOEolYOGVifXK2j41QI+0SO+/J4/2rfk/bAxBEBChEpCgU6JNtAq
dYlUNDqSJiIiIiKhlCpnM9JAU392X+xvU6BqrwuFSz3XB05shK+2vdjEq/Dw+Cd+eKIcCQNc4Fbr
HqdEuWukzgywIAu7uEY3+Bg3+tKYYZyzeg+PucSpklzrgvtX1bzk23PZLMZaPNQQkuKzN0VIHrll
RgLxyl8/nXAioBydr8JMs2x6rqQqkByfXr/t2jFqBWYPjcMevxZLHu8aqMIEl1EREREREE1ERCJjN
dG0EQvGanQz0r7a6DXoWHs2IwIysG17SNQEe9ql6l2AOTNFg7IQlXZEiDTb1awCtDY7F+YjK+v9q
AlAjpUG7Iq8QfeZ5dwgPpnMWJ61YXeg2ks2TrpE120SOQTtIpsGxsUr0D6Qsmttd5/H/8o2/zlLY
TEREREVHrFBbBNADc2CkS8lBpeo8oGEI8Kx0IiTolvh6diJeHxGJosgbTu0dh0+QU3N0jGkqFgOG
pWvw2IRn9DdIAdn62NWjHVGJz4frVhR4NwHonqLFkjAG/TUzG4/1ifL4+M0qJFdcY0CfB/7XEgiB
g7oh4DE7SQKsEpnePwo0dmZUmIiIiIqKmEzJl3nXJiFJiXFsdlp2q6mAdqwmfrHQgKBUC7u4Rjbt7
6ev/MKZFKPDNAj+tXF1U/9v2JcvxnaCyi5QuvG6nM7sINPxbigFFadn9z50i8PTyuOkP8ZH89AHh
07e6kV+K7MQa0qed+zd6kRCqxenwSnC6RGWkiIiIiImpyYRNMA8Drw+OhOQhMKFh7NiWkVW2y
XpWmREanEWWtVttjiEPHdiXLc2iUqYD/D4RJx2y/F2CprNHZNWx3ecgukL3iyf9We3LN2VAXUWQl
qLLoqEckRgRk7BtJEREREERNQcwiqYNuiUeGN4fHMfRshSKgTc1DkCc3aXVT82P9sa0GD6s8NWrJF
t8zUiVYOPL02Aykdg+3g/Pca3jUBeuRMj07Q+n0dEREREREBQuwmbNNNXPLZ2lgfOGvEocM3l2QW8
IURTx4cEyyWN9E9VYcEUidKraA+ReCWpcnqFjIE1EREREREC0Cg+kWplOsCsNSNJLHFhwJTCOyrQV
27C+pCcyVAvC/yxOg1/DXiIiIiIiIWhdGQS2QfBuxL49Y4XSJPp5df58cski+HtNG16gmYkREREER
EROGKwXQLNKlDBCLdyq7PWJz4LcdWyyvqZrS58O3xcsljd3QN3FpsIiIiIiIiKicMJgugWKUSswoZ1
O8tj8RpZ6f33UinJnTXY7M0qJKzK0jXpPIiIiIiIiKicMVguoWaJuvgvexkOYw2V4PeSxRFfHpYWuJ
9e9dIbktFREREREStFoPpFmp4qgZto2v2cq5wArN3mhr0Xt4ajwVyuy0iIiIiIiIqJww2C6hVIIAv7
cTRrwfnDAgr3Fdr/fy1vjsfQopY9nExEREREREtXwMpluwe3pGS7LTThF4dKMRolj/zt5sPEZEREER
EROSJwXQLFqES8PKQWMljG/Iq8dXRch+vkLI6XHh2SykbjxEREREREEck0apNgm82GZ555BseOHYN

```
Op0NERARmzJiB3r17o7KyEi+//DJ27NgBAJgyZQqmTZsWkIOm+hvbRocxmVqsOlOzNdazW0pxdRs
d4rS+51I25NnwwLoSHDU5JY+z8RgREREREVEjg2kAmDhxIoYPHw5BELB27Vo89thj+OGHH7BgwQJ
YrVYsXrwYZrMZt912G7KystCnT59AHDfVkyAIeHlIHH7NyYPtfFxcUOHCzG0mvDwkFhplTWBsdbi
wu8iOb46X48MDFsiLweO1Am5niTcREREREVHjgmmtVosRI0ZUf923b18UFxejoqICq1atwj/+8Q8
IggC9Xo+xY8di5cqVDKabQQe9CjP6xGD2TnP1Yx8fsmDeYQs66lXopFfhnMWJfSV2OH0sp24TrcR
Hl8YjNZKNx4iIiIiIiBqdmXa3YMECDBs2DDqdDrm5uUhLS6v+Xnp6OtaumtdnZ2YE8lGYTqp9
jfATwuVaHc7aa0m6nCGSXOpBd6qjllcCUNDseaGdFVKkZ2aXBPtLgCNVxac04JqGJ4xKaOC6hh2M
SmjguoYdjEpo4LvXTpUuXWr8fsGB68eLFWLNmDd57770Gvb6uAw0H2dnZIf05XouqwNSfiur9/Hb
RSrw1Ih4j08K74Vioj0trxDEJTRyX0MRxCT0ck9DEcQk9HJPQxHEJnIAE04sWLcLXX3+NuXPnIjE
xEQCQkpKCnJwcZGRkAADOnTuH1NTUQPw4aqAxbXT49LIEfJ5twcESB85anR7jwcESB85anR7PaR+jxKAkDYanaHF
jpwhEqdnwnYiIiIiISK7RwfRXX32FxYsXY+7cuTAYDNWPjxkBosWLcLAgQNhNpuxYsUKvPjii43
9cdRI13WIwHUdIgAAZrsL2UYHjpkdiNUoMMCgRqKOa6KJiIiIiIjq0qhgOi8vD3PmzzEFaWhoeeui
h6sdfffVV3HLLLZg1axYmT54MURQxcoUZGVlNfqAKXBi1AoMSMKMuQ+FiIiIiIgorDQqmE5
JScHmzZt9fv/5559vzNsTERERERERhSQuiCUiIiIiIiLyE4NpiIiIiIiIiIiLyE4NpiIiIiIiIiIiLyE4NpiIiIiIiIiIiLyE4NpiIiIiIiIiIiLyE4N
pIiIiIiIiIj8xmCYiIiIiIiLyE4NpIiIiIiIiIj8xmCYiIiIiIiLyE4NpIiIiIiIiIj8xmCYiIiI
iIiLyE4NpIiIiIiIiIj8JRqNRbO6DICIiIiIgonzEwTERERERER+YnBNBERERERERERZGfGEwTERE
RERER+YnBNBERERERERERZGfGEwTERERERER+UnV3AcQmw2G5555hhcO3YMOp0OmDFjO
Kykq8/PLL2LFjBwBgypQpmDZtWvVr58++fj0WLFgEAvXrhyeyQYulwv/+Mc/cPDgQRQVFeG
HH36AwWBSBols8XjoIxJuXl5bjvvvvtgs9kCAIIiyPx8MMPo0PHs3yGcNRMMYFAAYPHoyoHTCqVQ
CACZPnozrr7++iT9deArGmOzVubi5WrlyJ2NjYpv2AYfysLFizA999/Dwb
ISkrCs88+i+Tk5Cb+dOGroeNy9OhRvPTww8jLywMAdFi4cGH1e/J/J63zjBGBNe7xsvGOMC8HrfGME
YE17v/cNguhYTJ07E8OHDIQgC1q5di8ceeww//vgjPxWWLF8NsNuO2225DVlYWevXq1cLSBGBBuM3OzSq4ZDTCs3BSbGXSUUWH
7d2PRokWYN28eYmJi8Nhjj+GLL77An/70Hu3fHmDFjmvvjhaVAj4lWq8Vbb72F2Oh
oAMCaNWvwz3/+0+NET7UL9Lhc8Pbbb/MGtGtIECPSa9e/fG/Pnzq9//Vfew9GjR3lh9VOgx2Xz5s1
YuHAh5s2bB71ej2Xmm2AGi6XmgH27Nm2YM2dOc3/UsNQ/J+vN43XqDHhNf7wAj
0uFzA633DBXpMeL33D8u8fdBqtRgxYgQEAA9A3bF8XFxaioqMCqVasmZcoUCIAAvV6PsWPHYuXKlQ
KlQCAVatWYezYsdDr9r9RAEATfccEP19wRBwLBhwxAfH99snyycCBcWNMFApF9YUVAMxmc/X7U1y
ocYI9Jk6nE0uXLsXkyZMrVcOxZdkxZ4Hu2CMy+HDh5GVlQWQW9Xg8AGDFiBNatW4wfS0tLm+ZBhq
sLERERHi8J6/3jROMMe4HlvvGCMS7UOMEeE17v68Zgup4WLFiAYcOGQafTfITTc3F2lpadXfS09
uLgAgNzcXqamp1d9LS0ur/vYq7Re5d68/H1WD/PLqHqhZhNjUmQx4NcW5x3hZCqyjoHLYuWTxOanfKkjc
Efn5+03yAFijQ569169ZBpVJhyJAhwTZzt9d5qFFq/Q49KJrRq9s27YNehXHZYNhdASVWaXRpqkPmlyV7ZXhNp8wCqNyPYHlbbEp5kCnZbNeYxvuHoF
QY8LrfeAEclx4x4vq+vDS/J/dGWyYnlb2sYNYnZ3S5mvpyn8w6t30rP3bgYvzEWGRvbzYxtnJ2QzrTKvPAF
++2712h2qv0COy5IlS5CWlaHw4GPP+4YjZ76KObNmaxAa2xdngH++vbbbyFx4kQkYNJyPbahAjcv
AgQNx5513A34tFHH4VCocAIIlyCmJgYnr8aiNf7OMPrfWj9HrfPPPg/U4dFixbhyy+w+/xNy5c5G
YmAgASELJQU5O0TvVzzp07D1VPSE1Nlcz5phTcdVVOH78OE78CBG8D9BCBXpcLsy
mqlQqTJs2DQcOHEBFRUhxKZ7mm5udi6dSsmTJQbJ+gZQr0uEyePBPmfffYZPv30O0UycOBE
2mw2ZmZlN9GlaDn/HhYIvmGPC633DBXpceL1vvGD8rfB6Xz8Mpmvx1VdfYenHeHChZg7dy6SkpI2cyPq6Mi8ThyrtgP
zZgwWLVoEURRhMpmwYsWK6gWgZMyWK6gYjY8aMwoVK2AymSCKIhYKuXIirrqucT5CixPoSksLJSSsLdywYQ
cTicyMjKa9oOF9hM1mq36f5cuXY3LkyYiNjYWVlRWOHj2KQ4cOAQC+//57tG/fHrGxscF8SEQNpMvsbhkZ
EADgcDrzxhu48bfP+bYb+bfip4aMCeM1/f6f6X7/F6Xz+C0Wg8Ul6uvzZ
apKWlSRpWvPrqq4iLi9Li8OsWbOwa9cr9cuiKKb9cL++eaie+TlLw2qOb36pXccvatZhc7Eu1vz7/HfMMpIoK2nnMmjQgbxWbFJ
A06dPR05ODvLz8bBGATE4OvvqqYT9fOArGmOzbtw8vvqYHA4HBEFAkyZmeAWbzFkTTXrwx
LwRiXTZs24fXXX4cjcgBBFEenp6XjgqSwBfRr275viIYSdY5y+n04kJEyweMyMJKeW+exwHhwxW7rY3E5
LrbfeCrvdDrvdpEjR+L+HHWq1vU8s8Xrho6LoWFhbjjjtQWVkJswMxRED06FM888wM8
bIxhjwut94wVjXHi9b5xgnb94va8/BtNEREREREEfmKZNxEREREREREREZGfGEwTERERERER+YnBNBERERE
RERERREZGfGEwTERERERER+YnBNBERERERER+UnV3AdAREEwLJlyzBz5Bz5szqrzU
aDfR6PTp16oQ6vt9jx07hoIIaTxNMiNTURTTUREFEKmT5+OOz
MOBwOFBUVYdu2bXj11VexYMCvPLKK+jSpYtf73+ONAGHGDhwIINpIiKiAGIwTUREFEKmT5+OwOm
KPn36VH99xX13jdMuWLXj44aS3zxYfxyCOP4KuvoJOp2vgIyQiIqIIC3kUXXYS77roLUHPK5WL
yJQAgOzsbM2fOR+g8nvx0V8qe5bEH6HJ58EEXSGHnS8mfNrfEP4vD6mPPo9Mn2ZPp+NhR6gXXMp8VmxXL
/Z//+/ZgxYwGeBfNOtT8fWrVub9gMSERERERERPQmBTBEREREREdFJYuoC8
aa67BI488gokTJ2A245557QKgGcA5DBg5P6R+P6fqqQZ0w3QoYPCCb1ghRdeQP+QEEA27dvx1+/+i
YTCbcfffeOCBB1Bm0YtWqUqXI3sRWMiDqSLIx9rgDwKtZ9QUrOZLj6HYcaP6JtKpt1c/nPy5XDtD
S54wcORJ33UX1qj0ZAtG0aF9+vDgIGVw8SVE1bN2vMUfnv3xGtRuFxAwFJMnkS7Ww6FH9sw6TAltG
kyZNx66Y7JEoB5G+a7gMZReFEJgWaL2WmiZiIiwbIwROQh5yZS77pcGNOoHTzdGmDjzC6QQiFIyqdzoqmdjbRM
d75WdnY2TJO9izJgxKC0dhBFohlZXRGhpOkpz3hdFWAS4WdMGYTVrUsGb33mnqCmkK2SWM9aqaxeDg4PTSPQ
T3nnnHfz888888wUjyS51ksjrf6+TJkwCAl56eWbw0R4yMUe9dpEbTREmysXq+YHXTHMJQSAvLw9ZZF25
EADz11FPYPYtWsXpk2bkhbm7uiGyMhKCCO1pp5w+5lZkawAJbMQ+7QAA3H///ejRjUcBjRLRXgB8CmX+tUd/aQEQ2
wDKaJiIjCwoVKwBUZ1lMpmefNmTJ8+hdOnT9vQLt8+69fiu7Ul1rzuwlpnQuWHhyyzsyLL5YLSY/Pwa3Q3KmVfHyoyY
iImq5uGaaiIgfffQRU1K1vIw6gThpZMiWqDVcZA8+4cXp0FjnYZnKlZWzYVVjbJAjhGYzD1pTjTH0vUjgJxX+0KHmhm
```

```
WLFjgtSy8pKQkkB+BiIioxWFmmoiIKIRs3LgRZ86cgdPpRFFREbZu3YrNmzcjNTUVc+bMgVarhVa
rxcCBA/H555/D4XAgNTUVu3btwvbt2xEbGyt5v65du0KpVGLevHkoKyuDVqtFr169kJGRgaeffho
PPfQQpk6digkTJiA5ORkFBQXYvn07AODdd99tjv8CIiKisCAYjUax7qcRERFRMC1btgwzZ86s/lq
tVkOv16Nz584YMWIExo8fj6ioqOrvFxYWYs6cOfj/du7YVEIoCMPo34NdGImZPQg2YmQBWolgD5Y
hWIYdGF7Y4MHLb7KwcE480WRfMHNdV0op6bouy7Jknuf0fZ9t2/5nz/PMcRx5niellKzrmnEck/x
99d73Pfd9533fNE2Ttm0zTVOGYfjeAgDgx4hpAAAAqORmGgAAACqJaQAAAKgkpgEAAKCSmAYAAIB
KYhoAAAAqiWkAAACoJKYBAACgkpgGAACASmIaAAAAKn0ACck1zLUIiBsAAAAASUVORK5CYII=\n"
          },
          "metadata": {}
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "rcParams['figure.figsize'] = 18, 8\n",
        "decomposition = sm.tsa.seasonal_decompose(y, model='additive')\n",
        "fig = decomposition.plot()\n",
        "plt.show()"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 577
        },
        "id": "pNwsFgUtNfES",
        "outputId": "8b48d365-7387-4467-d30d-f09174626af9"
      },
      "execution_count": 16,
      "outputs": [
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 1296x576 with 4 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAABQAAAAIwCAYAAADZOcbUAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nOzdd3xT5f4H8M/JTpqme0DZ0DIEyga
RKRsHQwX1ioqKF6+iXu9173EdPwdXBTdXEQcoIKCAILJFVls2SFlldCZtmqTZyfn9UZr2dBe6+bx
fL1/IyTnJk3KanHzyfZ6vYDabRRARERERERERERVGzJGvoARAREREREREVHdYQBIREREREREREREETUj
DEAJCIiIiIiIiasYYABIREREREREREVjDACJiIiIiIiiaMQaAREREREREREEzRgDQCI
iIqIGlp6ejgEDBuDll1+ut8ecPXs2BgwYUG+P1P11i8/PLLGDBgANLT0xt6KERERET1RtHQAyAiIiJ
qjtLS0rB06VIkJSUhMzMTTqcBoMBnTt3xrBhwzBx4Rg4kRotdqGHmajcd999+HAgQN45513MGzYsEr
3veuuu3D06FF8+OGDhDwxYD2Nkw//HHiItfBAYEmTJkEQhPocIhERERRUVQ4sWLFiAzz//HH6/H927
dsWrrT7IKlJP1PxkOrQ+ODMpyNavdCmP8yKsY5CqnDoBZhBQHTQPp3nlVrAHmntNFzSmRqkbYz
3TtLNwlK53pRSX2A5kVPuhhHNRhqbfbxIl33A6HjF8B4uUCTHQ8s8wWYoHgnHdF/BhOm7M/s2q6c
F9CSwPaO6m/EwsppbspyF6Jgv3DabTcO8mtqRS8ViUHN5Nv6XGrF2+y/eqZvQOYT9R5HmM1KG9yO5Kr
du7CGzCpXZYPDxrrb1dEt5ucU9WGN0sxh9SQGlmzQwZpDRmAQYk7RdtSMeWS4jnY/PSsd/BpOAHFv
3KmvxZM+wTHsxK5lgR4eLAkjHT3Mn75XC1r0tI8IFhJ/6E5iEUfXmeuFyybq7MKYwFTeBnwmxRkAKNw
gMFnRH8cB8IGtt8sDcn2YDVWwiM/1M9vAadZLKc5nuV17sYJ5C1gHq4etQEPHJhP1J6+AL3DVk1ycQHpD
nSdAiEjIlkp3X8W79hV1fZjQ8qEwMHk+nSwaK1gGmLQ3X7d8R0rCObYuywQCD5eDIKNyv5lVqLQsOHS2
xxVyFyDatZeUYWzlzmiP2ZrDR30ZXXlLqoIqJZTm5BEXu5+2sWHjv3vO+jk9s+rMwv3Hqr10rfVSRwBBd
VSJTFrOekMuKb/hRrYbg/ffVWNCkWiW8OvlfDmPPSnyC6EL0Lr
uNWFg3ufAUnuYQq7f/DkyHFV5cqvMEdjZIEX0hZ+8iiFiMrjOXMRmrMZQ7WdDyOqY1uSu1Gz5AhbYZvl
uxMZWEJYRHiOrQpU08E6sKG4T8QlpEPoNaXtwN9VJYhn5vmDkJTAA6PqAcwDmn2mcM9X+pzZ/R4vs40Ep
KjSwBDO8L3wBTTRk6qgaA2j0MpVzzRrmCGGbWfxWfp5B3OAIvEIHK5aT0n7GBdrbg+yc1xU2MkVDi/
lyGtfbO+Px1cLm+KzPeL1xv5yZW2Jaq95UK54a69/DFUpWfSRNRVpTJHtQNN39Agz5aDa7nj9NgTcSl
TwZ7D75lTiIdF/5WmICSeKjOh5N/JxAHN34kWH9bqoFrxNTSHt9KJ6VQQmYu4hOUa2Fvt4iqYFB2jqfFj
bw2Ag/nEm9+7xiTi5yAtBz51Mc/+/C9HYq7EtqoHTtvVHFzG4QAWSKfyF9H0DBFOU6OvhCvQjmt7DpP/5
hwqLAO8P1MQ3
```

```
+fDmioqKwePHiMg1GzGZztR83KSmpzGOOGzcOjzzyCL755htJAPjNN9/g2LFjuPbaa/H6669DJpM
BAO655x7MmDGjRs+3Y8eOSExMxP79+7Fr1y4MGjRIcnvR9N+SQebcuXPRqlUryX5+vx+vvPIK1qx
Zg+nTp6NHjx41Gkd1JCcn46OPPkL37t3x/vvvIzg4OHDbmjVr8NJLL+G9997DW2+9VeuPTURERFR
TsoYeABEREVFzYTKZAKBWqr6ys7Px559/Ijo6Gnfffffbfbfktk6dOuGmm26C2+3G2rVrJbcpFIpyp72
GhoZW+7FbtGiBe+65R7Lt6quvRmxsLA4fPizZvnr1agiCgIceeigQ/gGFP4Nbb7212o9ZZMqUK
Kw74ip06dwoEDB9CiRQtJMFg6/AMAmUwWeOxdu3bVeAzVsWTJMEoiiiKefflo5S/gHAxIkT0blzZ2z
duhUFBQV18vhERERENcEKQCIiIqJG6K+//gIAJCYmQlqlUrl9wIAB+O6773Ds2LHAtvvHjx+OHH37
AtGnTMGrUKPTq1s9e/ZESEhIjR47Pj6+3BAxJiYGBw8eDPzdZrMhJSUF0dHRiFqyMZZ3PxsLHAtvHjx+OHH37
sBvLee+9h27ZtyMvLQ1hYGICKm3+YzWZ888032LFjBy5cuACHwyG5PSMjozZOp4DXr1ae/Pvb
atKnM7W63Gz6fD2fPnkXXrl3rZAxERERE1cUAkIiIiKiWFC2FvXjxIvLz8zXEFqPpp8WbCTFqUqlnZIDJJmrzV/dL9mbNM5CfQ/p8jpGFfeb+Gcb7YIB9SZh6Vxxzhpi7NEcv76Td8BxFJxAn
F+wHAP//5T7Ru3Rq//PILvvnmGyxatAgymQz9+/fHnDlzkJCQUK3HL3lL3RVkQul8Pv9wf+XlTdFh4
eXu7+FW2vjFqtxsSJE7F48WWksXr0ad9xxR6DSUS6X48Ybbwsa7VacffdyM9PR1XXXUVJk6cCIP
BALlcDpvNhsWLF8Ptdtd4DNWRn58Pn8+HL774dtL9SgeSRERERA2BASARERFRLVEqlejUqRP27t2L
3VrvhRkX0ej2A4mnFpRmNNRsl+QGFAN336dEyfPh1msxkHDhzAxo0bsbXbtWsyZMwdLliyp0VTgqgQ
FBQEAcnNzy729ou1VmTx5XoxoyVK1fijjjuuwKZNm5Cfn4/hw4dLOuuuXLkS6XbvPXDgABY
vXlytxyuqKPfKXebrVay2zT6/Xwer3lVv8RERENTZcA5CIiIiIolI
+VVWmFVXr7d+/Hx6Pp8zte/bsAQB06dKl3OOUvQ
AoNzKTZvNhrNnz5bZ3qNHDxPUFOD48ePVfhwiIhsIAkIiIiKhikshsIAkIiIiKiWtGzZErNmzYLX68UU//bD
0qNz99u7di9mzZ1d6XzExMRg0aBCys7Px9d
AQKPRXMrTqtTEiRMhiiLmz58vmR2sVw7j59j64XGmWtvOIU
VV9V+rHbt2kGv12PLli2BykoA8Hq9+Pe//w2gGv9dj8bqrY5U0bMG0z/DcuWtMR2wKuWugwBVtV
tbVPdvZTbPy0S1zObp2dtuSQhBWGI39gC4dbA2M/NUvMggMmQ3/nBsHdDbGsxJZDZBzpSgcsL8
dKmsF2tttCN+Wc6IPT
FOlRfxqEx65jR4aGOWudunN1ZqsYdGmOF1S9ElWH6eUXnzR28l/JUeMKfvRB6ilkK0jR+kJTyrLC8fovuXJSNPR
Ymo2XGnrUo0bGklmuoWm05UqNxtFbJJhVC9sfw
DDSCRzLpLIKeOQPnyI28=
lw78inx8rwNAWatzYTluPI6t7J/O9eGRHHrZnugPb/m+fFZPaafH3bkHoH6VieFMLdma58E2qHRv
OO5Hp8Je7j90rwh6oNPXhYK4HR/I8mD8kDCp50/g3yHX68PZ+K744VgDPxafZPliOm9rrMLWDFt3
C+KUPEdUcpwATERER1cC7B6xINkrXMruhrQZzB4fW+Qd8hUzAs30Mkm0rzzixz+iu4Aii+iGKIp7
dk49PjkjDv0iNDK310iq4h/7IQ1ozaQri8YuYe8CKwSuzJOEfAHhFYNlpB8auNuLaX3Kw+IQdPj+
nqF6qz4/aMGGNEd+k2isM/yry4ykHpm0wweqp2XH1zeEV8d8DVvRaloWPjxSHfwBw2urDOwesGLw
iG4N/ysL7B62cUk9ENcIKQCIiIqJqOpHvwfsHpevuDW+hxhfDw6GQ1U9lyQ1tNUiMUGK/qTiEnH/
Yhs+Hh9fL4xOV59VkCz46LA3/ItQyrBofCadXxLg1OYEww+IWce+WXKydGAVlPf3e1Da/KOLPLDe
e3JWPQ7lVNzdJMXowe1selp6y4/vREU32eTeUfUY3ntmdj8riLo0ckm7spW1Od+H6tUb8OCYC0dq
Gm5qdZfch1eKF0yvCK4rw+AGfH8h0+PDhQRsu2Ct5EhcdMXvx4l4L1px1YvnYCAQpWddDRFVrFAH
gRx99hN9//x3nz5/Hq6++irFjxwIA3G433nzzTaSkpAAAbr75Zvztb38LHPftt99i6dKlAIBevXr
h6aefhkqlqv8nQERERM1eUVdTd4mKjBitDAtHhkNdj9PKZIKAZ3sbMG2DKbBtW4YLoihyiiE1iOQ
cN947YJNsC1fLsHJ8ZGCq4kv9QvDs7vzA7XtzPHg1yYJX+ofU61gvh80LrDjtwPrzTmy44ER2BVV
oQ2JVyHL4kZpftspxwwUX/pNswUv9ms7zbmg2jx/3bslF6eK9IIWA4S3VGNtKg1FxarTWK+Dzi7D
7RNg9IjLsPty7JRcnLcWB2n6TB2NX52DZmEh0DKndj8Lnbd7AOVHypfiszYcDJjcOmDw4kOtBVg2
rF6O1Mphdfsl7T5Fd2W7csTEXi0dH1Ov7EBE1TY0iABwyZAgmTZqEV199VbL9u+++g91ux/Lly2G
1WjFjxgz07NkTPXr0wIEDB7B06VIsXLgQwcHBeOKJJ/D999/jrrvuaqBnQURERM3Z0lMObMlwSbb
9Z0AIQtX1X3kxoqUaWrkAx8VV7zMdfqTZfGgX3Cgu7egK8+lRafgXphawcnwkuocXr1P2j25B2Jr
hwrpzzsC2Dw7ZEKGR4aGr9JA3woo4r19EitGDLRkubE534s8sLXxiboX7h6tleGNgCKZ10EJEYcX
Zp0dsWHde+rrx34M2DIlVY3QrTR0/g+bh6V35khAPAN4fHIpbO+nKhF5ymYBgmYBgJRCjk2PddVG
Y9ptJsmzDGasP49bkYOmYCPSKvPziEb8o4vk9Fsw/bKt65xoIUQn4d89gzOqqh8svYnWaA8tOO7A
53RVoeAIAm9JduG9LLr4cUX+V6ETUNDWKWuGePXsiLi6uzPZ169bh5ptfvhiAIMBgMmDBhAn799df
AbRMmTIDBULjY9i233BK4jYiIiKg2mV1+PLsnX7X7JteAs1bmrfMI0MVHIBfaOki8DvzOI6gFT/jE4
ffjrtkGz7aEgYeoRLz09BEPDxkFDEleqK++JeCyatM+KcreHXBPT6RewzujH/sA3TfzOi/XcZGLM
6B68lW7A90w2fWHG4Mq2jFrunRmN6Rx0EQYBMEHBtnAZLxkRi79RotNRJP3bN3paHjGpM9bzSrTj
twKJUu2Tb3Qk63NU5qFoVb5EaOVaNj8SYOLVku9Hpx+R1Ruw3Xd7rptcvYva2vFoN/1Qy4KGr9Nh
3cyzm9AiG5mI349vjg7BsbCSOTo9FYoT09+vnNCce/sMMv8g1AYmoYo36a+LMzEy0aNEi8PeWLVt
iy5Ytgds6uLg6ZmZmV3l9qamrdDJSaFJ4HVBLPByrCc4FKKn0+/N9JJbIdxR+4lIKIOS3
NOHEir76HFpCgVGI7ise0PjUbfcWq1yKjmuFrQ+W+OqeA219cRRWn8aOD8zwq+rG91FGG2QfV8KE
4vNme6cbVyzPxZEc3xkdLQzGzB8hxC2ipFhFUy59cjG7goEWOQ1YZDlplOGqTwemvfgVVkFzEwFA
fprbwYmCoHXnnTKjoFaHoefsvPm+j048Zv17AvO4ucOZm+TKdAuakaIAS50o7rR/3RBiRmmqs0X2
90hZQeVRYnV18EpndIm5ck42PujuRoC8bnGW7BGwyyRGmFDE8wofSxd6H/krFs8dU2JJbsxNTJYj
oGORHiAJQCIBcEC/+CcRpREyO9aKlxg7j2WxU9Czf7gTcb9fgjKN4UN+dsMNXkI9/dfCAq0HUP75
XUJGGPBfi4+Mrvb1RB4C1raofBjV/qampPA8ogOcDFeG5QCWVPh+Sc9xYmpEj2eefiQaM7tmqvoc
mMVHrxP/OFa8DeNSpRXx8u4YbUDPE14bK+fwiVu3LAlAc2s3uHorOCcEVHhMPQB/twD+25yHPVRy
62HwCnj+uxm6XBtFaOY6ZPfjL7IXRWbjwWbhahuVja2fKJgC8uCcfHxyyVdpUojydQxQY00qDsa0
1GBStgqqa6V08gDMKC15PKW4itDdfjlX2GDzRy1DxgVcon1/Ew78aYfUVV+ipZMDCMTHoGXFp58A
3CSJe2GvBh4eKq/XyvQLmHA3CqhJT1p1eEfMO2/DeASvsF7vsRmlkuL9rEO7tEoRwjRz7j6XihTN
h2JIrnd4dqhLQLlghOa8MSgE9IpToGa5CzwglEkIUtTJVd3VbH8avycE5W/Hv35IMJVpFheGZ3oZ
GOa2+ueJ7BRVp7OdCow4AY2JikJGREZgenJ6ejtjYWABAbGyspOLvwoULUgn9iNrzDbQKr/fmjkhVP
yasro9OGxP82SD3Ptg+V4rEfFAUd96R+tggAExnbU7IXZ5W+QNQnpyrTuvFMSPmjkwB0JQVUeN6G
NFn9MUuHB7XnYlC4NUH5Oc5Z7TK7Lj79vzcP2ydGX3UF33Tkn3j9UvSmbURoZhrVQY1gLNdq4MzC
yR6dLftx/9QzG9kw3tpZYS/TNfVZcE6vGNbHqSo5sPMwuPzLsPhidfpicfuQ4C/9fJgDXttSgX5T
ykpoR+UURJy3ewkYZJg92ZbuxM1s6PffFfiFIvMTwDyichv5KPwNEEZhXYspursuPSb8a8fOESKR
ZvXh6dz7OWKWVqDlOP/6TYsV7B2y4I16HnRfUOGiVnrtt9HKsHBeJ9ob6+YgdFyTHirGRmLA2R9K
Q5t0DNvyc5sQTvYIxpZ2WQSARBTTqAHDcuHFYunQp+vbtC6vVirVr1wYahNw4PPh2vGwzh8Xkb3oY
HB+PHH38MdA8mIiIiAoDXkiyYe9CKYKWAe7sE4YGr9IjUyCs9xuUHVp5x4PsTdmw474S3VInQO1e
HQqNo+A9UISoZuoUpcDiveO20XdlujGvNxgJUP744WiD5+00ddAirZgDdMkiOZWMj8OmRAryUlA9
XNZbD+yvfi48P2/DwZQTwdq8fj+80V3h7tFaG/lEqDL0Y+nUNVQQCrdTUy1tfTS4T8OmwMAxdmR2
obPSLwKwtudg+KRrhVbw2leb2iViUWoDDuV5YPX5Y3H5YPCLy3X7YvSK6hynxxsAQtNZf/kc+l0/
Eg9vzsPSUo8J93kixoluYAncnBGFaR12VX0a4fCLWnnXi29QC7Mhyo6D0i20Jo+LUeKBb1eFyVQR
BwKv9DfBDxEeHi89fk8uPEauyy+20W5LDJ+LzYwUApP9WnUMU+GlcJFoG1ezf8HJ1DFFg+dhIXLc
2B/nu4p/f8Xwv7tuSh7f3WfFkr2BMbq+9pC/BiKh5Ecxmc4OvFPrBBx9g/fr1yMvLg1arhUQ
ffIC4uDi88cYb2L9/P4HBmcMVn5KRc1GtVkbqtyhkurbm91Bx4MGnf70WvMhjPMbEDgMD2RC89B
```

osHZiVAOOqHnha0PFnt5lxsclwpN+UUpsuD76ku/P5ROx4FgBjps9aBesQEKoAl1ClQhSCBj4Uxb
MJaqbLjWIT8334JoV0iqv2d2C8ObA6gVvtXk+vHBxDcIiCgHYPTUGHaoxfdTs8uOW34zYk1O9pj8
yAXixrwEPd9fXeHquzy/i79sqr/yrLoUAxGjluFCD7sex2sIAc3AdTJEWRRFP7MrH50fLhoAyAbi
3cxCe6WOAAODr4wX4+IgNGTIaaa4AACAASURBVHZpYDk4RoXFoyNgUDX80gtnrF68tc+KJSft8Jf
z1hesFLD1xuh6m6J8JeF7BRVp7OcCf/uJiIioWXF6RczcnFfpdDK3H1hVwVpjpXUJVeDWjjrc3Tm
o0a2vNzBGWs2RYnTD5ROhZltRqkMFHj++PWGXbLuvi/6y7lMtF/CPq8q/jxf6huCxP4un7f50xoG
70p0Y0bL6091FUcTjO/Ml4V+sVoZnejdMA47n+hiwKs0RWGvOKwL/SbZgwYjKg80chw9T1puqVS1
ZxC8CL+61ICnHjflDwxCsrN7rmF8U8egOc5nwL0ItQ3yIApEa2cX/5Dhi9mDdOSd8lZSWeEVUGv6
FqYVAo4ye4Ur0jFCik0FRZ2vYCYKA/xsYAr8ISSX14BgV3hoUih7hxV3WH+4RjNnd9Fh22oHPjtr
wV54bt3QsDI+1jWBJCABoF6zAx0PD8O+ewXh7vwU/nHJIgkCrR8R9W3Lx63VRl72OJhE1TQwAiYi
IqFl5fm9+mQ/H0ztqsSXdhUxH1dPNgMIpZzd30OLWjjokRlzaovb1oXWQHHG64ooapw/Yb3JjQDS
XRKGasXn82JLugs0rQhQLm8uIYmF6oJYLiNTIEa2VIVorw89pTlhKVORFqGWY3E5bZ2O7K0GHr48
XYJ+p+Pf68Z35+GOSutpdeH867cDmUg1H/jMgpMEqt1RyAc/2NkgqeJeddmBOd3eF06jP27yYvM6
EExavZHuvCCVmd9PDoBJgUMkQrBSw7pwTb6RYJQ2MVqU5ccycg3lDQqt8jRBFEU/tyseiVGnQ21Y
vx9qJUeWudZde4MO3qQX4OtUuaQ5TmVCVgGkddbgjXoce4fX/WisIAt4eFIJhLdT4I9OF4S3UmNh
GU+44VHIBt3XS4bZOOhw/noqEhIbtBF+RjiEKfDIsHP9O9OCNFCuWnS4OcJOMHryebMGL/bhUBNG
ViAEgERERNRs/pznKTOea0k6LT4aGweUDvjthx9yD1jIfTqO1MiSGK5EYoUIrbw7+1r9Dk6iQEAQ
BA2NUWF7iA96uLAaAVDMrTjsw54+8Stc5q8ydCbo6bYwjlwl49+pQjP4lJxBopeZ7Mf+wDf/sWXV
DEIvbj2d250u2DW+hxtT2dRdaVsdNHbT44JANB0t8YfFKkgXLx0WW2feUxYsbfzXifIH0tevqGBW
WlDMFNTFChT6RKty3JVcyffp4vhdjVxsxMFqFB6/S47o2mjIVdl6/iNeSLfis1GtpnE6OleMrbnT
RMkiOx3sZ8FjPYGzNcGFXthunrV6ctvhw2upFjrP4C5hhLdS4M0GH69toG7ypkkwQMKmdFpNqEGI
30u+EJDqFKPHF8DA4fCLWlFi/8b8HbRjeUl2jCloiah4YABIREVGzcNbmxUPbpU0/2url+O81oRA
EARoFcE+XIMxI0GHDeSfSbD60D1YgMUKJ2BLr+qWmZjWJ8K/IoGhpAPhnthtzGnA81HS4fSJe2Ft
1I4TKyARgZpfL785alb5RKtyVoMNXx4sr0t7eb8XNHbRVdrl9I8Uiqf5VyoB3rg5p8MpemSDgxb4
GSUOQjekubEl3SZp27DO6MW2DCdmlKphHx6nx9bXh0CnKr2Ic3UqDzTdGY8bGXEnICBR2DN+VnYt
2wXI80E2PcLUMSUY3UoweHDB54Cg1lzdKI8OK8RFoF1z1x0e5TMDIOA1GxkkDJqvHj/M2HyI1MkR
pG89aqs2ZIAiYd00ohhizkX5x/UIRwN+35uGPydGIbMDGJURU/xgAEhERUZPn8YuYtSUP+SUqXRQ
C8L8R4QgpVRmjlAmY0KZhK39qU+l1AHdluSGKYoOHG9S4nbN5cc/m3Go3kqjI+NYatKkigKstL/Q
1YGWaA3muwt9zu1fEuNU56BamRGu9HG30CrTWy2H3ijhr8+GszYtzNh92Zbsl9/NI92DEhyjLe4h
6NypOjSGxKmzPLB7jy0n5+L1FFARBwPpzTszcnFtmTdMb22rwxfDwKqdAtwtWYN11kXhshxmLT5Z
t5HHG6sOTu/LLObJYmFrAinGRl/0zC1bK0DWsca2jeiUI18jx2fBw3LDWGKigzXL48eC2PCweHcH
3CqIrCANAIiIiatI8fhEPbs8r8yH/xX4G9I0qfy2t5uSqMCWClUJg+qbJ5ccJi7fRBBxXKpdPxDG
zB50MCgRVs+lCffn9ghOztuQh1yWtKFPKgOvaaKGUAQIACIV/Orwicpx+ZDt8yHH4Ybl4rnU0yPH
2oOp10K0N4Ro5Xuobgkd2FDcESbf7kW53VXKUVBu9HI8lXl7DktokCAJe6heC0b/kBLYlGz1Ylea
EyenHv3eay3R0vb2TDh9cEwpFNSuVdQoZPhkWjls7OfHhIRt+v1D9n5dBKeCnsZG4KpyvJ03ZkFg
1/pUYjHf2WwPb1p134dOjBZjdrfH8PhBR3WIASERERE2W3evHzE25WHde+oF2TJwaD1bQUbS5Ucg
E9I9SYWOJBgc7s9wMABuI2eXHF8cK8MkRG4xOPyLUMqyZGInOoQ3772Fy+rDijAM/nHSUCcsBoFW
QHAtHhlcrNHd4RRR4C59bfVcPzbjYECTJeGmVi/83KKTCKbMNpV+UCje01eDnEp3JH9pe/pqMc7r
r8XI/A2SX8HMf0VKDES01OJLnwUeHbfjhpF3SFbmkaK0MQ2LVeKpXMBIa+Nyl2vFUr2BsTXdhd07
x7/8Le/IxKFpVYeMZImpeGAASERFRk2R2+XHb7yb8mSUNM1rqZPhoaNglfUBuqgbGlAoAs92YkVD
367JRsfQCHz46bMNXfxXAVmK6psnlx0Pb87Duuqh6PyedXhFrzjrwwykHNpx3wltBj4+xrdT4ZGg
Ywqu5HphWIUCraJi1w2SCgC+Gh+OOjSYczvNWfcBFGjnwcI9gjG/dOKf/P9/HgNVnnYFqv9Lhn0w
A3hoYglldL/+LjW5hSswbEobn+xjwxbEC7MhyQSEI6BOpRO9IFfpEKhEXJOfU0GZGIRPw+fAwDF2
VHeji7fYDd23KxZYboxGqblzBOBHVPgaARERE1ORk2n24ab2xTADQVi/HinGRV9wC84Oi1QCKp3b
tyipb4UV1I8vuw2vJFiw+aYengmqqPTkefJNqx531GMqesXpx83oTTlgqDslkAvBcHwMe7aFvUoF
5e4MC2ydFI8/lx1mbD+cKfIV/2rw4b/NBoxDQ9uKagG0u/tlKL4e6ivXyGlJCqBJ3xOvwdYkmJ0W
0cgELRoRhYi2vXRqjk+PZPoZavU9q3NoGK/D+4FDM3FzcMCvN5sMD2/Lw7ajwJvU6QEQ1xwCQiIi
ImpTTFi+mrDfijNUn2d4tTIHlYyMlHX2vFP2ilJALQFHjzhMWL3IcvisuCK1ve7LdmLHRJOkwW5G
X9lpwfRtNtavsLseFAh9u/NWIszZfhfsMilbhub4GDIlVV7hPYyYIAsI1coRr5OgV2dCjqR1P9TL
gh5N2OEv8s0VpZFgyOgJ9roD1TKl+TGmvwx+ZbnxxrLj799pzhetDPtIjuAFHRkR1jXW+RERE1GR
k2H244dey4d+gaBXWTIi6IsM/AAhSytAzQrpOV3nrvFH1ObwiThYIcPnKnze76HgBrlubU274F62
V4YlewdCWqDjLdfnxcpKlzsZbJMvuw6QKwr+EEAWe62PAvptj8Ot1UU02/GuuWgbJ8XzfkMDfE0I
U+O36KIZ/VOv+MyAEfSKl7xkvJ1mwPbP6DWKIqOlhBSARERE1CVaPH7f8ZsL5AmmwMa6VGl+ODG9
0C/vXt4HRKqSUaIywM8uN69s2zvXO6luey4+f0xz46bQDqflejIpT44leBsQFlQ2MPX4RC44V4M0
UC8xuLfQHMzC8pRrjWmswppUGkRoZntmdj8+PFpQ5tn2wHA93D8ZtnXTQKARo5AJeKRH6fX3cjhk
JQehXR4GOyenDlHXGMtN+r2ujweOJwUiMUHJdt0buwav06BupRLbDj/GtNVA14mnL1HSp5QK+Ghm
OYSuzYb64HqBfBO7ZnIutN0ZfsV+mETV3DACJiIio0fP4Rdy9KReHcqWdP6d10GL+0DAoZfyQfHW
MGp8cKQ6ldmVf2ZUcVo8fa886sey0AxsvOCXr8y08bscPJx34x1VBeKRHMAyqwvB4c7oTT+3KxzF
zcYBm84pYfdaJ1WcLO7RGa2XILqfq79EeejzXxwBFiXPxoav0+P6EHan5hfcnAnhshxmbboiCvIJ
z1uUTcdrqxXGzFyctXpicfti9Iuzeoj9FuP1Au2A5ekeo0DtSiavClSjwiJi63oQjZmn4d2NbDf4

3IlwyLmrcBsWwMpPqXhu9Ap8NC8e0DabAtmyHH/dszsWq8ZF8zSBqhhgAEhERUaMmiiL+9acZv1+
QBlrjW2vw0dAwfki5aGC0tKpsn8mDAo8fQcorqzJSFEXMP2zD6ylW2CtqewvA4RPx7gEbFh6349E
eeuzMcuOXiyFfZUqHf1q5gHlDQnFTB12ZfVVyAe8MCsGkdcUfsA/kerDgWAHu76aHwysiyejGjkw
XknLcOJ7vRZrNF+gEW5mtGcDXKGwYoZYDIaqyweTYVmp8MZzhHxGVb2xrDf6dGIx39hc3kdqR5cZ
/ki14sV9IJUcSUVPEAJCIiKiB+Pwivjthx2dHC+Dxi+gZrkTvSBX6RCrRM0IFreLK+dCenOPGtyf
sCFPLMLyFGgOiVYGOne8esJXpjNk7UokFwxn+lRSrk6N9sBynL66P6PEDv513YXL7K2casCiKeC3
ZgncP2Kp9jNHpx3N7Lm1tvlZBcnw7KhyJERVP6R3eUoOb2mux7LQjsO21ZAuWn3Yg2eiGu+r+IVV
y+coGk8NbqPH1yAhOISWiSj3dKxh7st3YklH8JdsHh2y4v5seLTgVmKhZYQBIRERUz0RRxK/nnHg
5ySKZanjM7MUPpwpDArkAdA1T4ub2WjzcQw9ZM163a3WaA3dtykVRsdY7+63QKQQMjlGhvUFRZq2
1Nno5loyOuOIq26pjQhsNPjpc/PNadtp+xQSAoijipb0WvH+o/PAvPkSBqe21iFDL8M4Ba7nTeEu
6rZMOM8JM0MW2wfrzTqw/70RSjgdFxXlDYlX4amQ4IqvR1fe1ASFYf94Jq6fwaItHxM46bNJydYw
K340Kh+YK+hKBiC6NXCbgi+FhGLYqGxn2wtdFnwh8f8KOx3qyKzBRc8IAkIiIqB7tyXbjhb35+DO
r8g//PhE4lOvBoVwP3H4RT/Qy1NMI69fvF5yYubk4/Cti94rYcMEFlJr2G6oSsHRMBKK1rEooz03
tdZIAcP15Jyxuf2CNu6ZOFAtPlNKNLERRxPN7LJh3WBr+GZQC7ukShKnttegRXtwA4/Z4HT48ZMO
Hh2xlpgn3jVTirUGh6BelQmqqEfGRKvSKVOGJXgbkOHzYkeWGVi5gVJy6wnX8Smuhk+Op3gY8uzu
/Wvu3CpKjU4gC8SEKtAqSQ6cQoFMICFLIoFMI8EPEoVwvko1u7DO6kW4vDjMHRKkYkBNRjURp5Xi
kRzCe2lX8GrXoeAEebeZfQBJdaRgAEhER1QKfX8RRsxd7c9zYk+NGSo4bRpcfXj/gFUX4Lv7p8lV
9X6W9tc+KES3VGBDdvBaG35rhwt9+N1V7CqRKBnw3KgIJocq6HVgT1idSibZ6OdJshSeaywesPef
E9I5l16drSkRRxH8P2vDWPgvkgoCB0SoMaaHGNTEq9I5U4YW9+ZIGKAAQohLw09hI9Cmn465eKcP
TvQ2Y2TkIb6ZYsOy0A1EaGf6VWNjBt6IPvFFaOSa1u7SKyr93DcLSU3ZJp2agsKJ1cIwKg2PV6Bm
uRKcQBfTVCO/Gty7+/0y7D/tMbggoDCY5NZ6IampaBy1e2JMfeE8+bfXhj0w3hrZoXtceRE2NXxQ
hoOyXn5eCASAREdFlWH/OiXmHbUjOccNWScOBiggApnfUYlpHHQ7neZBi9CDZ6MYZa3FS6BOBWVv
ysG1SdLOp5NqZ5cJtG0xwlgpEJ7bRIMXoDkxDKunjoWEYHMsPIpURBAFT22sx92BxJdzyU/ZGGwB
uzXDhjRQLQlQyvNDXgG5h5Ye7b+yz4v/2FS1SL2Jjugsb0wurQ1UylAmRQ1UCVoyLRK/IitfmAwr
XTfzvNWF4b3BonVe5KGSFY5p3yAabx48+kSpcHaNCK/3lX47H6uQYr7sypnoTUd0I18hxfVstlpd
Yr3RRagEDQKIGYnH78cEhG744aoNfBIa3VGNcaw3GttJc8kwYBoBERESX6JMjNsl0mZoa20qNF/q
GoHt4YehxbZwmcNu6c05M31DcOTTN5sPjO834dFj4pQ+4kUgxujHtNxMKSgWmbw8KwayueoiiiOP
5XmxOd2FLhgtWtx/3d9PjhrYMOKpjagedJAD8/YILeS4/wtSNKzw+bfFi2m/GQAj8R6YLX44Ix+h
WGsl+/z1QMvwrq3T4F66WYcW4CPSspDFHafU1xS1EJcOzfZrndH4iavruTNBJAsBVZxz4v4F+hDa
y9w+i5szjF7HwrwK8uc8Ko7P4IufnNCd+TnMCKJzxMa61Brd10qFNDb5I5G8yERHRJVhw7NLCP5k
ADI5R4efxkfhhTGQg/CttXGsN7u8aJNm25KQDS0/Zy92/KRBFEavOODBlnREWjzT8e7WfAbO66gE
UVrF1DlXi7930+G5UBH6eEMXwrwa6hymQEFJ8MegVgZ/THJUcUf9EUcS//jRLKkCtHhHTN5jwv2P
FU3k/PWLDS0nV79AboZZh1fjIGoV/RERUaFgLNdroiyuLnD406esOoqak6Dp50E9Z+PfOfEn4V1q
y0YM3UqwYuDwba89W/xqPFYBERNQouX0icpx+ZDt8yHYU/ukXgWtiVegUUntrwOU4fPgjV4bIGlR
IfX28AP/6s2z4F6GWoV+0Cv2jVOgfpURCqBIqGSAXBChkgKLoz2quz/VKvxBsz3DhSIlOwY/tMKN
/lAptg5vWW/gZqxdP7DRj/XlXmdue6R2MOT3YabC2FE0DfrNE1dzy0w7cmRBUyVH1a/lpR2AKb0k
+EXjsTzNOWbyID1HgyVIhu14h4ONhYTA6/Nie6cIfmS5kXuzm21Inw9KxkRVOIyYiosrJBAF/i9f
hjZTi949FqXbcd/ELOiKqG/luP+7alIvN5VwbVcbhE3Hnplx8PTIcE9pU/WV50/r0QEREzd7iE3a
8s9+KExZvubcLAG7poMVTvQ3oYLi8t7EfT9ox5488OH0aPHc8E/9ODMYDV+mhllcc0H2XWoBH/jB
LtqnlwNcjIzC2lbpWFugtolEI+GJEOEb+nB1oHmLxiPj71jz8MiGy0S30n+/2I0ghSMbl8ol4/6A
V7x2wllnvDwAe66nH44kM/2pb6QBwa4YL2Q5fo+iebHb58XQV3XBLd/MFAK1cwOIxERhycR3ImV2
CIIoiTll8SLf70DtSWa3mGUREVLHbO+nwZooVRXX6+00e7De5kcjKaqI6kefyY8o6I/aZPGVu0yk
EzOmux+g4DTamO7HunBPJpZqJefyodgjIqyQiImoURFHEa8kWzN6WV2H4BwAigB9OOdB/eRYe+SM
P520V71uZT4/YMGtrXiCUsnlFvJRkwdU/ZWHdOWe5x/x40o4Ht5tRcvKqSgYsGhmBca01tRr+Fek
WpsQr/UIk23Zmu/GfZAtEseZNR2qbxy/iy2MF6PljJtp+m4HIhelo9206+i3LwvjVORiwPAuvp5Q
N/xRCYeXf830MdfJzu9IlhCol08v9YuFaTo3BK0kWZDuKp7WoZMDrA0KgU1R8HqhkwDejwgPhXxF
BENAxRIGhLdQM/4iIakFrvQKj4qSvtd8c5zRgorpgdPpw/dqcMuGfTADuTtAh+aYYPN3bgP7RKjz
Zy4CNN0Tj2PRYvNBXup5wUQhYFVYAEhFRg/P6C9cDW1iDC0yfCCw8bsfik3bc0zkIT/QyVGsKryi
KpTqKSp2y+jB9gwlj4tSY1VWPNJsXR/I8OJLnxZ4ctyT8UwjAVyPDMba1ptz7qi33dw3C7xeckum
zcw/acMbqwwdDQhFcg+DjrM2Lw7ke9ItSIeoyqsFEUcTKM068mpyPkxZpumd2izC7vThRwdJtV8e
o8O7VoZyqWcduaq/FodziC8plpx0NPo1rd7YL//urQLLt0Z7B+MdVegyOUWH6BhOyHNI1b+QC8OW
IcIyKq9vfMyIiKjQjIQgbLhRfc/xwyo5X+odAW8kXNURUM5l2HyavM+KYWVrM0D9KiXlDwtA5tPz
r5FidHI/1DEakRoaHS8xK8lS8ZGAAA0AiImpQDq+I+7bkYvVZadWdACBSI0OUVoZorRzRWhkOmTy
S9fAAwOUDPj5SgCUnHXixrwEzEnQVdvT0+UU8sSsfC44VlHt7Sb9dcOG3CxWvwyEXgAUjwjGxGut
tXC5BEDBvSBiuWZGNnBILAv90xoFDeR4sHBleZZi2K8uFeYdt+CXNCRGAQSngu9ERZSqqqiKKIrZ
muPBSkgUpxrJTFSoTqZHh1f4huLWjllV/9WBKey1eLtFA488sNy4U+BAX1DDTgD1+EY/ukE6f72i
Q47GL6z/2ilRhw/VRmL7BhCN5hb/nAoDPhoXhOjaBISKqNxNaaxChlsHkKrzmyHeL+CXNgVs66hp
4ZETNw4UCH278NafMl+hDYlVYPDqiWrMaitZ2frjU0kSVYQBIREQNxuzy47bfTfgzyy3ZHqYW8MP
oSPSPlq434/OLWH7agTdSLDhllb5h5rr8eGSHGQuPF+DtQaHoGyU91uL249EdZiw/LZ0GqVMIeLa
jE6eEcHz5VwH81ZhVKxOAz4eFYVK7+gslorVyfDUyHLdtMEk66KbmezHq5xzMHRyKWztJL8x9fhG

```
/nHVi3iEr9uRIwzqLR8Ttv5vw68SoSsPDNKsX+y6u/7Pf5MEBk0cSQpYkE1Duz08AMLNzEJ7vW70
qTaod7YIV6BupRFKJoHbFGQcevKphqgA/OmwLBHtF3rs6DJoSFSWt9Qr8OjEK7x2w4kS+F/d2CcJ
IVv4REdUrlVzA9E5afHS4+AvTRan2egkAjU4fztt8yHH6YXT6YXQU/r9KJuDWTtpabQRHVF+sHj9
O5ntx2urFSYsPi44XIM0m/SxzbUs1vhkVDp2i+tfKNQ0BGQASEVG98/pFrDnrxOspljJl762C5Fg
2NqLcsne5TMAtHXWY3F6L70/Y8X/7rDhfIH3zTDZ6MPqXHNzWSYdIjQzHzIXTd0vvBwChKgE/jol
EaH4a4uNDcXfnIDy504wdpQLJktro5XhjQEiDVCRdE6vGphuicecmEw6XCFIcPhGzt+Xh7f0W+MT
CDspOX2F1pcNXcaJpcYu4Zb0J66+PKlMVlmH34eHteZVWQRaRC8Ad8To82cuAaK0Mua6LF+1OP2w
ePxIjVA1WdXalm9pBhyRjccON5afsDRIAplm9eDNFOu1+ekcthrcsW4FqUMnwUql1L4mIqH7NiA+
SBIBbM1w4bfGi/WU2YKtIjsOHR3eYseasExVduXx+zIbNN0TX2RiILofHL+Ivm4BDp+04afHhlMV
b+J/VK1n7uDzjWmuwcES45EvR6qpJCMjfHCIiqjcZdh8W/lWAhccLkGEv+0bYNVSBpWMjqwyLlDI
BdyYEYVoHHeYdtuHd/VZJ0CUC+O5E5esJttTJsGxsJLqGKZF6MR/pEa7E6gmRWHHGgXmHbHD6RHQ
NU6JbmBLdwhS4KkyJVkHyBp2+2jFEgQ3XR+PfO834NlX6HEtPI6iOC3YfbvnNiLUToxCiKvzGce1
ZBx7cbkauq+rFRG5sq8FzfQxIKBHYFk7ZZuDXGExup8WzJTruJhk9OGP1ol1w/V0C+kURj+wwS35
HQ1UCXuvPkI+IqLHqGqZE/yilZAbBB4esmDs4rNYf649MF+7bklvutWFJ+e7CZWN+vS4KShmXEqH
GIcvuwxfHCvC/YwUwubQA8mp0/A1tNVgwPBwq+aWf03cmBKFXRNXVsQwAiYiozp2yePFyUj5+SXO
iooK0QdGFa16E1mCKqEYh4N+JwZjesTDkWJVWfvfe0joZFFg+LgJt9GXfBgVBwJT2Okxp33jXudE
qBMwfEoZB0So8vtNcpsNuRcbEqfFQdz1WpTkl6yAeyfPijt9N+GZUBF5LsuDzKtZI1MoFDGuhwhO
9DGWmWlPjEhckx9UxKsk0++WnHXisZ3C9jeF/xwqwOV1aSfpK/5DLakJDRER1786EIOzJKa4q+vq
4HQ9000u+9LscflHEfw/a8FqypVpLsACFX2S9mWLB8335JRI1rEO5Hnx02Ialp+xwV6MBR3lu66T
Dh9eEQlELgXbPiKqvyRkAEhFRndpndGPKeiPyXBVf2d3UXot5Q8Iuubtca70CX18bgU0XnHhiVz5
S871l9pELQEeDAqPi1Ph3YjAiNE0/fJiREITECCVmb8srs7ZaEb1CwJT2WjzYXY8uFy/Yh8SqkWH
3YU2JxivbMt24akkmbN6y/049wpUY2kKFxAgVEiOUiDcoIOc3703GTe21kgDw+xN2/LOHvl4qWU9
ZvHhhr7Qd9NBYFe6Ib7wBOxERFZreUYe5B6yBdZd9IvBykgXfjoq47Ps2OX34+9Y8SbfhIm31crQ
LViBKK0OkRob9Jo/kfey9AzaMjNPUuJEZ0aXyiyLSrD4cyfPgqNmLrRkubM2oepmcIgoBaBssR0e
DAu2DFehoUGBAtAq9Iuv3i3QGgERETZQoivCJqJVvjCpjcfvxcpIFB0xutNEr0D1cGfgvViurNER
IMboxeZ0R+e6yoVKQQsC0jlrc00WPHuG1803yyDgN/pikxsLjBdid7UYbvRxdQpXoGqZEfIgC6ss
orW+sekaosGNyDE5bvHD6RKjlAlSywupIlUxAkEIoE9bJZQK+GB6GSb8aJVN7Sod/cgF4to8Bj3T
XM/Brwia31+Lp3fnwAVjoFQAAIABJREFUXPx2OjXfix1ZblxTxx+cfH4RD2zLg73EeRWsFDB/aFi
FnbqJiKjxUMkFvNgvBHdtyg1sW33WiR2ZLgy+jPeQw7ke3PKbEenlTPl9rKcez/Q2SK5vTU4frlm
RjcyL66iJAP6+JQ/bJ0ezuVgj5fKJOGfzwu0vXOKnIZfPuVRH8jz4/oQd2zNd+MvslVzPVEQvFzG
4hSYQ8nUwFP7ZWi+v889s1cEAkIioCdqa4cLsrbkwu0Xc3zUIz/Q2XNa6ERWxuP24aX1xSLQnx4N
lJbroRqhlGBCtwgNX6TE0ViV5c0/KKaz8s5QK/zqHKHBvlyBM76QLrDlXm1RyAbO66jGra63fdaN
W0wWxdQoZFo+OwNjVOeWuHdguWI4vhoejH6f4NnmRGjmua6PFijPFv7sL/yqo8wBw3mEbdmVLG+q
8PiCk3Kn3RETUON3YVoN+UUrsLfGF4Qt78/HbdVGXFOoczvXgxl+NMJVaZzhcLcOnw8IwplXZzu8
RGjk+HRaGyetMgQYhF+w+PPxHHr4eGd4kw6WmyuEtDPbMbj/yXCLMbj/MLj9yXX6ctflwxurFWas
P6XZf4N+qd6QS7w8OrdYU1YaW5/Jj6Sk7vjthR4rRU/UBF7UPlmN2Nz0GyTKQ2KVVHY7w8jSJK7D
Zs2cjIyMDen1h17quXbviueeegyiK+PDDD7Fp0yaIoohrr70Wc+bM4QsAETVr+01u3LbBhIKL30L
996ANm9NdWDA8HB1Dau9l3erx45bfTJIKsdJMLj/WnnNi7Tknro5R4YnEYIxoqcbeHA9uWm+ExSM
N/+Z01+OVfga+TjcSERo5lo2NxJhfcpDjLL4Qn9ZRi3cGhcJQBwEtNYy7EnSSAHBlmgNvufx1Vjl
xJM+D/yRLp/6Oa6Xm1F8ioiZGEAS80i8EE9caA9v25niwKs2JSe20NbqvisK/QdEqLBgRXmkTuOE
tNXi4ux7vH7IFtv2c5sSiVHugCyrVrR9O2vHoDnO1KuFKSjF6MPLnHDx0lR5P9g6GTiG99nD7RJy
2ehGmljVIE7n/Z+++49sqrwaO/662ZEmW98gezp5AyABCWAkkBQqqE0VBGS1tWBxSat6UUWnZLIYy
W2RYoJTQk7JFBAgkrkISQhGxnD9vxlLWsfd8/FDtWPGI78j7fT/1xuVeS73UeS8899zzN7S8ZzMmp6m/h2r6bZt7u8PE
6QUnM6GNucW3w44mqKhvLQnx80M/HBwOsLQ02qQlNilFhWIqeYUfKC43PNDA8QWWM2kqnCQDWZ8m
SJVxyySUYDLFU0h/84AcsyXXPzmWJ6xy0+cVfFLg57GJDmz6lmX2eI5k/tU
utgwwLkPPFQMsbCoPsak8xJaKMFX1tPRdW0/G4J2jbfxfA49ZlA
SD9TKyvvPdi83Dk1KyN9lVFXZVB7is8IAH+33s7E8/n3gsQkOi2dv/GOEEJ0V3862c6qp36
73RFe3u7lusFJvLLdy+Mb3TVz2CLepv49UgrKtKAGurVr2m3uDfTcOSePj
J7xXX3Az3hVWuXl7OsosycHTheoDVQStvWGV8piEh9eSCEZUH1rl4ZrOpn5hj528TkuMAtwMv
bvdz2lbOV6mfAmSYNRfc92y1x3h6k/K638S8GVRkC+LgiQbnFw5wMK1g5IYcYJBNlVV2eUK81
hkM8KA3xeGKzFhs6h8k5RmbnWZiOMiRFBUVkZ2
dXfOY3NxciouPZpbm5uZSUFBQ7+vs3bu3zvdFaw1XX/wP/ccf27DqN+sHGy9/b5mDp7ANFqDPZ7+u09hJnkG/IeueXDn
DvLHbVyeK3K6JZkn0K6noN+Mjd+NkeCfEO03t6jwLD3zqF
kxLjMltXkUVWVd/ZW7e7q3GOkxXp
b5b3Z9vB6F09+7+G
```

Qr24tYZVYw5AP9/sxaKizrPLGoc0L/lUbkKzjrxOSufWLowGpna4wP1lRzhvnpXWIRguJ4g1F+bQ
gwOIDfpYc8NeUaxnq0PHoRMcJdUHe6w5zw4pyvq1V525lYYDT3inmtlE2fjPShkmn8K9tHu5YVRn
3XJ0Co9L0pBg1OAwaHEYNDoNCbpKWPlYdfW1aelljzfc+2l/Fb1dV1jtGGlMZVHlhq5cXtno5M8f
I3SfZmzxf8YdVNlWEWFcSZG1pkC8KA/U2nGlIX5uW2QMtXDXQ0uXqFitOp7OlWZxtpqioiKysLBR
F4fPPP+e+++7j9ddf51e/+hW3334748aNA2DNmjXMnTuXefPm1fs6s6+R19QbYQQtRDVeG+fAMfFMd
/AE1JC/PIkCDvFul4fI+eQDR+wmPRqtw/KMjktPo/cL+r1PD+YR1bPBr2+BSi1D9hGmqN8I8RAWy
NfP5tdmv41wEdn5fHP+jG3kF+2jvcwLOEEG3tji0GPqv1d3pxVpg/5jV9qX9t/z2o48m9jU/GU/U
q80+qQlZmCSFE51cahEvXmqmKnliQ7cqcEHf0D9HSe8MNzY1n54a4vX/TGjeoKlSEwKCBjhbj+aZ
Cw+sFetY4NQTVhn9J52eE+XW/IOnNvI+3vFTLA/kGPJGGX7unKcrk1AjzCuI/wHWKyiNDgpzZwPV
FfTxheHafngWFOtQGrjeSdSqV4cYHxOkpEW7qE2Sw9WgIS1Vhv19hQ6WGTR4NW9xadvoUIo383uo
z0BJlUmqEM1IijLZHWzw221teXl6j+ztFAPBY1157LTfeeCMLFy7k7LPPZsaMGQC8++67rFy5ksc
ff7ydj1B0VPn5+cf9oxDdR0cdDxWBKDucIXZUhsmvDLO+LLasrrZTMmJLNquL6G6tCPGTFeVsPSZ
DUAH+dIqdX42w1txhza8Mce9aV1wdl4aMTtPz7rT0Ji+pWF8a5PmtXva6w/x4cBJXdJJach11LIj
20ZXHw+IDVVy17OjSG4tOYduV2c1u+LKlIsSU94obLZTdy6rlxckpTMhq3W7DrakrjwXRfDIeRLX
uPBYe+c7FI+vd9e4zaxVm51lYUxysUwqi2s+GJvHX8c3P/DtWIKIyY1Epq0vib2I9fZqDa45pCnL
IG+HDfVXsdIXZ54mw3x37Xt3MYmKWgbvG2jkjp2WfV4kaD6qq8thGT1y5juOx6xV+P9bOtYMsmLQ
K2mMyIKOqijuk4gxEqQxGeXm7Ly6LszkMGvjP2amc36t5zV+qrSkOcveaSrY5Qwx16Dklw8C4TAO
nZBjokaRluzPEf3b4eH2nr8FVBRDrTD0u08A3h4N8XRyk1N+Mrh1H2PUKk3OMTO1l4pwepkYb0DR
HR39v6GCx7roCgQA+n4+UlBQANm/ezKFDhxgwYADTpk3jjTfeYNq0aQC88847XHnlle15uEII0SK
qqvLBfj8PfOtie2XjGXN9bVpePyctroPW0BQ9n8zI5HffOHllh+/o6wL3rnWxtSLEPScnM/d7Ny9
t89KU5l0TswzMOyetWfVUxqQbePaMli0nFEK0vnN7mMi1aGqWwvjCKm/uruLHQ5reQTEUVbnl84o
6wT+7QeH0bCOTc2JfQx06WfovhBBdzC9GWPn3dm9cbTejFn4yOInbRtrIsmhRVZWVhQGe/N7DpwV
Hb2L/bEhign+xn6nw6tmpnP1+Sdzy0t+scpKXrOPkDAOLD/h5dYeXZYcCjTZ5WHU4yIzFpUzOMXL
XWFu73LgKR1XuXOXk5Vrz+GNZdQqeYybxrpDK71dX8vvVsWW6OiX2uzFqFaKqiiukHrfBxfReJi7
qY+K+b10cbqDWuFEL/z07jfN6mpp3YrWMyzSw5MKMBvcPduh58NRk7jnZzkf7q/jnNi9fFtVdpfD
ePj/v7Tt+IkNtZq3CxCxDzRxlVJq+Sy0Xb6oOHwD0er3cdttthEIhNBoNRqORBx54gOzsbKZOncr
27du54oorAJg3z3Tz3qQGWqA4dTIVGVXgbSpYhE3ZJJATAXvoFfNrVBPwtcurTaV8sEuL9ePgs6wTTzNNbsKZj2Sov5Isv8U+qUppxO4+9L2wFdU03dnl8tr+KxHEJh1BrnhlgaqyCKaHKjyNDj0q4q36/eShX9HbpOqjG9DSTF+MdxiPQi9xdV3IK/5c1VKWRZpiqgUZPSpB79Zyf4d5T3uX6fjiX8y40J9Al3qzVVpR6WfJgkwVNS5EvWK3uJC4rrXpmd1TQNoOc8F6q+d1kfQjWjzXpiMSHNJG3Ix1bJIP6oagQnlBvbwkIT9IWW5x8HpIHjrMKKjbvNIOm4HtbeeoUGR1czZH3w/Mp8nl3fWqoGozMwkU3MQK+t7sFeP4hR4kqA5XNhf9Df2uiZp0GOuvxHXZjeXD3e3XUW7L/J8dXWQjHqB/0Mx1B0dNxFnJqSqUs5nnHwhFmfPtcNHkKJsjdk4K4OD3Osks

```
gXxUrCwOoKvSyaull1dHLqqVnkpZIFDaUBVlfFmJ9WYgNZUEqgyo5Fg0vnpnK6dmSQSSaRlVVvjw
c5NnNHhYd8BM9pnP86pIgq0uC3PVNJZf0M3PDkKQuc0eqzB/hw/1+3t1bxWeFAUJRC6w+fMKv2zN
Jyx9PtnN5f3NN/b5L+1twBaMsOuDn/b1VVASjXNzHzA8HWhrsuCWEEB2RRadhdl4Ss/OS2vtQhBB
CiDZj0Wn4z1mp/H51JS9s9dZsV4G717jIrwzzaCPlMF7Z7uUfm+NLAY1K1fP85BSGJqDhnxDNJQH
AdrbDGeKVHT5e3+mLKzTaVIW+KJctLeXlKalMlxpfohGqqrLogJ9Hvmu48GxtnrDKq/k+Xs33ccc
oK384yZ7Q5hRtQVVVDnojLD8U4J29VXxeGCCiHv95x9IpkG46GrSr/jWkmbRc3t/MjUOtmHR1fzd
2g4YrB1i4coClpacghBBCCCGEaCdajcJfJzgYaNfxu9WVcckTr+zwscsV5k996j7vs8IAd6xyxm3
LtWiY38QOv0K0BgkAJkAoqvLm7ipe3OrhcFWUM3ON/KCvmTNzjHWW56qqyj5PhM8KA7y+08eqww0
XAm2qQAR+9Ek5z56RwhUSaBD1+KwwwP3fVrKmpOHAn12v1CwDPtZjGz1sqgjzwuQUkluYvaaqKlE
VwioEoyoF3ggHPBH2eyLs94TZ74mgU+CcniYu62ducrt6VVUpC0Q5eOS1drrCbHeG2FEZJt8ZxhN
uQcSPWNBvSq6RmX1jreal1pUQQgghhBDd08+HWelv1/GTFeVx10xfFAWZXW5iSkk5w1L0DEvRYzc
oXPtJGbUvQyw6hXnnSPBPtK9uHwD0haOU+aOUB6L4wyoj0/RNbhXuC0d5dYePpzd5OOg9WujrtXw
fr+X7SDYoXNjbzPTeJgq8Eb4+HGTV4QBFVc3P9DueiAo3flaBKxjlp0OtdfZXhVUCtVKfVGKBE5N
OafL5irblDUWpiqikGTXHrS3RkO9Kg9z3rYtPCwINPmZkqp6bhiVxWT8Lhb4I/833Mm+nj0Jf/Dh
dcsDPuR+UMO+cVPKSG09ZdwaiLNjt4387fWyuCBGK0uTMuzd2V/GX71zcMdrGFQMscZ2wIlGVb4q
DfLTfz/flIQ56wxzyRvBHGnnBRiTrVFLMOux6DTaDgl2vIcWoYWKWQYJ+QgghhBBCiBrn9jSx9KI
Mrvy4jH2eoxcgxUENb+yuAqoafO7zk1O6TFkl0Xl1qwCgMxBl8QE/H+yrYn1ZiDJ/LMBSm1mr8Lu
xNm4Zbm2wBXdFIMqLWz08v8VLWSPLdiuDKvN2+pi309ek43MYFK4cYGGAXceBI9lRBzxhDngjRKI
wLEXHmHQDY9L0jE7T83lhkN+sclJ9Bipw59eVHK6KkpesY0tFiK0VIbY4wxzw1B8hUYBJ2QauH5T
EjD7mepcx1ieqquxzRygPROlt1ZJuanmQSsTG1MayIBvLQmwoD7GhLMTOyjAqYNJCb6uOvjYtfaw
6+ti0nJ5tZHSavsHf+YayIH/b4Ob9ff569yvABb1N3DzcymlZhprX6WfX8ceTk/n9WDuLD/j5zSo
nxbUC1vmVYc55v4QXz0ytUwA+qqp8Xhjgv/k+3ttXVaf5RXPsdke49Qsnj25w85tRNnIsWj7YV8V
H+/2U+E8sgD7MoWNmPzMz+5rRluwlLy/vhF5PCCGEEEII0T0MceheZPiODaz4pb/JqvntPtjOjj5T
rEu2vWwUAB75eyPFWA1ZFVO5d62L+Lh9PTHJwaubR5hqHvBGe2ezh5e1evC1cVlifSVkGrh8cC8C
ZmxiAg1h78mSDws8/q4g7r0c3uJv8GirwZVGQL4uCpBidXDnAwuwjTQr8ERV/WMV35Gu3K8zmihC
bK0JsrQjH/Q4cBoVByXryHDoGJce+RqTq6ZmkbTBIFYioHPREMOsUcizdJ4AYjqpsKg+x9kizjbX
FQXa7G46W+SOwozJ8pCvt0Uy+oQ4dswdauGKAhawjqeRrioP8bUPdTlO1Te9l4u6T7AxvpOOUTqN
wUR8zY9MNXPNJGetKjy4ddoVUrlpWRn+7Fq2ioFFAAziDUQp8ic1u3euO8Ksvncd/YCPs+liHy/N
6xpbzDqrV7TK/5ESPUAghhBBCCNGdpJu0vDMtnbtXV/LSdm+jMYarBpi5bWTdFXpCtIduFQBsTsx
uS0WYaR+Wcv1gC9fkJfHv7V7m7/IRaiC+YdTCj/KSODvXyNKDft7f52+wqYdFp3BKhoGJWQZm9Tc
fdzllYy7tb8Fm0HDtJ+V1shmbqyKg8twWL89t8R7/wcdwBtWa7rG1OQwKw1P1jEjR0yNJy35PhF2
uMLtcYQ56IzVFVB0GhWEpeoan6hmeomdEqp6xaXq0TawD1xbCUZVtzjC7XWH2ucPs9UTY4wqzzxN
Gr1E4Kd3AhCwD4zMNDErW1QQ0D/sirCkJsqY4yJqSWCdnXwICyFudYf641sWfvnVxbg8jgSisaGS
p7+nZBu452R4X1D6eHklaPpqewW1fVfC/XUdT2lVgl6v5KX4aJVZbT6dRSDdp6H2k23Xsu5bVxUH
m5fua9bcKYNMr9EyKdczua9MxyKFjULKeQQ4d2ebuE1wWQgghhBBCtD6jVuHRiQ7uOsnOko17qEz
KYktF6MgqvFiyzOX9zTx5Wopci4gOo1sFAOtj0ECqUUOqScMhb4TKYHydvJe2+3hpe8NLeO16hZ8
OTeKmYVYyzbEsrAv7mHlsosoXRQHe2VPFloowaSYNk7IMTDyydLOh5cUtcV5PE29OTeOqZWUNNnH
QKGDVKRD7H9U/3RlMXCZjfZxBtSbD8HiP++pwkK9qpVH3t2l5bKKDs3qYGnlm6wlFVTaUhfiiMMA
XRQG+PhxstKHENme4Zrl3ilFhVKqBPe5Yc4uWStIpx802jag0mvE3Jk3PPSfbOSvX2KIPH5NO4dk
zUhiVZuCPayqbXMuvt1XLj/IsXDXQQo5Fi1bhuF2Ef5SXxB2jbDzxvZv/5tcfcLfrFab2MjG9l4k
hDj09rdoWNyYRQgghhBBCiJZKMWo42RElL+9oll9188OOlMwiBHTDAODgZB0z+pi5sI+Jgck6rDq
lJihS6o9w9+rKuEynhmSZNdwy3Mr1g5PqDT7oNApTck1MyW2b4NWkbCMfXpDB775xUuiNMDBZx7A
UPUOPdCLKs+vqre+3zRniPzu8vL7TR0WgecFAh0Ehx6JlnyeSkIy2Y+12R7hkaRlXDDDz0KnJpJv
apmPS+tIgf1nv5vPCQIs7yFYEVFYWNhyUO5ZWgSEOHaPTDIw6UuNxRKoeq07BGVTZ5w6zzxNhnzv
MV4eDfHzQf9xA3LgMPb8dbee8ni0L/NWmKAq3DLcyPEXH7V85G1yybNTCjD5mrsmzcEaO8bgBv/r
0semYOymF34yy8cT3Ht7Y5SNJp3B+LxMX9TEzuZ7u2kIIIYQQQgjRESiKglyuiI5IcTqdrZsC1oH
scIbi6n81ZGVBgN+sqqh3iWM/m5Zfj7Rx1QBLkxtmdAb+sMoH+6t4Ld/HlooQeo2CWadg1sa+m7Q
KWWYNw1NjAcXhKfqaun1RVeWQN0L+kTp1O5xhtlSE2FQeOm4ALceiwRVUj5vllmJUuH9cMlcPtJx
QMCs/P7/Bpg/eUJSHv3PzzBZPzdLk1pJp1jAuw8CpmQZOyTAwNr3p3acBiqsivLHLx7x8H1uc4bh
9p2cb+O1oG5NzTjzwVx9VVdnniRCKxu5sRdVYFqIK9LVpsekTm42nqmqrpc03Nh5E9yJjQdQm40F
Uk7EgapPxIKrJWBC1yXgQ1Tr6WOhWGYBNCf4BnJlr5MuZWTz+vZsnv3cTiMDIVD23j7RycV8zui6
YymvSKczqb2FvSpsYAAAgAElEQVRWf0uzn6tRFHpZdfSy6ji7x9Ht1Z2Cvy8PsakiRIU/Sk+rlv5
2HQPssa62Fp2m5nHVDUY2lIVYtN9P7RhcRUDlF184mZfv4xcjrJzX05TQZdSfHPJz+1fOuHbu9ck
waRiVpqevLXb8se86yv1RvikO8E1xrNZf7aXYeg2MStUzLtPAuIxYwK+3teHmKE2RadbyixE2bh1
uZUNZiLf2VOEORbligIWJWU2v8dcSiqLQ19Z2bx1SM0MIIYQQQgghDgx3SoA2BwmncJdY+38coS
Vcn/0hAM23ZFGUehn19HPruPivg23Pa/9uIuOtEdfWxLk119WsLkiPrstViewnCyzhqsGWLg6z9L
kwG5tzkCU/Mow+ZUhlh8K8Oae+pd9Z5o1nJ5tPPJlIK9Wc49jnZkbC7xFoipbnbFGJ9lmDaPTDK2
WLaooCmPSDYxJN7TK6wshhBBCCCGEEKLzkwDgcdj0moQvaRTHd0qGgRUXZ/LMZg+PfOeu0+H4cFW
UJzd5eHKTh1MzDAxy6DDrFCxaBZNOwaJTCEXBE4riCak13w86jRxaW0iJv4F2zkf0TIo1IJnagvp
5Wo3CiNRYDT8hhBBCCCGEEEKI9iYBQNFh6TUKvx5pY2Z34NMP
```

```
BP40CNw5N4g8n2bFK4FcIIYQQQgghhBBdgAQARYfX16Zj4XlpfFsa4r87vLy5pwp3KPFdOoal6Hj
6tBROzpDltEIIIYQQQgghhOg6JAAoOgVFUTjlSAONh8Yn895eP6/me/myqKmZf/EMGuhv1zHQriM
vWcfYdAPTeye2sYgQQgghhBBCCCFERyABQNHpWHQarhpo4aqBFva6w3xbEsQbVvGFVarCKlWR2He
9BpJ0Gqx65ciXBndxAacN6UNvqxatBPuEEEIIIYQQQgjRDUgAUHRqfW06+tqaPozzQ1H62WXYCyG
EEEIIIYQQovtQnE5n4oupCSGEEEIIIYQQQgghOgRpcyqEEEIIIYQQQgghRBcmAUAhhBBCCCGEEEI
IIbowCQAKIYQQQgghhBBCCNGFSQBQCCGEEEIIIYQQQogurFO1Qw0EAtx9993s3r0bk8mE2Wzmttt
uY8SIEQSDQR555BG+++47AGbNmsXVV19d89zXXnuNhQsXAjBmzBh+//vfYzAYAFi9ejWPP/44oVC
IjIwM7rnnHnJzc9v9v+BEWztMZ42LlzJ48++ijl5eXodDp69+7NnDlzSEtLa5dzFE3TWu8N1f7xj3/
wyiuv8K9//YuRI0e23YmJFmmt8VBaWsrf/vY3Bv8vPz0ev1jB07lv/7v/9r+xMUTdZaY2HBggW8+ea
baLVaFEXh1ltvZeLEiW1/gqJZWjoedu3u3axV/+8hfy8/NJT09nwYIFca8r88jOpzXGgswhO6/Wem+
oJvPIzqO1xoLMITun1hoP7TmP7FQBQICZM2dy2mmnnoSgKK1euZM6Ufn8/HMmzcPn8/Mn8/HW2+9hdv
t5pprrmHUqFmSjRs3nDhQl555RVsNhtz5szh9ddf57rrrsPv9/PHP/6RJ598kiFDhjBv3jz
eeeQRnnrqfY+VdEEiR4PBoOBO+64GDBgEwd+5c5s6dywMPPPNDOZyqOJ9Fjodq3337Lzp07ycn
JacezE83VGuNhzTzyCNbbb+FwOPj5Ej4UDBw7rOr5Ej4UDBw7w7LPP8tZbb+FwOPj666/+xOLFi9v
7VEUTtGQ8OBwOfvGLX+ByuXjyyjyySfjXk/mkZ1XoseCzCE7t0SPh2oyj+x8WmMsyByy80r0eGjve
nWgJsNBo5/fTTURQFgNGjGjR1NeXo7f72Mnz695pe4ZMkSpk+fjjt1uR1EULrzwQpZUrizKCg7U9ONEtrjAU
Al8vF3Llz+cMf/tAu5yiapjXGQ8phNq7f9yYlmaY2xoCgKqqri8/kAcLvdZGZ
mts8JimZp6XhIS0tj1KhRmM3mOq8t8shO4ZMozOUrbsjBzjMuSRjqeGDqmRm
EzdUVERAEVFRWRnZ9fs69GjR82+wsLCuH0dfXuX0MoYYuCeRewbPqq
dzM/CG8p+wEj2GGlqpTXKzs23ZXfIlBrD4j5L5p0TYs7/zRYvLHzoZ4SqBCuz1TrD4T/7y96kt
UX3CaZH8SmyLo4L3fHS1IdbfeeXc7A4S09CNzEA7F8CnNuQU2iWNTYKNNDFLv
88pp9X331FX379mXIkCEAXHrppaxbtw7n5Ep82/VV7SSnZU0XYrBj4zq9CdpwpTgFcc
Al8vF3Llz+cMf/tAu5yiPm8XhWHSDORu/9vuhGl3oPjqBZxRyPRlvEiXBOBaojHNIfcy1o
mts8JimZp6XhIS0tj1KhRmM3mOq8p88jOqTXGgswhO6/WGA8g88jOqDXGgswhO6/WGA/tPY/sVAH
AY82bN4+JEydiMpkoKiqKu7OSm7sOSm5tLUVERAEVFRWRnZ9fs69GjR82+wsLCuH0mk4mUlJSa/aLzSMR
4qC0cDvPGG29w1llntf7Bi4RK1Fh46KGHuOGGxkKsEZZ9UmHsRW8zp6OGBQzMaESpDhrojv3ZoxV
0qW1PRJywRIyFnj178utf/5rZs8cUX89hjj3HPPfSbMzeCiuWtnTP7TSCN7hkSIoe6Dm5JWo
8yDyy80vEWJA5ZNeRiPHQ3vPITrcEuNpbbe73Fp59+ynPPPPPDfehyIc6gSEyUh3334/D4YircSU
6vkSNhffeew+z2SoSyT904uUeMhHA6zbt06nn32WUaPs3q1av5wW9+w9Ks0SUlKCjla0pkSNhcO
HD/P+++8zb948cnNz+eyzz7r7jjjjt44403MBqNCTpa0dpkHimqyRxS1CbzSFFN5pCitq4yj+yUGYA
LFy7kf//7H8888xY2srCwKCwtrHlNQNUFBzRzYY7OzsuGnvo0KEG9/n9fioqKmnRvYWMOK3KW6/0ROVOR
SiXDvvffi9/t58MEH0Wg0wg65Z9Jt5TIsbBmzRrWrWFFnDzJkzmTlXBxcyZM0fqHUiiRwPubm5jB07l
Vi9GjRwNw6qmnYjL5Yto2fVNHi9ZPnkxVVRRUHDhxoq9MRJ6i546ExMo/
s3BI5FkDmkJ1dIseDzCM7t0SOBZlDdn6JHA/tPY/sdJ9K8+fPZ8GCBTzzzDNkGTUbJ82bRoLFy5
EVVVcLheLFi1i2rRpBkCGRkPgNBdyrpbfz7OjGWd0SjNKlhDAFZaZqMEg0XuUr9JMewmbYzBprofo
2RI+HcDjMH//4R8LhMA8++n12d4i9tn3ybt/yM+dx39wPRxAxTQkz81+fn4NH
nJ5on0eNh4sSJeL1e9uzZA8D27duJsmRgpFgZfJDfq8WVj2bBCegwXThwwKycDyDzFgX14nBFJMl
RkvHQGJlHdl6JHgsyh+x8WyPU59ODlJXO3Z8LWFPg9I7QXMl5qlOhubVPSPH8zOURNmbNehSaDb
rtWb7448/jsPh4OGHH31/OYVk7lbfeegtRVVR9u1oqWVcblL3K6TXo+5dvsp5gi2ruZGRsHoGLdmUHfPYQsvH
5cwmHw6SlpXHvffXSo0ePNk4Yru2qmLwWOhvkQXfcZnSQWBCu5R5d+mAzXAdqLQ9tfwZj3L
cyQMPPMDIkSPb5Jx7Jvb7Ey0Jbfg0bKrkfsEw8dNYsq4GXjE9KgmpiQ6J9jm7DpKLSvd6s
Hp9Nxyy23MGHChDY/P1P9E8LR0PaDWlXH/99QSDqQmNXQyHyclBbRCGY7BrM0bemkIpUlanhKZXJ9ON0
31CbzyM6htcaCzCE7p9YaD05j+xUAUAhhBBCCCGEEEIIIUTzdLolwEIIYQQQggghhBBCiKaTAKA
QQgghhBBCCCGEEF2YBACFEEIIIYQQQggghhOjCJAAohBBBCCGEEEKIIIQQXJgFAIYQQQgghhBBCCC
6MAkACiGEEEIIIYQQQgjRhUkAUghhBBCCCGEEEKILkwCgEIIIYQQQgghhBBCdGESABRCCCGEEEII
IIYQQoguTAKAQQgghhBBCCCGEEF2YBACFEEIIIYQQQgghhOjCJAAohBBBCCGEEEIIIYQQXJgFAIYQ
QQgghhBBCCCG6MAkACiGEEEIIIYQQQgjRhUkAUghhBBCCCGEEEKILkwCgEIIIYQQQgghhBBCdGE
SABRCCCGEEEIIIYQQoguTAKAQQgghhBBCCCGEEF2YBABFt5Kfn9/ehyA6EBkPIkbGgkgk40HEyFgQ
UVEwBqIJPiLRXuS9Qm9QdQm40FUk7EgapPxIKp9s8g
ffWEVgFMy9Px0iJUf9DVj0intfLRCCCGEEKK7kwCgEEIIIIUQThaIq/9jk4ZH1LvyRhh+3tiTE2pI
K/rC6kmsGWfjx4CT62GTaJYQQQgh2ocsARZCCCGEaIJvS4JMea+YP33BvtrJAlCe+9zBm4WPn4WF
+/Gk5pU19ohBCCCGEEGEEAkkAUAhhBBCCCiEa4Q1H72sn535ahkKcKOPTWpgua8KvL23WN
rHKUQQgghhBANk7UoQgghBAN+L48xA+XWzhgfAxsNH9zL8WocONQK4OSdfS16ehj05Jq1FLLF272
8lu+jMqjGPafAF+XCRaX86RQ7tw63oihSH1AIIYQQQrQrQ+CQAKIYQQQtRtRtRjZUGAaz4pwxVS6+y7YoC
Zh05NJt2krbNvYLKeh051cPdJdhburuK5LR621MocDKtw9xoXqw4H+cfPGEEEK0rg4
x43zmmWe47LLLGD9+PEuxLq3ZHgwGue+++++7jkkku45JJLeO211+Ke99prr3H66adz+umnM++
CCCFEF/Tmbh+zPi6tE/zra9Py4HEBlrQE8z3epVXmnP8C3nnylJ7yJ9ynx2xnXA08sc2PS19YUDdVQ==
RQoiOTlVVXMEoe91hdnoVinwRwtG6N4iEEKKj6hAZgKeffjozZ87k/vvvvj9s+b948fD4fb731Fm6
3m2uuuYZRo0YxcuRINm7cyMKFC3nllVew2WzMmTOH119/neuuu66dzkIIIYQQXcG8Qzrm7qmos/2
nQ5K4b5wdi65590/1GoX7xiUzcAzZ9XxC0L3ueJcOFHpfz3nFSm5JpO+NiFEEK0jKqqHPJGyK8
Ms6MyTP6Rr5KqCGWBKOWBKKFo9aPN8F0RGgXSTRqyzFqyzBr62nScnGFgXIaeAXadlHkQSI
AOGrUqHq3L1myhDvuuANFUfRUbDb7UyfPp3FixxNFUBmgGi1krqgbJ70yVVEz1ZZBCm1jmbKAQgghhGi
RqKpyzxoXT2yrivs7TE5qlu4Pj8Bbavf+Y8Z7uuy6o5qvlK8lbcY2n7t8EB2vhy1XZsjpXvjvKFl
i+AIRlV2uWHBvzMUF/DzpuX0b4VCMUDGtIDfFavEcoqemuWOIRGbY6vqGNfT4Pc69y+2Lmk/8WhzbmC1ZrghNcYcHOoUgW
UQrSfDhEAbHEREURE5OTkfLX/52bmk7fcV/T5SlVzPC9ZI/
```

K265T4JkzUrhigCUhP6evTceSCzP4w+rKmotEgGAUrv+0nLmTHFw/OCkhP0sIIbqrqKpSXBVbrls
d3NtRGSbfGWKvJ0JbrNytCKgsPRhg6cFY53eLTmF8poEzcoyckW1kbLoenUYCgkKIttOhA4CJlp+
f396HIDoAGQeiNhkPopqMhe5LVeHR3XoWFOrjtlu0Kn8dEmBs9BCJHh43poOtn465tbINVeC2r5z
sOFTM9T3DyMqxjkHeG0RtMh7anqqCNwLlIYWKkEJlSKEqCr4IVEUUqiLgiyiUBBUOBxSKAgrFQYW
wmvg3UZNGxaFXMWmIHUu46T/DF1b5tCDApwWxgKBdpzIpJcLk1AgTUyJYu9WVedcj7w2iWnuOhby
8vEb3d+i3maysLAoLC+nRowcABQUFZGdnA5CdnR2X8Xfo0KGafQ053i9DdH35+fkyDkQNGQ+imoy
F7u2R71wsKHTHbcs0a3jj3DTGpNddDpwo9+bBoJ4+fvFFBZFa2SjP7DOANYUHxiWjkShgu5L3hs4
jGFHZ6w6zxx0hGFUxahQMWjBoFAxaBYtOIceixWFQWryUX8ZD6wlEVHa7amfrhdhZGeawL0qJP0I
wevzXSAS7XmGQQ0desp7ByTryknX0tGpJN2lJNWow62Jjp3osBCMqh6siFFdFKfBF2FAWYm1JkG9
Lgrjr6SBfmyussLhEx+ISHXoNnJFt5MI+Jq4YYMGm7xC9OkUTyXuDqNbRx0KHDgBOmzaNhQsXcvL
JJ+N2u1m0aFFNo5Bp06Zx7733ctVVV2Gz2ViwYAFTp05t5yMWQgghRGfywhYPj6yPD/6lG6IsuSC
LfvbWnyb9cKAFh0HhxyvK8UeObn9ms5cyf5S/n56CXpaICVHDH1bZUhFiQ1mIrc4Qu1xhdrnC7G/
isk6zViHHoiE3SUtukpZ+Nh3DU/SMSNXT16aVoHsrK/dH2F4d5HOGya8MsaMyzL42WpZbrWeSlkF
HAnyDHDoGJesZlKwj06xpVoDYoFXoZdXRywonAzP6xOq4RqIq25xhVhcH+epwgM8LAxRVNRzFDEX
hk4IAnxQEeGCdi1uGWfn5MCvJBgkECiESp0MEAJ966imWLl1KRUUFO3fu5KmnnuKpp55i9uzZPPz
ww1x66aWoqsqsWbNqGoaMGjWKSy+9lOuuuw5VVRk9ejSzZ89u5zMRQgghRGexYJePOd9Uxm1LNig
8PTzQJsG/atN7m3lrajpXLS/DVatD8PxdVTiDKi9NSWl252EhugJfOMqm8hDrS0NsKI8F/bZVhGh
mn4Y4VRGV3e4Iu92ROvuSdArDUmIBwTHpBsak6RmWom9W44Zyf4RdrggFvggF3giFvthXgS9CVVg
l2aDBYdDgMCo4DBpSjBr62XWMTNXTx6rtUl1ji6sisX+7siDry2L/fge9dX/vrcVhUOhp1THQfiT
QdyTgl5esI6mVM+y0GoXhqXqGp+r58ZAkVDXWeOTzwiBfFMWWAZcH6g8IVgRUHvzOzdObPfx8qJV
bhiWRatK26vEKIboHxel0tuG9FiHaV0dPyRVtS8aDqCZjofv5+KCfHy4riwskWHQK70xLI6Vyf7u
Mh+/LQ1y2tJTiY7JEJmYZeP2cNBxGCQK2NXlvaDvuUJTvy0JHAkVBNpaF2F4ZbtOssPoYNDA8Vc/
YNANGfwVZ6elolViAR6eAP6KSXxlmpyuW0VbWQFCnKez6WNBoREoscDQwWccAu47sZmaltYdIVGW
LM8zXhwN8Uxzk68PBhAb7zFqFDLOGDJOGVKMGm0FDkk4hSa8c+a4hzaihp1VLzyQtPZK0WFsxyHe
i7w2RqMrqkiCL9vv5aL+fna5wg4+16hRuGm5lzmibdBHuoOSzQlTr6GOhQ2QACiGEEEK0lTXFQa7
9pDwu+KdT4D9npXJqppH8yoaf25pGpupZckEGlywtZW+t7KRVh4NcuKiEt6amk2WRLBDRualHurN
uc4bZWBZkw5EMv12uMCca6+th0dLfrsVm0BCMqAQiKqEoBKIqrmCUQl8UXzPTB4NR+K40xHelIcA
A+1wneJQNc4VUVh0OsupwMG57kk6hn11Hf5uWXlZdbPmyRUOOJbaMOceibfNSAZGoyvflIT4rDPB
ZYYDVxUFcx6l5dzzVy3IH1mTr6elj05Jh0rR6xl5b02oUJmYZmZhl5L5xyeRXhpi/q4oXtnriMsE
BPGGVv21ws6LAz0tTUukl3UKEEC0k7x5CCCGE6DZ2u8JctayMqlpdNxTguckpnNvT1H4HdkQ/u47
FF2Rw2dJSNlcczQjZXBFm2kclvD01vU2XJwvREqqqUhaIst8dYZ8nzG5XhPzKEPmVYfJd4ToBjub
qY9UyOk3PqDQDeUey5PrbtcddKq+qKpVBlYIjy3IPeiJsdYbYVB77cp7gcbUWb1itOcb6aBUYaNc
xLEXP0JTY92Epia1pqKoqW53hmoDfl0UBKlvw+9JrIC9Zx+BkPXmOo8tyB9pbf1luR5aXrOfuk/T
8coSVF7d6eWazp84S4bUlISa/V8yLk1M7xOeVEKLzkRmkEEIIIbqFMn+EWUtL6yzR+8v4ZGb1t7T
TUdWVbdHy4fQMfri8LC4TaK87wvSPSnh/ejp5yfp2PEIhYlRVZZ8n1vn0+7IQmypC7HPHGnJ4T6R
QXy0D7FpGp8Xq8VUH/VJauBxeUZRY7T2jhmEp8X9DqqpS6IvVHKyuWbe+NMQhX/OWsRq1MMCmo5d
NF5ell2vRYtUrVAZVnIEozmCUikCU4qoomytCbC4PtTiDLqLC9sow2yvDvL336Ha7XuHkDAOnZBg
Yl2HglAx9k2vJqarKblekJuD3RVGAEn/zljcbNDAiNfbvVv1vODRFj1GWsTYo2aDhztE2bhqWxL+
3eXl6kyfu914RULn84zLuHG3jd2NsaKVJlBCiGSQAKIQQQoguryqsMnt5eZ3C/3eOsvHzYdZ2Oqq
GOYwa3pyaxo8/LWfJwUDN9qKqKBcuKuW989MZ4pAgoGhbwYjK2pIgKwoDfFUUYGN56ISz+aopwKB
kXSxYlG5gdJqekan6NuuCqihKTWfgqb2OZldVN7LYVBHiwOEy7I4UIiqEVZVIFBQF+tqONpfolaR
tUVBGVVX2eyKxTL+KEDucse7Gu13hFgcGXSGVTwtiDSeq9bbG6uNlmbVkWzRkm7VkmDVUBKIc9EY
44Ilw0BthvydMRaB5PzfVqGF8poEJWQYmZBoYk26QYF8LWfUafjXSxjWDkrjps/jPARV4dIOb1cV
B/jUlhXRpECKEaCIJAAohhBCiS4uqKjd9Xs43xfF1ta4cYOYPJ9na6aiOz6LT8N9z0vjFFxXM31V
Vs724KspFR4KAx2YxCZFo25whlh8KsLLAz5dFwYRk9iXpFPKSdQxN0ddk9o1I1bdq04aWyjRrmdo
rFhTMzy8iLy+5VX6Ooij0senoY9NxYR9zzXZVVSn1R9nlCrPHfaSjsDfCoSMdhg95I83KzNvvibD
fk5jmHClGhdOzjUzOMXJGjpHByboO36yks0kxanj93DSe+N7DA+tccU1xVhYGOPv9Euafm8ZQ+Sw
QQjSBBACFEEII0aXds8bFu3v9cdvOyDbwz/aBuvOHU+STqGPVUsvm44+Vi15Rxo7DEzWk2vp+F1tOwpFUcgwa8kwa5m
QVf9jKoNRtlaE2FoRZktFKLakuCLUohp9jbHpFSZlGTgjJxb0G5GqT1iNQdEwjaLwm1E2TskwcMO
K8riA735PhKkflvDSFKkLKIQ4PgkACiGEEKLLen6Lh79v9sV9sRtG+LQ8erZaRg6ydI0jaIwd5IDnUb
hX9u8NdvLAlEuXlzCO9PSGZ1maMcjFF2BJxTl/X1+3tjlY2VhIC7TqDF2vcLIND2jUmP1+QYn6+h
j05JqlCBfW0k2aJiQZWRClrFmW1RV2eUKs6Y4yNqSEGtKgmyuCDX53xXArFUYn2Vg8pGA35g0PTq
pOdduJucY+WxmJj9ZUR5XH9YdUrliWRmPnJrcIUtaCCE6DgkACiGEEKJLem6Lh999Uxm3LcusY3
z0nC0sIlAe9EoCn+bkIxWgRe2Hg0CVgRULl5cyrvT0hmTLkFA0Xx7XGFe3Obhv/m+JtXzyzJrODP
XyJQcIxOzjPS1aSXQ1wFpFIW8ZD15yXpm58W2eUOxOn9FvghFVdHYd19sCXGyQUOvJC09rVp6Jmn
pZdWRbdZIk4kOJsei5b3z0/ntKicv18oKj6ow55tK8ivDPDw+WQK1Qoh6SQBQCCGEEF3O4xvxd3Pe
tK25bkk5h/rlp9LZ2zumPoij8ZXwyOg08s/loELAyqDLr4zIWXSDdgUXTqKrKioIAz231svSAn8b
CfhoFzswxMrWniSm5RoY4pM5Wk1zDYoWGwNBDq1PSaWFZ4nkPP3asr4/5+X9zmZbc7zEtTUrG
3UQMdIUTn0TlnwEIIIYQQ9VBVlQfWuXhYxwXhsY/yyX4MGXj4rtdNnySmKwoPjktEpOnQp9jkcmGpE5Ck9tOnqOpf4olyw

pY/EF6fTspAFO0fr8YZU3dvv4xyYP2yvDjT52VKqeKwdauKyfmWyLdBkVoiNRFIVbh1sZYNfy0xU
VeGo151l+KMBFi0pZcF4aWfK3K4SoRWaIQgghhOgSVFXl96sreW6LN267Wasw75xUzurRNQqkK4r
Cn0+xowJP1woCHvRGuGRpLBMw3SQXfeKocn+Ef23z8sJWb6MdY9NNGq4eaOGqgRbpKipEJ3B+LzO
LL9Rx1bIyDnqPdnfeWB5i6oclvDk1jYGSGS6EOEICgEIIIYTo9CJRldtXOflPrZpIEOtaOf/cNCZ
lGxt4ZuekKAr3nWLHGYjyav7Rc86vDHPZ0jLePz9dln8JdrvCPLs5Vt+vKtLwQt/RaXpuHJrEpf0
smHSyvFeIzmREqp7lF2Xww+VlrCs92rV7nyfCtA9LmX9eGqdkdO7sdyFEYsjMUAghhBCdmjsU5Ue
flNcJ/jkMCu9OS+9ywb9qiqLwxCQHM/vGZzZuKAvxw+VlVIWb0e5TdBmRqMriA1XMWlrKSW8e5sV
t3nqDf1oFLu1nZskF6ayYkcHsvCQJ/gnRSWVZtLx/fjpTe8Z/3sW6xZey5IC/nY5MCNGRSABQCCG
EEJ3Wfk+Y8z8sYdExFzcZJg0fTM/gpC6e9aDVKLwwOZWzcuMv+r4sCvKTFeWEoxIE7C7K/RGe/N7
N2DcPc9WycpYdCtT7OJs+Vjts/aws/j0llfFZRmnqIUQXkKTX8No5aVydZ4nb7gurzF5exqs7vA0
8UwjRXTRpCfC6deta9OInnXRSi54nhBBCCHE8q4sDXL28vE5Ns1yLhnempTOom3S6NGoVXj07lR8
sKWVtydHlX4sO+LljlZMnJjkkwNMFqarKjsowSw/4WXLQz9eHgzSW9Jlr0XDzMCvXDk4iWZaHC9E
l6TUKfz/NQY5Zy982umu2R1T45ZdOinwR7hxtk88EIbqpJgUAb7755rg3CVVVm/Sm8fXXX7f8yIQ
QQgghGjB/l49fflFB8Jh+BqNS9bx+bho9krpXEwyrXsOC89K58KMStjiPdnd9ZYePHIuW3421t+P
RiURQVZW97gjry4J8dTjIxwf97HVHjvu8k9L1/HyolUv7mTFo5aJfiK5OURTuPtlOTpKGO1dVUvu
+wIPfuXVpNagAACAASURBVCn0RXl0QjJajbwfCNHdNCkA+Oyzz8b9dzAY5OmnnyYUCjFz5kx69+4
NwL59+3jvvfcwGAz88pe/TPzRCiGEEKJbC0RU7v/Wxd83e+rsm9HHxHNnpJCk757ZTSlGDW9OS+e
8D0riukE+st5NjkXLdYOT2vHoRHMd8kZYXRzgu9IQ68tCbCgLUhls2pJuoxYu7WfhZ0OSuvwyeCF
E/W4YYiXTrOWnK8sJ1LpX8O/tXg5XRfjnmamYpe6nEN1KkwKAxy7lnTt3LkajkZdeegmD4eik4ow
zzmDWrFncfPPNrFq1ivHjxyf2aIUQQgjRbW2pCPHzzyrYVB6qs+/OUTbuOsmGppsva8qxaHlzahr
TPizBWStYdPsqJxlmDRf0Nrfj0YmGBCMqWypCfFMcZHVxkG+Kg3FB3KbKS9ZAL1wyykGbqXlm
wQoi6ZvQx8860dK5aVhZ3A+HD/X4uWVLK6+emkWLsnjfNhOiOmhQAPNaSJUv48Y9/HBf8q2YymTj
//PN56aWXuO222074AIUQQgjRvUVVleE2ePnzt5e1XcwWQy3R6+rQUrhhgqf/J3dBgh57556Yxc0k
p/iO/r6gKN6yo4N3zNZya2TW7IncG4ahKkS/Cjsowm8pDbKoIcJkoaCYvLMHHCNTe5q
Y2tNEP3uLpvZCiC5sYpaRJRdmMGtpWdyNha+Lg0z7sITXz0ljQLK8dwjRHbToL72qqoqysrIG95e
UlOD3S6txIYQQQpyYAm+EW76oB3Y6mPZO0vDQllXGZssTxWOOzjPzrzFSu+bSc6kbAVRGVK5e
VsfTCDPKSu0eDlLZUFVYp9EUo8EUo8B75OvL/q7cfropyIo2Z7XqFUWl6xqQZmJRt4MwcY7dd8i6
EaLohDj1LLszg8qWlcXVid1SGOev9Yp6fnMJ0yRAXostrUQBw3Lhx/O9//2PIkCGcffbZcfuWL1/
O/PnzmTBhQkOIOMBKJcO2119b8t6qq7Ny5k8ee4zKykoee+wKxbnNza/Y/8cQTGEEKJ
9qKrK/3ZV8V8ftvnHFLWatd3t/MoxMcOGTpUoMu7GPmsQkObl/lrNlWEVC5/OMyll2UQbosEW2WikC
U/MoQ+ZVhDnqPCfL5IlQETiCyVw+DBsamGxiXYWBMup6xaQb62bXdfpm7EKJleiRp+eiCDGYvL+Q
rw8Ga7a6Qyg+Xl/Pb0TZ+N8YmzUGaan06NEDgEOHDlFeXsHj6PHj2
48847E3KAWq2W1157rea/V69ezT333MPEiRNZvHgxJzvnkrIzxJCCCFE+9vnDnP7V04+qSf
rz25QeHyig1n9ZclvU/x4SBIFvgiPbnDXbNvrjvDDZWW8d36GFIA/Rjiqss8d8d8VIt8ZZu0BA8X
5JeRXhin1R4//Aiegh0XL2HQ94zMNjM8yMDrNgFG69gohEshPpWRqlx+rFC3zXxe17dIOb9aV
BXjgzVeoCCtFFtSgAmJmZyWuvvvcbbb7/NV199OVRFREQADBw7ktNNOY+MnmHiy6
6CJ1O6hQIIYQQXUkkqvL8L8L8Vi8PrHPhq2cuoUP6mP&#x202F;

0IyJqw4QUPTbMTcXNG2uw4VjzSn2PDNzxjRWH6iQ8dFIcD+UiikCdyuCNHj0aBQUFXd0WIiIiUpn
Pjzrxp7xa7LOGL8c8I12P5yZbMDIx8ktPI82do2NxpF7CGy32fPqxyoNbNtbg3XOSImK/u701Hvx
usxWby9xhH78+14z7x8dhYFzPTVoTEUW6ZKMGH5yfjL/sqMPzuxoCHvvbrnocrPPilWmJMGn7/uc
EETXr1PTuwoULsWHDBqxdu7ar20NEREQqsLvag0vXVeLyz6vCJv8yzSIWTUvEqtkpTP71EkEQ8PR
pCZjRL3BPp7VFTjy0tbaXWtU+dq+M//dDLaZ9XB42+TfKosW6OSl4aWoik39ERJ2gEQU8cnIC3jo
7EYagnSE+POLA3LUVKHeo8wR5IrXq1IjoD3/4AzweDx577DE89dRTSE1NhUYTul/MkiVLTriBRER
EFDnKHRKe3F6Hd/LtkEOrMWHWCrh7TCzuGhOLmAgqM1UrrSjgzbOTMGd1JX6qbj4t97W9NgyK1+L
2UbG92LrwPity4oEtVhQ2hF54mjQC/jAxDneMjoUuAlYwEhH1dZcMMqNfjBZXf1EVsI3HDxUenPt
pBZael8yJPKII0akEYGJiIpKSkjBgwICubg8REVGvUxQFxTYJpQ4ZdW4ZdW4FdR4ZtS4ZCnxJLLN
WQIxOREzj9waNAJ3oO0lPL/q+BwCXpMAlNd7KCjwyEK8TkGgQkWQUEasVVLEnmcOr4NU9DXj+x3o
0eMPvw3b1UDMePikeWTGReciEWsXpRCw5LxnnfVqOY/bmi7sHv69FuknEJYPMvdi6ZsdsEh7casX
HR5xhH5+ZY8QzpyVgAFf8ERF1qUlpeqy/MBVXrQ9c1V/UIGHWal8SkCcEE/V9nRohLVq0qKvbQUR
E1CsURUG5S8AvBQ7sqPJgZ6UbO6s8YQ+r6A46EUgyiEg1aZAdo0FOjAb9YnzfZ8doEK8XmxKOpsb
bvrSyab/Vg/cO2PH+ATtKHeH/zaZm6PHEpARMSNH3cOuovbJiNFhyfgpmr6poSuAqAG7ZWANFAS4
d3HtJQElW8K99Nvxlex3qPaHJ5SyziKdPs/D0aCKibjQwTot1F6Tipq+q8VWLw0Fq3QouXleFd85
Jwvn9jL3YQiI6Hk6REhFR1CmxS9hU4sKmEhc2HnPhqM0EoLpX2uKRgTKHjDKHjJ9blGC2RS8CFoO
IRL2IRIPo+94gIt0kItOsQYZZgyyzBhlmERlmTZcnDGtcMpYfsuO9A3Zsr2y9zUPiNfjzKQmY05+
JmUgwNkmHf5+dhKu+qGoq35YU4JZNNVAAzO+FJOCOSjfu+86KnVWh/UwUgNtGxuChk+IRx3JyIqJ
ul6AXsfT8ZCzcYsVb++1N9zskBVevr8Kr0xJx+ZC+sWqciEJ1OgFotVrx3nvvYdu2baipqcGjjz6
KcePGwWq1Yvny5Tj33HMxaNCgrmwrERFRp3hkBV+XuLCm0ImNJS78Uhv+RNpI4ZaBcoeM8lZW3LU
kAEg1icgwaZAZo0GmyZcUjNEJMGl8pctGjQCj1le6LAqARvAlVwQIUKCgsEHCwVovDtR5cbDOiwO
1XoSp8m1i0Qv4w8R4/N/wGOg1TPxFkhk5Rrw6LRG/+bqmKQkoK8CvN/l+7qkLuzq3jCe21+GNfba
we0lOTNHhhckWriolIuphOlHA85MtyDBr8NSO+qb7vY2fFdUuGbf1wf1jiaiTCcBjx47h1ltvRW1
tLYYMGYLi4mK4XL5lwBaLBZ9//jlqamUzERFRezm8Cr4sdmJlgQNri5yodbeKCLm0sERFRezm8Cr4sdmJlgQNri5yodbeRsQojRit
gaIIWCXoR8ToBCQbfrSgIsHtl2LwK7B7Fd+tV4JYVuCXfHn9uWYGn8XwCgwYwaAToGxNtGgGo8yi
odkqodslwdvMBegqak4W72rnCsLNitAJuHG7GgvHxSDRwRVakunKIGQKA24OSgLd97VsJeEU3JgE
VRcHHR5z4w/fWsCXl8ToBfzrZl1zW9KFSeCKiaCIIAn4/IR7JBhELttSi5Qjr99/XotIp46GJcVz
9T9THdCoB+NJLL0FRFLz//vuIiYnBzJkzAx4/88wzsXHjxxi5pIBERUXvVuWV8dtSJTwoc+PyoC/a
2lqm1oBcUTEgxYEKKDhNT9JiYokNuvLZHEgx2r4wqp4xSu4xim4QimxfFNglHGySUOiTYPL4Eo0N
S4PD6Eo7hVkT1FgHAtEwDrh5qxtwBRsSyFFMVrhhihigAt24KTALe/nUNPLKCa3NjuvzvPFLvxYL
NVnxe7Ar7+KWDTPjLqQnINPMQGSKivuCWkbFINIiNnw3N9z/7Yz3qPTKeOjWBSUCiPqRTCcC8vDx
cd911yM7OhtVqDXk8KysL5eXlJ9w4IiKi46lySlhd6MSnBQ58dcwFdzvO7tAIwEkpOpyVacS0TAO
S6wsxZni/7m9sGGatCHOsiJxYYFI7nq8ovmSg1aWgxiWjxi3D6pJR7ZJRYpdQYpNQ4pB9t3YJFd1
0mElughZXDjHjyiEm5MRyS2E1umywGSJ8JV1SiyTgnd9YsavKg8cnJXRJibdbUvDy7gY8s7MeDik
0uz0gVoPnJltwHjeXJyLqc+YPNsNiEHH9l9UBE6+L9tjgloC/TU6AyCQgUZ/QqRG7y+VCXFxcq4/
X19cz009ERF1OURTk13qRV+FGXrkbWyvcCUbRMzpb8Sc/kZMyTAgXt+8Ui0/v/va3NU
EQWg8Fdh3cuvxuCUFZQ5fMrDE7ksSljskOLwKXBLglJSmL4+kQIYvyeP78v2cYdJgSIIWQ+K1GBr
vu7WwxDcqXDrYDEHwnQbcMjf32l4bdlR68Nb0JGS3ox+GoygKPjvqwmM/1GKvNXRfTp0I3D0mFve
Pj4NZy/5GRNRXnZttxMczU3DF+krUuJo/LP693waXrOAfZ1i4bQNRH9CpBOCQIUOwY8cOzJ8/P+z
jGzduxIgRI05oYUREFN0kWcHBOi9+9qvHN0aXzJKSZaNqXa6RTeZwKF8fzRUykVCKL0kqRGSLm6VGUvnn6+MpZAovzWz7bOe2TPB/TWpwacPf5rqUNpBYQ89y88Uf6iz3sjHeyMYZkuYE6CG4xFw7pC2S9vV9J4Y5qQi2WbuCRKnG9Bm/tUWIFT1B6g7qYjkO9kGrdypTWP+TipNUv8Yp5dMPVVZaNKUnwOlLwbPBQQ4QjCYA5V/UaN5xl5HdoY8EV0FVHlfoHv1ebSjyZkZ0wM8frQeJG2Yb3QJZN7qi0X6Kdj9VMmYWcHmFemfi86QoSwSXKXJeMTJ2PUrjGjjW9yrGdQ2dLWXZnTnFkIk5F/pz/ZCzvIX2sl3X+MCV+5lVRB8UGNRUrlFX+2IKp+uiywGKsJxGy0eJTF/Fv3pMnyF6nXUmkvfcmrO/jhyEIm+VhsJa6YgpPcVp5RyoINgfEjHpX9BZq3QgwfxUdO7cspHTNjmPXWUV7KDgT35fQ8Gl2X5qyjYfFbgvyBm

```
pgROfFjiwsx1bFo1N0uGiAUaMF8p7oIWd16kE4AMPPIC77roL//jHPzB//nxkZmZCFDmgIyJSK0V
RUNAg4ZtSF74tdeObUleHk31+Bg1wdqYBFwwwYVaOEWmmvj1TRkTtIwgCJqToMSFFD0zs7dYQEVF
fIQgCbhkZi0lpevzfhuqAkmC3DDy5ox4rDjvw4hkWnJ5u6MWWUrTyyAq2V7ixutCJTwsdAX20NSe
l6HDxQBMuGmjCwDhfai0/v6y7m3pC2p0AXLVqFSZOnIisrCxcccUVUBQFixcvxuLFiyEIArTawF8
lCAI2bdrU5Q0mIqKeU2yT8P4BO947YMeBOm+nfodWAMYn6zApTY8z0g04J9uAWB0njYiIiIiiyfh
kPTZclIbffWfFskOOgMf2Wb2YtboS/zc8Bo+eEs8V49Stalwytpa7sbXche/L3dhe6YHd23Z5LwA
kGgRcOcSMG4bFYFRi5JWutzsB+Pjjj+Oxxx5DVlYWzjvvvB7dk+n2229HSUkJYmNjAQAjR47Eww8
/DEVR8NJLL+Grr76Coig455xzcNddd3G/KCKiE+CSFKwudODdfDu+POZqOtWqPUQBGBSnwUiLDqe
k+k7pnZCig1nLQRwRERFRtIvTiXj9zETMzDHiwe9rUeGUAx7/934bPi104LejY/GrETGI46QxdYL
Tq6DCKaHcIaOg3ovD9RIO13txuN6LI3VShw4wEwBMyzTgxmFmXNDfBGMEH1zT7gSgojRfAT766KP
d0pi23HnnnZgxY0bAfZ999hl27tyJJUuWAABuu+02fP755yHPIyKittm9Mr4oduGTAgfWFjlR5z5
+1k8vAqek6nFamh4jE3UYYdFiWIIuoj8UiYiIiKh7CYKAywabcW62EX/Kq8X/8u0Bj5c7ZDzyQx2
e21WPW0fG4vZRMUg2csuYaNfgkVHhkJsSexUOGeVOyXfrkFDhbLx1yKjzdGAFQxg6ETgr04ALB5g
wO8eomoPLOrUHYF+xbt06XHLJJdDr9QCAiy++GGvXrmUCkIioHeo9Mj4rcmJlgQOfH3Udd9m7TgR
OS9NjaoYBUzIMOCVVDxOTfURERETUCYkGEf+cmogrhphx33c1IfuuboVPPtj7e3YAbhplx84g
Y5CZEXtklBVIUBU7Jdy3S4FFQ55ZR71FQ7fIl8HzJPQnlzsZbh4wKp9yuEt0TkW4SMSXDgAv6G3F
eP6Mqy9A7lADszdLa1157DW+99RbS09Nx8803Y+zYsSgtLUVGRkbTc7KyslBaWtprbSQi6utqXDJ
WFzqwssCJDceccm1Vj9fvYJB2uzTXj8sEmzr4SERERUZc6M9OAb+el47ld9XhldwNsQYkeu1fBoj0
2LNpjw6hELS4ZaMLFg0xMBnYhl6Sg1C6hzCGhxC6j1C6h3CFBAaARBGhFQCcK0AqAtuWtiKbvnZI
Cm0eB3avA7pVh8yq+xJ5TRo1LRnXjl9Ulo5tzecclCsCoRB1OT/NtWXRqmh4DYjWq305OsFqt7fq
nP+2006DT6dr9D9KVh4CUlpYiPT0dgiDg66+/xp///Ge89957uPvvu3Hfffdh0qRJAIC8vDy88tpyLFK
LWLx4cdjfk5+f3Kgx9qRUjK8eN4ok7BjBQv5qZ7MTy2lz+hiYiIiCg
qWD3AshItlhzTodbb9pg11yxjeooXZyVJyI1RoPLcTZdQFKDUJWBfg4j9NhH7G0T8YhNZAjc
NFFh0QJJeQbpeQT+TjH5GBf2MMrKNClS4wA+5ubltPt6hFYCjR49Gdnb2CTWolZVVWJ6enqDP
yMzOxf/9+pKenB6z4O3bsWMv3vH4PULz8ncod8/D8
IIMss4sIBvqPsJ6fpoRGjZxSlxr5Ancf+QH7sC9QS+wP5sS90r0mjgEc8Mv6z34aXdzegxB5+LJt
vF5FfqMfrhUBOrAazc4yY09+IKRkG6HpwHNvX+0O1U8LnxS6sLXRiQ4kTNa7InNzXiUCaUYNUk4g
0k4hUkwZpxsZbk4hUY+OtSUSSQYTYCxnhvt4XOpQAvOSSSzBr1qzuaktYLpcLdrsdiYmJAIDdu3e
juLgYQ4YMwcyZM7F06VLMnDkTAPDRRx/hyiuv7NH2LZjU9TkrR9coHJRERERFRS7E6Eb8dE4dfj4zF8kN2LDvkwNclLkitDHOLGiS8vteG1/faEK8
TMDXTgDMbv0ZatKov7Qz2i9WD1YVOrDvqxPflbsh9LOenE30nQsfpBMTpfbcWfWBiL83UItln1CB
BL0Td/2NX6/OHgNhsNtx7773weDwQRREGgwFPPPEEMjIyMGPGDOzfvx9XXHEFAODss8/mASBEpGq
yomBnpQdripz47KgTu6o86OjnQiLFnMHmDB3gBFjk9q/tQMRERERUU8yaARcmxuDa3NjUOmU8Gm
BEx8dcWBTiavVpFadR8HqQidWFzoBAKlGsfEQOz1OSdVjdJKuR1cKOhBrSrS8YHh+14N9+O7ZWeDv1
ZUfAdgpFu0iDDrEGm2ZeA04kCvLICrwLfrQx4lcbbFvdLCqDXCDBrBcRofbcmrYA4nYhEg29FXpJ
BRJJRRKJehJEHCfaKPp8ATEpKwjvvvvBP2MVEUcc899+Cee+7p4VYREfWcBo+MTSUsUurC1yYl2RE2X
tLOn1i9UKOD3dd3rvnP5GDLNww2QiIiIiwpRg1uGh6Dm4b7koH+JN+GY0442zjYrsIp48MjDnx
4xAEAMGiA8Ul6nJSqwympepycosfAuMg8AEKSFWwsceHdfDs+LXS064C/WK2Asck6jEvSYVyyDuO
S9Rhh0aoyKUqB+nwCkIgo2nhlBdsq3NhQ4sKGYo7klbs7dFJWvF7Aqal6TMs0YGqmHGqGAeOTddd1w
IL04PLwAAIABJREFUiIiIVCLFqMENw2Jww7AY2L0yNpe48XmRE58ddaHI5u3QXnmKc+7gJSUthG0Mwu9eKrnmWJYeEyprj
BAJIMIk5O0eGkVN8qwZNSdEg2anrglXScW1KwqcSFTwocWF3oRMVxXi8AjE7UYlaOEbNyfNv+RNN
e39Ss3QnA77//vjvbQUQUtWrdMnZUuvFtmRubSlz4tswFh7fzl7Sz4OIFQ5UnD9FXM/VUZBndMeYa+jOc7xP4zEAMhI4EzYgvbTkfrxo6mtowIxm2ZmJ7uPhnPFb2
Ywsby0xXsu/fctPpH9B7kQMz3L9LhTcpHbmpYE/XuojuvdaWpDUWp3xCqDaG0dVGK5qu8jYHh0c6QgQ
nIIQkzSSmJJR+2+PprJJRJhjb+oG31Sqjrv1orqpP98fyG8iWpnCIkJoBpVJRhM6HT3RvPmpPfEU0
UPPAJWUwqOQOpD+y3OlKrz0QWamVNQtbvuLUMJ69ts/NK7Ok1AmT0v+j5w1GqXzYJvzKbnfgCaz8/FZfh6vpvDxtjKpDIpFoWqSS6KOnDCRYqQXzZRNfHwk5YsbRNQt+6BlftcfHxYPu0yXoZvwWXb5TTRzu9pqF8qhTMH0T0pd4tXuvgt3T6Pl+2zRCDkUcavq3LyJwdO9OZBsJeUcnJe2jc7Ju9Uw4q+7iSGiNq7ggAWHkxW+fvkqyDmYuZzCn0xRnhqXI/IfdqE+
jT40He2q82FPjwX6rBwfqvCixByf6NACOn/yL0Qo4N9uAy4eYMaOfEQYNk35EREREREQUOZgAJKK
I5ZEVHKj1Jfl213iwuzHhV9QgnfDvHp6gxfn9jDi/nwGnpxuY9CMiIiIiIqKIxQQgEfUoWVFg9yr
wyr7vJQWQFEBWAElRIDd+7/9ZUoAql4xjNgkldqnp9mCdF/m1XrjbX73bKqMGGJukw4RkPSak6DA
t04D+sQyPREREREREpA68wu0B1U4JJXYZHlmBRwbcsgKP7Et0aARAIwrQCIBWEKAVAbHF9xoB0Iq
+lUeS7EuIeBsTJpKsQK8RYNYKMGkEmLS+7/3PJ2ovp1eB1S039UsFrSfkZAXwKr799OrcCho8Muo
9Cho8zd/Xe2Q0eJTG++Wm24bG57XjfI1uoRGAgXEaDI3XYmiCDomuKswcnYMRFi10fN8QERERERG
RSjEB2AMWH7Dj4by6Hvv7DBogQS82fgmwNH3v+zlBL8Ji8H2faBCRadYgK0aDOJ3YY22krqMovsS
y3etLvNV7FNS5m2+tbhk1LgVWl4wat4walwxr41eNW4bVpcDRniNvI4goALnxWoxK1GFUohYjE3U
YbtFiQKwW+halvPn55chN0vViS4mIiIiIiIi6HxOAPUDfwyuLXBJQ7pBR3oGTTQEgXicgK0aDfjE
aDInXYoRFh2EWLYZbtEgxarqptRROjUvGzko3DtV7UWqXUeaQUGaXUOqQUeWU4fAqcMsKXJLSJSW
wkSzDJGJUog6jk3RNCb/hCToYtVzRR0REREREREAQwAdgj9BFyeECdR0Gd1Yt9Vi/WF7sCHksyiBi
dqMVpaQaclq7HpFQ9LAauGOwKTq+CH6vc2FbpwfZKN7ZXuHGo/sQPsejLTBoBOo2vJFeEAI0IiAA
0ggBBaLxf8P0sCkCCXkCmWdO0WjXTrEF2jAYjLVokMTlNREREREE1CYmAHtAkkHEyMY9xvQaQCc
K0Im+xIYkNx+CICm+gxG8CiDLvr3+vI2PywC0jfsBahoTJBpBgEf2HajgkBQ4vApsXt8ebV2t2iX
j61I3vi51AwAEACMtWpyapsfJqXpMTNFjhEXL/QfbweaRkVfhxrelbnxb6sK2SjdcvZzv0wqAxSD
CIAYm4HxfQmOiDhAaf9aKQKxWQJxeRKxOQJxORJxOQKzO/7Pv+/gw98XquE8lEREREREREUU/q8wl
Al8uFhx9+GIcOHYLRaITJZMK9996LMWPG4PXXX8eyZcuQlpbW9Px33323F1sb3kUDTbhooKlH/i5
F8SUDa90Kat0yal0yat2+Ax5q3XLz/W7fHnCVThnHGk9W7UgpqQJgj9WLPVYv/vOLHQBg1goYl6T
DhBQdxifrMdKixTCLFmZt9K4UlBUFB+u82F7pwbYKN7ZXurGz0gNvFydptQJg0gqI14mI0/uSbXG
N31v0IhINYvNt0/dC08+xWgGCwKQcEREREREREkRr1+QQgAMybNw9TpkyBIAjYuHEjFi5ciNWrVwM
AzjnnHDz44IO93MK+QxB8JwGbtUCmuf2lkYqioNIpo9gmoaBBwn6rB7/U+sqBD9R64GzHCjW7V8G
Wcje2lLsB2HztAdA/VoMRFt+egsMtzXsLxqro0BGvrKCwQcLBOm/T136rFzur3KhzdzzbpxGAERY
txiXrkW3WIN0sIt3kK31NM4mI0flWkRoaV5WKTN4RERERERERERUSv6fALQYDBg6tSpTT+PHz8e1dX
VcDqdvdgq9REEAakmDFmFYuSrKCgQUJehRvfl7mxpcyGSpPWWUgAUNPiSiuuO0Bu4
rmBPr28NtUJwWGWYNMswaZJpFZJg1SDdpYNYK0Ivo9ZVpsqKgxiWjwimjzC7jqM2LogYJRTYJRxs
kFDV4UdggndCqvpxYDU5L85VSn5yiw7hkXVSvnCQiIiIiIiKirtPnE4DBFi9ejMmTJ8NoNAIAIANm3
ahJ9++gkmkwnz58/HnDlzermF6qMRBQy12JwvBZXDjEDAKwu3z522yrc2FHpxvZKDyqcHTuqh
BQlGDBMDV5vOMGsCgEWDUCEG3rd3fvNeitnGvRI3oW1VXU62FxVbn21dRBryKb49Fp+TbS9HuUWD
3ymjwKk0l0pVOGVIXl+wOjtNgSoYBUzIMOCNDj/6xEfdWJCIiIiIiIqIIIVit1m44MqJ7rFixAu+
99x4WLVqE5ORkVFZW4hxWf8Hnq9HgUFBbj77rvxAMPYNqmgTeBhG
H7AIO20UcdQpQEN3lqXEaBaPiZIyKlTEqTsboWBmphoh52xEREREREREERFRH5ebm9vm4xGTAFy+fDm
WLl2Kl19+GampfxWGf8+KLL0IURdx999Dpqjc0OrIL/Wg/1WL/ZZPdhn9e2Nd6je2y2nFfemNJO
IIfHapq/B8VqMSdRhcLym18uYKbz8/PzjBkmKDuwL1BL7A/mxL1BL7A/kx75ALbE/kF9f7wsRUXe
4ZMkSrFixAq+88gpSUlKa7i8rK0K0N6ejoAoLq6Gps3b8Ytt9zSzSWFjmxvp/
Oi/1WD47ZJJQ6ZJTaJZJTYJZTaJVQ6ZTglpUMnE3eneL2ANKMGqSYRWWYNcmI16BejQU6sFjmxvp/
jVHSoCREREREREREREGpR59PAJaVleG5555n7n7nnnnqb7n3/+eTz/eTz33HmoLCyERqOBoiiYN28etWDY
//F5sLbWXUStgTJIOY5J0bT5PVhS4ZFMAlKXA2frkCbhH0s+/W4fXXt7eeVAalxnz9JBiqrq5GSnNS
4NyCgFQVoBECv8Z2eHKP1n6IsIl4vINUoItWkgUHDFXxEREREREREREEFJn6fAIwPT0dW7duRx88
808OtoZ4mCgJMWt9Kwq6Qn1+G3Nz4Lvld0REREREREREREESRIGL2ACQiIiIiIiIiIiIiIIqKO46Z1REREREREER
EREREKsYEIBEREREREREREkYoxAUhEREREREREREKRiTAASEREREREREREREREGpWJ8/Bbgll8uFhx9+GIc
OHYLRaITJZMK9996LMWPGwO124+mnn8aOHTsAAJdddhmuvfZbapj/77rvvvYnY5QCACRMm4MEHH4R4MEHH4R
erwcAbN26Fc8//////w8Hg9SU1PxyCdf+dfIHVId/SHAwcO4Nlnn0V1dTW0Wi369A/uLaU+LAEFVTW0Wi369++PhQsXIjk
5uVdeI7VPd8UGv5dffhlvv//023nzzTYwdO7bnXh1Snf1h8rKSpvztb39Dfn4kKsN39xt39Dfn4+dDodJk6ciN///vc
9/wKp3bqrLyxbtgwffcfffPABNBoNFNoNBEHwAXfeiNicmTP76P76178iPz8fn8fKSkpWLZWLZsWcD
v5Tgy8nRHX+AYMnJ1V2zw4zgycnRXX+AYMjJ1V3/ozXFkRCUAAWDevHmYmUKBEHAxo0xo0xQbsXDhQqx
evRqLFy+G3W7W2ppPBHvihlOzUF9fj+uvvx4/zuPhk4vvJvLaPsQfvrGjk4ct3WWFjspsWvXLixxfvhxvv/+024uLBhzhizhQ3cnu48cYb4XQ
68ac//0yQlQKUtfOABhCPiMCAIdLqhEXvTKfcW/T2S6V+26ru+2f//992PYsGEAgIdjeeAEvvPACnnj//+//
iiV5+pXQ8Xd0X/LZt24YDw4gMzOzF17ddVR39dJV8RYXF2e+bfevNNuWgifWsg1O3o93wBwK2TPWwMR+Se2
KFStWwGKxYMuWLXjsccew3u6MjHnkjj/nX7Lx3T3H5Hvl///Q8Wiw//hnM/f/3hzvw/jyMjV1X2BY8jI1t9z
9wY/jyMjTHX2BY8jI1dX9obfHkRFVAmwwGDB16lQ1/+ZKqKrrq+F0OrFu3TpXtcEttllBBEAcB8fH
xmD17tdtM/4rp16zB79mzEx8dDEARckcfnvlTY999913GDw+HwbebDe8H9zgYH9zcKEZgGMEsOAu8P7F9+x3HZrde8bdeZ3HHUbt3
RH/r37980cAOAcePG4PDdixYz3/4qhDuqhMvAFBBdXR1rfeOEFPPGPf+yV1oWd0x39IS8vDw0NDbjjsssu
a/p6U1lJSef3HUId3RFwBRgKIosNvtAID6D6+nqKkpaX1zgukDuCIasa4ceNgPmlClCfifCHkZGpOO/o
Cx5CRqzv6A8BxZCTqjr7AMWTk6ko7o+0NvjyIhKAAAZSkt
LkZGRR0fRYdnZ202MlJSUBjxmNRiQmjY9TpGjKQ/pDS16vF0uXLsX06zaxs55JO4+ea
b+SEd4bqiPxw6dAhpaWl4/PHHcf311+PWW2/FXUc+O60OD/Ezz//3LMvhE5YVV/SFfv364uPmoE5IYrD4DdiyZy6CM8
99xweeeeSRnn0h1CXa2x/fe/99zdrY0PfRZhhvXn7YTUJiQ4Cs3P7FP5ZfOO7jqDmoJHYvQw2PNUdIS7y9gWjCEjS7
++uorLFq0fq0qLebQn1AV/cHWZbx+OOPw+Ywf+4YxbvQnA1F3ZMvJ1RX1fV/WFlStXwmQYCfAe4bqqP3i9x9fXmzmfzfvh2vvyo
qxo8fj6j6lbt+3+ssJ3v/sdPvzwQ8sTExHRRa6k7dVVfVKfCsrrlRFVYTEIEBxFLIM6J/cQ+wajc0zSdMwzL4mu+lS5fCCYVDB
0UWupu3EcSX4cQ1JLHEeSH8eQ1JJaxpERuQbaA9+cbp6OkpKSpuuc+c0cO3
```

```
2IyMjIBtbXFzc6mNOpxM1NTUBs7nUt3VlfwAASZLw6KOPwul04i9/+QtEMSLfJlGpK/tCXl4e8vL
yMG/ePMybNw/l5eVYuHAh9/mKIF3ZH7KyspCTk4Px48cDAE499VTo9XoUFhb21MuhE9CVfWH9+vX
o169f0yEPZ555JhwOB4qKinrq5dAJ6mh/aAvHkZGtK/sCwDFkpOvK/sBxZGTryr7AMWTk68r+0Nv
jyIj7VFqyZAmWLVuGV155BampqU33z5w5E8uXL4eiKKirq8OaNWswc+bMpsfWrFmDuro6KIqCZcu
WYcaMGQCAM844A4cPH8a+ffsA+DK7EydOhMVi6fkXRx3W1f3B6/XiT3/6E7xeL/7yl79Aq43YRbJ
Rp6v7wuOPP45PP/0UH3/8MT7++GOkpaXhmWeewaxZs3rl9VHHdHV/mDx5Mmw2Gw4fPgwA2L9/P5x
OJ3Jycnr+xVGHdHVfyMrKwk8//YT6+noAwM6dO+HxeHjqa4ToTH9oC8eRkaur+wLHkJGtq/sDx5G
Rq6v7AseQka2r+0NvjyMFq9Wq9Mjf1AXKysowd+5cZGZmIjY2tun+559/HhaLBU899RR+/PFHKIq
C+fPn4/rrr296zn//+1+sWLECiqJg/PjxeOihh5qWWG7ZsgUvvPACvF4vkpOT8eijjjyI7O7vHXx9
1THf0h7Vr1+KRRx7BkCFDmmZtk5KS8NJLL/X466P2667Y0NK8efPwxBNPYOzYsT3ymqjzuqs/bNu
2DS+++CIkSYJOp8Odd96JU089tcdfH7Vfd/WFRYsW4YsvvoBOp4NWq8Udd9yB008/vcdfH3VMZ/t
DZWUlbrrpJrjdbtTX1yM5ORmnn346Hn74YQAcR0ai7ugLHENGru6KDS1xHBkZuqsvcAwZmbqrP/T
mODKiEoBEREREREREREETUMRFXAkxEREREREREETtxwQgEREREREREGRijEBSEREREREREREpGJ
MABIREREREREREakYE4BEREREREREREQqxgQgEREREREREGRijEBSEREREREREREpGJMABIRERE
REREREakYE4BEREREREREREQqxgQgEREREREREGRijEBSEREREREREREpGJMABIREREREREREak
YE4BEREREREREREQqxgQgEREREREREGRijEBSEREREREREREpGJMABIREREREREREakYE4BERER
EREREREQqxgQgEREREREREGRijEB2M0URUGJXYKiKL3dlF7lkhRUOaweby8/N/N79e+vcclweO
7LwBAmV2CJPf+v0Nv9gfGBh/kZOr4JqxgbGhkaldglyH3hPMDb0PsYGn2qnxNg
AxgaAscGPsaF9mADsRqV2CXPWVGLkklKcsqIMZfbe75C9YcMxJyYuL8WQ90rx643VfSJI9zRFUfD
szjqMXFKCfv87hvcO2Hu7Sb2izi3jqvVVGL6kFCOXluJArae3m9QrSuwSZq/2xYZJK8pR7ojO2PB
VcXNsuHVT9MaGZ3bWYURjbFhyMDpjQ61bxpXrqzBiSSlGLS3FwVpvbzepVxyzSZjVGBtO/bAcFVE
aG74sdmLiB6UY/F4pbo/i2PD0jubYsCyaY8Pnlb7YsKQUh+qiNzbMXOWLDad9WI7KPnCR2xvWH3V
iwnJfbLjj65qoTHgoioInd9Rh5NJS9PvfMXxwKDpjg9Ul4/LPmmPD4SiNDcU2CTNWVWDkklKc/mF
5n5hU7w2ft4gNv/0memNDe64jmADsJiV2CXPWVGLkklKcsqIMZfbe75C9YcMxJyYuL8WQ90rx643VfSJI9zRFUfD
AskMOrDzi7OVW9SxFUfDoD3X4y456OCVAUoD7vqvpEzMUPanWLWP+Z5VYW+T7/y93yPjd5uh7Txy
zSbhwTQW2lPtiw4E6Lx77oa6XW9Xz1h914qovmmPD0oMOfFIQfbHhT3l1eHJHPVyNseHeb62occm
93bQeZXXJuHRdJdY1xoYyh4z7t1h7uVU9r7gxNnzfGBvya734f9uiLzZ8VuTEVeurUNIYG94/6MC
qwuiLDX/Mq8XTO5tjwz3fWWGNwthwybpKrDvqAgCUOmQ8sDn6YsPRBi8uWFOBrRW+2PBLrRd/jsL
YsLbIgWu+qEKpw/c+WHzAjtVRGBse3FqLZ1rEhru/jdLY8FklPi9ujg0LonDcUNQYG/IqfAsq9td
68XgUxoY1hYGx4X/59qbrzWihKAr+8H0tbttUc9znMgHYDfwX+AeCZiJWHHZE1Uqfz486cc0XVXA
FveRFexp6p0G9QFEUPJxXh3/8HPianRLw9i/RM2Pnv8D3f0D5bSpxYXd19KwC9F/gH6wLfFMsP2S
PqpU+64oYGxRFwUNba/HP3YGv2SEpeOcXWy+1quf5B/HbKgPjwIZjLuytiZ7YcLTBiwvXVOBQfeC
bYtkhe1St9Flb5MB1X1bBHXQtG22x4cGttXhld2AcsHsV/DeKYkONS8bF6yqxPSg2fHnMhf3W6Ik
NRQ1eXLi2EoeDYsPSg/aoWumzutCB67+sjvrY8Pvva7FoT2AcsHkV/C8/umLDvHWV2BEUG9YXu/B
LFMWGwgYvLlxTiSNBsWHJQUdUTSSvKnDghq+q4QmJDdHznlAUBQu31OK1ve17zUwAdjH/LF3wBT4
AuGXg3/uiozOuLXLg2jAX+ACwpdyNnZXunm9UD/MP4l/eHX5w8sZeGzx9YK+r7tbaBb7fa3ujY/D
mn6ULvsAHfLHhrf3RERvWFIa/wAeAzWVu/FgVHbHhD9/X4tVWBif/2muDNwpiQ2uDeL/XouTCrrD
BiwvWhF7gA4BLAv6zPzomi1YVhL/AB4BvS93YFSWxYWGYC3y/1/dFR2yodkqYt7YSO6taiw3R8Xl
ZUO+LDcEX+EB0TSR/WuDAjWEu8AHg61I3fo6CiWRZUbBgSy1eb+UC//W9tj6xf253q3ZKuGhtJX5
sJTa09u+jNkcaY0NBQ2hscEgK3o6Sa4qVR1qPDRtLXNgTBRPJsqLg/s21+FcHckxMAHah1mbpWvr
3fhtckroD9JpWZulaelXlF3atzdK1VGyX8GmBowdb1fOOd4EPRMcsdmuzdC29uc8Gt8pjQ2uzdC2
pfcauPbN0R22S6ksejzeIB6JjFrug3hcbwg3i/d7c16D6yaJP2rjA92vvzHakkhUFD2ypxb/aeJ1
FDZLqSx6rnBIuWleFXW0kdd4/aFd9yeORet81RWEbseGNveqPDR8fceCm48UGlV9T+C7wrXijjQv
8wgYJa1Re8ljplDB3bSV+aiM2vHcgSmLDmkoUtRUbomCy6OMjDvxqQzXaOgcnGmLDfd9Z8e8OJny
ZAOwivtK+0Av82TlGxOmEpp/LHTI+PKzepM+awvAX+FcMNgX8vOKwQ7WHoviTf8GzUEkGEedlGwL
uU3Oyw+qSw17gT07Xo1+Mpulntc9iH20If4E/p78Rsdrm2FDmkPHREfXGhtWF4S/wg2PDB4fsqt0
qwZ/8C56lSzaIODckNqh30GJ1ybhoXVXIIP6MoNig9lnsosaVf8EX+Bf0NyKmRWwoscv4WMWx4dM
CB371Vegg/vKg2LBcxeXQSuPqnjeDYkOKUcQ5WdETG2pcMuaurQxZ0TUlQ49sc3NssHvVvVWCf9I
w+AL/wv5GmFvEhmN2GZ+oODasPOLA/wVd4AsIjQ3LDql3IllRFPzuOyveCloJnmIUMT2KYkO1U8J
FayqxuyZwe62pGXpkmpvTGTavgv+quBz6SL0XF6yuxFFbaGwwaZpjw1GbhE9VvK/2x42xQTpObFh
y0K7aPfcVRcG931lDrqFTjcdP7zEB2EXu+bYm5AJ/3kAj3jknCdcMNQfcv2hPgypPpql2SrhtU03
IBf6LZ1iw6MxEDI5rHrx5ZHQ4Wx0pVhU6Q5J/yQYRK2el4IlTEwLu/77cjR0qLYf+U15t2Av8Zec
n49aRMQH3q3kW+65vrSEX+BcPNOHt6Um4Ojc6YkNVY2wIHsT/Y4oFr05LxMAWsUHN5dCfFDhDkn8
pRhGfzE7B45MCY8PmMvVulfDHvNqQC/ypGb7YcMuIoNig4lns335jDRnEXzqoMTaEGTeoUaVTwu1
hYsNLUyxYNC0RA2KbY4NLAt5S6TYqKwucIcm/VKOIT2aFxobvytRbDv3Q1lrsCbrAn5ahx9LzknF
z0LjhdRVvlXDn1zUhseGywSb8Z3oSrhoSHBvU+Z4od0j4zdc1IRf4/5zqiw05sYETyWrdKuGjIw7
8J+gCP80k4tPZKfhzUGz4RsXl0A9urcUea2BsOCvTgKXnJ+PmEbEB9/9LpeXQiqLgzm9qUBy0gOb
yxthw5ZDA5Jdaxw1l9vCx4eWpvmuKaFlksuKwA+8Evbb0xthwPEwAdoG8cjfWN55C5HfJQBPeOCs
JOlHAbaNiIbR4bGeVp+mUPzV5ZY8NdZ7md6P/Av+m4TEQBQG3jgoM0P/ep75yaEVR8Ned9QH3+S/
wxyTpMMKiC5mxU2M59JF6LxYfCAxK0xov8GN1Im4YFhMVs9jfl7nw1bHA2DB/kAlvnJXoiw1BFzT
```

```
bKz3Iq1BfbHh5dwPqg2LDS1MtuGFYDDSigFtHhsYGtZVDy4qCv+4MPJnNf4E/KlGHUYk6nJWp/tn
8I/VevB8UG85sHMTH6ETcODwmKmaxN5e5sLEkMDZcPtiE189MhFYUcNuowNjwQ4UHP6gwNvzz5wY
0eANjw8tTLbi+MTb8OihGqnGrBFlR8NcdgbEhzeQbN4xM1GF0kg7TMvQBj6sx6XOozoslBwMnXAQ
FAAAgAElEQVRjw9lZBixpjA03DTPD2Hxdp9qtEr4tdeHr0sD3+hWDTVg0LXxs2FrhxnaVxgZbUGx
4ZVoirs1tHDeMCI4N6ptIlhUFzwRdU6SbRHw6KwUjLDqMTdJhSkhsUN+44WCtF8sOBV4jTM8y4P3
zkmHWirhpuBmGFrGhsEHCahWWQ39T6sa3QbHhyiEtY0PgWFqte+6/9HMD7C1igygAi85MxDW5MdC
GGTeocc/9cLEhozH5N9yiO+6fZwKwCzz7Y+DA7fQ0Pf7VeIEPAIPjtZiRYwx4jtoGb1aXjNeDPnQ
emhiHG4Y1vwmvGWoOKIeucMpYobJy6LVFzoBVbwKA5ecnY1Ri85vx9qAA/eFhB0pVVg794q76gJm
ZIfGapkE8AFgMYlTMYj/7Y2Bwnpyux2uNF/gAMDRBhxn91F0WXuOSQ/a0evikeFyX2xwbrs01h5R
Df6iyhPCaQmdA6YoA4IMZyRgZEBsCBy1qPDn++aDYMDRei/fPS4JZ64sNiQYxKmaxnw0auJ2Rrm8
axANAboIuzJYR6vp3qHZKeCMoNjxycjyuaREbrsuNCSiHLnWorxx6VaEzYGWLLzb4LvD9gscNajw
5/vld9Wh5jTYsQYv3zk1uig1JRg2uCBk3qOs9AYSOG6Zm6PFqi9gw3KJTfVl4lVMKWRH72CnxASu
jrw8zkbxSZbHhkwIn9raIDaIArJiRgmFtxAY1nhz/XFBsGJ6gxeJzk2Fq/P9PMWpw+eDoiw3TMvR
4ZWoiNI2xYWSiDmerfJFJpVMKqSD8fyfH48oWnw03DAucSFbjnvsrjzixvzYoNsxMQW7C8ZN/ABO
AJ2xHpRufHQ2cxX9wYnzTB7Xfb4Iu7D4pcOBoQ+BS5ki2aE9DwOq/ZIOI34wO/FCK14u4VsUlj4q
i4Jmg4DxvoAkTUgJn587vZ8CQePWWQxc1ePFu0AqfB8bHNw3i/dQ+i72tInRl8ENhYkPw4O3jIw4
U29QzeHt1T+DqvxSjGJLoStCLuCbKYsMlg0wYlxwYG2bmGDEoqBxaTSfHFzZ4sTg/MDYsmBAXJja
oexY7r9yNL4NWBv/xpPimQbxfcGz46LADx1QUG17ZYwtY/ZdqFEM+FywGMew2KmqhhJnFnz/YhLF
JgYP4WTnGgHJotW2VEG5l8ILxcU0X+H63Ba0WV9vJ8d+XubChE7HhwyPqmkh+ZXfg6r80kxhSKWA
xiKreKsG3widwgcllg0wYHRQb5uQY0T9oqwQ1lUMfqQ9dGbxwQmhsCH5PqO3k+M1lLmwKqhp4OGx
sCJ1IVtOe+y8Hrf7LMIn4dVBsSDSIuGpo8ESyej4vZUXBM0GLzy4fbApYbHQ8TACeoOCB2+lpepy
ZqQ953lmZBoy0aJt+lhS0eZpTJKl1yyEzDL8dE4tYXWj3um1kYDn0j1UebFFJOfT6YlfIabcPjI8
LeZ4ohJY8vqWicui//9QQsA/kwDhNyKasgPpnsYMTPpPT9ZiaERobpmcZMDwhMDa8uU8d/w5Wlxz
yf3rXmNimlaAtBceGHZUebFVJbPjsqCvkMJz2xgY1nRz/4q6GgL3eBsdpMH9QaGxQ+yx2cNXAlAw
9pmQYQp53TrYBuS1ig1dRT0I4XNXA3WNiQ5LBAHBr0AXNtkoP8lQSG8JVDdw/LjQ2aMTQbVTUVA7
94q76gNgwNF6LS8PEhtFJOpwZslWCOt4TQPgVPpPTQ2PDef0MGBrfHBs8MkJWzEWqGpccso/23WN
iQxI+AEK2Ucmr8GCbSiaSw1UN3B9m3BB+qwT1lEMHVw3kJmhx8cDQ2DAmSRcyxlbTyfHBVQNnZRp
wWpjYMKOfUbV77lc7pZCKorvHxsEYLjYEfV6qac/9VYXOgL1yBYS/pmgLE4An4Mcqd8iR6wsnxEE
QQjuiIITW5v9nvw12b+QfVf6vvTbUupujc6JBwC1BH0Z+g+K1mBlSDh35F3ZKmP29LuxvxJik8Nn
4a3LNiA8qh/7gUOTP2B2zSSEn890/Li5k1ZufWmexd1a6sS4oNvy+Q7HBDkdb59pHiNf3NqCuRWx
IMoi4eUT42DAkQavKcmglzCz+RQOMrc7UXZtrVuXJ8UcbvCEn890/vq3YoM5Z7HBVAwvHx4d9rii
E7hP61n4bnCqIDeGqBn7VSmzITdDhfBWWQ7dWNTCyldhwXZitEtRwcny4qoH7x8eFrGzxC44Najk
5PlzVwMIJrceG4MPU1BIbwlUN/Gp4+NgwzKLDuVESGy4ZZGp1b6/rg7ZKUMvJ8eGqBh5oMzaEbpW
ghnLocFUDCyeET/ioec/9cFUDNw03h32uWvfcb61qoL2lv35MAJ6AvwUF51NSQztbS1cMMSHR0By
0rG4FSw9GdoCu98h4eXfgv8Mdo2IRF2aFj19wOfSnBU4URXg59IZjLvxQEbjCZ0ErwRkA4nThyqF
tEV/y+Pef6uFukdPOidXgqqHhgzOg3lns4Fn8U1P1IQc8tHTlEBMS9M2xodolY1mEJ4Tr3DJe2R3
4YXvn6PArg/2CB28rVbBVwpfHXNjWjpXBfvH68CWPER8bfg5cGTwgNnQ/r5bUOovd3qoBv6uGmhH
fIjZUqSA2dKRqwO/20aFbJUR6OXR7qwb8EvSiKk+Ob2/VgN/MfkZVnhzf3qoBv6uDJpIrnTI+OBz
ZsaEjVQN+4fbVLonwyaL2Vg34qbUcur1VA36zg7ZKUMvJ8e2tGvBT4577Haka8FPjnvvtrRo4HiY
AO2l3tQefBJ1MuGB8fNgVPn5mrYgbhwUmvyJ98PbmXhtqXM3tj9eHzjwEOzPTgFHB5dARvEw73Mm
/s3KMGJ/c+sANQMjp0LuqPdhcFrnLk0vtEt4Os/pP18pMHaDOWeyfqj0hJxO2tjLYL0YXJjbsjuz
Y8MY+G6wtVv8l6EPLVIKdnWXACEtwOXSEx4YdgbFhTn9jyN5/wdR2cnyJPczK4PHtiA0qm8XuSNW
AX6xOxA256ho3dKRqwO+cLAOGBZVDR/JWCR2tGvBT28nxHa0aANDqyfGRHBs6UjXgF6cTcd0wdU0
kd6RqwO/c7MCJZK8Kxg0dqRrwU9vJ8R2tGgBaK4eO7K0SOlI14KfGPfc7UjXgp7Y99ztaNdAWJgA
7KXj13/jk0NM8w7llRAxaHEyDfVYvvi6NzABt88h46efAAfjto2KRoG+7W4UreXwn3xaxJY9fl7p
D9jFc2I5a/IFxWswKKocO3tsgkrz0cwNarrTvF6MJmZEMJ9wsdiSXNf0taKbupJTQEpVwbhkZg5b
jmj1WL76N0IRwg0fGP4Niwx2jYxHfntgQdGH39i/2iE0IbypxYWtFx2NDuJPjIzk2/OOneriCYkP
wKeDhhJvFjuSypo5WDfj9Oig27K7xRuxkUWeqBgD/uCFwsP+f/faITfp0tGrAL9zJ8ZEcGzpaNeA
X7uT4TyL4lMeOVg343Rq0d+5P1ZE7WdSZqgGgcauEoNjw9v7ITfp0tGrAL9zJ8a/vjdxJko5WDfi
FOzk+kk+A7WjVgF+4PfcjdbKoM1UDQPh9tf8TwbGho1UDbYnoBKDL5cKCBQswf/58XHvttbjlllv
w888/d/vfu9/qCUlQLBzf9kydX79YLeYOCFy+HKmber+134YqV3N0jtMJ+M1xVv/5XTHEHFAOXeN
SIjbpEzxTd362ASelHj84A6HLkz8piMx9riocUkg/vndsLAya478nws1iR+p7Yk+NBx8f6dgKH7/
+sVpc2D8w6ROp/w7/3mdDdYvYEK8LTey15sqhoeXQH0fo4C14ZfDMfoaQU8FbE7xVwsoCR0Tuc1V
ml0LK8343Lg76dsSGcLPYkfqeCFc1sPA4VQN+A+K0mBOUEI7UWezOVA34XTUktBx6ZQSOGzpbNeA
X7uT4igiMDZ2pGvALd3J8pK766kzVgN/AOC1mq2Tc0JmqAb+rh4buqx2JCeHOVg34hTs5vioC98D
rTNWA3/9n77zD5CjO/P/tybM5B62ytMo5GIkghACDgEMkG2NhI4JN/uHD3J05+7CxzwfnczjrQNg
GTJRBZIHABmNEEIoC0W0WqVdbQ6zaXJ3/4Y9U53dUvamenuqlrxfR49j3eXnVnXVH266q33/b5
GneOf5PR5mU7VgCIjz31eGZlO1YCibxv4ar99lFM2pFE1cCJxHQAEgCVLluCVV17BypUr8Z3vfAf
/+q//avl7/mZHD9Sx4ylFblxCPIBAPJjKdfc0R/mrSQ3EZy4kMn+9PzEahd2BTyu8SsHSsvmsVb/q
0KYJ1RAbnQG7xFS2o9Oi6PJIPPR70yK5ehFQ3KpVZDlxfPTA4A8BNhMHzxtYovmjn76bqN8Qt/rQ
```

```
iNy4amgobCA+8w/wFhINxSceGWyfloGCAbMhy6YM+T3KY4bKuKYL1zSQbTl62odaCSr0/5nP7+fN
3emS3NjN4SJb+8z2ZyOflhpYodnXETvBfsysy+29GsRsXDqBqQBG52V19mL+AsFHVwO0DqBpQlO0
2ONhxeKBJt2pA0cIhXo0/ZlQCnq/hjw3pVg0oItnwWXMUu7lkQ3pVA4puIcbhjcMh7hofpFs1oCj
H7dBljj7BIRvSrRpQtKjKq/PH5JEN6VYNKLqJWBOfNkWxt5NHNqRXNaCIZMPrHAaE060aUJTrduj
mDo9sSLdq4ETKKAC4ZMkSXHHFFSn9u/LKKzN5S428Xi/OPvvs/kj49OnT0dHRgXA4fIrfTF/v14f
xCmGk+S8DzP5TdHaFB+M5DvrIsoyfbe5CSyiZ4ZPtEnDH5IHd4isiu3ptbo1hO0ctugMRCfd/3qX
53nlDvPha2cDhLAiCLvj1zP4g4hI/6clbWqN4nIDpPSdoy34ijc13YyHxUOPtFvvdurDOZHegN3W
KjALCz3G0eZNlGT/d1I22cJINOWmwgfeAsBEbzq/yYs4AM4OBROkC6W/y1Jd9EDliw+bWqC5A84O
puQPKDFZUne/Ggkq+2fC3upC+aiBlNui9bHg62MmyjAc2d6ddNaCIZANvAeHOiIT7Pw9ovpdK1QA
wONiwsSWSdtWAovEFbl2TDN6agbxzNJR21YCihUO8GEUGfTi6LJJkGQ9s6k67akARGfT5rDmKPRw
FfTrCIu7fqN03pFI1ACTYQDLyz/v6IHHk/fZ5cyTtqgFFEwvdOLNcO268Zc2vORJKu2pAkVFAeCV
H+wZJlvEfG7vSrhpQZBQQ3hfghw3tYRE/JtiQStWAkYRAIJA2FR588MGUJqKiBx54IN23PKlWrFi
Bmpoa/O53vzP8eU1NTUavv7bNiX//0oO4nPz/PDpLwgszwxhAVrJGqxpc+PXB5AdX5pGwem4YKcR
MqEiSgf856MYrjdqU0+9WxXD3qNQX0927vNgQSMJpSXkcP6lm/6DfGQPu2uXD/j5tDP3xqWHMyJd
O8FvG6okDizf6EZGSH/6vJ0ZwbjH7tzTbuhz45z1e9InJv73ILWP1nBB8zpP8ooHWtjnxr/uSB32
fQ8ZfvxZCjuskv8SIPmhz4scEG6qzJDyfBhteOObCbw8l2VDulbB6Thgp7H2oSJKBX9W68WqTlg3
LhsZw58jU2XDnLi82qthwZUUM/z6W/Qd2RxS4a7cPNQQbnpgWxvS81NjQHQcuIdjwm4kRLOCADVu
6HLh3jxdBFRtKPBLemBPGAJNB+/VBmxP/pmKD3yHjHU7Y8H6bEz/50gNRzYZsCStnhJHq9ukvx1z
4nYoNFd7EePLAhodr3XidYMNNw2K4fUTqa/r2nV5s7kqy4eqKGH7ECxt2+VAT1C6AJ6eFMS1FNgR
iwGWbtGz43aQwzi5K7XVoaEsgsW8Iqf72Uo+E19Ngw/ttTtyvYkOWU8Y7c0PI5oAN77U68cB+LRv
GZ0t4Lg02PF/vwu8PJ9kwxCvhNQ7YIMrAQwc8WN2s/cBuHhbDbWmw4badXmxRseGayhj+bQz7bGi
PAnfu8qGWYMNT08OYkps6Gy7d6EdUNa/+d1IYZ3HAhk0BB35IsKHsOBsGmAzar/danfjxl0k2ZDs
T+4asFM8mNPReqxMPfOmBqHLxm5gj4pnpkZTZ8Fy9C8tVbKjySXhtdupnE7slysB/HfDgTYIN3xs
Ww/fTYMP3v/BiW3fyw/9mZQz/wgEb2o6z4SDBhqenhzH5JGyorq4+6etmFABkSa+99hpeeOEF/OE
Pf0BxcbHpr/9ybRC3fdIJ0jfyhfOLsHj4iVuSn0hdUQkTVzUhqDK3f35RES4bkfpr2SVRknH3pwH
85YD29qDY68CGK8tQ6k+dqmuOhHD9Bx39X2e5BOz5ZsWAywVTVU1NzSkXxanUFBRxxbtt2BeIa75
/0TAfVl2Q3ty7e12nJtPr/CovXv16SUZ/p9VaeyyMb/+jQ1P6CwC/m19wys5MRopLMqa93ISGYBJ
ovzojP+WbnlRkxnxYVRvE7Z90gky+eOmCYl0jh4EoEEmwQT2ufzm/CJekwRm7FJdk3LWuEy/WarO
cSn0JNhSnGg1Govz5u2uTbMh2CdhzbcWAywVTlRlzoTEoYsnf2rC/S8uGxcN8eCFNNty5rlNzY3t
hlRcvM86GfxwL43oDNvz+zALcMD51NsQkGVNfakKTKuv81/PycUuKGSKpyIz58MKBIO5cp2fDKxc
W44IUrAEUdUYkTFzVqCmbfPGCIlw8jG023LGuEy8RbCjzO7DhijIUpcGG1YdDuEHFhhyXgL3fqhh
wSVCqMmMuNPSJWPJuG2oINlw63IeV56fHhts/6cQLqv3YRUO9WHUh22x4vz6M6z9oB1mFtvysAnx
3XHpsmPJSE5pVbPjt/AJdxoeZMmM+/KWmD3d9GtCx4bWvF2NRVeps6AiLmPhSk6ZsMt09iF2KSzJ
u/6QTLx/UsqHc78CGK8sHbCmk1uuHgrjxw87+r3PdAvZeW3HKZgHpyoy5cKwvsW840K1lwz+N8OG
5Remx4daPO7BKxdyLh/nwYpp7ELv09/owvmPAhkfOLkjJUkhRVJQx5eUmTbXa/55ZgGVp7EEGKjP
mw/M1fbh7XQBkcOb1rxfjvDTY0B4WMYlgQ7p7ELsUO86GVwg2VBxnQzoxglcPBnHzR0k25LkTZwq
W2VDfG8eSd9tQ261dFEtG+vDMeZmtZ+49AAHglVdewYsvvogVK1ZYEvx7dn8fvv+xPvj363n5aQX
/gISB8TdGa3+XZS+bmCTjex93Ggb/XruoOK3gH5B4KFWprmOCcRkv1rKbnlzXG8cl77Tqgn/zyz1
4fEFh2q9Letn841gEB4nNAEv669EQrn2/XXfA/5fpuVg2fuA+HWq5HILuAPDkvj6m29Y/82Ufbvt
Yf8D/7fyCtDfeBV4HruaMDbd8pA/+lfgceO2ikrSCf0DC+LoyK/mI6ovLWHWAXTYcPc4GMvh3VoU
Hfzo3AzYQm9X3j0VwuIddNrxzNITrDNjwbzNy8d1x6bHB7RDwXYOyJpbZ8NS+Pt3FgIBEEDTdjXe
h14GrRvHTFCUqyrjpww5d8K/U58BrXy9JK/gHJNhQ4U+yoTcu4yWG9w1HeuK45K+tuuDf2RUe/NH
EfcN79REcYZgNa46EcN0/9Af8+2fm4jsp+IKq5XYI+I5u39DLNBue3NeLO9YFdGxYflZBWsE/ACj
yOXHlSGLfwHDJY1SUceOHHbrgX5k/wYZ0gn8AcOlwP8pUbOiJyXi5ll3D/8M9iX0DGfxbUOnFY+e
YyYYwjvayy4a3joTwbQM2/HhmLpam4Auqlscp6LjC+pni8b29uIsI/gkA/u+sgrSCfwBQ7HNiCck
GhvcNEVHGjWs7dMG/Mn/iTJFugtA/jfCj1Jf83e6YjFcPMs6Gv+qDf+dWerHi7PTZoMiSAGA8Hse
BAwewfft2bN26VffPTK1atQovv/wyVqxYgdLSUlNfGwD+uKcX/+9T7WJ0CMCjZxdknH1A3lCubYi
gtos9QEdEGTes7dD5m5X7HXj7kpKMatBdDgE3jNcfaFgE9KHuOBa/04aDPdrFeN4QL165sHjAZsV
GmlHiwewSbXkUq142bxwK4TsfdCBKZB7/dHYefjwrNX8KUjeMz9aUrHzZFcenzWyWhD+2uxf3rNe
z4bFzCjPOPiCNe/9xLIJDDAaEw3EZ3/mgQ+dvVpnlwNuLSzA1ze5UwHE2EAe7P3/JJhsOdsdxyTt
tOESwYdEQL16+sDijzKRZpR7MVLFBRiK4xKJeOxjEdw3Y8OCcPNw/M0M2jNOyYU8gjs8YZcOK3b3
458+0Pm9OAfjDgsK0MiDVItnw93o2A8IJNrTjTcLDSGFDup3rAOOA8JN72WRDbVccl/61DYcJNlx
Q5cXLF5ZklH0wu8SN6cVaNjzN6L7h1YNB3LC2AzGCDb+Yk4d/m5EZG5aNy9KUs+3ujOPzFjbZ8Mi
uHvzwM62Xk1MA/rSgMK0MSLXIM8l7dWEmA8KhuIzrP2jX+ZsNOc6GyRmwwePUXyQ/wWhA+EBXDJe
+04YjvVo2fH2oF6suKM6IDXNLPZr9lyQnLqxZ1Mu1QSwzYMN/zs3Dv2TIhhvGZ2vYsLMjhk2tbLL
h/3b24F826Nnw+LmFukuOVEXuG96tD6OOwYBwKC5j6T/asYboil6V5cQ7i0swqTBTNuibgbDIhv2
BGC55pxVHCTZcdJwN2SZkLZoaAJRlGStWrMCFF16I66+/HrfddhvuuOMO3T+z1NzcjN/85jcIh8O
45557sHTpUixduhTNzc2mvP6bh0P4N8LI3SUATywoxNI0pFJTS/2YG6pdjKzaFL6w88CeIdYjEO
znXhncSkmFKS/GBV9d1y2xvtwf1ccnzSxBehgXMLl77ahvk+7GC8e5sML55uzGMmg0fM1fQjF2QL
TxpYIbvqoA+Sf9fAZ+fjnael3I1JUmeXEpURHbRY7wK4+HNKZNbsE4M/nFqXUxfBEmlHiwawS9tl
```

```
w72cB/K3OmA3jTWKDOuizL8BeQLgvJuHyv+nZcMnwRNlvlssENown2RBEmDE2bGiO4JaPO3Vs+NU
Z+bhnauZsqMp2YjGRVcvimnj9UBD/bsSGhUW4NoUuhifSrFIPZnAQ9PnB+k68Wx/RfG94jhN/vaQ
U40xgg1FAmOysS1u9J2CDUvbrz9D0WRAEXabPczVBRMhyFcpa3xTBLR/pq2j+Z14+7jaBDUNzXLi
YZAODlySvHgziJ5u0HX/dDuCphUX4hglsmF3ixrQiLRueYbDB4D2fduI9Aza8c0kpqvMzZ4NRQHg
jY2zoOc6GY0EtG/5phA/PL7KGDc/uDyLKGBs+bYoYVtj9Zn4+7pqSORuG57jwdSLjnsXst5drg/i
PzXo2PHNeEa4ZnTkb5pZ6NJduiYAwe1nzd3/aifePadkwIseJdy4pwVgT2HDD+GyoV9bOjpiusy5
t9cQkLHm3TWOHBQCXH7cESKXB5slkagDw2WefxTPPPIOvf/3r+NnPfgZZlnHnnXfiRz/6EUaPHo1
x48Zh+fLlpr1feXk5Nm7ciNWrV2PlypX9/8rLy015/d/t1Lad9jiAZxcV4SoTFqOimyZob+xWMhb
0aQqKuo5Bo3ITi3FMvjkuyxVZTp33IWubt1cOhlBHROKvGOnHc4uKTFuMV43KQoEn+VqdEVmXWUV
bv9/Za1jSdpuJPn03E2virSMhNAfZanrw2y/0bHhuURGuGGWeFxe5eWMtINzQJ+osAUbnOvHXS0o
wKs8cNgzJ1geEWWQDecC/apQfz5xXlFJHy5Pp6tF+5KvY0B6RsPoI+2xYflaBqR6et0zUronVh0N
oCbHGhl7N114nsPL8Yl0JTibSBX32sxX0qe+N6ywBxuQlbvBH5prDBqOAMGsHu5dqQ7oD/tWj/Hj
aTDaM8iNPxYa2sIQ3mds39BiWtH3PRA9PMsPljcMhtJH1hJT1G2Lf4HMCKxcV43KT2CAIgo6RzzL
GhqO9cbxElN2NzXPhr5eUmsYGo4Awa2xYdSCoO+B/Y7QfTy0sSqnb7cl0zWg/8tzJ12oNS3iLsX3
D/37RY1hhR54DMhHJhtcPhdDOGBvIM4XPCbxwfrFpfQEEQdCNwzP7+5gKCB/uievKfqvzE2wYYRI
bhue4cBHBhif29Z7gv6ajF2qCaCTY8M0xfvzZRDYAJgcA33zzTZx33nm4//77MW/ePADAhAkTcMU
VV+Dpp5+GKIrYtm2bmW9pmfpiEr5o10aFX7ig2HQj/itH+lHoTX6ggaiM1w6xE5X/vCWqgfOw47d
0w01uv0geaNYcCaGJoaDPBiLraMlIH544txBuE9so+V2CLrP0SYbAJMuybhyWn5Weqf/JtKDSg2p
VcDkuJ3w4WVFPTMLODi0bVl1QnLYf6InEekCYvFFXbvCHWcyGNw+zFRD+rFl7W3nlSD8eX2AuG7J
cDnx7LLveb7Is60ruHjk7PVP/k2lBpRdj8pK+cTEpkQ3JirqjEnYZsIHccGYqw4Aww2wYkZPICh5
qMRtWHw6hlaGA8AaCDVeP8uNPJrMh2+3AdUT2GEuZsZIs6zIzV5yTeUkbqYVDvBiVm2RDVAKe388
OGwIRCXs6kyV3AoBVF5SY3qTDKCDMUtDnc2IPOfJ4QkFVtrmtWUk2sBYQJtfEN0b78YdzCuEykQ0
5bgeuHav3wGNFkizjc6Ic97FzzKmwU2tRlRcjCTaQiS001RmRsFflK+8QgJcuLDG9Scc1o/3IJQL
Ca1hiA7EmRuc68fbiEgyxgQ0sBYRJNnzTAjYAJgcAm5ubMXfuXACA05n4wKLRxP8Rj8eDxYsX4+2
33zbzLS3TtvaYJiV5TJ4T56dpwHky+VyCrrsRSwe7zQScLx/hR6UFPdTPrvBgPMNBH3Icbp6QY/p
iBPSlfptbY9jexkbpwuEeEe2R5K1EjkvQBSXMkCAIunF4Zn8QcbLTBiVta4tpMp2q811pm/OeTEY
B4T8zFBAmfVSWjPSjwgI2LKj06gLCzzG0eSPLB26emA2nFWwgNi2ft0R1gWhaOtgtohzi0xkAACA
ASURBVEPFhjy3YEopPCmHIOiy5v+8rw8iM2zQXpiNz3dh4RDz2WAUEGbpYEey4YqRfpRbwIZzh+g
DwiyxgRyHmydYwwbyQPNZcxS7GWHDga44uqLJVZHnEXDtGPO7VifYoPeMZYUNW4l93MQCF84d4jX
9fYwCwiyz4cqRfpSl2UTwZDqP8YDwphY6bFjfHMWeTjbYsL8rjm4VGwo8gq45phlyGJwp/vxlHyR
GvN+2tOrZsKDSfDbkuB341li9Bx4r2kysiStHWcOG86u8GJGTfN2IyFZAmGTkLROz4cjAB/NEMjU
AmJub2x/wy87Ohtvt1vjxeTweBAKBE/06UyLhPLc0/UYXpxIJpi1t7AR97BoHwWDz9vSXfUwEfTo
jkqazp0OAzp/NLI3Jd+E8YlPIym3+RgJKs0o9lmxYAOC6sVnwq1Kd6/tEvEt4zdESTTZsao1hRzu
bbJhjIRtuHK9nAwsHu46wqOne5xSAmcXWsKE6341ziU0hKwFhIzZYsWEBgG+PzYK6eWx9n4i/H2O
DDWTm29wyC9lgEBAmsw9pidzAzrFoHBwGbHiKETa0hkRNUyCXAMywaN8wrsCNcyq0Y8zqvmFOiXV
sWDo2C14VG472ivgH4SdFSyQbrFoTgJ4NLAWEdWywaN9gFPR5ipGgT0tI1DT+cAnIqJniyTShwI2
zCDaw0kRMtyYs3Dcsrday4XCPiA9YYUMrvTPF+uYo9jISENY9K6xkwwQ22dAUFDV2Y24HMK3IonE
w88XGjBmD/fv3J17Y4cCkSZPw6quvorm5GU1NTXj99dcxcuRIM9/SMpEPqa+VmR+NVzQqz4Xzq7S
vz4Kpd1SUsb3dvgPNt8ZmIUvlp9cQlPD3evoHO/J2ZlKhO6PuXKcSCaZXDobQS7bHoiDyduZrFj6
kCrwOXE3cBLJiZq1ng3XjMCbfhYVEQJgF496IKGNHh33j8G2DgDBpFExDZPbf5EK3KQ2BTiSSDS/
VhtDHAhts3MAWeh24apT2FvspBtYEoB8HK9dEdb5blyXAwr4hIso6+xQr58PS6mxNQLiuV8QHDSy
wQTsXphS5TWkIdCKRnlmrDgQRjDPABhuD4kU+J64k/PSeZmTfYCcjxxe4cTYR9GFh/xSKy9hJssH
C+UAGfY70iviQATaQl6fTit0ZN/04mW4mgj4v1gaZ8JPWrQkL50Kxz6nz4WXheQnYy8iJhW6cWc4
eG4JxvX2K1WzwqB7Hh3pEfNzIABuINTG92G1anwFSpu5GLr74Yhw6dAiRSGIQ77jjDhw9ehRLliz
BFVdcgbq6Otx+++1mvqUlkmXZILvFmptbRWRU/u/1EeqtqXd1xKAuix+S5TDdp0OtfI8D1xBBn7/
X01+Q+tsZa+fC4mE+DMlKLs1gXManDHRF1t3OlFk7DmTpwseNEeqdT43ZYN1DCtCz4b36MHU27Oy
IIaJiw9BspyXWAIoKvA5cpWMD/csBHRss3LAAic7ClSo29MVlrGegK7Iu883iNaFjQ0OEutG9LMv
YRASErWYDOQ7vMbAmdrRHEVXFnYblOC2xBlBU6HXgSiIgzMI4kBt5q9fEpSN8KPcn2dAb13v20pC
d2S2APhD6YUOEutG9JMv6+WDxs4JkAwvPy+3tUU2X+BE5TktK/BQV+Zy4ggj6sMgGq58Tl43wo0z
Fhp6YjM9b6J+t7KymAfSB0A8bIohRzhaXZBmb2+jun1hgw7Y2re3aqFwnSnzWsaGEVTbYeL40NQB
42WWX4emnn4bXm7iVnjFjBl588UX84Ac/wL333ou//OUvmD9/vplvaYmO9IpoDSd3sNkuAZMKrQ1
2XDDUp7nFPhYUcbCbriml3YdbAPgnouMRExF5mx9SLoegayhBexz6Yga3MxaPw4xiN4aqAs5hUT8
n7ZaRD+LEAnON7UldONSnucWu7xNxuIcyG2xeEwDwTyO0Xmq01wRgPxvcDkHX3ZD2OPTGJOzuJAN
f1j4vZ5W4NZckIVF/wLZbtd1xnQ/iBMvZ4NXcYh/tFXG4J37iX7BBJBuszBRXRLLhExbZYPH+iUU
29MQk7O3Uzkergx1zSt2o8GsvULdQttQx8kEcl28tG74+1Ac3keFytJcuG3RVJKfrmcLGTHEA8Dg
FXDSULTZ0RSXsC2ib4sy2mA1fK/NoAqG9cRlbKe8bjHwQx1rMhouG+aBOKqvtFlFPmw0U4g1kh+W
PG+lfmOnYwEsA0EhVVVX41re+hW9+85sYPny41W9nisiN28wStyUNH9TyOgXMK9eW83xEGdDkgrR
64wYA88s9GjAd6I7jWB+9YIcky7oSYDvARJZ20Z4L24mmOKNznSi28HYGSHi/nUOMw8eUyzfs9EF
U5HcJuocA7fmgY4MNa+LMci9UVcDY3xVHA0U2iJKsM3a3IxCqYwPlNUE2xRmb50IRBTbQHgcy+89
KH0RFWS6H7nlE+2BHlsXbsW84s9wLNYb3BeJootgpPC7J2Npm74UZwN6+YWurtinOuHwXCrzWHjs
```

```
EQWCOkXb6ICrKdjt0a482G+zy+FLr7AotG/Z0xql2Co9LMra12c9I1tYEyYbxBS7keyiwgfaasNE
HUVEOi2ygkFRwTqUX6pHe1RGj2ik8JsnYTrLBwrOVZastGAz2e/+R/1iX3bczikiD948a6aaj0rj
JNwLTRw30xuHLQBzdMeJ2Js/a2xkAOKfCwxSY7M5mUMTamrDTB1Gtc4lOorQ3bzTYkOdx6Jrv0Ny
07AvE0aNiQ5HXgdF51ga+AOgCXzs7YuigyQYKFySAng20N7B2Z4Mq0jGSdiCUQpZPgdeha75Dcz7
sDcTRp6p1LPE5MDLXejaQh9vtbTEEIvR8AMmguF1sWDCErUO+nR5faukYSZENRvYpdrFhOkNs2N0
ZQ1DFhjK/Q9OR1CqRbNjWTpsNdJ6XrAUAaWS+AWwxUpZlWz1SFRV6HZhGsIFm9cDujhhCqkybCr8
Dwyy0XTM1ABiJRPDoo4/ioosuwqJFi/q9/8h/rMtufwZF5MP6k8Yota40zUERR4lONFZ1qSJFgon
mw9roISVYfDsDJLxLphZpwbSOYnoyKw/rrW0xdEfpbVrs9kFUpGNDU4QaGxqDIupVmXceB3QPUat
0biU7ZSz6DYvbFjaU+JyYomKDDOATih6htAJfJBu2tEbRQ7EhCi1GGgVCaXmEHusTcUyVeed1Qvc
cs0rnsrRvMMjqsIMNpX4nJhUmLyhlAOuaaI6D9r1psWFza5RqsyS7fRAVGa0JWmyo7xPRFEp+Bj5
nommWHWLpsogWG8qznBq7GkkG1jezMw60guKbWqJUmyWxcnH4CUU2HO0V0axig98pYLJd+waG2GC
UDWolG0wNAP7qV7/Cc889hxkzZuCee+7BT37yE8N/LMuwS5VNC3J6sRt5nuSH3RHR+67ZJfIwM63
Iuk40pMjNG81NC62HFGB0U0Un+02mYGCtaEi2E9UqPwyR4qaFhg+iopklbuS6k+uvLSxhTycdzw5
yTUwvdsPrtIcNupJwimzQe6Ra1yme1IJKNso3aLJhaI4LY1QZl3EZ+IxSIJSGD6KiWaUe5Kieza1
hCXsDdNhABsVnFHvgsYkNRpkd1PYNlKpIAHYyXIya4tj1vBye48IoVcZlTAI2tNBhQ3fUfh9ERbN
LPMhSsaEpJGF/FxtsmFlClw20ZKfHFylWbDMM9w02jcOIXJcm4zIqAZ9TapZEwwdR0ZxSD/yq9dc
QlHCgmxU2uOG22FpJEUul8eQ4WL1vMDUAuHbtWixZsgT//d//jeuuuw6XXXaZ4T+WRXapGpnrRKm
FXarUcjoEnF3BxoOKhv+forkMg8muhxSgv72lBaajvSJaVLczWS7BtptbgJ0SNxo+iIpcDgFnfsU
GnFHm0TRLqu+j1yxJV9ZlU8AH0GdC0loTR3pFtBENs6xuiqOWbhworYmtFHwQFbkdAs6sIG0z6Iy
D3R3S1TqjzKtpllTXS69ZEs1xYKXs82C3qGmKk2tDUxy1WNk3bGsjvM5s8EFU5HEKOLOcDTYYZbf
YpXnlHk2zpMM9Io5QapakY4ONlwOsZDsd6I4joG6K4xYw3k42MFL+SvogTrDBB1GR1ylgPqNssPO
cPb/co2mWdLBHRB2lhih2e6SaOtMEQcD48ePNfEnbpfOosHEiAsZpuTSk8/iy8SFlBCYaDyqj25l
ZNoNJnXRJC0zkTZ0dTXHUMsoIpSGa2aAAO5s3mtktPpeAM8roj0MgIuHLLnpsOLOCjWZJ5HNils1
sYKXsk5aPjyJmGEmRDUbNkmiMQ0dY1FxYOgTovEut1FkV2mZJX3bF0UihIYquYVaJ9Q2z1GLlkK8
LfFHeN9AaB1oZX4BxsyQa49AWFnFQdSnhFKDzLrVSZxENUfYG4mimwQZiTcy2ofGFWqysCRpNcdR
ihZG0qkgA42ZJNMahNaS9sHQJwAyL9w2mBgAXLFiAjRs3mvmStouW/58iciO/vimKmGRvGUuMUpc
qtVhIy91C8XYGSDREYaHDI83bGUDfqWk3pS5uNDewgH7Tsr4pgrjNbIiKMra1fbVpIQM+EwtdyHX
bx4Zct0NXKkKDDbSD4ucQmW87O2Jop9AQhTYjySZBn1Jiw/Z22mygnxFKdkGeVOhGjo1sMGqWROM
imWb1BKAvd/yiPYZOCk0PdGVddu+liefluqYIRJvZEBFlfEFaK1EOhLKwJiYXupFtIxuMmiV9QsE
jlGYVCaBnww5KDVFoNQdSZLQm7PYWD8UN2ED5bEVlL02siSlFbmS5rGWDqa++bNkyHDt2DL/85S+
xa9cutLW1oaOjQ/ePVdHqUqXWhAIXyvzJj6U3LmMrMTGsFtmJptzvwHAbulSpRR7yaTQ9oHkrocj
I88xu0d7IG3VqstvYnKbXmaKJhS6U+JJs6InpA/VWa3dnDOr4SmWWA0Mt7FJlJPJygMamhaaPjyI
W2EA7KG7YLMlmH0DDDnY2s2FyoQvFqrLC7piMHe32smFnRwwRFRuqspyosp0N+qC43T6AZFYHDTa
w4HmmC4rbvCZKfE5MJhqi2B30MfJBtDvYMbXIjUJv8gq1KyrjC5u9xXe0R6Hu3TY024nKLLr7Bhp
soH2+BNhIrqBZYQYAZX4nJhENUew+U0gUfRAVTS1yI1/VdyAQ1QfjrNYOwnZteI4T5bTZ0ECfDXY
8L00NAH7zm9/E/v378eabb+KWW27BZZddhksuuUT3j1WRXars7ESjSBAE6ps3oyxIO7pUqTWNAFN
nRMZOmzctNH18FNHu8Gh4O0Nh00Lbz4e2DyIAOBhgg91dqow0o9iNPFVDlPaIhN02N0Sh6eOjyMj
ry042BOP6pjgsMNJuNhzuoeuDCCTYoDN4p80GmzqkqzWrxEO9WZI+u8X+cVhg4BFqJxv6KDbFUYu
2RQBtH0TgOBtI/2CbGUmrGYxas4lmSc0hrcWPHaIdDAboV1D0xPRNqmiMA5kZa/c41FL2QQQSfQd
INtjNSNpBUCAx/2g3S6IxDqYGAG+++WbccsstuOWWW3DzzTef8B+rIg91M2zsRKMW7UM+TR8fRUY
NUew0s5YMsjpojAPZEKUxKKHGRjCRtzMjcpwos6kpjlq0Ny20fRAV6YMd9naGpl3OBCQaopylO9D
YNw6SLGNzG/1x+BrREOVYUEStjc2StrfFNGwYZWPDLLX0bLB3TZBssNsHURHtQCjtTHGAfrMkUZK
xhYF9wxllHk1DlPo+exuikE1xxuTZ1zBLLbJJkN2HW9o+iIpoB0JpW0UAxs2S7BwHUdJXc9Fhg1f
TEOVor4jDNjZE2dqqZUN1vguFNjXFUYt2SThtH0RFX7HBuFmSneMQl2RsJaq57GCDqeHm73//+2a
+nO1iIRIN6MG0qSWKYFyyvB5c/X5q0RyHt48mD3MfN0Zw99RcW977QBdxO+MRMC7f3tsZIAGm+eU
efKA6zH3cGMG4Antu01koWQCAeWWJTk2x45fph3pEHO2NY3iOPZ8JK2uCvBzY2BpFKC7D77Jn40C
7rEvRgkov/lqXZMMnjRHcNcUeNuzviqNbxYYCj4CxFNjgdQqYV+7Fhxo2RDE23yY2MGCRACSbJSn
ByNpuEfW9cQylxQaKa0Ktz1siCMdl+OxiAwOHWyAxDu/WafcNd0zOseW9v+yKoyeWZEOhV8CYPPv
ZoDRLUh9iPmqMYJRNfwsLwWAgwQanACiONvu74mjoEzHEptJ02tYAikg2fNYcRUSU4XXawwYWLtO
BhG3Ge/XaNXHrJHvYsDcQR6/qxqzY68CoXPuD4n6XgK+VeTRWGR83RjAy9/Riw5nHG6Iowch9gTi
agiIqbCo/ZZUN65ujiIoyPLTYQPFs9f4xFRsaIvjeRHvYsKczhqCKDaU+B0bYYLtmWURJlmV0dna
is7PT9lrqdMXKgWZErkvz4Ucl4PNme3yN2sIiDtncieZEIlO0FTDZIV0ZdAmd2xmAbvYb7aY4iow
6Ndl5Q8PK5cDIXCeGqdgQEYGNLfaMQ0tIxJFeLRumF1O6HCDWxKc2NksysgagxgZdtrh92W+sBMV
zKLOBDHzRGofReU6NH2dY1P9tVqkpKKJOxQa3A5hWxMYFqp3NkozWhN0WCYpoZoSyclGU53Fgdgl
FNjDCyLF5LgzJSh73QqLef8wqNfSJqFd1qPc4oPNttUvkmlhHkQ1zyk5TNjDyvMw3aJZ0OrJhXL4
LFaq+A8G4vvrNKtX3xtEQTFok+JyJ5hc0pPMWt7FZEi3bNdMDgHV1dbj//vuxaNEiLF68GIsXL8a
iRYvw4x//GHV1dWa/HRoaGnD77bfj6quvxrXXXpt2F2LDLlWUFiRAL+hDPqTs6ERzIo0nwNQXl7G
lzR4wsZLVARinqtsBJqPGF7RubgFjzzM7xIoPIpDwCNUHfeiwYVqx27bMQ1ITC1wo9dFplsTKRRF
gxIaoLQ1RDJviUHxe0vLzYcUHETD2D7aLkeRcmF7sti3zkNQkollSt43NklhaE7QaqRk2xWGIDXY
d8o18EOdS8EEETsAGu/YNxFyYUeyxLfOQ1JQiN4rUzZJsbHrAQuMwRUZln3Yk6siyTL3zrVq0ur8
```

```
a+SDSYqQgCNQYuZnwxJxR7LEt85DUtGI3ClR9B7qi9vUdoFVpZ2pkp7a2FjfccAM+/vhjnHHGGbj
xxhtx44034owzzsBHH32EG2+8EbW1tWa+JR566CEsXLgQr776Kn7605/iJz/5CcLh1LMgyC5Vw3K
ctqUCG4kWmFjawNLctLBySwUYd2qyA0z1fSIaidsZuxtfqEWrixsrPoiKWNnI0wp0AJQPNIzc3AK
JQEueig0dEX1Aygod7RXRTLlhllpGnaHtYMO2thhEBnwQFbGyJmiywajpATU2UDzcks2S7GqIcqR
XRCvRFGfSabhvIH0Qx+a5UETBB1ERtcsBBpoDKUo0S9KuSdsuUBnaP80s0TZEaQ3rA1JW6FCPiHZ
VU5wcCg2z1DLy27eFDYQP4rh8Fwoo+CAqotV3YGOr9n2o7xtoJVdQYoOpM+7RRx+Fz+fDiy++iIc
ffhi33norbr31Vjz88MN48cUX4fF4sGLFCtPeLxAIYPv27ViyZAkAYNKkSRgxYgTWr1+f8muxko6
riJyI29tjCKjAaZVY2sAC+nGwI1W9JyZhbyf9LlWKaHVqIm/xZ5bQu50B6HVqYm1NkA/rrW0xdEV
tYAND2aAAnSzprqi2c6CAhHkzLRk1S7JjHPRsoNMwSxHZLKkhKOGADQ1RWMp0AvRrYktbFN002MD
YONixJgIRCV92adkwq4TeONBqiEI+L2k1zFL0tVJts6T6PhEHu61viKI/1NELfAHAuUO0DVE2t0b
RG7OeDXqPL+8J/kt7RDaGsWNNdEa0zfscAjCL4nxwOwScRTREseNspWuYVUqnKY6ir5V5Nc2S6nr
taZbEUjAY0CccbW6Nos8ONrSwUVmliEZpfEdYRK3qeeQQoCtNt0qmBgC3b9+Oa665BsOGDdP9bOj
Qobj66quxbds2096vqakJBQUF8PmSQB8yZAgaGxtTfi0yFZX2Rr7M78Qk1c2IJCf8KqyUUSca2uN
AbuQ32QCmra1RqO+AaN/OAHQONGRQnPZDyqhTE41NC+01UZHlxASCDZ/awAayjI72fNA1RDneLMl
KkWwYX+BCvocyGyhkdrB2YeY93ixJLTvYwIrXmaLKLKemWZUoA+ubrR2HmCRjO8kGxjbyn7dEEIp
bm9lB2pNMLHAhjzk2WO8RylL1BJBoiDKvnH4g9GtldANfVdlOjFU1gYnLwPqm6El+I3NFRRnb2hk
LhBJrYkNzolmSlSKDoBMLXMh102XDgiH2B0L11RN054L/eLMktWwZB8Yu04fluDBa1ZAmJiUaBVm
piChjeztbzwrynK00S7JSm4jY0+RCN7JtYoOpubeiKMLrPfFDzufzQRStj66fSDU1NSf82foGH9T
x0IpIM2pqUg8kmqlpWW7sCSQB+dbeZoyPWlfetb9XQF/c3/91oVtGrOkQapote8sBaajPh/pw4rO
JScBr2w5hXmH6B/2TzQMA+FudC0ASROO94VP+jtUaERUAJD+bTxvD2PNlDazkxLo6L4DkQ2FovB0
1NS3WveEANMntwvuqz+adA+04z53ZOj3ZZyvLwGeNWjaUhxlgg9+NfSo2vLm3GdUR69jwZa+AoIo
NRW4Z0cZDqGmy7C0HpCFeHxoiic8mepwNZxRYyIaj7LNhXWMYe7+sgZXWrTo2iAywwePCB2o21LT
jXJfVbPAjkeuVUHm4CTU1dJueTctyY3+Xig17mjEmbB0b9vYKCInJ+VfikRBuOIQaeokdkGWgwut
D03E2RETg9W0HMddCNrxLsGEcA2wYGdOy4RMKbKhiZN/woYYNbTjH2ZDRa56KDRuatGwoCzVSZ8P
0LDcOdCfZsHpvE0ZZyIbdPQ5ExGSgqcwjIdR4CDRXhSwDZR4fWqKJRRAWgTe2HcRsK9lwxA0gOe7
jvSH6bCD2DZ80hLB3fw2stG5dV6fdS1fF26izYbLHhY9VbHh7fxvOcljLhs8bGWRDthsHe1Rs2NO
EESHr2LCrx4GolGRDuVdCX8NB6mwo9fjQepwNIVHGG9sOYla+fWwY5wmaxobq6uqT/tzUAOCECRP
wxhtv4PLLL0dubq7mZz09PXjjjTcwceJE096voqICnZ2diEQi/YHHhoYGnHvuuYb//YkG41ifiJZ
o8iTrdQKXTh9NtdwRAC73hvBiQ0f/1ztCflRXj7Ts/T7Z1wcg0P/1GRV+jBs31LL3G6jOb+nEM/u
D/V8fEErwner8tF6rpqbmlIvi4OE2AMlboPPHlqC6Ojut9zNLY2UZFXub0HTcdyssCQjkD9eV+Ji
liCjjy/Xah+Dl00egkqIvJgBcVRjF8sOt/V9v63Fj9JjhaZcRnGo+1PfG0RpNRsB9TuASFtjgCeG
lxiQbvghay4aP9vYC6Or/el4lI2xo7sRzNUk21AoluN5KNhxikw3l+5r6PflCkoCuguG6rBezFI7
L2E+yYdoIlFNmw9WFUTyiZkOvG2PGDk+7Q/Op5sORnjg6Ykk2+J0CFk8fQ7UUGgAud4fwSqN9+4a
1ewg2VGRh3Dh9FYjdOr+lEyvVbHCU4NsWsqGWUTaU7m3q9+QLigJ6Codblo0Wisuo+VTLhiXTR1D
1xQSAqwuiWHEkyYatPR5L2XC4J45OFRuyXQIunj6Gaik0kGDDq02qfUMoC9XVZZa93z9INlRmobq
aATY0d+KFA6ozhbMU36rOS+u1BsSGgyQbSqmzYYwso3hvU78nX58ooK9whGWVHX0xCTWfai/kLp8
+EiUUfTEB4Kr8KB47ot43WMuGg91xBOJJNuS4BFw0bQzVUmgAuNwVxOtNnf1ffxG2lg1/361lw/z
KbCbYsKipA6tqQ/1f1zpLca2VbKjVsuGC6jJUj81K6/1Slan3gLfeeiuOHTuGa665BsuXL8cbb7y
BN954A8uXL8c3vvENNDQ04Pvf/75p71dQUICZM2fijjTfeAADs3bsXhw8fxvz581N6nS1kB7siul5
nis6q8ELNhH2BOJqD1mVQ6rw6KKclK9KVsVhc7shaOThgf6emnR0xTVOcodlO6sE/INGpqdBrX6e
mLURpG80uVWqdTbBhTyCO1pCNbGBgTQDG3eyskizL2NzGVskCYH9DlC86olC7MAzPcVIP/gHANKJ
ZUmfEYjaQ3S0p+yAqOqfSC/VfsbszjvawhWxo42TfYDEbyPnAwjgYs8G60i6yYdZIyk1xFM0waJa
028KGKEYeqbSDfwBwToVHw4adHTF0WMgGck3QtkhQRLLhE6vZwCAjHTbvG3a0axtmjc51Ug/+AYm
1STZLIj3gzRTJBto+iIpIv/0vLO478BUbEmzYSrLBxjOFqQHAWbNmYfny5SgrK8PKlSvx0EMP4aG
HHsLKlStRVlaG5cuXY9asWWa+JX70ox/hgw8/exn+PnPfw6/f6/4bd2uz/33/j/f/OHaqGdDOyBtBRy7Ubkmyh2ZTYckjMrchoTQiTdj/zkp5bVQz6E5qrDrr+RhJP+hoLnGuDzFCHqFkr6oczPBcrtIzFVbQ0G/lrZAqGjpRBwn5jOEpxPmLjb7FP1aQbeyB7n0rTXq6BdBfIRL1yKo1CtvAKZZcvVmOwJCOYD/FaHZKK2ykpojOVUXjxo2MEoGUqpUsUPLY4t+2zutCl5m9HWqYRP1xWuGfxKrUjwkBkySTu4jngn/ahgt60dWr+lz2bOUSA+Imzx1I6Ih0k7vp7TUFeBxjchmZi6cyupT4BDqAndhk1ZLR60Z6KBb8lqzTF6N3OSX+NtwNmlvtfNT4N69tR6FP1Bs+HugmBEuGmxFwhgoPk+s69ex0Lhk2CKSRsWNqHKuEJaQcRhftlN92h+YQtiKQ/cHF7+d/A9
```

```
tu/aRtOMNpo74hx9+iBUrVmi+9/zzz+Pcc8/FwoULcd999zEZCCRv61i5wQagM2ve1m5NOmodEYm
uynYy4XWmiGzVbtWCPMrwXBAEwbZgB3kzwdI4TC/2wKMiV12viEaLvDFZfliTG9htFqWqHyVubod
mO5nwOlNkVyDU6LaOFTlsZQObGV9AotRO3Rn9SK9omW8uebBjmQ1bW2OISdazYViOkwmvM0V2BUJ
1a4Khi0OHIOgOuNbtn9h9Xs4scWs6nB7qES3zzSXPFCxdDpBZX1vaYohbwAYyCDos28lMNQ1g3+U
Ay8Fgp0PQNf2wjJEMj8OsEjfUx93abhFtFnljshxv0LGhNQrREjYQF2YMVdMAdrKB7lwwNQD47LP
Poq2trf/rvXv34pFHHsHkyZNxxRVXYP369Xj22WfNfMuMFRVlHOsjo7DsLMiqbKcuHfULC9JRWc5
uAfQReatS1Q+T6foMbeQBYK5NKdrkw5qlcfC5BJ1fhBXjECHYICDxoGJFicYsSYSHRWCXJWwgb27
ZmQuAjWzgbBzs2rSwNA5+l6D3xrSgnMcoY54lNgzPcaLcn2RDSJSx2xY2sDMGgH4jv6nVJjYwdKg
D9BkuVpWEszwOWS4HptqwbyAz5p0CMJQhRo7IcaLUl2RDMC5jd6cNbGBoLgD6y4HTlg02lYQfZvg
CNdvtwBTSG9OCcSAz5l0CW5V2o3KdKPYm2dAbl7HHgioz1uMNun2DZWtiEGUAHj16FOPHj+//+t1
330V+fj5+//vf40c/+hGuuuoq/P3vfzfzLTNWXa8IdYC7MssBv4udSDRgT4aLLruFsY38pAIXclS
fS3tE0jUmMEO6mwmGHlKAPTcTRkFxlg63gD4z1opxONobh3o7OCTLCR9DbBAEwR426B7WbM2FyYV
uZKs+l9awpNtgmCHWx4G8vbUqKE4GvoYxxwbrx4F8XrKWMU+PDWxt5CcXuZGlYkNzSNJl65khnc8
XQwEfwB42hOIymkLs2sgA9thFkM8e1jLmjdhgDSPZXhNTi93wq5jdGJRQ1/cVG6wKircwbCMD2LN
vINnAWsa8MRvML4dmPd4wrcgNr2p6HguKqO81PxCqu0znOQMwEonA5/P1f71hwwbMmzcPHk9iQo0
bNw7Nzc1mvmXGYjkdV5EdNzQsp2cDiVT12eTmzYJbbH15G1vjMKvEDfXz4kB3HO0mp6rX94mawFd
lloOpwBdgzw3NISLAPJIhb1BFdmR2sFwODiS6uM0qsT7ri/VxmFXq0bBhf1ccnRFzu6XXGQTFvQw
FvgB72KD3/2OPDUYZLmaL9TXhdgiYWWJ9ZgftUp5TaXapB+pVui8QR8ACNqjFWlAcMDjkW7AmdGx
gyORfkR2MZH1NuB0CZnzFBswu0c6FvYE4uqLmsoF1GxnA4HLAAjawbA2gyI5AKMtWEQDgcQqYWWz
H/onuOJgaACwvL8eePXsAJLIBDx06hHnz5vX/PBAIwOv1nujf0JgfkWc9TR0wApO
54yDJMtNGtQCQ43ZgSiGxaTEZTKyXdQH6ubC9PYqIyf53fLDB+kxI2ka1A5HV4yBKsm4Ty9o45Lo
dmESyweRxIOfCcMY2boA9vrm6ywEG2WBHBiAfbLD2QCNKMuoMOn2ypDyPAxMLtXN0s+n7BrbHALD
HN5fsDs6SH6Qie7KD2W0WpYhkg9njEJdk1OuqadgahwKvAxMLkmyQkfB+M1Os26cA9vjm8nCmsCc
Tkr94g9lsiEn0K+1MDQAuXrwYq1evxg9/+EPcc889yMvLwznnnNP/8z179P48PzNQnDTImHXqYg
kTy1yw6f6sxqCkunpqKynqQPWg6kpKEF98ZXvEVDgZaczkSKrb2/JbFAWD/mVWU5NeVFUAna0mzs
OPNzWTSt2axqi1PeJuodKpuJhI2+1/11jUERMxYZCr4A8D4NssPgWW1+6wd5csMM3V8cGBrN8phM
NUaxolkQ2zWJxI6/r8GjymjgWFBFXnROLvQ7kuL9iA4sXh3b45h7uZp8NM4o9moYoVjRL0p8p2Bs
Hqyso6vtEqOPLpT4Hshlkg9X7J9btUwB7fHPJSxIW4w12NEviId5gdeVlPWE/V+63337OVBItW7Y
My5YtQ0tLCyoqKvCrX/0KOTk5AICuri5s374dCxYsMPMtM5a+lIc9MHmcAmaWWAdo2SDzjcWbCfJ
hvaczjm4TU9VZL2dSZPXNBC/jYPXtLQ8Pa68BG8x8UMmyPvONxfkwt1Sb+ba7M4aemJlsYH8MAOs
vSfRWEXyMg+mM5KAE2OcSMMPCpgeyLFMvYRmIyLmwqyOGPjPZwHg5kyL72cDeONjhjclDlo/fqJG
aicGvxJmC/flAzoWd7TEE41+xwXQ2cHCmoMEGFgOhVjdLkmQZRzmYD+SF2Y72GEJx8zJCWVgTpgY
AnU4nbrvtNjz33HN47LHHMHPmzP6f5efn429/+tuuuOEGM98yY/HgAQhYe4sdiMrojiUnts8JzU0
IKyrwOjA+X5uqvrXNvHHgoZwJMEhVb4shbmKqOi/jYPUNDQ8becCIDeaVxndEJPSo2JDlEjQdBFl
Rkc+JahUbJDlRwmGWeCiLB4zKWKIms4H9DSxgPRt0ZX6n4Ti0hSX0qTbE2S5B00GQFRX7nBij8m8
V5cQz0yzxsiZINmxpjUI8HdlgcdaXng2n3/6pJSQhpEp9y3ULKGSQDaV+p+byJi4D20xkgy4rltE
1QQY7NrdGIZnYEZmXM4XVvrmHOMgOBqxlZHNIgtq2Po/RSrvyLKdmnsblhNWUWWJhTbA36jbK6JZ
qFING/4C1NzRGvneCwJZBqyIrb2h4yIIEEqAoUwVog3HZ1DIWFm4mBiIjbyfZpE2LLMtcZAACVrN
B/5DihQ1meoTykOkEJOZoiSpA2xuXsafTTDbwMQ5W+uZKsqxjJIuVA4C13pi6Mp5cltlg3ziwuib
G5LlQpDpo9cRk7A2YZyWjGwdGD/lWromEVyy/+yezZBT4YpcN9p2tRjJ6cTg234VCb/Lz6Y7J2Gc
mGzg5W1l5vjTykWb1WWFldRUvawKwmJEMXJiZ/o6RSARr167Fvn370Nvba3gg/4//+A3z3zYttRI
32DmM3mADejB9cTwd1YyacaONPKuaW+bBczXB/q/NvLXkZRwEQcDcUg/ePhru/96m1ihmEKWg6co
o6MOiphS54XcK/TfNTSEJdX2iKUaqzcQNdh6jN9iAng072mMIx2VTOjeTG/nhjG7cgMTt7Uo1G0y
8tWS9OZAihQ1/rdOyYVqxWWzgYxwU31zlplnxzR1qwt/bGJQQUSGSVa9YQH+TrzRLMqNzMy9zAUi
w4YUDSTaYWUGh80FkdBwEQcDcMg/eVbOhJYopRe6T/NbAZGQjw2q2k+Kbq7jHKL65VdmZ73MagqL
GR7rI60A+g16xgJ4NSrMkMzo387KHBBL7p1W1of6vrbwcYNFPGwAcx/cN79UnL8o2tUR1TcXSkZG
NDKtnK8U3V3GIUHxzK7My/3vr+7ResSU+B3IZ9IME9NnB29qiiEmyKZ2beTlnA4lxeOmgPWygMQ6
mzr7m5mZcd911+OlPf4o1a9ZgzZZo1+OSTT/DOO+9gzZo1WLduHbZs2WLmW2YkMh13ZB67t1Rlfqf
mtiCRqm7OZOSlvA0wiMibmKrO9TiYBKbuqISOSHIH63bAlAIfFXI7BMwsscargwsgagFU2VGQ5NZ3
lYpJ5qeospKkPVEY3+WaxQW9ize44WHV7G4hICEST4+l1AhVZbG5grfTN5cE3WNGQbOuaJfHQ9VW
RUeMss7LFeRoHPRvMyYzlxUYGsNY3l5wLrGb4AMDQHJdlzZL05eDsjoNRdrB5bODnTEGOg1n7Bl5
sZABrfXP1lkLsrolhBs2SdrZbxAam14R1VWYsjIOpq/D//u//0NXVhSeeeAKvvPIKZFnGL3/5S3z
00Ue4/fbb4fP58Oijj5r5lhnpMCclC4qs8ifg6XBbne9CvkeVqh6Vsb/LnFT1oxxt3qzq2qW7scx
xwmnCrY9Vsqp8g6eNPGBdZ2heSjcAYHyBC3kqNgSiMg6YxAZeyuIB67yddNmgOS44GA2KA9b55vL
iG6zIKkbytCYmFLiQ607O1Y6IhNru05ANlu0h+bGRAazzzdV1B2d4LgD27SNZLvObVOBCjqpaoj0
i6Xwc0xUvTUAAK9nAj40MYN3+SW8pxO6aUCpJ1DJr/8TTmphc6Ea2ig2tYUnHtnTFwjiYGgDcuHE
jrr76akydOrV/gcuyDI/Hg2XLXmHGjBn43e9+h9tvv4XcE9+Z+ZbZiReDDkVWeVPwItRLZBIVScDoWZsWiKijIZ
gckEKAIYxvGmZcTxVXdGRXhHNwczBxFNZF2Dd4Za3jTy5JsxjAz8ZgA6DTYsZ4xCOy2gMJrNiBUC
TVcWaZpa4oa7+PtQjojVkBhv4mQuAfZcDrHqDKrIsAMjRfHA6BMyxgA3BuITmUJINDgGmlJJapVk
lbqgrPGu7RbSFTWADA+VMqci6NcHZvsGmcWD5TOF0CJhtARv6YhJaw0k2OBlnw+xSN9R3/TVdcXS
```

```
YwgZ+bGQA6zxCv7o4TIineIPLIWAWUWVmBht6YhLaiUq7IRQq7UwNAIZCIVRVVQEAPJ7E5AkGk74
rM2bMwNatW818y4zEUykPYHwzYUY6qj4Vld2HFGDNrWVdbxzqkazMcpjik2SV/C4B0wjvHjNuaHg
q3QD0t5Y7O2Loi0kn+K8HLh0bOLsc2NRqERtYZ6QFt9ikqXtVttMUnySrlOVyYKoFZSw8HeqAE/v
mZiru2GCwJk5LNliQ2UF6W1VlO03xSbJK2W6HzvPPjHFgoZwpFZ3INzdTkZljrO+frKocIKurWB8
HK9hABsWHZjvhYpgNOW4HJhOef5taMy/75MlGBjixb26m4r2q6HS8OASsYSQ5BkOz6VTamRoALC0
tRVtbGwDA7/cjLy8P+/fv7/95Y2MjXC52Nga8LUijdFTy/0OqEiUZdbqHNTufkZGs8L/T32CzPQa
ANYDWdzRkexxK/U6MVq1bUQa2meBVcbibMzYUuZGlYkNzKPNUdWM2sD0O9rCB7TEA9EEfS9jA+Mb
NKt9c3oIdU4sTzZIUNQYTzZIyUVySUd/H10beEjaQe0jGxwCwaN/AWeDLKt9c3i4HphW54VV9VMe
CIup7MyuNj0kyjhFsGM74urDCG5Mn+xRFekaefuNglW8ub1VF04s9UPcvqu8T0ZDhviEqGrGB7XG
wwhuTlTVhagBw5syZ2LBhQ//XixYtwsqVK/Hkk0/iiSeewKpVqzB37lwz3zIj8ZYBaJSOmmnWVyP
RtazQKzDbtUzRrFKPJlV9f1ccnZHMsr54y4IErLqZ4OtwC1hze3uYI18nwLghSqbjcCyo7VpW7GW
3a5mi2aUeqO/R9gXiCJjOBrbnAmBwyLcgO5j1Uh7AGt/cQ8TlwKg8tp8VboeAGSazob5PhDopotT
nQDbrbCAaP+wNxNEVNZkNHK4JayoHOBgHC/YNujI/xveRHqeAmcXmMrK+V4SkYkO534EsF9tsILO
+9gbi6M6YDXwFxQFrqqt02aCMrwnAmksSfRMQthnpdQqYYTIb6npFXaWdz8VuViwAzC3V7p12d8b
Qm2GVGStrwlQqX3fddVi4cCEikcStwV133YWpU6fiT3/6Ex5//HFMmjQJ9957r5lvmbaCcQlNIa0
/w1AOwGS2PwFvGV8AkOt26NrTZ7p50xtysj8O5M3EtvYoohmmqrNgTJqqyDWR6Q1Nb0xCi4oNLoF
tzzdFZme48DgX8jwOTCzUrt3NGW5aeBwHcgO7rc0ENuieFfyNQ6Zs6I7qvVuqGO2SrpbZnaF5XBM
FXgcmFiTZIAPYkikbOLsoAvRrYmtrDDEpUzZwOA4m+74FIhI6I8lx9DgS2USsy2xG8jgXCrwOjM9
P/p2SDGzNMFucx3EgnxNb2mKIZ8oGzgJfgPkBwM6IhK6otkt6RRbbQXHAiA2ZZYTyuCaKfE5UE2z
YkmFpPCtrwtQZOHbsWCxduhReb+JAnpubi0ceeQTvv/8+PvjgAzz22GMoLi425b02bdqEm266Cdd
eey2+9a1v4Re/+AXC4fCAf5+8nRnKuHeLIrNvaHjM+ALMv8XmyZhUUVW2U3P4jIjAFx3pg0mWZS7
HwWxvTJINw3LY9m5RZHbXLh7nAvAVG4DE86xStcEMi8CuDNggybLOC5GHZ4X5bCA7IbPdJV2R2d6
YPK4JwIr9Exs3+aloeI4T5f4kG0KijN0ZsMHQKoKDcTDbN9coC5LlLumKzK6g4HFNAF+xAUj8jaW
+JBuCcRm7O01mAweXRWb75pINR0fkfMUGgH1rAEXm75/YWBO2hKBzc3ORnZ1t6mvm5eXhwQcfxKp
Vq7By5Ur09fXhySefHPDv85aOq8goHbUng3RUVlJRU5XZNzS8GdUqMvP2tjkkQd30K88toMDD/kN
qUoELOao08vaIhIPd6XtV8MoGci7s6sgsVX2wrInTkQ2CIJjKhqaghIhqGPI9Agq87N9gm+2bq/c
N5pMNX7Rn1ixJf3HI/poArGADX02zAPPZYGQjk8e4jQxgvm8u+busdwdXRAY7dmTYLIlHqwjAfDb
w1jQLMGZDJuNgZCOTw7hVBGC+b67uTMG4N6gici5sz7BZEo9WEYD53pisJF6ZvhKPHDmCBx54AJd
eeinOPPNMbNq0CQAQCATwi1/8Art27TLlfcaPH49hw4YBAJxOJ6ZMmYLGxsYB/z7ZrYuXh7VROur
WDKLRrKSipioSTFtaoxmlqvM6Dvpby/TBZLRhETi4pXI6BMzWlfOkPw68mfUqKvY5MUblRybKwNa
29G9vj3K6Jkg2bG6JQsyEDRyWLQAGGaEZbOR5HQMj39xMgh28+QYrKvU7NXucTJsl8dg0C9AHOza
3RiGlmdkhy7KuCzAv68LMJkE82sgAxr65mYyDLsuHk3Eoz3JqAviZNkvi0SoCMPbNzYQNg2YcMmE
Dh1YRgLFvbkZs4LBZFABUZjkxzMRmSbw1klNklB1s7r5hEGQA7t+/H8uWLcPGjRsxdepUSFLyWrC
goAC1tbV49dVXzXxLAEAwGMTq1auxcOHCAf8O9261DIT0ORE5OEGG0gEbEtUqep9cRl70kxVD0Q
kBFT+DF5O/BkAY9+3dFPVdYc6TuAMGJfzpCty08LL5QCg94XMJGWflTT1VDUmz4UiVXZab1zG3kB
6nQ0DEQndhHeLunyOZZHemGauCZ7YYOY46Ez+OVkTgLlND1i5wU5VY/NdKPQmL7W6YzL2pcmGzoi
EnliSDVkuQVM+x7LMbBLE61wA9PunTNYErxeHgME4nIbzYVy+C/mqipfuqIz9XemxoT0ioU+VKZX
tElDMQcY8YHIAkFOrCMBcT22e4w1mMpLX+TChwIU8d5INgaiMA2myoTUsIahiQ65bQCElNpg6+o8
++ihKSkrw1FNPIR6PY+3atZqfz58/H++9996AXuu+++7Djh07DH9266236n/bW02n4pprrrEAhMNh3HfffZg
/fz4uuOCCk75mTU1/e3eQFkNy8e3tbUVPTNKC/jbZGyE4AyUPN2sMBXJHdnNZrHej0QRMHbq9
HTTAz01e7NCnLg4/DySm8Zvcx+CtPvSjV8wAAvuwVAPj7v67wSKg9cMC0v9NK+SXA6/AjIiXg1Bi
UsG5XLSp8qX+GW466ACRhny/2oKamw6w/1VJVxRwAfP1ff1LXi5qatgH9LjkfdhFs8HDEhpGy9jN
ce6gTl2el97cf6PQDqp66cns9avp4YYMX6yLJz3DNrnp402DDXoINlR4JBzhhQ5YEuAU/YnLiM6z
vE7Fu1wGUe1P/DLeSbIhzxIa4lg0f1/WgpqR1QL9Lzoc9zfzuG0ZC+xl+cKgDl/nT+9trA1o2oL2
OKzZ8qmLDW7vq4K44dXkXORd292jnVYVH5IYN2RLgEvyIH2dDXa+I9bsOoDQdNhxxA0hmy+TFu1F
T027Wn2qpqmLavbSZbPD0tKCmZuBVSTQ1gmTDwQ5c4jOJDW11qOnlhA3ZXnwWVbFhZx2cabBhF8G
GSo7YkCMCTsEP8TgbjvSK+Gx3DYgm6gOSjg0xftgwNK5lwyd13agpTpMNLSQbmrlhA7lv+MehDly
cJhsOEmyQ2+pQ08MPGzYEtGy4PA02fNFt376hurr6pD83NQC4Y8cOfO9730NOTg4CgYDu5+Xl5Wh
tHdgC+vWvf/Z/Rf/YcDuPee+/F6NGhbRvMt/Vvb/JpeRvF/HDuVrRiPtALDEVA8pwRzr8Dbbe9X/Z6
wduywlMs1w3EZresa+r8WAJw9eQw8TvbLPgHgvFAPPu7o7v/6KPJRXV100t+pqanRLYq9h0MAkof
Z6iI/qquHmfq3WqmZB1qxQXUr055ThXNG+U/yG8bqbe4EEOz/esawElRX55jxJ1qu0oiEH+xJPlA
PhhyoGDkGuafwGzGaD61fNAFIgv3M8cNQXeQGD7q0OIaHatVscKfFhlBcRruKDQ4hwQYeGiUBwHn
BHqzrVLFBKEB1deFJf8doLuwh2VDMGRtqWjXZPe05VTh7ZOps6Gki2DCcHzYUh0X8857khrU26MC
QkWOQnQYbmncQbBjHERuKovjv2uT+a1/Qg7Fjh6fMhr6YhI51SdY6BeCsyWO4aJQEAAv7uvFpZ0/
/13VCYVps2HUoCKCz/+vq4iyu2DCjpgWbVd0M23OrcOaINNjQ2AEg1P/1zGGlqK421/PbKhUNE/H
DvUk2HAg6UDVqDLJcabBhu5YNZ40fhupCTthQGMX/HFSxIZQeG3pjEgIqNrgdwJmTx3DRKAlIsOE
zNRsc6bHhi4N8s2F6TYvGPqYjZyjmp8GG3gaCDcP5YUPBUBH3qdhQ0+fE0FFj4XedfC4bsmEbyYb
hqC7ghw2/Ppj5vqE7KqFLxQbPcTbw0AwFAM7t7caG7ZmzYXutlg3jKLLB9LxDj+fEQbSOjo6T/jw
VBYNB3HPPPaiursZ9992X0u+Kxksyt5xugT0ftjMhpNT2o69OOQVW2k5vgH6Cvy9+cZmtuXtOSFek
6NaVZvsFr6QYAFHgdGEd4Y25Lw/9OlNjxZ0hHEw0aoqTT9IDs+FrFSZd0RXo2nH5rAvhqHICEb+7
```

YPIINafjfxTntaKhokkFDlHSaHpB8HJrNR5d0RTqP0LTXBL9l8YDeI3RzmqVdvNrIAAnf3NGEN+b
2NPYNMUlGfR+/4zClSNv0oDmUXtMDck0MzeajS7oinUdommuCV/sURTo2pMtIjseh1O/U7P3jMrA
jDf+7iCjjGMkGjvZPU4rc8Kk+tqaQpPv/MxCRc2FYjpOb4B9gns0US2vC1ADghAkTsG7dOsOfxeN
xvPfee5g6daop7/Xiiy9ix44d2Lx5M5YuXYqlS5fi5z//+YB+l+xaVuR1IJ+DrmWKHIKAWSa0pdZ
1t+QIzgAws9gN9d7iQHccHeHMNy28GJMqmmPSw1rf4fL0G4djQRHq5pglPscpswhZktNhDRt4WxM
zS7Rs2N8VRyCSetdT3hlp2kaeezZob9vTOdjV94kQVdUqZX4+Ohoqchk0PUhnPvDaJV3RzBKPukA
R+wLpsoHvcSDZkO6BhvdxmGNCQLiuV4S6z1SF33HKLEKWZMiGNBjJ+1yYVaplw95AHN3R05ANJgU
7uB8HExhZ1xuHush1SJYDvlNkEbKkRLOkzJNteJ8Ls4kx2BuIoydmAhsoBoNNfUItW7YMn3/+OX7
5y1/21z23tbXhs88+wx133IGjR49i2bJlprzXTTfdhA0bNmDlypX9/x544IEB/a6uIw9nhxnAnGA
Hz1kdAJDtdmASUWKxJY3bW3IchnMGJnIubG+PIiqm5qsQFWU0BLXrYjhn84Ech3TMag91DwY2aNd
EOpsWo47QPCnX7cCEAu3fvCWNzoa8ZwfPJubCtrYoYil2RI4QbBAADOOdDWYEvjgbA8AcRrJ0g52
O8jx6NqTT9ZTnplkAMJuYC9vaYoinyIZwXEZjMHkIEpDI+uJJc0rMZwNPJv+KzGAk72si36OtJJG
RJhs4zw4m50I6bAjGJTSHkmxwCIlKEp5EMjKdc7Y+3sAfG8jglylnCs72TwVeB6pNqDJj6WxlagB
w3rx5+NnPfoa1a9fi7rvvBgA8+OCD+MEPfoCDBw/iwQcfxPTp0818y7TEc7cuRWbcTPCe3QIAc00
IdhzmfNNSle3EEFXX4ogI7OpIDUz1ffobbJ5uqQCDLJ/W1Dsi6zbyHLJnb1XUAAAgAElEQVTBjMu
Bw2Tgi7M1ARgw0oSMBt7GYWi2ExWqrsVhEdidIhvIG+zKLAe8HFlFAMZrIlU26C4H8viaC4BJbOB
8Iw9YExDm7XJgeI4TZSo2hEQZuztTY4ORVQRPNjKAQbZTSxps4Pw5AVjEBs7WBKDPCN1kQrYTb+M
wIseJElVH8764jL0pdksnrQF4s5EBjGwSUp8Lh7r5znwDzLGS0QXFOYw3mHO2YmccTM9Rv/jii7F
mzRo8/PDDuOuuu3D77bfjv/7rv7B69WpceOGFZr9dWiIjsHwe8rXBjl0dMQTjqaWjktktPIJJtyB
TPORLsqzbxA6GcUj1QMOzJ6aiSYVuZGXoccV7mjqgnws722MIxVM70OhLPvkfh1Qf1pJs5PnG1zg
IgpA5GzgfAwCYbILH1WC8HPiiI4ZwxmzgbyOf6YFG5NzzDTBmQ6rjwHs2KABMLszc44q8TOcxA5B
cEzvaY4ikWElCzgcu2ZDh8zJuxAbOAsKGbEjxbMW7tRIATC1yw6v6s48FxdTZ0EvuG/gbBzOqzAZ
DvCHTpIKYpPeDHE5xXVhiUuHz+bBw4UJMmjQJdXV1eOutt/D444+jqSm91tFmi0zJ5XHTYoZ5Me9
p6oDBDU1bFFIKt7fNIQkR1TDkewQUePnxblGUqdcXuXEbzuGaMMPjajDYA5T5nZq1nI558WA42Bk
d8lNhQ2NQ0njFFnoFrrxiFWXq5zMYnhNuh4AZGXpckVk+PG5gK7KcGKb6/GIS8EVHqmzgO7sFML4
wSyXrq4Hwii328uUVqyjTAw3vZV0A4HEKmFGcmcfVYMjyqcxyasq3oxKwM8Vs8aODYD4YBb5SYcM
xwiu21Oc4Zcd5FpVpIJSlUsd05XEKmF6U2dlKbyvE3ziYUWU2GPaRmVaZHSMq7cope8Vm/M6PP/4
4zjnnHHR2dmq+v2bNGtx5551466238Nlnn+GFF17AsmXL0NjYeIJXsk+Dwa8DyNy8mLyZ4BFMY/N
dyPckMzu6ozJqugaeqj4YMr4Ao7KFFOfCIBmHTA80gyHLB8gs6CPLfHdJVzQuX9stPRCVUdt9GrI
hw5v8wTIOmR5odFk+HAbFAaNxGPhGPsEG/i9Jxue7kKtiQ2dExsHugWd28N4wS5E+AzC1Q91gHYe
U2dA7ONiQiUeoLMuDIgNwYoFL0y29PSLp5vnJNFjXxOl4vgSAOWWZJRXoKg55jTdkwMgEG/ifD5l
WmbG2l844ALhlyxbMmzcPhYWF/d+LRqP47W9/Z//Z66vtZZHFzMmhTW9cWRoU7VR4I+86vp6bnVPD5PZHbNZDyazOljpgNYfZQ/Xtx/FrRwoQPLOWXRv82UtibKYQcWx3wm6UthFcAqn+Z2vnVMi33Ft5ckmxQCanVzlR5YbW2mNkEIPUT9dHOHDXmhc+DcilxdXbNaWbHX0JTqwyr0lsaE4fmw6Q9eZBWjodeCEQ5ttlAsZBEtDK0Nx5X0c08M2nO/QuRVLMFC8bGSaJ2Y24+2G11mlQwGYaQuzjQuNmnJyeWZFg7ekpJOPPgoFbcaUPIAHZqLHCumMf2aeU9DbwsUymqP5MI1Xs6s/5TIlAdUojmk0qBN98xZrIiwTgaZZtx8hfnbBHbV2FZ7t6rMLTmOrCaqyuNUuohqYgTSUHUoG6VbdcWCbZEQBwDA8h19t6V5vN/mCdgYcUZbtgUnnQ2q4z4T8ERKI/hANbc4zIEepG6qv9GAtqC0HXJcN9H3JWZLdJmHPZqMKCPudTFEMLUmqaHyzwHDjHeZd4/DUC6LYYjpNhVRKjLjo6WMsdvE0HlGOOnMjlzXuMCIK9KXMUjV8XTFSRtNmbhcdqK3sFOcV6XOHUHNdTaO4qERNfvzcbc9O3r2U6U8MUU39tZ7sP3HKYrX/mVf46EuB7dZFHvxkyTjqAMBtDwi0bmnyFSXuewSGP9XGE6uklg/8+p8Kqhj7FqR48rbZfU4yFlTZ0XL9VQhKRc+pzW9BOK2e5yCqjbeSkAMWx8N2JGORUT9ClNWXMM6hzpsy4QR4qfhkzjB3xG97rHRRRC8ZSpnMsaUwU3QzCkrQ+1u7YWIfHrGAOn+4KkvfTZZj3LE0Ku5DB6Jg0YYKeTFoCjKNVLS/NVuQxMsDE1qNwdT6Yz8D5wHB2LwoR9B6wxBCpNrgzTt+6knzyn5kMAgxMtzFA9+nZUg9xCglvwlvmtKU2Y4tShGdf9Y1ZHcg7vsCXqX4wF/r0AXnlc/F0a3OFpG+f8XsPjj4tSCT4jLBrYn6qXgnQ/CbhuAGKI79rA9NBd+WC+/9i1V5QfQIWMmj7Q9bIwDJAOhMXD/6TkG8NTOdxKRXwfTcE56VXsS2MUcw89MT6lbxzQqLYFbbkAT+5tZzVbHoGzr0Mr3m3T7Swr2rvoPoyEf/dfJoLEGE7pmgWAhkEpVb3uhv0sYIvRmmZbRY2GVLv+5gbbK6aqIBHtqCVCu0TiBjJ6cPbzvvZMPoKeq5HmIHGbmcT+Bv13I5SkN6sllNDqqTd+2dTHKIq3G2fOtBn2ZuNgW03npEGWRQhhXKCl0FDLXZvVT3ZJpQs+OIZBjdRyWsKAVIi+MJExdVqijB/k2qLpGcRPrOF2wQ4tlYk1nAzqrrgG6qdtbO6uVUJgBbLLvXn38/HhOwLtGK1ITd9UIwqSbHrJI4mcHPHSwSx2DHRVWpVEW08Yfl3rBdAKGsKZ+8F0n3sc1XwaXBcKcRX6MCXqCs0NgPoywbw7kOJHymL+SOovXbtTR0+VwR8xkiKvHnMk0hk+CHJXiPbOXdL0c+k3xvpNs/WMsZ3NFVxmQV/19Wu0o8RSkc6+nwyTfTrG6nWk4YYAiMOU04vkILTq2vfoDEbpcAs8mZl4sH6o46kxwOGjH8I9Jsn6WJQvlwbJYi5ydWZsvRC0ZsKdvVjPHNkR4UwKA2TuK95fzSbJKVZOPaVYVkNIWcCbq6Nbu+C7qgY0YzBTlM2pBr2y8X2iOldxwdZsfEJbD+MTjNvQLM1jjhyfRTcVeb2rbKzJnzXzZVkBF7U7vTpL6AR1wZ7xRzsrQNAwUcwuHNCG4DRkLqUcsq1Oe4NW+yEu8wpbt6qwvsD8EMsSOLR1t7URzB3f4Cs2XhPH0Tu7pdhwLfOa5IpN5REuASgf7pDTRJBV9SdKPFSL4DbCsjBIlYOzj2qOqpJWYsxa0ZZvwOsi6v8pKVlBu1YiQX/YM8ZsaIiwnUhOUcyFx+gsNrJhSn4w3X+Yh3vp2+oE6DmZJLN9bz+ISq8H8KTQ78bQYvKqXSbJCmdgfkfAiBJ5xPMA5bNZpqBUfNQxRm6Vz1K3E3o0uuC2c7vuEZPcPxSGcHLOTWy/qkPJrvVPLzU63npqxBqbUZnjBa34Bt0u9Y6DcTZaDAAhojK1xb4wqZYlGiW1ISqzDsbmCkhGC4IKo6Ee8LBvRyhMfWW5rkbC17wrWPXIwDeppVSdYqOJoXNKr6Ydq1Et5YlbyTzpmRmBqNvaEHFUKDl+Qlu74W9OAJH6PspF1hVvc8WWCU+FhjLoEfQm5E4NpJtL2Yd+7h0vPgPOjGGAoB5Hdce8NbJEIYe4IIo0F7DpbfCB1bYoVbEXgkdVgqj4w7B1PTL20ULa8PPHTUFvaL3D7y5WGcu+oK5sG60PvmEBW01ZUswexwoKfjYW2BxMmAo5UGxyZR+TtDvFyaP7ZNdeMyB9bOkOG8dhz8DKdDcDHnB5l7M4X9vS90g/Fk+UC4YnaQxAhRjP6aQhoXpl+qrBI5TBahQ8k6zoQ1tIDGQKpEpHsGZGcbtyS/KvTJjH1NLowJgbDNu5V+6xwydwjyYTVCNvp8ZRUFZMeeGjO0WTJGtYLmNTkmMSe3zTtIHXLcqRBrzmAKjRhD3w3bGexhOvdcZcZYMzOODfFfBhI3twV7hgRwx/FQ1nZkj+k4Jmzd1qLtZbMS+9PLfWSy7T0XeyOmqIXQB8cHWjHPJpX/I5GP7Qv9DOvqoeQqcvRqL9M69dz7WnjMvrsh+9Y2ghmqkCZTNh8oHNnNxOeAhoNbYm2LfgUBePU9Wu/RMY8jKa1SQWbKDMe9n+7FqDhQkJ2bu+5kFPntgD+nt3kHtTFnjm8fDbWYxQSNjs4c/ihlDvQLm4OzmpzOUeJnNVqFmEm4DKIUhbAqxfkdYMTRjmJH2nEB2qfjGLUL3XN+vj0K+XGZZoiAKc3b5JP7KzDXMM8nfJ2zmJSxZIRcO9TQbBtO6Bsh5fS0kW0HMj8gkaC1MaHKUTvw0U1eiFCxj02Y0ocdd+M3o8iT/NStyqIMjVYdSeLnUBcvqL6q/VfqeKIVjQRFvNHwnIabI/y94QCBcIIi+9iqHTn7iRSxWvWKtHtE2ZLjzGYDlXOUBGHVEnLS6qrm8YIA28O9i8F6ZaPz1PtM0P7IY7+tZmJ+P9sqWAKk1bVjQr77XC9tyaP7cMqnJNh5TMz/qlQQd4dLbAaUPo+GfwJtVMwgvaDKuBWVjbBaeFp2yQwKGeRlpCqMRlmk6b5gQcBRJsRSYP9DFNHVRM79qpOOz/Q3dvP0wmUmAEH8Uqj/FoOGELgxYSEQP6b7FhPtHumI7V/Tdk/wg3CYV7mbRVN3kYhfDjKVrIxxVxzmL4cQjFZHtuYZJDjbl3M/N6p1RGcG6DyjNIc7r8hJVqkmKvZxTZKmSgt3Zd2RVChOqBxrs2kC9S87TB1IiNTlS4T8+z4VgmyvkE3sTqh0GW3Oly2/g4Qt2DLjOQBdjqmT4z3OKKmXvY5jmb/Q9O3XdyNBQ9iYC9+iAcdYnGBkOoM+tB8z7ehFBBIDe1lS7fDcZHy2bN4aFMdGpeYMf/wO7h1vcgFeXYwzrGF0qACUCzsWwvwqmq4+JXX/zrzzTPP0KRnHh8lCWI9NPYtPj0V8YjBJvXwDJ3OmR66eXZNnt8CQJm+JWKSK/8/h69gMPF3RP7tFJP6oa+Q7WDWyhvGoyBEGbehuvv1FQTTF1uAMNwoCn5FvzyRqqs7cqZTDX6u+c9RGFqSRIWKAp3xvcmxLP5qJFpE9G3hfpMQ4a3ru8JFbxTOn7uU1O+8u1VKSC5TnjvWpyTj6SNjMY8sTbkbyPVbNtrETbdbT9uCh9GZNcRUgOM7Sd62u4OTLMkwQcUvqnfTYNP0XbJpqTZqkdD6+CG5YUzsKyttc7gy9jEVNhQkGD8bMnS7I7FCKa6x3+ZBZvvtUYQxN+5pqkMzgYdSRMzmS6CjHeQGkTzPCYUfgVGAKyV1gRDXFtc5HPbBjqN6GuvT2LGdgn3oxDiJPPaY0lswhjcNZcz7AGQZtd4sHnwNNaS0oCTuUvUSjygUPZDyMhK0mx0QVs4LAN/dGprfovkDRmHyuh3TPZmQ2sqKtH6WmHrJTm9Wf5xrUUIkXVOQBeh/eyRX5uWQmlXHg2Q6aMtz6e5o1QGh/GEXTPGNpZXUU1GrNmm4K8Km4O1TnwnGdV0aQ9CIrT1qRy8kRAQW69aqsvG5nMVqJ9FbDj3FTWZ8RmT+YJl43hbUchXd6UPDpGr7Y3EQbkkhNaNqBfmuukFH5x5c+CjAh2y1+ZS5HdsbmY1CbUTh6Co5/+KtrjaUZEMObIw5oyLbK4oD7/YmRb0RT/KwPjj4SSkQIOdZlG0WAyX5/dGExPJS1XOkFSAt2CTTNaHOhzfYXnCM4pFlGa/J1ZqpkAUL1bV5skSf6ni+Aqmp4VfbZQ/eyJF2t7nPY4uNaOqpwdqnqVaZGPf3fXh4j+tuFh2mBxUN56fLVqTzAZUzd+yBZpxRr8kWpHLnx2idZcAUNkzuBm7C4JbL/LxdFdUP1nJtY8u/3R6wbtEN9hb9hbKn3Rm1sG0mjoq9l7TkeCx3FvBtL7LMJ1txYPD7ynMW+Ix9gUqr4yHMoVNfLVJUBkEWc3Ou0MmKc2Hdjuj/I5g4UPf0iHHBU6KyTcjMtPexCtIO4oMt75trRLLfGDjbnyNfv8tJSR8W6dM4KlVRpqX7kAnzXT6qnGwNpK/YZGpbaOQeaJ+KfvQAzvzQMlBCWt3Y0I7LPM8oPYZNJgD1EwMhTFnzK2HYhKY3yCtc1L7cgP1+E1O3bO0KMq4X4sZ3Ip3NrdZ0lEu0JUNO2UmCs9T1M8jlDLJSGFmqhhoKjOjz5PUbhRyOfCgN4BAVH+Pxy3jCH7Jx3RyXLTQMgXKOfzOhKJE+gZVpYn/ju38FhDLHVH7+Gd9DvJ2xHnTr8Xp1nngPS4VMl9kFPJfU8FZrL87FAt5Vr2rfGIvSWBC6x0XUQqTCVGRG4lhi0jkjR1lxAZNbQcCm3R3LvCLuEgVgqpLexCJebcJ4iyqLLsgkMdfQ/qJfHBwMvcQ0r56pTKfOD8z7bJcBaYQdqnNHqHcB9E8U4bjDBVaEtDNlIt2aXbaOlz+7Eb8CoyMkSu9kbNaCP8WSNoyNtJFfwHfnGg2lUzH6pu6Ud3gpZdZdu9X0tZ8o+Kws7pq1UNq3ZD8H4/UMyyt1YZj2idXCGG8ZW9cj6t9mHAs9xhmn3DqjVHtSzMhQkiSq+RD8lwOWBEBJ6hlz17UqVGRmJAmKdwr0GzJGFdz/bjhJvpN+NJzhsBb4gJ+vC0Mu1mYl/PoL2wt8SdYjJjKKaYwaHvn9i2uQD6Y8CmT7eSsRVOqB3PuQ4nJvIa8kiN6trr8LHc46H44bXh7qB9vUnbDCc8bXEDjT13aAwjGECdDFzgcQ5ogNtUchDjWuYL0u4E4bLzlcIA2VQK9A2p5cvS/5Tw6tOkHLk4W3OYfTkNQ3dp02L3lg2t3vw7GCW3JvE5gyZ3aQ2ORK8IS1nWPbGRMG1R1qOKWMpiB1Tq4Y1LnYvA+Zb3Z2dA4PYNa3A/A+TYLr1v4tsb7i/8oKHvUdrkdEXQlm7yOvn4xBvNTEVdPRBR4UbsjyBz7o3/Y0Bsuy1pQT0Dr7FhGHMqdMvB8s9PzddtUMG61m83Nk+JdBCWtMS3LbwDQsz7r4MQ2iZF4FWqY6ETwRWZo4NqGN38oBaQDTAAFAdGrNvJOtmw/bdjT+Q8F2Gd4SoAiJZ/XMM+i6V6XJwXGtmwnqUIJXM4O/pBKKrNk+BOPpx2Lw6ExpMmqnNqYaUm7YjOaXumuI5KnVbS6Wi3iD0gg==

```
MdEN4aV6fqm/CXryW5weLnIZzcUFrpwZqDktxg9fpTGPHw15JanKg/nxscFs8NPds6kOGzW3pFg4
gDlSpyw+f+uSHWa51FKpzny5P1Tfi/gNxg8fMQxiyzA2c9eOc7m+WGMHLk69/U4pRkpl1ujGfaRd
wA2L17dyxevBh9+vTBmTNn0L9/fyxatAj9+vXze9/OnTuRmpqKq666KtI/GbZHiytw1OehTgAwso
srZuXRgzQQQy1e3NQs4rf/ORMwPPuKTvY6D6Gmdh2u9uCx4gq/167KdcGVaN1eKkEQAkb6hLpRrS
qtC0jOo7om6162aApnjatHtlUENPiM7GLt86B18WJvbpAuaD68s81yQ4hr4vsqDx7/9Kzfaz/Ndc
Fp+dyg7TysOFCH9d/X+71m/dygbY0rURTx8LazKKvzzw0/t11uaIRHRW7w3QE43SFgmMXrDVpnUH
xX5cETktzwszwXkiw8Yj5BZopbqPPw5re1eO+QvXKD1jWuRFHEQ9sqAhqDr7Z4bpA2fIXKDZ5mEf
duOROw2cHlnZyKv2MFWht9vqvy4ElpbuiSDIfFc4N0iaVQz1Z/+7YWGyS54RqLXxMDspPgW/07UN
kUdLd0URTx4CcVfo3BiULLvcLKtM4y8zSLuOc/p/1yQ0aSgKEd4ys3HKz04Knt/rnh6rzkmM+002
VSfv/+/bFgwQKsXLkSCxcuxGWXXRbwnssuuwxvvvkmhgwZosefDMsqSe/1A/3S8aO2sV2TMFK5aY
nI82lFDrV48ZIvq7G13D9Yn7ksw5JbcvuSm9qltCNysyjivv+c8dvVMTkRmDO4raFljAYtFfkfqj
14RNIIWpCdhLt7pRlStmjRusbV26W1eFvSe/1gv3RcZOE1AAHtixf/6ctqbJPkhpmD2qKt0+q5If
BmHTQ3fOzfQZKSKNgjN8hM51FyuNqDKZLcMLB9Eu662Nq5QesaV6tL67BW0kHyUL826G7x3NAtPR
EdfHJDXZOIL88o54aX91ajWPLgN6uwLTIsnhsCRvkcV84Nch0kqQ4BzxVmGlrGaNAySvpQtQePSR
o6Lm2fhDt6Wjs3aF3jamVpHdZ959/Q8VD/NpYf5XNBeiLa+6zXVuMRsS/ITJKFX1Rju+Q8zS5siz
ZJFs8NAdeEcn5sOtcIKu08fa7QBvWGgPqT8nn4XqaD5LKcJIzvmWpI2aIlLSkBfbKkuUH5PMh1kD
xS0MbyI9+0zjJb8HlVwHmaXdgW6TbLDcGes5uaRdwr6TxNcwh41gS5wdrfQgT6ZDnwxMCMWBdDF2
p3s/vydCNmflbp99rP8lyYYPEGHwC4JCsJqT7dtyfqmwMWJfZa+lUN/lPmf47+eFlb9My0dmMwIL
9mh5xmUcTkjyv8FvF2JQJ/HpFl6dEMXmofaI67BTy0zb+ho1+7JDw+wPq5QW7xYqXcsPd0I2ZLcs
PP81z4/y62dsUNaNktPcWn+7a8rhmHFXZEXvxlNT6R5IZnBmVYvqMIkHugUc4Nch0kf77CprlB4Z
oodwsBHST92yVhyoA2hpUtWrQsbP75qQY8u8s/N4zq4sKdFn+oA4A+WUlI9pmFU1bXrLhb+mLZDp
IM9Ghr7Yc6QP0DjTc3SDtI7FJvUDslvMwt4FFJbhiQbePcoHCv2HOqAc9JcsM1XZMxPt/6uaFfuy
S4fHLDkdomxdywSK6DZFBby3ciA+pzg9zsspREAa9ckWXpUZBeaqeEH6sXAjpIBrZPwiMF9sgNam
eZ7T7ZgOd3+08Fv7ZrMm63QW6QzjI7XN2EMoWZJAv3VuNTSW54trCtKTqR47IB0JkAvDqinaWne/
pSU5F3N4m45z9n0OCzyV2mU8CfhmdZenFWL0eCgIHtQyemAzUCntnpn5yvzHVhYm/rN4ICwMD2Tv
h+m/srPKhwB+5s+Oq+Gnx0zO332owft0UvGzSCAuoqLaIoYmaJE2d9GkGdCS2NoFae7ulLTaXF3S
Ri0pbTfrkhyyVgkU1yQ1KCgAHS3CDzQPNtjYCZO/0fZn6a68JvbNBBAgCXSnaM31fhQWVDYG54Ra
aD5GmbdJAA6nJDsyjiGUlucCUCr/7ERrlBxdSuek/LyBbfjbPbuRLw8jB75AZnooAB2aFHuJTUCJ
gl6SC5Os+Fuy0+ItZLOs1vX4UHVTK7pS/5shofB3SeZiDfBh0kgLo1rppFEX/8xhmwA7JdGkEBdT
my3iPini1n4Ls5brYrAS8Py7RNbihoF/rZ6ptqAbMljaAju7hwlw06T4HA2VVfnmlEjUJukM4um2
mTzlNA3XN2syjijyVOv0ZQO3WeAupmmdXJ5Ib2yfbJDWpnmX1dLQR0kIzqmow7TNJ5GpcNgFMvzU
CfdvZISoC6m/WcXZUBU4PnD81E5xgvQqmnUA80DU0iZnzjgtunob6tU8Di4VmW3tnSV1tnAnpl+v
cs7Drpfx6+rmjE0//1bwQd0dmFey+xx8MMoG6Nq/+3vwbFFf7xP+3HGbgkyz65QU1P/rOfVeKrM/
7D+BcMzUInO+eGE4G5Yfo3Ltt2kABApisBF/uMVhIRmBv2VzTij5IOkp90dmGSTTpIAHVrXL22rw
bbJblhuo06SAB1U7tm76rEV5IpPgsuz0RHG+WGUA807iYRM7522baDBACyXAnI98kNzWLLdeFr35
nAGSRX5brwGzvlBhVrXP35qxr896x//M/4cVtcbOvcEFhvmPlZZcD0v5eGZaJDio1yQ4fgD/nuc8
8Udu0gAYB2yYno4bPrfbMI7JLsemrn2WVe0pFvO2V2RF76VQ12SnKDXWaXeamZZfbMzrP4WrKR1M
LLM5Fjp9wQIkfWe1qeKQJyw+XmaQSNuwbAoR2dmNwnPdbF0FVBtjPo4sXF5W4s3Ou/w+nYi1Jw80
XmaIXWS6gLcu7uKnxd4x/y84ZkWnqXLjnBHmgam1t6ZnzXr81IErB4eKZtGkGB0GtclZxtxPQd/h
WWYZ3slxukixeXVjXhlM+Xv7XMjZclueFXF6Xgf7qnRKuIUREqN8zZXYkSSW54cWgmcu2WG4KMcG
loaskNvh0kGU775YZQa1x9U9GIGZIOkis6OfFbG3WQAC3TknwHJXxb6fFb2PzjMjf+JMkNt/RIwe
gL7ZUbQj3QPLerEiW1/rnBbp2nQPAc2dAkYpIkN9it8xQIvcbV1wodJPfYLDdcmpPkN5Pkm7P+M0
n+c8yNJV/654Zf/ygVN1xgs9wQot4w+7NKfCvJDQsuz7RV5ykQvCPZ7rPLvHpkOJDpu1t6o4hvfB
q59p1ptPXsMq9Qs8w+OurG0q/8dz++LT8Vv7B5bpB2HM76rBKlktzwksk6T+OqATDdIWDpFVkx33
lFbykOAf1khqPWepqxaG8Vxn1wGr6dmLmpCZg3xPoLV0tJb1Kfn25EvUfEoWoPHvzkDOZ/4b8ewU
0XpmDsRfZKSoDyA83HZW7cuOkkdkt67+YOyUTX9NivR6AnpTWuGptFvP51Df5n0ynUNfnvWrf0Cn
s9zADKixfXNDZj0RdVuO1fp+Dbh5mXmoi5NswN0mtiz6lGuJtEfF/lwQOfnMFLX/g/zIzpnoIxNu
sgAZQrLf855sbozSexR5Ib5g3JRJd4yA3HG9DQJOL/vq7BTZtPBXSQLLFhbkhPSggY7bzzZEtueP
mLKoyX5IYuafbMDdJY2H2qJRa+r/Lg95+cwUJJbvjlRSm4qXsc5IZzD/lbjrXUG76QzCCxYweJ0h
pXDU0i/rK/BjdtPumfG2zYQQIAbZIS0DvLP+/vPNmAmsZmLFTIDXbYKEsqcCZJSx3yuyoPfvfxGS
ySdp7asIMEkK83iKKIj4625Aa7zy4DvLkhMEc2NIn43/01uPkfJ23fQQIozzKrbmzGS59X4c4PT/
n9rFt6oi02w5FSmmX2XZUH9398BoslHSS39kjBjSbLDfaq2Yfw3OC2lt+hS8llOU6/KRsvfVGNB7
ZW4Hhd4FoNi4dnWX7XXzmdUhPRNT0Rh89t/tHYDNz579P415F6SJes6JSSgPkmGoqrJ2liKj7egB
s2nghY1wsAbrwgGbf0MFdS0sugHCc2+OzE9frXtVj4RTUOyWwOM2dwW3SzWUOH16AOTnzuU0Fb8H
kV7v/YgxP1Muu4XJFpy9zQOTURXdIS8cO5RbwbmoE7PpTPDZ1TEzBvqP0aOgCZ3FDuxvUbTwSs6w
UA/3NhCn5pww4SoCU3bDp8Pjf83zc1mP9FVeu9w9fzNuwg8RqUk+T38Db/8yrcd1Y+NywenmX5Hc
Hl5KUlIjc1AUdrW47Z3dSSG97/od5vDSOgpfP0BRs2ggKBa31tK3fjFxtPBGyKBAA3d0/BWBt2kA
AtOXLzD+fXR/7fr2sw73P53PCCDTtIvAblOP2WBpm3pwolZz04KckNAoClV9gzN3RJS0SnlASUnX
uOqm8Cxv9LPjfkpSZi7mC75gb/esPWcjd+sfFkwJp/gD1nl3ld1sGJfx45nxuW7a/B87urWuuVvu
w4u8zrshyn36yJuXuq8E2FB6fc8rkhw4a5wTvLzNvGUusRcbtCbuiSlojnTVhvsMW3MnXqVBQWFu
```

```
LkyZNB32eH3WeUSHsmPj3eINv4d0/vNFyVlxytYkWd9DxsPhz4gC8A+NPwLGTZsKEDAC5u60CbJJ
+h6g2ibONfp5QELLBpIyggPyJUrvHvhguSMe5H9s0Ncg3Ccg/4v70kDT/JtW9ukJ4Hpdyw2Ma5oX
emA2k+60WcbRBlG/86pyZg/tC2cZMb9pxqlH3AH31hMm61aQcJEHgetpXL54bJfdLxk1xXtIoVdd
LzsOlwYCU+QQCWXGHPzlMAuCQrCak+uaGiQZRt/MtNTcCLNu0gAeRHi8vlhpsuTMGvbNpBAsjnBm
njHwDc3zcdV3S2Z26QGy2unBvs2XkKAH3aJSHFZy2ZM25RtvGvS1qiLWeXeUmfL3efapRt/BvT3Z
6zy7ykOXJbeUNA4x8A/L5vOoZ1Ym5YYtIOEvOVSKO///3vSEtTN8ferg8zQGBikkpzCHh8QBs8a8
OhuL6kF6RUXnIz3ry6Ha7uYt+GjsQEIWDHT6mrcl3Y9IscZCfbs4cKCFzjSsqZANzSuRGvjWhn89
wQfAHidIeAJwa2waxBds8Nwc9Dl+RmvHV1Nn5q4w6SltwQ/Dz8NNeFjdfloJ2Nc4N0jSspZwLw69
xGvGr73BD8PpHuEPDUwDaYOSgjSiWKjVDnoWtyM976WTautHEHiSNBwMAQueFneS5suC7Hth0kQO
AaV1KuRGBcbiNeGWGvNc6kpA/5Um2SBEy9NAN/vMzmuSHEeeia3IwVV2fbuvM0KUHAgBC54eo8Fz
Zc1962jaAAQj5XeXPD0ivsnRtCPWe3SRIw7dIMzLB7bghxHrqlNGPVz7MxwqQdJJYeu37kyBGsWL
ECf/7zn7F27dpYFyemLmyTiGxXQkArfJpDwMTeafhd33RbN/Z4KV2QF6Qn4tEBbfDj5qPo3dW+PT
Neg3Kc+OiYO+D1n3R24fGBbTC0ozkTkp68a1xJ1ydJSgDu6JmGP/RLR92xg0h22PdGDbQsXpzlEn
DG7d81leYQMKl3Gu6P89xwYZtEPFrQBpeKR9G7q30r8V6DOjhlRwRfmevC4wPaYEgc5AbvGlfS3a
+d3tzQvw1qj5bClWjv3PCjtg60dQo42+CfG9IdAiZdkob7+6TbuiHYS7o5jlf3c7lhYLzkhhyn7K
i/q87lhsFxkBu8a1ztqwjMDXeeyw01cZAberZ1IMMpoFImN9xzSRomx0tuUKg3dG+TiCkDMjCg+Q
h623hAgdegHCe2yYz6+2luyzNFYQf754ZMVwIubusI2OHWmQDceXEa/tCvJTc4bZ4bvLPMqhoDc8
O9l6Rjct90W3cSeSnlhovO5YaC5iPobeIBBZZtAPR4PHj66afx2GOPISXF/g06oQhCS0PfnN0tG1
2kOgT8plcaft8vHe3j4Cbt9eOcJFzaPgmfnVsPsVt6Ih4paINf/ygVSQkCSkpiXMAoub1nKhZ9Wd
W6KO0VnZx4YmAGLrfpcGwlk3qn4fefVABoafi7PT8VD/Vv07qmVzyEgyAI+E2vdLyw53xumNgrDb
+Ls9wwqIMTA7KTWjfB6Zbe8nB/a5zlhjt6pmHxl9WtuWFEZxeeiJNOAV+Teqfjwa3nc8P4/DQ81D
+9dU2veAiHBEHAxF7pmPd5S26Itw5Dr8IcJ/q3S2pdK9XbYXhLj/jKDXf2TMOSL6tbd/T8ybncEA
+dAr4m9k7HQ9vO5wZvh2G85Ybf9ErD/M9bFrOPtw5DryEdnOjXLql1Exxvh+EtPVLhiLPcsPSr6t
ZlU67MdeGJOOkU8DWxdxoeKW7Z7de3w9C73l88hENigoAJvdJaN8+Ltw5Dr8s7OtEny4Evz3Ukez
sMf2WR3CBUVFSIod8WfY888gj27Nkj+7N77rkHp06dgiAImDRpEgCgsLAQGzZsQPv27RU/s8Ts30
aEmkXgP6cTUe0BhmY1oV3w0am2VeMB/nkyEVlJwOVZTUiyf0eErB/qBGw9k4je6c3olxG4PkM8EE
Xgv2cTcLReQGFmMzonmzLdGY65oYVvbhiW1QRHnOeGS9o0o2+b+M0NO84moOxcbugU57mhpqklN2
QFn+llW9Ue4H3mBhyuE1BckYhL0pvRJ45zw/azCSivF1CY1YxOrvjNDVtOJ6KWuQHvn0xEu3PPFM
wNcZ4bKhJQ7o7v3NAkAltOJaKuueWayIzj3PDPk4nINmFuyM/PD/pz0zYAhjJx4kSUlZUhIaHlbB
87dgwdOnTAM888g0svvTTGpSOzKikpCXlRUPxgPJAXY4F8MR7Ii7FAvhgP5MVYIF+MB/IyeyxYtg
FQSs0IQCIiIiIiIiIionhjosGKRERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERE
NsACQiIiIiIiIiIiIrIxR6wLoIXb7cbUqVNRWlqK5ORkpKSk4MEHH0Tfvn3R0NCAOXPmYNeuXQCAsW
PH4rbbbmv93b/97W/97W9YvXo1AGDAgAF44okn4HQ6AQDbt2/H/Pnz0djYiJycHEyfPh25ububRP0DSxI
h4+Pbbb/HCCy/g9OnTcDgc6NatG6ZMmYLs7OyYHCOpY1Ru8Fq8eDFef/llLFu2DP369G9YvegVFYjI
qHkydPYt68eSgpKUFSUhIGDhyIxx57LPoHSKoZFQurvVq3C22+/jcTERAiCgMmTJ2Po0KHRP0DSJN
x4OHDgAJ5//nmUlJSgffv2WLVqld/nsh5pPUBEAuuQ1mVUbvBiPdI6jIoF1iGty0ah4nGPGyZ9y0lIgNgA
AwevRoDBs2DIIg4KOPPKsKUKVOwYcMGFBUobF2ShfmvWrEFVVRXGjx+xevVqvP/fvj888+xevVqvP
7662jTpg2mTJmCN998E3feeadSkdfwYJSkHarQHw3ZdHttde2sTNmzfj2muvRUZGBgRBwCC9
/9wNZoAsKhttTuUr4FVO3gVwIMVZvt2Me/Gsmfj8UHCmrNyzfwAwZQcGQcvWPh0x/Fc/ivWV3m
xaS/4KJtRnCFx8VgcbvrmP9dQ6QYqbnKbRd5HLxZa/b69wWHhm8T0FG5Qco3f0Q04jIAbNcXFmoc
QJe1xnUKq+9vbuSKfvamKcgPdKrfUuAZ5hUpdAesnQQQtCgvbNrKsvbkUeeKzrVVM5hoWJ6T6WG
SjsrtQiZv5PqjJgLBjbBZ6r7bPVnqjQG1NFxSwNyJdl2S+4jC+WNfi/ycyYQuYE41t1Nfe0P/Tt
DxpMi6y6+4z9ZnVqbUaZ1NL8YnV0/qbHYFC3D5tbnh/Bv+EsAzFrb1TjUcx2EGbqhVx/kdhDWkVn
kYv6GlVzDFmGmR4VUbVyp3mN/vYnJoaLWOd1bUBBiXhP03Y5Lq6Vs6GPiYi1ZRtrq3jCl6I+/wVdh
yMppfGEmOuVWzxCLL9zNtMG/HBxt5ExmKNKNcpHaxKnpqpZOJ7Kh4OA3r7QaqpvrGVMDWTUsT+xjQwh
eR19Hz0R69Fh/zR+zNZ1zKaYdwaB/EUFYaWmeLPEiZNGibDelHDhdQR0zRK4aoX8ZJiCbwWSeBTuU
yxUhkVE3ECsXKWdIT+Lfsvq6nXTCAVZwCe0FVXdwaj4UYeWS/t9HiyRhH9EJZljXVhsOWDl1zZ6k
UD41wXjZhMIyj1tTFqhuVi3sN2jBZ+jIZ9kZHt6Kz7ZvxeCFxT/3yrzGsTwCm2yvdHuOiMU79BNPh
/9c/Gkkj7oznsu21Mw8hYUgu/SEhm6xWcaJqIfoDjy5Xbbn30pbOYc47Pxnz9zDcWlgXP7yCsa/sEx
8gxXIh1RfmsMa5pEjKJYcFC08F7tJAhF2FoZW3zSStVZuzkmGowUi9bH5VfBqUoaNkMjqoW38VObLq
uTT24WfIUAFCSxe+A+GIsG+mWGWQhBm7ELx5BEb9j+w43W2wzEbn1VzyVHN6oxJ9xQuVF49/dVdzd
Px8cOWe7x3vORYVdiqhUHBtBNeuZqdOOfuQm3/bmfvrbW7cD7Ipn/BN5r0TzmHBcZTGlA1ZhNLcS/
EKcKP6AxrsQWhbdvhHcwmWo4CAfdWdoWO/fOfUTUbvlz97X1yGmOtBh3l18T0LsDWl9X74/9g/MTqD
uMzE8yx/u6MFiJDfEycn1S+rHS3UbVM6vgCi1JCDvFLxlxFXTzOM+YbJgoZvbVxD0jYHoK2bvOGZ
qNjz2eOh//yDr2+uD3Wf2FZn84DZdaR4Hv3NthEO4eVrBIKQqzebs1FnZe3DM8NN5iKnt6l3Yk3kQO
XMtjP7iyGDhDOFWYkbFoSxBBfRCuuHvMbdAdX6N9kSTyD1EaD8qOyW8HjI3iAjK/0ETIHi3DDKfI5k
+c3ErTupjL2ZnRDGvt3tEfAv6LL5T8lq/PAW9qp0TtOdFdzVVAWTg0aq0A6x/eyDvvQDZnQKtDqMky
yQDmWxdWNKwcNUWbNVF5mS2xBnzpd8gN3ZTS9fErsZOe5gctI+TKHdCoOdOx5BtAdHMbvN8bzNk5Kc
eZY2cfNGHDhVXdgqxhWSu3SQxsWnsrJlvBDZZFs0flVEXxeZ7Z5wf3E8SoJSH0LGqx+3Bw0/s48HsJl
Qy5Axnf8z8TJmW7HiUDqwE4aDzlntl1CzYjQRzLoxNO9qBWcBH2tDemdLpw5OEuibu1hUXiEhKEtvE
```

```
EiHjZt2oTp06ejR48erb227dq1w6JFi6J+fKSeUbnB1+jRozFr1iz069cvKsdE4TMqHnbu3ImXXn
oJTU1NSEpKwuTJk1FYWBj14yP1jIqFV155BR988AGSkpLgcDhw3333YciQIVE/PtIm3Hg4efIk7r
rrLjQ0NKCqqgrZ2dkYMmQIpk6dCoD1SCsyIhZYh7Quo3KDL9YjrcGoWGAd0pqMiodY1iMt1QBIRE
RERERERERE2lhuCjAREREREREREREGpxwZAIiIiIiIiIiIiG2MDIBEREREREREREkY2xAZCIiIiIiI
iIiMjG2ABIREREREREREREkY2wAJCIiIiIiIiIisjE2ABIREREREREREEdkYGwCJiIiIiIiIiIhsjA
2AREREREREREREENsYGQCIiIiIiIiIiIhtjAyAREREREREREZGNsQGQiIiIiIiIiIjIxtgASERERE
RERERERZGNsACQiIiIiIiIiIrIxNgASERERERERERHZGBsAiYiIiIiIiIiIbIwNgEREREREREREREERD
bGBkAiIiIiIiIiIiIbYwMgxZWSkpJYF4FMhPFAXowF8hXreCgud6OopAaflrtjWg6Kfsyq4
uxQL4YD+Rl9lhwxLoARERERNRiSnEFlpfUQgAgArg9PxVzh2TGulhEREREZHEcAUhERERkAsXlbi
wvqUWtR0SNR0StR8TyklqOBCQiIiKiiLEBkIiIiMgESis9ECSvCQAOVHpiURwiIIishE2ABIRER
GZwEUZDoiS10QAPTK4YgsRERERRYYNgEREREQxIN3sY0hHF27PT0WqQ0CaQ0CqQ8D4/FQM7uiKcU
mJiIiIyOrYpUxEREQUZUqbfcwdkokx3VNwoNKDHhkONv4REREREkS7YAEhEREQURb6bfXgtL6n
4pGNzR1fofEREREZFeOAWYiIiIKIq42QcRERERRsbAImIiIiiiJt9EBEREVG0sQGQiIiIKIq42Q
cRERERRu7momIiIiijJt9EBEREVE0sQGQiIiIKAa42QcRERERQunABMREREREREREEdkYGwCJiI
iIiIiIiIhsjA2AREREREREREREENsYGQCIiIiIiIiIiIhtjAyAREREREREREZGNcRdgIiIiIpMrLn
ejtNKDHhkO7hxMREREJqZvgHQ7XZj6tSpKC0tRXJyUvDggw+ib9++aGhowA7XZj6tSpKC0tRXJyMlJSUvDggw+ib9++aGhowJw5c7Br1y4AwN
ixY3HbbbfFuMREREE+plSXIHlJbUQAIgAbs9PxdwhmbEuFhERERFZiOkbAAFg9OjRGDZsGARBwE
cffYQpU6Zgw4YNKCoqQm1tLdasWYOqqiqMHz9e/fv3R/WJdZCIiIqKIFZe7sbykFrUesW15S
W1GNM9hSMBiYiIiEg1068B6HA5ZKO9cqjg4s5XZO7dDmNfrvxb2kG3Cu+nwpoo+TlQ0c9GWX5KpKk9
FWSz4HvQm+Yr0ZzG1u9cWb0XwE8R8JBm+dsu+ZcdHzJdyZWSKsLtKtxhYi2yVn+mYjUg8MtZ7Nq
IgBj7MBKEcSpvemLKvbMfsnzYbThX8R8zVeqVyG9E3TjYL3qALzZIMn0PZYOIK5YrAOPRVKVYm+9Ve
PxQ2fHMUwx3N9JmMUOfbTKd+7eJU7KM3/5tYMKZGX3n+Zx4lEYyJXhnzQtG/ufcs0w4GpoenqvNhc
UCWPeJKM0jxVZEU
```

```
vfvn3405/+hIEDB+KCCy7A+vXr8cYbb3CqcJjUXuhKFebgjR0tlpfUoqy2ya8SzIcUewnVEKTmAc
vqDUmAfGVsQHaSKaew6MHMoyrUCmcqT7iVVT0fQpUrAS18rx9peZXyuZFTV9kBpD+75M1YUPswof
bhR8soklDfm3TTB7UjC5XWRDJrLCjlw02H63WvL5qlQSyaD9N2WYpAehwigJv/ccrvPUY15qrZpT
nUOnlG0SP/R7txR+6cxbqBKRa5waz1IbtubGMX0u+nMEf++dKMz2Cas/OhQ4cwatSo1n9v3rwZbd
u2xcKFC+F0OuFwOPDPf/6TDYBh0HqhS29qahs7mkURmw7Xo6EZsp9L1hdpQ5AdGpKUN1fIMMUUFi
MYOaoiWhUkrVN5IqmsapnmF4yWSsDbB+tky6v0kGLEw0usK/hG0hKnese0HfKmmWl5wNYyikTL96
Z1ZKHSmkhmJPfg/fM8F/5p0ANnrBvEYtFgb5dRM9LjiFaDjZpdmtX+XTPkfy0dh0bxPWdm6cSKZm
4IVh+KZcNgLDa2MWtDqBmoGWC17XhDQA4w66h/zSWSrvNXXFyMIUOGwOl0AgB69uyJ9evX61fCOB
Hpha6lsaNZhKl3xjJDAjJDGSIRaUOQmabnhEupMpbsEAIafaI9hcUoRq3zGM0GI6VjkJvKo0dlNd
g0PzW0VALqPWLMK9dmqeAbQUucGhHTdsibZqblAVtLLtTyvYWzPpWVGn2kD94HKj14/4jb7z163h
tjeW7iocE+WnXZaDXY6JVjteb/SNbz7ZHhULVDuBlGD5npmohGbghWH1LqrI2WaG9sE+2OYSs9Z6
sdYOUQBFx27nkykkEF0aA5Yjp27IivvvoKo0ePxqFDh3Dw4EHceeedrT+vqKiAyxW9Az169Chmzp
yJ48ePw+Fw4OGHH0ZhYWHU/r5eIr3Q1TR2SHt0fRs7zLL2SixGpmgpg1zCMmMSi3QIvVmm52jl+1
0Eq4yNy0/za/QJNYXFDN+x2jLosc5jrHul1U7l0auyKjfNTy25MvhWAnyvn2RH7DcFMFMFX09aGj
aNagS1at60Cq0P/2pzoV6NhVZq6AOU7ym+xyEi+DqGZrg3hsvuDfbRXo8uGvGvR47Vmv8jXc9Xri
FJbodwM4wesvs1IaVUH/rHD/Ux7yiN5sY20e4YNuuoSzlaBliJAJ4cmAEAYQ8qiBahoqJCWv6gli
1bhtdeew3Dhg1DaWkpqqursXbtWqSnpwMAHn/8cZw4cQLLli0zpMBSv/vd7zB8+HDccsst+Oqrr/
Dggw9i/fr1spuQZGaaeIpT94HAPa8CrtTzr7lrgX3/AXpfAYgiIAjAp2uAtc+p//1XJgLf7QYuHA
DkXACc+L7l3zc9CQy+KfTnGslbBogABGDfx0Dv4crHoEX3gUD7bueP99xrMxYtwx/vv/v8a1rKMP
A6//d+urbl59LX1j6r/VyoPYZgr8uRfu9aRfr70ST9Ln2/H4U437FjBwYNGuT/+9L3yn2uXt+xHL
nv14gyhMo53r/13W7gwgLAlXb+ffU1wJrZwI51kZUhUqHyntLvyFw/frGgVxkA/+snnPLqzQxlMM
Kg0cCYp9TFqYr3hh0PgLXyptUYWXdR+N4CYsEM9adIabmnmOXeaIQwvsuIckO06JHnzfz9RpJjtd
wrQpxH2VjwLZsgyP/+R28APxkfWIbv97T8vhmezayU37Q8F0l/T8v3E6Leq3tuUPou9K5jaLkmIh
XsmpJ7zo51zgl2brr0UbxWYn2fqKioCPpzzU36d911FxobG/HJJ5+gU6dOmDRpUmvj39mzZ7F792
78+te/Dq+0GlVUVGD37t144YUXAACXXHIJLrjgAmzduhAmzduhU//elPo1IG3Rzc1RLo4V7oSr/vff93u/
1/d+2zwK4NkSWQcBOu93cH3+SfAPr8pKXsvkSxpYxaPj9IQ9DcA07g3tdaXtu9UX0Zel8R+N4hY1
q6ipJ8GpsH39RyXiNNyEoVL6XXlb4L6feuVaS/bxTp8crF0+CbWm4oauNc7ppQ+lw9vmM5ct+vXJ
yGUwbpOWvfDYFjOwSg71WAw+nze5cisItUaPmcWAuV96S0PtCoyXFqcm+45TVCOGWIJNdHy8lDCA
hUpTjV8t5Q5M6NWfOmGUQaS0p1Fz1iVO33pkf9KZa03teifW/Uo/NTLat/l0rk7u9a6tPRrvtoFU
mO1ZL/wzmPvmUbNFr+91v+aGAZNi1u+f9YxqPVrolI6nVK9aH9HwM/ucP/97wxEs36kNJ3oXcdI9
Q1oecxK11Tcs/Zsco5vscb7NzsWGeta8WH5hGAZrJ//348+uijePfdd1tfmzFjBnr27Inbbrst4P
0lJSXRLF5Qeyo3cLhOQNcUEQUZzSFfj/9Fw4kYX25o3X47o0dPXi0R6Pqz3ivPBFzDhR13
z+onIKIkQAjeL511ISRCzq60ZBRrOqz91TmYD797pQL/lcAGiQfO6tuY1462iSqjLIvTfp3OdK3z
ulRwOu7+izm4QKvscGIOAYUhJE/L57AxYedAa8PjSrCVvPJMp+F1YSSez1Tm8OiKdwvwtfcnEa6n
PDvQblYlcp9rQem9w5G5nTFPD3kgQRCQLglvytvm2asLcqsfW1G3SMsUhzltJnSF9TOr/e/CIVLM
dFehx6HHOk9Mj1sSY9hhcOJOHd8vP9msHiVMt7lZj53JiRUeeL34M24dzXjPgMOUrfJb/j4LTc7w
CEVUcP5/s1w71Ojtr8r7XeoOX3/3EiMeJ7ULzTq14nF6dyMQLAtnlI6ZrQO/ca+ayjB7njBWC5az
U/Pz/ozyNqADx06BDOnDmDHj16tI4CjCatDYtmH1wcrl2xquNKQ92PBduRFbWqY8qx3Jj
dUt8F9bqSezzz8+hpgy19benjUlGH3psDyNtQHjgAMZyqd9NiUplzu2gBcep3/6+46IDHRf8SWFa
fzqb0mlOJm3QvA6Ec1T3cJOURb6zSaSK5tpWHmSnGq9jtWM9zeG+ve2Fczn
TUc0uvVqFwo97k/fB
l0ioNfLJh9mmy0eqDNfB6UYkfL1Jgg79U9N8Q7HafrR/y5GsV6Ko8u1Nbh1J4zI867zvd3o5guHk
LNGPGty0KIvJ5vhynEgPp7RZApsapiIdiUWi4XEVl9Rsfp3LLUTOf2+f2IcoMe9bpIP0Maj0bdX+
WuCbnnbDMtkQNoulZjfZ8INQU4rAbATZs2YfHixThx4gQAYNGiRRg0aBAqKiowYcIE3Hvvvvfj5z3
8eXok1qKiowPXXX48PPvigdeORiRMn4te//rVppwAXl7tx8z9O+S0mmmeoQsHZkdkwXigxWLukCtt
4dU2t83pvmEHDvJWlY+lVNwGdcnee/YKl3sU/pIqDBdvqUK5vc597cPSXgva7Elna6ep8OBKVjUy
oDELho6fj81NYNEqS/7z2noTYMkTs2pfLOuiwDU/9b6fdeZ0LLZjG+701zCHhhSFuMy/e5IZqYlm
uiqKQGjxafDYi9F4a0xe5TjYrfhZKSkpKQvSRKcarmu9RybWu5BtUcm1ewczYuPy0g1pWON1i51S
zYK/1c7/Wrdy5UOo9y14/v3/ONhVDnLJaM2L1QSSzOg5ryGn0fLS53Y1uIBZzNHCNmpPV8qY3zaH
wPau4TZiZ3LgHluku4nxvOZ/hS+i69O3Ka5VozIh7CzdOhcqHv/V0EVNcxIo0Rsz7rhEvumUDNfS
LY71PkG9Xo9fygJo+o+f1wc4MeG/YYsYmmkfdXtc/ZkdxTtNLzeM1eb9C8BuC//vUvzJgxA4WFhb
j11lvx8ssssvbx4YNG6LSAJiZmBAwfinXfewS233PFGTHTNAwhdwBK1hnJVSSzhEDDWRvexMBCZqSkXUbzEXPm6CxAIPhgmCHAQnLEvNtXLdfbgmaBK9/mn
LY71PkG9Xo9fygJo+o+f1wc4MeG/YYsYmmkfdXtc/ZkdxTtNLzeM1eb9C8BuC//vUvzJgxA4WFhb
```

Mt50tLnPN7CC7YuRyXnxbxedLznqv0Xcp1ftrlO440T2vJhVrq+ZHGSDg5Opada1rKoOU+Qcr0ai
RW2xER6b1Cr3uN9BmoMCeywQOA8rl8dlel37O60nOYUn0tFvdXpcZ2vevdSp8bL/UJzbXl7777Dg
888IDiz7OyskLOO9ZTXl4e/vznP0ft72klF2BmDC6li1yuLWLZCgAAFWhJREFUUS9YA5X0M/QYoS
ZXtlCfq9RQ166yCfkGn2u5RCyKCKgMqRmVJBcXkYzYilSwRKymR6tXW0fA6+8fcQdM5/aeA6W/F6
3jlaPHCDMvucqmEY2YRpyzSEduJggCRnaR/94joVcjsfSciQicPhXtSr+WhyqjKtdy50Gpg0PLuT
FDo4/WcxbrXGQ1as+X1sYDfg/KzNrprETpu7TrdxxpntaSC7XW8yOJEa05Wm3dMlb1W192G90YK3
rmpnDrdVq/r0h/X67BX2lWnZbzIHcuPaKIT483yAw8CXwOC1Zfi2Xu1aMjW+vn2vVeI6X5KktOTk
ZdXZ3iz48cOWKezTZizMipF0ZQCnotDVRGjVAzw8g3teQSsSAE9rBqHZUUTDSmFwRLmHLrRMpVkH
afapR9/ZquyZjcJz1g7cdgiT9W37Ve05CN6tVSovc501J5U3oYUFpvM1J63cB9z5nS9KloVvpDPV
RFOtpc6UEr1HlQ6uDQcm7M0OgTztTvWOYiK1JzvowatRyPrDhCUum7tON3rEcjiJZcGE49P1zByq
V2CmS0Rt2bYQR6vIlVboo0j0Ty+3KNx0qz6rT8Dblz6V2v37d2Fuw5LFh9LVa5V8/Gdi1LT9nxXi
OlOVtddtlleO+993DrrbcG/OzEiRN45513MGLECF0KZ3VGT70wglzQa33QMmqEWixHvmmhdFPzVn
KiOXVVT3qsExlsLR/pOTBrL2s4I8zkNgwxolcrmrRW3tSu+amXWDZ4GknpPEY62lxtg7TWDg61zN
Do41sG3/ViYz31Ox6Z9f6ul2hOd7T7ubQyvRpBtORCPer5kZRL7RRIpbUJjagnhTsCXbquOGljZG
6K9pTy4nI3tpUn4vIMt+bR7Uqz6rRQmq0h/VtWWlNVr3q31qWn4oHmI/3tb3+Lu+++G3feeSd+9r
OfQRAEbN26FZ9++ineeecdJCQkQkYOLEiUaU1XLM8sCoh1j2mMTicyOlZcMQq9BjnUgta/mY+fqJdK
OMSDcMMYtIOwesxEwjaaTnMdLR5loapMPp4FD6m9LKuRkaKrxl8K4Xa4ap3+Eww3pakbJyvghG63
RHNZtp2Tn32p0Z8h4Qu5kkWjYWNKqeFO4IdLl1xUkbI+Iu2jNs1G4Ko2YAQSSk59Lqa6rqUe+Wyz
nBlp6KF5qfpLt164bXXnsN8+fPx2uvvQZRFFFFUVAQA+PGPf4zHH38cHTp00L2gVmSmB0aKPrmbmp
Ur4XqtExmtBXuNpua7NHLDELNQM+LRLszyoCYV6Whzvda+UzPda3BH5bU9AXPE0+CO59eLNcPUb6
2i/fBD6mlZSiPY5jr8ju3FynVDLbRMgTRibUIl4Y5Aj8a64qRNtGfYaN0UJpr1SDusqaq1rNL6ol
L9Vm7pqXgSVhbt3r07Fi1ahMrKSvzwww9obm5GXl4e0tPTsX79ejzwwANYu3at3mW1JCtdZESh6L
FOpPc1NdeC1a8fPTcMsQq7P5ia8UEt0tGy4fy+2g4OuakX7x9xq66cxzqezDwSWY4dlhewMy1LaS
htrqN1EXcis9A6BTKa9SSr1zejwQqdu9FePijY3xPP/TyWawcr/S0z1mWVKJVVTefyzd1TFOtwVj
oHelNdg21sbMSWLVvwww8/ICMjA8OHD0dOdSxcgyW048Bx
jZj1w8G1lxsvL1o9eGIVbBxofYiHS0rFGjbeXiYdPheiRIastKlXMzxJPZRyJLmXXtVGqhZSkNpc
11gi3irvTASWQGWqdARrtRzsr1TaPFujNOrWh32in9vU2H6/2mmJr1fFmVls5lK9XhokXV1XDixA
nce++9OHLkCESx5cQ6nU68+OKLcLlcmDp1Ko4fP45+/frh0UcfxZVXXXmlkmYnIhFhxChTOhiFWxs
aH2In0QcmIBy25eEhomSHjR6lybpZ4stLIEDONWLTCaJFo07KUhtLmOkqLuPOBMz5Y/bqKp7WD7S
KczrhYxWm0O+3kNoX5eZ4L/9Qw04G00dq5bKU6XLSoqhEuXboUR48exfjx4zFgwAAcPXOoUy5Ytw5
w5c3D27FlcdNFFmDVrFgoKCowuLxGRpcTTjcdMjQ/xKNIHJb0ftOTiUEQMLKLusWXzRRPVnkINc
uIRauMFokFLUtpyG2uI7eIOx8444Ndriur5FNqobUzLtZxGu16t3RTmAOVHrx/xO33HnaG6yeczm
XmHH+qatHbt2/HDTfcgMmTJ7e+lp2djeSeeeAJXXEBMMMKSURkZfFy4zFwPEWDfnEHIF0a53+24KIy
8kE5+Wuy2/EZBZRfvhRzrSwwTt81Oy1IaahZx5wNnaFYfOcfrimJFS2dcsDiN5hIF0a53+24KIy
Jw+QZ2husn0s5lUtkAeOrUKfTt29fvNe+/f/GiLAMTXiEcKTSke4mUjoFiJ1sOP3EiPAd
lJppi6bUVqN9eRvs4HzuBiPSJJD2ZZEoHij5bOOKU4fXZXpd9Oz1a8BtVi56WxIu1cJpUNgM3NzX
A6nX6vef+dnp6uf6mIiMiy4mXEI6ljtqnJpA+lkR692mawMSrK+MCpzC4j58y0JALFH7WdcXJx6h
FFfHq8IWBHc6tdg1qw89JYkXYuxzvVd40jR47gyy+/bP13dXU1AOD7779vDvpDxP8eoprWnmq/UV6HLyuv+aaqO2OTQqpkqa4tMvtKDOdDiHIOSbNi8mm/EXVWeTo8QnZW9tN02IUu5lgWiQW0jYVkDrvdXsZKIz+YZyMEIYVCzkS9fxeqM7DsPAgVDh3Y2B4kgZudFljI8DMWPCWLPu
FGXgf//gNg3gUx4u1LUWPUYiUnF0rIiR7ZARoEP1n9mHEaLpLmNVPiJb9nqVExSybqQqOO5+XnhTxCd
gNeO2bqPZjzL3PCcFnO4ejVMNaVdGKnxRXRTeBZCBZnx1j/jE7FMfl0EGDUXzIYy9QLg0xoTWkjAhh
0jNFNw3AGXyMZCBFMIxdyPcHrLh0hNe6B1+Mt/fYlTJb/w0g0hfDXPRKVCjRxMv/gBi0i40ChwGQ9MZSy
FREREREI9UpmeD64awUSv4c+ZHLU4kTYawl8dMlaEl97B0jM30vtOPvO7e9yyAA8uF/OpYlvDUbLTb5yV
0ZREREREpEfrYTgtNEQGhTAdlRCQXb4xXE/SJVbD3fQPWWb/HIj9H/+y7x65yF0/W5+HP5kgGH1jFMNs3
uUE20yqRzBCA3Ix/3WRhDDzBgfvtTOPhu2g7KNRdbRRf9K2FU7IdPIUu4Op4yoEVsaOCtD8Pq3WzflH
+CunEj9p08UZmIgj1V9sfOtSK3Dw3d0yrYYuB/OfRYe+rm8GvhsuTvCcWHLsY8V2upwXEbSf2q5L6Iqv
zwYqHnXsOhYxPd2uUemf0hJDDxwoDmNqe2nIMF4sPA7RB3U1ta2ISQWPF+9qZMC5UZvbMtTQGYzIEhQi
7D2cwN+H6sYDbAGzvT8bdfAnUEVIzMoYwxWi3NvWa6XMwrPL9VG0+oNydNPaDk7ZCjGkt5HVKXP0bsiQB
IV3RMhrzd+E3iPLwRCmK2xs/iF20TMqBXCqr+nRXh0X3iBcXh9Do4W6bn/5Hh2e8fhU1cGJQIiiZF3B
Go+fZ/eBDsRFI+LHbbLfCMx1pwpCyTgGrsOTB4PJ/qsiFk4nFbZnPPSWzrVb3Bm+K/fhWmrqk0SfUBm2Ii0
Z/W6vlcTH8kwXe9Ox6VfsQCl92tkjkwhBZmrezP7JedC2MD8zBGPrg7RFsdl6fNwYv5yPAyh4FswDsHQ
fe3l02T0eS1aflWeG0DthGwyS4JpuhpxcKRXp0SrYFtc1HhB4VQRU8xpZUHvqbJOAH4WHfu89XNvhpEIE
Gu/LS4PnU3uenb35rBe9POzjRvpOpPqGC5+lLWLSpWqQU9KWiF0MtHFXyo6Ek1vi11YYkGzBCT5IwhkRyz
bwZ0pTJj5QrN4FKJwZFFBAv75j1sgxh/I5QVZM7n3HUM2I8e31OBkDjnCxsbRtpodxcPffPbQuZxtVgYCiwyJRRVZEpUm8QJAp9C0VdvW/L
UydrtQhH7/i9RdoTk0iQZQzvRzfx3rbGHqTP3cX55m/Pzx1kbcfX7hMxy98JZNdV6zxD/RtYuPIQeZ
pZfVUU1E0+6RZb6pzqqKOC4zyTgUaGIaHGhvg5o/U5zf9xlOGvXfvXU3/OHuThUfPmbKYZlxemlfW
PJhfZCD+Op9TzaqzQ5eJpkw6DdElMpxOTWYu6mpFORIWx5PXfjFGFwfgQqXB0Cg6/LxxQp0T4lEZUU
lJK1iw7R09V31pHrPmE1TqFZaQzFg+zffFkv2uJSOvyUPNmgE8ghG32cABITRmEDpdC34h46YAMC+O
4zD4ai40pMwLxlT8mPM6Q/pyNI6FBW50ktS3T3FxZf4MFNvTPXkE5Df+PtLdqfcgvpTOOQtEkjDZn6
AvlvzcMYtuhxeIHZTE3f7bB7SdMP85pw5Iw+cy9IrbUNMN5Pd+qlWi2EFgOIrwL63j6bRTEDpjTqyXT0
1kYm/k1hn1ZV4+Jo/D+PXCg3tP/nH3Hd2RnZ8wP/8gDz9sAWTwHN6NVGoNbUnEyYhKMg4ScxEwI+
y9bprrqNBkBx0Sk92+IsfPmg/+OjkNGxvYjKfGe5VIsPs2FnKTeUs7++1Sbsy/fMxR4tA0v/ThCf+ca
OqUbLIBBtMhbPYqCJcbAIX2oM/F79JjbSxDq63W/WsBeuxpJP78Tt5zZVnzTVZ62X5MNfDeHNPH0IkN
ulhiNtmsaKGOjsErMbg5e9ehSpBgFtQbPXtwjXCyOcG7V8bUxwBhHhHRoVhE+DgRMNy4Sr2YTu3lOuD
ydCWzgnJbUI+SDNNQOvO7AQhmYoqOjfZBjxDBChPYKzQDaOWA6C/YgTTrL3SUUEXAiyOmFjNbMfjl1
Gbg9ABU5mGrSrAdtsUXj2a8FzqHbtHFDY4w8lpEq2B9FJXJbRlWeRu1fD9Z5nI9lS+IN6Jb1ZNRlQbD
KJpIozzPNrVE5SOrDoYlv4RkWnZSJmtWfFbixq9ZY4g54S5o8Jz4mDZqIg4l6tA58s2lXTnYeX0DFzHH4
b3KXUZTqvsSHmWvOHpB1/IXFp5+4gSaK2BBYnlLMmtcnMZIRFCN/yyQq4oyhSO8i6+tPnTKpqPOrKH4
mXZIkoqDZwXjE4ROUA3Xdw4U0CYLoUaTiWf1+KdT64ydEO+7jh20ubN4FIgj5ZwNb7OxQoKqSsX5
xGCy5Yeu7Zrqhl3JGoz8ttBDGK9wSpHs7uibfJcbFDPQ0+AVqpWQ5hpihf+2nEebG7Dkb6WPqDZsITRb
WQlMd8tb7ZLVjDY4bGJEPz7hjAVXgR74WWgaHXeDE+gPr5/SqgDRnudjg4nCvpLvSmNaGWhaf8ZX5
mOJ3QUgi+uG3/uakG7T6qJ5BqQSbzDSjDlkqTAgtsdCiNzXM5z/WpPuanwMhISv2DbLiu9nxQPmhp4j
5DB3HJ2Lhn5u5PDMZUHe5JvDwuBd4e3uLpWuLzmhpAeh9P66QvjRuozhvVCIuugIBoGfzKVW/HmlZOG
3DrKsxiBcmmVFTdFLnpRwTHBg2RK+6jRL/MOoBv4XOmD1DghPP/7FqxmtmBSD0kcqrqQf/i8QcrCcn
nxm6nCDXyIqDEsj8DOU4N+dcVTdxTp/lRwKrNwK0qmgC4qLRBukTJj8t7qjJm/hE8c1hPSZJeIfn0e
Q35k0FhpPq+/oZ8zhgFuUfUBfrsCAI36V+rzx3RGx1GOKmb8GefUwmCQNt6iCoJ9G4ss5gWk4gtwjwq
3ZaZB89khM0VyTmPhZmA3nrEqQoHGCnKnGVIvwn0rTgdFI5tHjkUXvMr0AFXYQszlAAfH42f3xbGSAHB

```
QNgERERETkh9N9iYiIiOwlIdYFICIiIiIiIiIiIuNwBCARERFRnCoud6OUU32JiIiIbI8NgERERE
RxaEpxhd9mH7fnp2LukMxYF4uIiIiIDMApwERERERxprjcjeUltaj1iKjxiKj1iFheUotPy92xLh
oRERERGYANgERERERxprTSA0HymgDgQKUnFsUhIiIiIoOxAZCIiIgozlyU4YAoeU0E0CODq8MQER
ER2REbAImIiIjizJCOLtyen4pUh4A0h4BUh4Dx+ancCISIiIjIptjNS0RERBSH5g7JxJjuKTjAXY
CJiIiIbE+oqKiQzgAhIiIiIiIiIiIim+AUYCIiIiIiIiIiIhtjAyAREREREREREZGNsQGQiIiIiI
iIiIjIxtgASEREREREREREZGOW2gXY7XZj6tSpKC0tRXJyMlJSUvDggw+ib9++aGhowJw5c7Br1y
4AwNixY3Hbbbe1/u7f/vY3rF69GgAwYMAAPPHEE3A6nQQCA7du3Y/78+WhsbEROTg6mT5+O3Nzc6B
8gaWJEPHz77bd44YUXcPr0aTgcDnTr1g1TpkxBdnZ2TI6R1DEqN3gtXrwYr7/+OpYtW4Z+/fpF78
AoLEbFw8mTJzFv3jyUlJQgKSkJAwcOxGOPPRb9AyTVjIqFVatW4e2330ZiYiIEQcDkyZMxdOjQ6B
8gaRJuPBw4cADPP/88SkpK0L59e6xatcrvc1mPtB4jYoF1SOsyKjd4sR5pHUbFAuuQ1mRUPMSyHm
mpBkAAGD16NIYNGwZBEPDRRx9hypQp2LBhA4qKilBbW4s1a9agqqoK48ePR//+/dGvXz98/vnnWL
16NV5//XW0adMGU6ZMwZtvvok777wT9fX1mDZtGhYuXIhevXqhqKgIc+bMwcsvvxzrQyUV9I4Hp9
OJhx9+GD179gQALFiwAAsWLMCsWbNifKQUit6x4LVz5058++6Ny5cwyPjrQyIh6mTJmC6667Dn
PmzAHQUpkj89M7Fg4fPoylS5dizZo1yMzMRHFxMZ5++mls2rQp1odKKoQTD5mZmbj//vtRWVmJhQ
sX+n0e65HWpXcssA5pbXrHgxfrkdZjRCywDmldesdDrOuRlpoC7HK5MHz4cAiCAAAoKCjA6dOnUV
9fj82bN2Ps2LEQBAEZGRm49tprW0/i5s2s2bt2KCy+8EL169QIA3H
zzzfjss89QUVERm4Mk1YyIh27durVW3Acgf//+OHr0aPQPjjQxIhYAoLKyEgsWLMBTTz0Vk+Oi8B
gRDzt27EB1dTXGjh3b+nfat28f/YMjTYyIBUEQIIoiamtrAQBVVVXo0KFDbA6QNAk3HrKzs9G/f3
+kpKQEfCbrkdZkRCywDmldRsQDwHqkFRkRC6xDWpcR8RDreqSlGgClioqKMHToUCQnJ6OsrMyvZy
U3NxdlZWUAgLKyMnTq1Kn1Z3l5ea0/o7gQTkZWV1fpzsg494sGXx+PBypUrcdVVVxlfeN
KVXrHw7LPPYsKECbxJW5we8VBaWooOHTpg5syZGD9+PCZPnoy9e/dG90AoYnrEQpcuXfDAAw9g3L
hxuPHGG/Hiiy9i+vTp0T0Q0oXaeAiG9Uh70CMWfLEOaW16xQPrkdanRyywDmkfesRDrOuRlpsC7L
VmzRp8+OGHeOWVV2JdFDIBveOhubkZM2fOrGZmpt+aUGR+esXC+vXrkZKSVwsq7xekVDx6PB5999h
mWLl2KgoICbN++HQ899BDWrl2LtLQ0nUpLRtIrFsrLy/Huu+/iqKgIubm52LJkCx5++2GsXLkSLp
dLp9KS0ViPJC/WIKckX65HkxTok+bJLPdKSIwBXr16NWC0uWLGldWLdL9t956C0uZWL
PbqVMnv9bYI0eOKP6svr4eZ86c8evNJ9MmgCTNmzEB9fT1mz56NhARLXiZxSc9Y2LFjB3
bs2IHRo0dj9OjROH78OKZMmmcJ1vixEz3jIzc1F165dUVBQAAAoLCyE0+nEoUOHonU4FAE9Y+H999
9Hly5dWjd5GDFiBORq6nD48OFoHQ5FSGs8BMN6pDXp2c8sB5pbXrGAuuQ1qdnPMS6Hm
m5u9KKFSuwatUqLFmyBDk5Oa2vjxo1CqtXr4YoiqisrMTGjRsxatSo1p9t9t3LgRlZWVEEURq1atws
iRIwEAl19+OQ4ePIj9+/cDaGnZHThwIDIzM6N/cKSZ3vHg8Xgwbdo0eDwezJ49Gw6HZQfJxh29Y2
HmzJl47733sG7dOqxbtw4dOnTA/PnzMTg4cGDBwEAX3/9Nerr69G1a9
foHxxponcs5Obm4osvvkBVVRUAYPfu3WhsbOSurxYRTjwEw3qkdekdC6xDWpve8cB6pHXpHQusQ1
qb3vEQ63qkUFFRIUblL+mgvLwcN9xwAzp37b09PTW1+fPn4/MzEw899xz2LNnD0RRxLgxYzB+/P
jW9/z1r3/r3/FmjVrIIoCgoK8OST8T7YNNmIC8vL+rHR9oYEQ+bNm3C3C9O
nT0aNHj9Ze23bt2mHRokVRPz5Sz6jc4Gv06NHm+vXrF5VjovAZFQ87d+7ESy+9hKamJjiQlJW
Hy5MkoLCyM+vGRekbFwiuvvIIPPvgASUlJcDgcePTRR3N/QnRcA8GhpQVV
WF7OxsDBkyBFOnTgXAeqQVGRELrENalxHxALAeaUVGxALrkNZlRDywHml1RsQD2uRlmoAJCIiIiIiIm
0sNwWYiIiIiIiI1GMDIBEREREREREREkY2xAZCIiIiIiIiMjG2ABIREREREREREREREREkY2wAJC
IiIiIiIisjE2ABIREREREREREREdkYGwCJiIiIiIiIhsjA2AREREREREREREREREREENvb/A/Ib259z9t
zOAAAAAElFTkSuQmCC\n"
          },
          "metadata": {}
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "sc = MinMaxScaler(feature_range = (0, 1))\n",
        "df = sc.fit_transform(df)"
      ],
      "metadata": {
        "id": "mN_QttdiNfB0"
      },
      "execution_count": 17,
      "outputs": []
    },
    {
      "cell_type": "markdown",
      "source": [
   "**TRAINING AND TESTING**"
      ],
      "metadata": {
        "id": "FGke_VXXQedm"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "train_size = int(len(df) * 0.70)\n",
        "test_size = len(df) - train_size\n",
        "train, test = df[0:train_size, :], df[train_size:len(df), :]"
      ],
      "metadata": {
        "id": "Sudww0wuNe_V"
      },
      "execution_count": 18,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "def create_data_set(_data_set, _look_back=1):\n",
        "    data_x, data_y = [], []\n",
        "    for i in range(len(_data_set) - _look_back - 1):\n",
        "        a = _data_set[i:(i + _look_back), 0]\n",
        "        data_x.append(a)\n",
        "        data_y.append(_data_set[i + _look_back, 0])\n",
        "    return np.array(data_x), np.array(data_y)"
      ],
      "metadata": {
        "id": "xPUeoQzYNe8w"
      },
      "execution_count": 19,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "look_back =90\n",
        "X_train,Y_train,X_test,Ytest = [],[],[],[]\n",
        "X_train,Y_train=create_data_set(train,look_back)\n",
        "X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1],
1))\n",
        "X_test,Y_test=create_data_set(test,look_back)\n",
        "X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))"
      ],
      "metadata": {
        "id": "pXyfLTmPNe6Q"
      },
      "execution_count": 20,
      "outputs": []
    },
    {
      "cell_type": "markdown",
      "source": [
        "**LSTM LAYER**"
      ],
      "metadata": {
        "id": "DlWYJeeeQjRx"
    }
    },
    {
      "cell_type": "code",
      "source": [
        "regressor = Sequential()\n",
        "\n",
        "regressor.add(LSTM(units = 60, return_sequences = True, input_shape
= (X_train.shape[1], 1)))\n",
        "regressor.add(Dropout(0.1))\n",
        "\n",
        "regressor.add(LSTM(units = 60, return_sequences = True))\n",
        "regressor.add(Dropout(0.1))\n",
        "\n",
        "regressor.add(LSTM(units = 60))\n",
        "regressor.add(Dropout(0.1))\n",
        "\n",
        "regressor.add(Dense(units = 1))\n",
        "\n",
        "\n",
        "regressor.compile(optimizer = 'adam', loss =
'mean_squared_error')\n",
        "reduce_lr = ReduceLROnPlateau(monitor='val_loss',patience=5)\n",
        "history =regressor.fit(X_train, Y_train, epochs = 20, batch_size =
15,validation_data=(X_test, Y_test), callbacks=[reduce_lr],shuffle=False)"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "id": "WII6cXOpNe3T",
        "outputId": "ac95c5d7-7914-4d05-ca17-6edd2379451a"
      },
      "execution_count": 21,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "Epoch 1/20\n",
            "212/212 [==============================] - 23s 88ms/step -
loss: 0.0047 - val_loss: 0.0251 - lr: 0.0010\n",
            "Epoch 2/20\n",
            "212/212 [==============================] - 17s 82ms/step -
loss: 0.0122 - val_loss: 0.0478 - lr: 0.0010\n",
            "Epoch 3/20\n",
            "212/212 [==============================] - 17s 82ms/step -
loss: 0.0115 - val_loss: 0.0505 - lr: 0.0010\n",
            "Epoch 4/20\n",
            "212/212 [==============================] - 17s 81ms/step -
loss: 0.0163 - val_loss: 0.0461 - lr: 0.0010\n",
            "Epoch 5/20\n",
            "212/212 [==============================] - 19s 91ms/step -
loss: 0.0193 - val_loss: 0.0461 - lr: 0.0010\n",
            "Epoch 6/20\n",
            "212/212 [==============================] - 17s 82ms/step -
loss: 0.0174 - val_loss: 0.0605 - lr: 0.0010\n",
            "Epoch 7/20\n",
            "212/212 [==============================] - 18s 83ms/step -
loss: 0.0275 - val_loss: 0.0047 - lr: 1.0000e-04\n",
 "Epoch 8/20\n",
            "212/212 [==============================] - 18s 83ms/step -
loss: 0.0040 - val_loss: 0.0032 - lr: 1.0000e-04\n",
            "Epoch 9/20\n",
            "212/212 [==============================] - 17s 82ms/step -
loss: 0.0029 - val_loss: 0.0021 - lr: 1.0000e-04\n",
            "Epoch 10/20\n",
            "212/212 [==============================] - 17s 81ms/step -
loss: 0.0023 - val_loss: 0.0017 - lr: 1.0000e-04\n",
            "Epoch 11/20\n",
            "212/212 [==============================] - 17s 83ms/step -
loss: 0.0020 - val_loss: 0.0016 - lr: 1.0000e-04\n",
            "Epoch 12/20\n",
            "212/212 [==============================] - 17s 82ms/step -
loss: 0.0016 - val_loss: 0.0015 - lr: 1.0000e-04\n",
            "Epoch 13/20\n",
            "212/212 [==============================] - 17s 83ms/step -
loss: 0.0014 - val_loss: 0.0014 - lr: 1.0000e-04\n",
            "Epoch 14/20\n",
            "212/212 [==============================] - 18s 83ms/step -
loss: 0.0013 - val_loss: 0.0014 - lr: 1.0000e-04\n",
            "Epoch 15/20\n",
            "212/212 [==============================] - 18s 83ms/step -
loss: 0.0012 - val_loss: 0.0013 - lr: 1.0000e-04\n",
            "Epoch 16/20\n",
            "212/212 [==============================] - 18s 84ms/step -
loss: 0.0011 - val_loss: 0.0014 - lr: 1.0000e-04\n",
            "Epoch 17/20\n",
            "212/212 [==============================] - 18s 86ms/step -
loss: 0.0011 - val_loss: 0.0014 - lr: 1.0000e-04\n",
            "Epoch 18/20\n",
            "212/212 [==============================] - 19s 87ms/step -
loss: 0.0011 - val_loss: 0.0015 - lr: 1.0000e-04\n",
            "Epoch 19/20\n",
            "212/212 [==============================] - 17s 82ms/step -
loss: 0.0011 - val_loss: 0.0013 - lr: 1.0000e-05\n",
            "Epoch 20/20\n",
            "212/212 [==============================] - 18s 83ms/step -
loss: 0.0010 - val_loss: 0.0013 - lr: 1.0000e-05\n"
          ]
        }
      ]
    },
    {
      "cell_type": "markdown",
      "source": [
        "**MODEL TRAINING**"
      ],
      "metadata": {
        "id": "UR1fiR-nQnPK"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "train_predict = regressor.predict(X_train)\n",
        "test_predict = regressor.predict(X_test)"
      ],
      "metadata": {
        "colab": {
```

```
"base_uri": "https://localhost:8080/"
        },
        "id": "iMy9uuDuNe0J",
        "outputId": "3177cb11-0d0f-4fb7-cefb-91c89a609797"
      },
      "execution_count": 22,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "100/100 [==============================] - 4s 27ms/step\n",
            "41/41 [==============================] - 1s 28ms/step\n"
          ]
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "train_predict = sc.inverse_transform(train_predict)\n",
        "Y_train = sc.inverse_transform([Y_train])\n",
        "test_predict = sc.inverse_transform(test_predict)\n",
        "Y_test = sc.inverse_transform([Y_test])"
      ],
      "metadata": {
        "id": "-8jlSRIINew6"
      },
      "execution_count": 23,
      "outputs": []
    },
    {
      "cell_type": "markdown",
      "source": [
        "**PREDICTION**"
      ],
      "metadata": {
        "id": "-9FXpty9QtPZ"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "print('Train Mean Absolute Error:', mean_absolute_error(Y_train[0], 
train_predict[:,0]))\n",
        "print('Train Root Mean Squared 
Error:',np.sqrt(mean_squared_error(Y_train[0], train_predict[:,0])))\n",
        "print('Test Mean Absolute Error:', mean_absolute_error(Y_test[0], 
test_predict[:,0]))\n",
        "print('Test Root Mean Squared 
Error:',np.sqrt(mean_squared_error(Y_test[0], test_predict[:,0])))\n",
        "plt.figure(figsize=(8,4))\n",
        "plt.plot(history.history['loss'], label='Train Loss')\n",
        "plt.plot(history.history['val_loss'], label='Test Loss')\n",
        "plt.title('model loss')\n",
        "plt.ylabel('loss')\n",
        "plt.xlabel('epochs')\n",
        "plt.legend(loc='upper right')\n",
        "plt.show();"
      ],
   "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 390
        },
        "id": "WZJaxXOePtpE",
        "outputId": "99ce555f-59ae-4a96-8e66-cc6bb33de164"
      },
      "execution_count": 24,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "Train Mean Absolute Error: 2.3165036988408305\n",
            "Train Root Mean Squared Error: 3.285617879896689\n",
            "Test Mean Absolute Error: 2.3989636110004624\n",
            "Test Root Mean Squared Error: 5.289593391043789\n"
          ]
        },
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
              "<Figure size 576x288 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAjYAAAEwCAYAAACkBzu1AAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nOzdeXhTVfoH8O/N1nQvXWhpWWTtIoj
sIlioAm4gKo7gyuDgAIKKMDPiDztoEdFxXFAWV1QUEaSKMCIiWimyDrKN0mLZhNKN7luSJrn390d
p2pt0S5s2afL9PI8PnJN7b85JsX17znPEYqLiyUQERERuQGFsxtARERERE5CgMbIiIiMhtMLAhIiI
it8HAhoiIiNwGAxsiIiIyG0wsCEilzV79mwMHz3Z8zefJkTJ48udXvS0Ttj4ENERE
RuQ0GNkRERQ2GNgQERGR21A5uwFE5DxZWVm48847MXjwYCxbtgyrV6/G3r17/G3r17UVlZib59+2LevHk
YNGgQdDod3nvvvPezatQsFBQXo2rUrHn30UYwbN87UUlbevevXrjjjzs
xadIkCIJgc8/OnTvx6aef4ty5c/Dx8cF1112HefPmNdr2w4cPY8OGDfjf//+6H8vJyhSoUfj
LX/6C0NBQh31Grenb0aNH8emnn+LUqVMoKiqCn58foDMXjwYDz55JOW6ysqKvD5559j165dyMn
JgSiKCAoKQnR0NKZOnYohQ4a0SX+I3JHAs6KIPFdNYNO3b194/
odDoEBARg4MCByMvLw48//giNRoP3//32//giNRoO
1a9fixRdfRGVlJYYNG4aKigrs3LkTJpMkTJpMJ77//PgYMGDBmFFgB5nslkwhNPPIHDhw+je/fu
GD16fkqeeFVgZfX1+cPn0ahw4dkt3z8cc
fY9WqVQgICMCoUaOgRRxnAbEheHn1PFxcvLDRmuG1uBB7VUcK/QiNz8CBA+jcz2WBCCjeebWs66t
j8uXLOHbsGG699Vbb21dfyH/97ttcdw5Hd0ugcbRJC3Hy8SqXW9cnRmBvn+NOZeDcPHqFfHtPaT50
9+/fjqaeego+PBPD264vCNRxXX5/7vcdHLe3zj1Z1PzPUNXWUoDzFPP7aBFkOVE2Xdj//NE7OJYt
omrr71mdRkDi8qdpNzE/mh33r33ZnXRFF5IQ+4xg7ODAwzLuSJ9Hd8Ex9uN1FgHzjsLMI24xjpWSgiX
AfaeBDyN1Q+uZaCWD2nnHyjbGChOBD0ayPWuKHSN1xF+cinxn4zIGuqm4hr4XtrSaYbiH6wLqGVp
sfWDl4BDD+1A6cyvYoGbGywhoiILq95w3Xbre21y5PDSDt8Sw+UOR4QhxV4VfWo3XfvGrs1I0aK9
6qXDMhSRjMtuTjBvNiu8Ds2+p7czbLxCKjfPLJaLJ5yJtdw+BbuFgsQtfv2UjNNDaNN/LhyaiCkHf
wz01OUFhAPwJ1cg4sBn9F5ytqkZ5HpRxawAbHawd4k9gqX0VtJtSEqTMGkwVZ0kw6xPXKn2dPmnn+tSPq
m8/SS32bwbhw/vJoMtDV9p24z6JaTvNzEddZeALrPrQFdRENaGxMOrw2RaOTHPAzVfuD3lQ0gkPm
4vpGwfSxBi9h4qT4XHt0+OWFbXqqMT+e5wyWs18S9pGyfmxMOajd4YbmtKVUfsh2CUkuy49g8w7Jp
qQP9h+mz8JAshLC0ASMsP9Rl8Io+lcx+F15pSPQLUDkc3FTMkZKVpBMGGD0+bvwDMzMoBuNCD1/Px
H3lJOtoH8Sx7oR8D7Gd5T8D3H8MDP9w5uWcxRyXLOZVXuCJQg0L3XLkwlVHrz6c7Krvxmo5Ix/jumsw
3l4x9Oapl37XggwdCQ98z6SRA+KbYq03QeFbAw7QLjbmwd3UHZeAD2z0hP3uELyJu39w6yzZHbuWgfoe
Y9yAk4GPZUwGfG2nGIIBN+lbaEBkG6qQh+WTowamuF5XR3RxqVyKhJ7L/0qxlx7Ea/JW58h9Nbj
uPkzvq4LtmQyw4XPdMdUoZBZSIQ6NAuK4J01VlQNb4d6TMcqv7w40wv0hIWABmfmbbLBDMsw6xA1
cH2zBwQGsjVqC4ZcrkYHQvN+r2RaxQBsTaFcnXYeNG4GKGEcCy5ZoYKnwy8Z0t4Edtz8tbDoIK0h
c53c0k5l6TtXBWFm8WU1wZH6iFPIxJOkBXRgsx6GncCvXGSYTWq9Lae8ujqLqi5mV2f2TU9CmhX3
M3Wch7gj+zsU6A0dFlr9aIkaPIQBmxCDSHnOPZI+F9o+oWmlLQAMLBRkKTewnsxl0sBHmKqGl1zxB8
O6QVcV0++c3wmmA6qRF1b1ao1gSU9mAqgQ6L4Q1hEQEM9gIhsUrpzOdLm1Ab23LD0bOsTAJhyD9U
8Z8rk4FpwrqvGVZO1ksBHWlh9hiIWbA+2dOvxppKJFqMMz0oNZFvsq0u1oq0+hCv76J5QWs8pHf2tlou
O4gzoqavOKpkIKOMPQJfTN2ldWXozZfYU31h3BKD3mYrHs5dvkNoEJOXFs+FUDX4hE3+8tR8H7+ZAI
7OsWbSMAEPuQLv1WSYbHTiFE7uPoc2wnD1JAYYdeZG6qcnG6uvlqFgLgsZcJCZDCDugaIqvd1BYjjJ
7U4d27PyJAE0pAP4vSY6qK2/8DKl9ouLsu5lt6WqGiCdgmX6eZCHDE2qPoIx3TGCrD5RXRX8KBnYBpvMz
4fdXrUY4sgY5kCIUl3ubqiN0/lZRaPfWsxW9YqTjAEYmTjNAVxxv9//sVb+7Yi+Z0xZlk6m2Vz8kjQkqV
tcp7lbvlHc/PyrXb09rXXS7wGxotRdxwbCXN6KSbq8ZCkexnqq7ufRMGSDYsfUc6NkItnocmUn/iEAa
WrLwPLsD8x5KYcS+c8gfGCmaDPk0WV7rGXt4QE2FZfhzkaXmQtSaQXggYwLKMRyvxjkNYI4dGs8V
RVu2yGw5YE3Jk6ZmmsrgZWiZt5SkR+58bQ+Zn+kyOcQ6dGMt0dSyeJqq0QYiGK4vzl6DXbMTvf/4
Ewru9tgZxH4eR5H2cG0rMRTE6+UUR2VSrMjPMKLrpVYYs1HDodutUmJgNgfjfY8OFHbQZx03fmFr7c
O+kTIYZcnU9JyZjHcebxjFVfJjyaC9ZWgomk2BZ+/jqoT+uN7Utuke7Tsz2sE6z66LHLkRvBvXyVrd
h1FaGT4GatU3bNbwmLYUYX/MWV4rx2+hMp6gI7wDbM4o26GrJTYsTh59+xgwt2Sj2eNG5lfZU3IyTd
AKkrwFyYDvB1tunzQYgpKbUi4hQ4nMRZwyrOj2RHfnrcvy2pGvvJqdCYFJk85OxBCdf5J+tktInknqyxAiH
gEoeQ+KXWpW5N6jfo8Plt8fbHtAIbgjfAS0KTYJnSQxGvyg+4gsobUF1YVBmAK4VlfXIOyD2sJ+lG1O+ie
EZjdBZlmLZqdfXoU+1nKX2rjt4SIv6MJJZe9ITaRoNwN3atvJVq1qrVxXbzyZZZdTDHQFyMZIUBkXvSnrX
Xou9npf0/CFSMudHt3Tkq9yVbkfzkmdm3R2XxsLg7Qdx5Zv2rsgaL0rnLAqJRGnYxU8HUaetd8TdZ7A
iIkLvtNTCWlQiFtagXKumPQoR1ap6aNZU2hhNRi6LlPpYe7dyQ2gUbq1KC3qjDGuoc3MsfQEUtqHx1HZ
pSkUhPeXx3RoZbqPM41aMpHX0hlSj2xiNsmy9JmNhzkIsYF6LMm0/2J5KS3mRWohwNeXdEm70UX+gmf
jPVgXRgHiOh9WJ/M5K/s2sqfJ6w/MwiY2pqfHBeEdS8K0CIxOqc6NQrl/bEZlZqpf+jZJTqVvj8JIq
3RhJc8LJzGnwbbVlnjmvKKdSo77BSbJ8bs8dbR/FXGcFJV2tGh7+Pf0YE1vnqLGRHvK+S0y5B9ZZRcv7
5ZdRxEHnLQoGQ/5wtPtIqDEgzlYx7eSGx98Xj/JCXY37Qni2uAdxM5U3IyqwhRGIZ4lzPgE+tvhqA5Go
Y2LF+ExmE5OrLFbNj2obv8X2jHXVxBe2TSKlcbYNMiDbq5fIMSuTRx4fMFhd+RvUX9TJnP5V8xBNiaiM
9HwM3Ln7pb2ozkKT9o3k2CzjaTtkIRYSYSnexaiB19k7NRH4/fCBmKXa43flVnDTxoZn8CWOjkMDnXZV
AURXw2dHDnxxTtsXyFsoeqB2NGCCykRN39BsIeQP7ID9nbxpZ+5xnUxEH+h7gfqRcUMRi8Lo8QtFuQDO
IcyNNUZvjYrqVc5MxbCuF8pJwx8qm+NZzTA13W8ciW+CQn6vhN0Yh5vKUnpXT/X2GcOddMbb2Rb6hFrn
KMCRMrEF82bSOEoONBgDBaY46MSM+JvXs78F9Dg7WNVa3zYtgB9Rgq5JeJuyPndrTU+Va5CaT3WZ3GmZ
K9S6DpCkm4wCwLGGIC0LzUCcDMGnFI16PxTldhFj8JQBFUeB8FQdO8nDN75YuTF5nmDUDWE6zTqnTeFg
5uBhwFbW9FZGpNgMfQmd8aZiAd0vhGaJL6QDy6cshQCqqqH6Lxul9R5Ws3rP0Vjm7NwjaTiRuBjmvQIO
BCzX1yjAj9KPXBWNBiHhyxOJA9myL4yIhY9TRCz+0XjEoMemNzjTF79Dq9vvPA1TOGg2CU2Fs+5Htv5K
lQAJZa6aM0YlvbgvHs8ZyNPOHQ2l2t8O6glc0Bw6uFsVhuuNsiRN2bBpnpCO6j9R7V+Hh5QTVhUYrl6f
jQ0WhyRgjxfm0Ar72F2KyHGojfuqpVbhDTYWVCfB+39+Qs1dOfNfy+HmHHVhHJ7sh3qw0vuJV2uVIi1l
Ll0ZhYLk+H6PNV3pRcOa2hHGhp1JhpaYvThnRYHUQAs8N+LBn8OwC8c7/kfbqTvkHtXXnq5zCj6lgvk/xc
vz+UFdg6pQTKXubbDkCT9+cO72xjYx+06jK0vGIZD3dPp8jNvz6nHmO5dEgjkzYVR/fQ7q2ub3hj9jd6
Yk3Df/PkxJtG5VoJMfWeTYjz0Z6DbaMg0EKyWj+tu+3LcA99wZ9N5d8eqMr0+Xfq8HSvZQ76n+MOxSwR
MPZwHdb8YadVC6kRgfyYYt6x44X2QDzLyfN4jyCt2jr4M29WQFD6hMvEJ8DvKhtRuTKqJR3gHlfqhI7iu
k/vlE9PGXj0WODK6Sd/OZWHgJfqoejj2EPUHfb+TqUR9q+m5gk+ZOxQf37v9cRdR1GYWOLNEpXnBoZjC
0B83SiW1EH0nWahTSSIbvzRTE8aApL6JMy6XVlvDR4hv9oXSEuuTD1eLHzM61hiY94XVQv2yNWV4ZjQU
oxEbuBwu8YgF7OtrTU1zz5cTrkcO4SyJhnBv3tJ4Wq9pw4jdeiamcLyFuSsj4T0bTYkfBlrNBcXpNW+
U+wF71IXqKYWAtzUzyJ+ZSh/PjzcFyrWwmvx6RiLqkmJF0dmlTxDtw6X9+ReLHsFoeaPpeQ7DHenixkw
ABeByBrmT3vyHLCQZSJ7uTKZZ3+p/2dg0+6Jg+2xB8D9mZ4i3xhV24VeCfI4FU/rj6lzWeUYvtj5nwpg
oF8DLbF0Y3rNhWzEhP2SHbqF8hs5HY9oEYNbN/MO4oJbMajYL+M14wcWaFrdsVrzo5fCpr2ZC/fyrnv
fqcd6pUl2tMTLVNCDIKOD+aL1WnjP9Z78ni2kBiQVAlLpIBjbRx9qT36Eey5yw9Z7Lbcl/BlvgVwoIq5A
p0BbBtq6AFQYZhsXlO9SCftD4gCIQvQQwjGuSIaFLk6iDWJgeJ5YYKB28GT6p4HTmYEgfWsh8KgW5ltI
RnoCWH4M/tzyhELGWj6Y82I2wlj1g1UOt2d71djp6i3FwLV1l2J07uxLubIXS01ZSZqthrQFJzkLqx
/DCaBsBNOBPdrVc09HugX5kDzTzBfx9HQBFIUEaUOMbvtFdPKsHK7eBNGRmehiFrTAUZRDXsCWzqX8Le
ZIb6GTaJ2gbtzOt00WfpKvVQ/GNFfBDiQFCbr0SZAqJL5ygqJffHRxUyJqYMEkNK8AIoKOWtlmoMPX
FKqk3EBhNlh1OwSw2tyzDAoubnw8BIFbMTi6qPtpF1GO6o7GP1JIEjvJJ9HTv9jGJCEwH06TFtXldq
/Nvss2OCvsgNkW/xovQ2/ATbMGx0qPs3HIdUxJFAzbY/LsNEjM3kTnzvbDmtVKMjVtX8Y7UQWW3hn
d5olDEH5zPGQMGYsVMOlL1TIcoSpDfTAqBZcQZ4Lfj6xq4QszQnm7+/aEuhSKJXtlxYwVG13sYTmRV
```

```
1NWpGfWoWC1+6dEnW9xqVlZXQ6XRQqVT1LqImovpxKoqIHOb+++/H/v37sX//ftx3330YPXq0Zdo
oLy8Pt912m2xTv8jISMydOxcrVqzAQw89hHHjxiEgIAAHDhxAWVkZ+vTpg9OnT8veY+jQoXjiiSe
wcuVK3HPPPbj++usRFRUFg8GAnJwcHD16FF26dMH69eud2rdXX30Vubm5uPbaa9GlSxdoNBrLXju
BgYG46667AFSn2j/99NOIiYlBz549ERYWhtLSUuzduxelpaV44IEH4O3t7dC+ELkzBjZE5DBqtRp
vvvkmNmzYgO+++w6bN2+GIAjo3bs3Zs6cWe+J2Q888ABCQ0Px6aefYvv27Zadhx9//HGbzfxqPPj
ggxg4cCA2btyIY8eOc+ePfDx8UFYWBhuueWWendEbu++/fnPf8bu3buRlpaGw4cPAwA6d+6Mad0
m4f7777dsIBgbG4vp06fjyJEjOHjwIEpLSxEYGIirroK8+fPb5O+ELkz7jxMREREboNrbIiIiMh
tMLAhIiIit8HAhoiIiNwGAxsiIiJyGwxsiIiIyG0wsCEiIiK3wcCGiIiI3AYDGwfIyMhwdhPaFfv
r3thf98b+ujdP6299GNgQERGR22BgQ0RERG6DgQ0RERG5DR6CSUREVA9JklBeXGbTarU
oKSlxdjMcRqvVwsvLy657GNgQERHVo7y8HF5eXkbBoNM5uSrN5eXlBq9U6uxkOIUkSKisrYTKZ4Ov
r2+z7OBVF1BCTEYr049DmZjq7JUTkBKIodqigxt0IggBfX1+YTCa77nPqiE1WVhaWL12KvLw8qFQ
qLFy4EMOHD7e57tSpU1i2bBkqirg6+uLxYsXIzo62vL6xx9/jG3btkGtVgMA1qxZg6CgoHbrB7k
hSYJ2zVKoDqciFoDeVAHT2InObhURETXBqYHN8uXLMXbsWEydOPnzsXXrVtkwkwmiRJSEx
MxLx58xAfH4+UlBQkJiZi48aNEAQBmzdvxsGDB7Fu3Tr4+PigtLTUbYbhyHmUJw5CdTjV5UtZ89wU
DGyKiDsBpU1HFxcU4duwYJk+eDACIi4tDjx49sG/PtlI6enp0Ov1iI+PBwAkJCRAp9MhPT0dQPV
ozZw5c+Dj4wMACAgI4NAhtZrmmw2yspB7CehACwiJyD3NmDEDzzzwAB544AEMHz4c06ZNs5QrKiq
a/ZwXXngB+/fvt/v9n3/eOOnDZik5OTg6CgINnoSmRkJLKzs22ui4iIkNV16dIFOTk39g2CgvMPPl
56NatG3Jzc3Hixm88P+ureW9re+jJyIDQWOaJKNnPtCmnXdmjVrLH+Pj4/Hq4/Hq6++ipAQ+b3l5eVQqRr/8f6
3v/0NAKDX6+1qp9lshtFotPu+1igtLVUeXp6l3Ldv30av79BZUWazGP3LvSNsL
CwnDLLbfUe09TH0hLZGRktMlzXZW791e7/aN663sH+EDs4779ruHuX19r7K97a01/S0pK2m1pQ0v
fpyYLavbs2ejduzfS0tJQVlaGL774AitXrsThw4dhNBqhUqmwYMECDBw4EAAwe/Zs3H333ZgwYQK
ef/55KJVK5ObmIjs7GxEREg835ObmIjs7GxEREg835KpRJqtdqmvcXFxXjllVdw9uxZiKKIMWPGYPbs2VA
oFNi6dr18PlUoFs9mMuXPn4oYbbbmiw3lpAQAC6devW7M/EaYFNREEaoQqvUbnhkohsgt37SYEeC
ZI7suPDwcOTk5srqcnByEh4db9G3Q9n/wqLBiAVrAQaQR+N//ehOrK33tcU+Tk
Q+1zdzi0iImqe06dP4/XXX0dgYCAA4IEHHsC8efMAAdOnEBiYiK+/vrreu9bXz4h/HXUfHt/FVybx
YsABfffUVpk+f3uz3fvXVVxEUFIQNGzbZAp9Nh9uzZZiIyMxJ133onXX38dmzZtQlhYGERRGVlJQA
0WN9aTltjExQUhEGDBmHLli0AgLS0NBQWFuLyy+WjSHv/ghJcbGwuNRoPU1OqFnCkpKdBqtYiJiQA3Hb
bbdi7t/oHkcFgwOHh2UZU0T20Gz/vMHXhILcdmwJEZF9br31VtnU2eHDhzFz5kxxMmzYNL7/8Mr
zs1FeXl7vvvfHx8fDx8YEgCBgwYAUAuXrxo13vvbvv9emDRpg4cOAAAGDFiBJYsWYL
169fj7Nmz8PPza7tS+tZw6FbVo0SKJVh06ZKre4vKKS5ndos6KKgg7eejt3N5KTk5GZNZbjTJLmoKgGQQ9mqvmZ3J9XnrpJaSnp+P
o0aNYvHgxgxJk2ahAcffLTB+tZyamAATFRWd955x7RamAwd+Ruauun7yrvuOnz55ZdYsGAB9Ho
ov/sCgrl2IyhJECBIkqWsyM+p7zYiIpdTUVEBpVKsLAwAMCGDRuauKN1rrvuOnz55ZdYsGAB9Ho
9vvnmG0yePFZ2dmIiYlBTEwMzGYzTpw40WC9I3ToxcNEDlNeCvP22Rx79Fn/tDz5fHGm6xHZ9pHk7c
E1FH07t0bEydOxP3334+goFH99ZPNDhv2Rx99hC+++++MJSvuuuu7Bw4UL861//wn33ddc
AZDLhhRdeQElJCZRKJXx8flLBo0SKIolhvvSMIxcXFUtOXUWOYZdDxqb9eB68v11rKYmAwdEvehu+
Cey1lktYHFW9/AzQy5OoO3PHr2xj21721NiuqZiFuR6HX691uk1p7vw48K4rIofm+2RZfHmP0E
KDoOkqZZ13FvSVQGX9C++IiMg1MLAhj6fe8y2y2EshJLWfL2hTFhEiAIkeI6y67lOhsiItfGwm9k
E9bcbZVXGG+8AfKqzBsTgcNlrzIwiInJtDGzIoUnInJDG3Ioro6kO/SQbhZ2HUahgn3FNbDpUHNgouICYicmkMbMh
zSRLUVoddmkbdAimo9twVMcRqxIZTUURELo2BDXks5F48Ylyx6qRCb5q5UZuYHLRYTpYWWOylLgo
CqW6fKrpFC5QewcsSGiMi
1cR8b8lia7Z/JyqahYyBFdJXV2Y7YMLAhIueZMWMGqqqAFSnsvfq1QtKpRIA8O6771r+3hz/+c9
/EBsbi969e9f7+j/+8Q/s2793ejl3wgnE
a/j58+HCsXLkSoaGhlq9Xt/sZ/3/3nnPSKjsj2SsaAVFaRF
QZQDq7G9DRJ7Db/rYNnlgtLYXZbMadd96Je++9F6+oh4vXXX8fBgwehUqk
gCAJee+01/Pzzzzzz3ZNN97AsWPHd+fD61Wi59//hm
rV6+GIAgwmUyYyNm0a7rrrrgbr2wIDG/I4QtYfUB7dK6szTrlF7+fEHjTzzm1vbBovkh03+TvEMe2rkJnvh
EIqJmKy8vx3PPPYfnwcHBkCcQJjzzyCK6o6HvVanhw4EFs2LABS9qUGZRzvJU5Ua2+a+rbf4Tk
7774bEyZMsOv91q5di4KCAnz66acoMWzHyaajKUF4Bnz66a46acAgKeffhpr167Zc6f1GntWZ
D5HE0326UHW5p7tEP5rghDV5fFRgsKyu4zoaIXMiJEyeQk5Qk50Dv//973Mz4i
KioIoikhKSsLXX3+N8vJyaDSar3fgQMHcPfdd0OlUkGUmHkXlCk4cOAAAGDEiBF49dVX8dFHH+H
XX3+1HIXQUH1b4IgNeRShMA+qb7+Fl8bCn6KKvFsCpUNmlrfrPovlhx7PgKzbjuwvJvlCcv1HHRFo75vBXgHdATuTeF7lpSNdwpvmWpL8uOWs
asGEDjh07hiNHjuDhhx/GSy+9GfgJVfyxg7p6vKqmzyoN8BrRfJD/1+mO42ZnR/WOQo25kNaFHhEzbrGvsaiOhJUY5/hFn+Q5Fs1BSLVOrYkj+Dq0mC+HB/HSmeJ6r8+IH
m3n77bYiiiKCgICxZsgSlpaV48cUXYTKZoievbsidtvvvx0AcNddd+GNN97Al19+ialTp9a7eHj
i/m4zBLPJUhbDo2AeKNxg3onTzClWjjtmZtP3zUbkCq6IAh5Xg4YhX4FNP4fXXX8cDDzwAoDqwmFxq9dNhFbBgzZo1+OOPHzYDe2PGlW3SRc21+WgTw
Hjh3Yu7d2TeKoUaPw1FNP4fXXX8cDDzwAoDqwmFjBgBgzZo1+OOPP6BWq6FUKvHUU081+Wt8WhOL
iYqnpy6gxGRkZ6Nu3r7Ob0W46an/VWz+BV/IHlrIY2AmV//68yQynnB1foc+GFZayOWYgdM+saOS
```

```
Ojq2jfn1biv11b63pb0lJSZuuBWkL9U1FdXT2fh24eJg8Q5UB6p3JsirjhHualbbNERsioo6DgQ1
5BNWeb6EoK7aUJa0PjAl3NOteY4A8K0oougyIZoe2j4iIHIOBDbk/swmabzfKqow3TgZ8/Zt1u6j
xguRfOwwqmM0Qigoc2kQiInIMBjbk9lT/3Q3F5WxLWVKpYbz5HrueYXNmVAFTvomIXBEDG3JvkgT
1N1aHXY66GVJQSAM3NPAYm1O+81rdNCJyfZLE/Bpnasnnz8CG3Jry1/9CeeGMpSwJAqpum2r3c8T
gzrIyFxATuT+tVovKykpnN8NjSZKE4uJiy348zcV9bMitqa0OuzQPuaFF5zxZn/Kt4FQUkdvz8vK
CyWRq03ONHK20tBQBAQHObobD+Pv7Q6WyL1RhYENuS3EmDaq0o7K6qtsbOOyyCWKIfCqKIzZEnsH
e0QJny8vLQ7dunn1IL6eiyG1ptstHa0xxgyH2imnRs2xHbBjYEBG5IqeO2GRlZWHp0qXIy8uDSqX
CwoULMXz4cJvrTp06hWXLlqGiogK+vr5YvHgxoqOjAQDPP/88Dh48iE6dOgEAwsLC8MYbb7RrPzo
kYxVUh1Oh/OVnwEuLqtvvgxTZw9mtchgh+wKUv+yR1Rlvu6/FzxOtAhshPxeQpEYPzyQiovbn1MB
m+fLlGDt2LKZOnYqTJ09i/vz52Lp1q2w7aEmSkJiYiHnz5iE+Ph4pKSlITEzExo0bIVz5oTJ16lR
Mnz7dWd3oUIScTKh/2gb1zzsglNXOG6sO/ICqO6fDeOs0wM75TFek2f45hDqr6c09+sLcf2jLH+g
bAEmjhVClB4DqPytKAb+Otd06EZG7c9pUVHFxMY4dO4bJkycDAOLi4tCjRw/s27dPdl16ejr0ej3
i4+MBAAkJCdDpdEhPT2/3NndYJiNUB1OgfXkBfJ9+EJpvN8qCGgAQTEZ4bX4f3klzoPgjw0kNdQy
h8DJUe3fK6oy339e60RVBsJ2O4jobIiKX47RfzXNychAUFCQbnYmMjER2drbNdRER8oWbXbp0QU5
ODmJjYwEAycnJ+O677xAYGIiHHnoI119/fdt3oAMQ8rKg/mkbVHt2QFFa1Kx7lH9kwPv52TDedh+
qJj8MqDVt3ErHU+/cDMFsspTFsEiYhsa3+rliSDgUWX9YykJ+LnBVv1Y/l4iIHKfDzznMnj0boaG
hUCqV+PXXX7FgwQKsXLkS/frV/wMnI6NtRiPa5M5sQmHECoUd2I+DswzGIG/H4O
6stxSL5jN0Gz7FOb9P+DCxOmo7Nrb5l6X6a8Vpb4SV//wtawuc2gCCs6ea9VzMzIy0E2tRWiduoL
0X3E5MKLBezoyV/36thX2172xv+6lqdPanRbYREREoKioCAaDAV5e1ScsZ2VlYcyYMbLrwsPDkZM
j3zMkOzvbMooTHl47PdC/f/ftG/f+ruF8rQ7Vh8M9fuS8QsjVHtp2ZwhjjJgwUf6Ayyr+/vVxjXu/ePqjbaBaT4j5JE/V2zh93ya2kxRRmnHvFqKJiNuMQbMx4aDLvAr0kKdYc6DHzR
3uRrGhEkwDU+AVuMFQ2kx8OmbUB/8UXadd342+n38Mow3/wlVdz8CeFWPrrlCfxui3rYeyivrYAB
ADOiE4LsfRnAzTvFuSE1/1b2igSOplvrOghlBLvo5tIYrf33bAvvr3thfz+O0NTZBQUEYNGgQtmz
ZAgBIS0vD+fPnMXLkSNl1sbGx0Gg0SE2t/oJJgpW4UwCrMxeUuaReKPeIGcjRgHYs4+NIMRcm/tKJcA9ZBQh1yStY5IZ2jDs4xGFJrT+dxJZdNqEZQVU6Pr2p+D4Q0RFYw4uZ9LIEx
cWxxQZvUzd5jE2VtbN47HBSkf2k6aPkIa/SbA1Cf+Bx5vTpCiHZt2jLXlqd0JHfVBH8EBGhP4kL4
```

```
Zs2cP8vLycOedd7Z9ox1IefIoBKPRUhZDIyBF9nBii8heYghTvomIXE2HzisuKCjAqlWrsHr16mb
fk5GR0SZtsfe53XbvgHedcsFVscg8fdqxjWpDbfU5uqr6+uunN6JvnbLh0h9u87m4Sz+ai/11b+y
ve+nbt2+jrzstsImIiEBRUREMBgO8vLwAVC8mHjNmjOy68PBwy+hMjezsbERERCAtLQ1FRUWYMWM
GAKCsrAyiKKKwsBCvvPJKve/b1AfSEhkZGfY9V5Lgc/6krMpv7K1t0ra2YHd/O7iG+isE+ADra8s
+FSVu8bnw6+ve2F/35mn9rY/TApugoCAMGjQIW7ZswdSpU5GWlobz589j5Ej5OpPY2FhoNBqkpqY
iPj4eKSkp0Gq1iImJgSAI+O677yzXvvvuuygoKMAzzzzT3t2xi+LCaSiK8i1lSaOFOXqgE1tELWG
d7i0UFwAmI1BnfxsiImpfTp2KWrRoEZKSkrBp0yaoVCokJSXB29sbycnJyM/Px6xZsyAIApYuXYr
ly5djxYoV8PX1RVJSEgRBcGbTW0VplQ1l7j8E0Hg5qTXUYmoNxKAQKIoLAACCJEEovAypc6STG0Z
E5LmcGthERUXhnXfesamfMmWKrBwbG4t169Y1+by//vWvDmtbW1Idt0rz5m7DHZYUEg5cCWyA6qM
VzAxsiIichjsPtzOhtAiKs+myOvPA65zUGmotMdR6LxumfBMRORMDm3amPHEQQp3NBc09+kHqFOr
EFlFrSEz5JiJyKQxs2pn1bsNmbsrXoUkhnWVlRSGPVSAiciYGNu3JZITyf/+VVzGw6dA4FUVE5Fo
Y2LQj5akTEPSVlrIY2AniVf2c2CJqLeupKEUBp6KIiJyJgU07sj7N23zNdYCCX4KOzGbEpiAPEEU
ntYaIiPhTtR3xNG835O0LycfPUhRMRgilRU5sEBGRZ2Ng006EnItQ5GZaypJSBfPVQ53YInIUrrM
hInIdDGzaiU02VMy1gLePk1pDjiQFywMbRQEzo4iInIWBTTuxWV9zLTflcxe262y4gJiIyFkY2LS
HynIoTx2XVfEYBfchhXAqiojIVTCwaQfKXw9DMJstZbFLd0jhUU5sETmSGMqUbyIiV8HAph0wG8q
92Y7YMLAhInIWBjZtTTRDeeKgrIqHXroXKdR68TADGyIiZ2Fg08YUZ9OhKCu2lCUfX5j7DnBii8j
RJP8gSGq1pSzoKoDKcie2iIjIc6nsvaGqqgo6nQ6BgYGWuuLiYmzZsgVlZWW46aabEBcX59BGdmQ
201ADhgMquz92cmUKBaTgcAh19ilS5OdC7O7XyE1ERNQW7P4Ju2zZMpw7dw7r1q0DAOj1ejzyyCO
4dOkSAGDDhg1Ys2n2YNBg4c6NiWdlDK41b71zAbyi2JoeGyDRiFglyge28ntoiIyDPZPRV17Ngx3HD
DDZbyjh07cOnSJbzxxhvYvn07evbsibVr1zq0kR2VUJgH5YUzlrIkCDBdM9yJLaK2Yr2AWMGUbyI
ip7A7sCksLER4eO038T179mDAgAEYOXIkQkJCMGnSJJw6dcqhjeyorDflE3tfDfgHOak11JasU76
5SR8RkXPYHdj4+PigrKwMAGAymXCuHl5YWKigrHtbADY5q357BJ+eaxCkRETmH3Gpv
Y2Fh8/fXXGDp0KFJTU6HT6TB69GjL65mZmQgJCXFoIzukKgOUJ4/IqswMbNyWFNJZVlYUcCqKiMg
Z7A5s5syZg3nz5mH69OmQJMkmC+qnn37YbBG1JZupKG7SR0TkFHY
HNtHR0fjjiiy9w4sQJ+Pn5YfDgwZbXysrK8Kc//UlW56lsTvO+diQgCE5qDbU1qVMYJEEBQRIBAIq
SQqDKAGi8nNwyIiLP0qINVYKCghAfH29T7+/vj2nTprW6UR2eJEHJ9TWeRaWC1CkEQuFlS5VQeBl
SRFcnNoqIyPPYvXj43Llz2L17t6zuyJEjePjePzxxzFjxgxs2LDBYY3rqBQXz0JRWLt4VNJ4wRw7yIk
tovZgk/LNdTZERO3O7hGbFStWQBAEAEjBkzBgCQm5uLBQsWQKPRoFOnTlixYgX8/f0xceJEhze2o7B
O8zbHDeaUhAcQQyOgzPjVUmZmFBFR+7N7xcY9PR2DBtWOPxnz77bRr0j/HHxo0bMXr0aGzevj
hjexomObtmaRgq8woLiAmImp3do/YlJeXo1OnTpby/TpbDvZhbFLd0jhUU5sETmSGMqUbyIiV8HAph0wG8q
SpUuRl5cHlUqFhQsXyvbEqXHq1CksW7FixEdHQ0AWLNmDVJTU6FQKCCKIiZNmoT
777/f3m45TlkxFGd+k1XxNG/PIIZa72XDqSgiovZmd2DTqVMnZGdnAwBKS0vx22+/Zx22+/4cknn7S8XlV
VBUmSmvWs5cuXY+zYsZg6dSpOnjyY+fPnY+vWrdBqtZZrJElCYmIiIi5s2bh/j4eKSkpCAxMREbN26
EIAh48MEHHMWfOHADVQdd9992HAQMGYMAA55ygrTpxCEKEFDnJdCyS
R2LRpE9avX4/nn38eAGQZUdUAgAUmfPnpUdudCQ4uJiHDt2DJMnTwYAxMXFFoUePHti3b5/suvT0dEolJvhkTUUZTkRIUEmM9U2MxBZ+Ue
7JCQkQKfTIT09HUB1JlYNnU4HfmmFa7uM/yBN99
8E2q1Gk888QS6dOkCADAYDPjhhx9w6623Nv6JG0ZYdNvmcnJwcBAUFyUZnIiMjLaNBda+LiJD/JtylSxfk5OQ
gNjYWAJCcnIxNmzYhMzMTs2fPRv/+/Rt834yMjGb31R4ZGRmA2YQBx+QLh8+FdEVlG72nM7XV5+i
qmtNfRZUedbemFArykHHqFKCw+/cHp+PX172xv+7N3fvbt2/fRl+3AJDg7Ge++9h/Lycnh5eUG
tVltekyQJS9ebtaIjSNNmTIFU6ZMQU5ODv7xj3/gpq1ev8r9Ql1ymbFZWF0mhiE/ZWFQMxVGcmYcS9hD5FFCdMG2UpOhKUZ9cXN7/6IafnTBL6nVSYB2PN0d1SIW/SJ1PqwsXW4F/GGMfD6aHJ3Cq4nzO3UMAzt4Yy8mJ36g7K1mtj9DGhcVmF3wrdT+kxtnA3lhfBDYPn+cKYTWyhtEf9yZpnBAjWe1lEMnCvWu5yEQ8/MEAe1Ja2pxZKUAWTURq/t+Q6AOBT4gD+PaI+1AMTuEnJpYORxHAARr9YlKxZ9SbW4GS6bOpqyJjnDqnjEzUUOmxY1I7E6oS+rmUI9Gd5KGUWIOyxTYcMu8b0SqfA3VdmG6BMm06TiKC2HvdKFJJ9aAT5bIKxPOeZdhvdmXP51e4qUMSC4wW4IlLdLMRZtIyZohUkK8JUwNRA/wpSW6jT0+dwDTgBhyiWVaqjKdB6bOqYUaUTQgJZ8DCYVGCbHUJhaCgEL+8IhN2E59ivECLqZ5KThq0K2RwoxnJ6yMtCNMkZ2Wd0NNVxJF4lNUVrwC5RIIjEzZiHNfJLZjXbYnUlPMjzHhe6zdK4gQjdWBYWAbUCSVZJe2sZ1lFiOcxT/oMtzdmLXdSF9zrMl2d7C8POaAVLxzaH+LAiKAjXSzTh3ViiwhrVqoVcBKKCJRIbmVa6Ez3rwXWfl93FYfJBJJ0+43NYl9QjAP48ZBuD/ZAirV5R/1ZYx0UAZmeg5yR9pQKxUOB2L6X4BtfHLm6Gn8r6gxAw9nNEBxdQIEFZyjsl4A+4FGl8nrsYOaZKkE5rxQX1rAGvOt+8DGeGnGqlKq0iAv5dw+cgZhgvnH7w9++WMbqS5Sd2/ZQQTZl+pMxV1aiPB2v5fTZ8hh2DeUHQ7zfgcfWYAbM6Z2MMWbVJRqqmnlfGQ2/X8ndN5kPS+3+vhgDCwdqZ6mcr2XN8yG1szsPOe8mzBEzdl2u27gXUTQNPgKvPcXyBrV9WyWeO3GdGtLHMaqzEHUWP5y3vEhaa0UC6yUV9M78XZKiZ2rMO1LZmAe+vhDWAfcfK/RCYCP3pAZjhRVbJ/RQ97hF1fwkoGbDXySEvPZX+XKzvGGuVQyLOMoLkUrkXxZ2xhjeuVSVbJWcPYLbzSU7jyHGLy5eGUGAtQ1kqxedT1LQhvjQJqH9/FgIw4QsMVTxNwtFyI1cE4GTvgTsF0nkHVrIcybYDTE8JsffZK5BdJa7cjKwUYYZ8zG26LkIEVTPhyzgn2jRDVRFuKYPgD1ohpgtyY2R42RjgFdqrq8WqwbUxhwdLBBTHD3Of37NzAkUXo8+uyf3E+V7QP2cUkXXTW2Abl4HuylLzAEm52H4iM8gFddPjMY7uIZ0FPHxQ7oDp9MlsVmsOVhc+nHK3iBncZ3xYmMHX7vQgVDXsh38hkUJ+LmUHSHH4cxovv2ozgWJdlPhrmwk6NJ0HUUc8yD92mJu54/f4RsBHGEq1bPLp2WVwwChmtKHyA+5A4XMFCHhkomL6CdrHQLzs8gb4j6owUy2/QiMF0BOnh1DmXJ0JeQ7DX72+KLahuQlUkPjgcERu12BL3xsc7DHTT1Vkuwxqt5Ox6zfI7i1jQmMxD8ZXEjcmNuFaJzSUL/wUl/xhNNWwvs3mGgqkgLMNUNQIikDC/mwlUK85BKFxSh5J8Wki5hUFi1TTlzu9zk1CfjtFd6VEUN1i3UfApcP0S6g3pfotlyhD/IMtP73zoB7tT9NZvnGwE3Gg5aBGwPcyKaPi6EaJ3CjBz3uH4vAHX7mC2hIFl6GW9Om4YPi6ve9/gSrf4NiGD6sYU6h5m3bzJrR/fsqbFHyeHUVnx0nY86HA5lbv0/GAZ46T7ihA+OrsZzgzdtXnFnfZ8OW4EGE8i3Mj6m6XgxI5mtI/vaAqIljzmjM2Tsf8EAX1tsWbeFZUjpv91rX3j2g1Sbc3t7zQKntDyfbEv2aTDSwFMkuwlOWzTQvZmOKeaqdjYCaqr0YNuPnvyGhAs+/2e6ybeVT+TqmbWLTbnoXGwxQEF0IA5idwT5BlXFAVkaXwlb1ZpbSJjqSNFdr5mhoOYY+sMbQcQ+Bg+Z0TvaGY/Dk2c82eC5k49tSE95uVnGqKVcGyi0hp6Z8uOTdqUfBKmHZ3AFBDHqGqUExFHt8V6JqFAuH5BmYpmG3sHk+VR4c/Ae6HrhKeIJ7SdqmnTkBMmm4Oxq5tD1X/YS0jxiczemE1TG/Ozd2zWrLdsP3TKehPJKd7xwrc8GpXODLp+NfbIbEWrMWOFwJJ8SXEgohmXnZPuF9HKpZ9EAGHPMTmUk8j5Mv9GQRpLOhfH5Fha+9eZYJlD/2Wl2wAQEORvJxC5kZMLQXvj1c+t+2QbZzAMOnJmuTzOgTXw1CJXqHI3b8L8cMXC3aVmQJd3DUuLmSb6j+nMkiRtFdCrE3jgWqoIPdCBaHMWzmgYbY09Fup5GMxQ+1/Ye7n+Cgc8ou+yZm9zW4J0YmxRnbHN5Vb3+dJrZoXEvQM1yeLEhKhpMMfq6dHzkEiz2RBqwBD/SrJaPUlMqtIJ3U4PUJSPWRmFqPFbD5WQ55PJDOd2ZNzlQiKzwhQ8M4ZAddKgYtLCVsBnZXmKkSd+FxgEghOYEG9iMBY8gP6lf2/pMJL/JCw2zD+uBy6Ia6Ejhye2MWx2XA3oXvF7J+XjPH26LUN4WMfjk9jhnmwMKgmzDE5Y6H+3b4xtRjy+A0sZ62pR9HnB+WQ3Je1z5zR+HEQnF/Q6F8OPtpbgHeo+9uJzzCIcRfFDMCeqlHBYWSijjQxI5xr8bPwzb6XlmyrRcI6DfXLY6+XUTgQDFuMa0WyWSG2VLLTYTDfNDO+c5xzrvTeU2dYcRprZZxVK/ChLXq1h3b+rGS/FYnKEu6y1+3QXzWMk2WxVhVDqR4fy7x7d+3jSqI9NrMROS0PAzHnKJXQv5jSzE6ySFUMkqWnZF6jBaVPHAQVnz9SCeE/BTUdb6d1HAPH1/nYWJ/7oL76s8jrt9kVKxj5t0mvNSv1EzdnNkvnmCMz5CiPmNnTUJGL3ZEYGwObZOzVGR5ubXOG3n8DbZrEygN2s3F+5KxeW/sdQHnn6cXOg8CnfwuDBajD1ElQO17wnFe1iXJzQqKPydQEJbU2zjgTJjVnMnULcbGbfhFWL9aBgnnSZhTqCjzAKoubONOIEgUeaNbqrYcf6kWcm0UzmZ/nvnOmzNtSdFKXqEIaN7NgmUa8pmvqUmfWv5ynFZHJOKxOF7LCeGZdRRQnsuFfKAkKZG0RUHZ7J6/kJGCrpW8YhIRn3VDLB4DAUgEr8lrs5xsk6FVzDBwGA+7ve1IxHW2LcV6m6Ra0OXPFb0aGJgSl1M09ywANmG7LXQcepWjSbCPZBKsUkT1sycQPHsrD66nrMSxC1ytpVnm5zVbzXcs/PTJWGIrkTbP8RQ0ftMaZ5PdPQqD1D3wKFuA6J97ZWQNE1hTrHGFO3GgTgUrYVONLDU+Qoyv9iXW5LZtNYXwFgaB1hEsGe/M/K6MD89V3n3TTAAr2X26Nb5ZHqhZ2lCEm+MszO5CaxpnHUGjQVLMB9iT9y6xu2UV3qJhMkhXqczhd7jwQ9OGS80cCdF4AGNRH7xhCOVPbJqoehSEj8yoGsfATKNJFX3eF9y2DMwUJjrJ8ruR1y4Bv6Dpr++0oAAAy2pOkAIAuDJ8+zdHyGwy3tVUwxiQHsG/RLu/ULk2D+ESGkN5wTRtH8HvjSX0mR9lvXZ6++yaXPSuv5J2k9a1jUrX0bqZ9cAzEdEHsdlpV5rEHU1VKOAIjmcbE5Lg6lkJ7xvhBrGJnzlEtLcKdzgD8hnIXVf/Y2kXUeiy3tw7Sbeqc4aAkwexD6QnhLWpJqB0XT/8IqtZDZerPxpYrSLKwCK5DcXJHRMpqQl3cnqrrlxQn6qJWrXyJdF9KkVUxVbPhtwgL0S65YZLoQaddUjr/RD4EFQQ5qQE/7JrLe0ygcI6eeR5eS+ZWzcZYJmahUi+h0yXXsRk8vv1Sbk5ofRIzZGwemLIXqQ/p7A3V5L4ATK0AsZZWVb1mbdxCrEJmN9mrq0/gMzCbqbdDGGJvcqM70M1zI4mztl0pZTEoHbzTnsQPhsXkdbmL+TaZqFYHCCcF+4TUgbMzVrsHlThlZDgJgCAkGDfdfTMzUY0T+VwE/gFOVUfEAzTuT5ZbxTVWyWjfdWmIUTnUDjDYgHJPBPKymj9kYg6DqIUiHOdpO2BD8DwAH2BfVBaLpI+3ZVpb3+t3Ah20NXdTcApS+dlajn0L4C/2FJsXKc3ezgAZWPbFQGu9K+w3PY7/yOpD/KZDrYcZhKmbrNwOPJwaEq+DfwRIACNIKPkJ9kG0vsEcXIhSXHYxC2Ey4j/rE3s4PJVSm+YDA7Sj7t0u2zgq8Zz3SPlOTRoEavfUMRzhRdYoMnP9dFXsyMFH7Lj8YtQ7G7/Zu8U6gXSm7qeZlUBqd3Oeo44yRP/C3WfT2BADOWHUbmOwnX/Y1b/Bm4nnFq9b12cRzZgx8Z4l4Pt0OMJNRFDUYXNdFerXfB2LD3kVQkCLmWX/EXSbkDv8lPbHj7bmBSQd0nUQnEV7gHAqMMXZ5v4/sYVxnG2jqJdG0pTRfTGJb2FgO7R2yNkOqRMeRQQvfmTq0t7UM0wrz+Q7JiZcEgO+AQP9Mv0HEU7Rh3O7OCWJWH7oCRaM8LOnRZpXvBd+z+RErZ1
```

```
a7TYs9oqBFMCUXU8nhsqnooSCXMzfJ8+C8lEJWDm6E7ae18kCm10MbIiIWqXFp3sToDpuNQ01kNN
QBEhWuw9X5eZg1yX5FNRzQwJwlb8K47vKFwunZlcx7ZuIqBUY2LSCue8AmHvG1Ja5voYASEEhkBS
1/2t5V5bA21wb2FwfrsHMWF8AQEyQCl19lZbXdGYJe3MbXodDRESN4yrFVjDeNg3G26bh3NHD6Ft
ZCLF7H2c3iVyBUgUpOAxCnQMwuxkK8LtPJLyVAlaN7gTFlQX5giBgXJQXPvq90nLt95l63MS0byK
iFuGIjQOY/AJhGjXBZtt88lxSsHw6qoe+eiuEJUMD0DNA/vvEOKvpqF2ZHLEhImopBjZEbaAiSJ7
y3V2fj5HhGvz1yhRUXWMivaCu839iTdo3ERHZj4ENkYNJkoTtlYGyut5VBbIpqLr81Qpc11l+oOz
3mfo2bSMRkbtiYEPkYBtOVyKlKkhWd7tPCXoFNLykzTo7ahcDGyKiFmFgQ+RAWRVmLDpUggvaUFl
9tKmg0fuY9k1E5BgMbIgcRJIkPLWvCKVVEi54yQMbRWFuA3dVY9o3EZFjMLAhcpCNZ3T47kpG0wV
tiOw1ofAyYG54QXBN2nddXGdDRGQ/BjZEDpBTacbTB4stZZ3SC0VeAZayIIoQihufjmLaNxFR6zG
wIWolSZIwf18xSqpq18RolYBXuNWZUfmNT0cx7ZuIqPUY2BC10qazOuy4KJ82enZwADSd5YGNIj+
n0ecw7ZuIqPUY2BC1Qk6lGU8fKJbVDQ/TYE6cX72nfDdlgtV0FAMbIiL7MLAhaqHqLKhiFFtNQa2
6IQhKhQApRL77sKIgr8lnWq+z2cO0byIiuzCwIWqhzWd1+NZqCmrx4AD0DVQDAMQQ+XlRQkHjU1E
A076JiFrLqad7Z2VlYenSpcjLy4NKpcLChQsxfPhwm+tOnTqFZcuWoaKiAr6+vli8eDGio6MBAEu
XLsXx48eh0WigUqkwa9YsjBo1qr27Qh4mt9KMfxyUT0ENC1PjsTg/S1kKtV5j0/RUFE/7JiJqHae
O2Cxfvhxjx45FcnIylixZgmeffRZ6vfw3YEmSkJiYiJkzZyI5ORkzZsxAYmIiJKl6eD4+Ph6ff/4
5PvvsM/zf//0fFi9ejLKyMmd0hzyEJElYsL8YRYbaKSIvJbBqdCcoFbVnQdmO2OQCUtPTSkz7JiJ
qOacFNsXFxTh27BgmT54MAIiLi0OPHj2wb98+2XXp6enQ6/WIj48DQkJECn0yE9PR0AMGbMGKh
U1QNP/fr1gyAIKCwsbMeekKdJPqfDNxespqAGBaBfkFp+oa8/JK23pShUGYCykiafX1/a97lSpn0
TETWH0wKbnJwcBAUFQaut/e00MjIS2dnZNtdFRMiH9Lt06YKcHNv1Clu3bkVERAS6d+/eNo0mj5e
nM+PvVllQQ8PUmHu1n+3FgmAzaqNoRmYU076JiFrOqWtsHGnPnj344IMPsHLlSgiC0OB1GRkZZbfL
+bfVcV+UK/b2gE7DzshJeCuC6Tmb08ZHQyJe+VTIyMiBJwD/SNSgy1P5voxEk/KNbKc6eqX8kppf
WD4F1yjm/HkOJqenfJwZpVdiD2uDm698LkaDObuQOx3KFr297Yn/dG/vrXvr27dvo604LbCIiIlB
UVASDwQAvr+ozcrKysjBmzBjZdeHh4TajM9nZ2bJRnN27d+Pf//43VqxYgR49ejT6vk19IC2RkZH
RJs91Vc7sryRJOJhXhbd+Lcf2C3rUrFh58zwQ6aPAuK5ajIvSYmykFwI0jhmQrOnvl2cr8VNBkey
1xUMCMWFA1wbv9ereCzjzq6UcpVagczM+u/tCjXjzfG16+JFSFbr17Aatqo0itzr479m9sb/uzdP
6Wx+nTUUFBQVh0KBB2LJlCwAgLS0N58+fx8iRI2XXxcbGQqPRIDU1FQCkpICrVaLmJgYAMCPP/6
IV199FW+99RZ69erVvp2gdmMWJXx9Xofx31zGLdvz8U2doKZGVqWIdb9X4uGUvT6LBsTv72MFf8
rw8kio2WxeUtd1pnxtwPyUZkhoQ1MQdUhhtqf8g0w7ZuIqKWcOhW1aNEiJCUlYdOmTVCpVEhKSoK
3tzeSk5ORn5+PWbNmQRAELF26FKMTT4sWLERwcjMWLF1ue+8wzz6B///////7
O6hY5UIVRxPqMSqw+WY7zZeZm32eSgJ9zqvBzThWWHC5FV18lxkV5YXpKAhfu
LUWgQLXUaBbDqhk5QKRofQZFC7E/5Bpj2TUTUUk4NbKKiovDOO+/Y1E+ZMkVWjo2nBxbp16+p9xv7
9+9ukbeRcOZVmvJdWjg/SK2Q7+1q7JliNSF8lUrMNqGxkh97MCjM++r0SH/1eCbUCuD7cC+O6Vgc
60YGqRtdl7cpXYusf8sW7zwwKQIx1FlQ9bEdsmhfYANVp39aBzUsjmn07EZFHcpvFw+Qe0oqMWPV
bOTadqUSV2PB1E7p6Ye7V/ojvooEgCDCYJezLMeD7S3p8n2lARknD6dFGEdidbcDubAMS/1uKbn5
KTOiqxbgoL8R38YJvndGcyzoz/nVGnqE0OFSNx/s3PgVVQ2pBVlSNmrRv45XP4UypGedKTegzwP9
tiYgawu+Q5HSSJCE1uworfy3D95caXkeiUQD39vbB3Kv9ENtJPlripRSQEKVFQpQWWLw4HzpeZ8H2
mHrsy9UjNroLO3PBozsVyMz5Ir8AH6RXQKIBREV4Y11WLCV29sOxIGYpNtaM5GkX1RnxNTUFZ+hY
UDEmphGCunkYTyksBgw7w8m7izuq075HhXkjNrv1Mvs/U469xzQuqiIg8EQMbchqjKOGrczqs/LU
cJwqNDV4XpBEwM8YPj8b6ItxH2eB1dV3lr8KjsX54NNYPelP1wtvvM/X4PlOPM6UNr9WpEoGULAN
SsgxYfMj29UWDAmyCqkYplJCCO0O4XJuqLeTnQoq6qlm3j4+SBza7LjGwISJqDAMbanelVSI+/r0
Cb/9WgUuVDQczZV/kr8VicHx7o6yObHrKXViXgpigtborS4qURwNnS2tGcPTkG6Ju5JvnaEDWeaOY
UVF1iSDgUdQIbRX4uzM0MbMZ11SLxcKmlXHPad3ukfRMRdUQMbKjdZJab8PbJCqz7vQKXzeUl
GqfF4f39M7K6Vnb3kKL0CVJgV54dZcX7QmST8nGPAziujOQ1lXqkVwOpmZEHVR2phyjdQm/adWVH
drpq0b2ZHERHVj4ENtbnjBVVY9Ws5vjynQ0OJSwKA27trMa+/H0Z01jSapeRI3ioB47tqMb6rFpI
k4UypCd9nVk9b7c01wGAGFJDwynWdEGfPFFQdLU35BupP+955kWnfREQNYWBDbeZYfhX+ZhbA+ebhUtkb
EmlYJPNDXF4/F+aF3oHP/OQqCgD6BavQJVGPO1X6oMIo4WWRCWc4F3Bjt2+LniiGd5e9TmNfAlfW
zTvvedYnnRhERNYSBDTmcWZSw4tdyvHiktMERn+N9cVfYnwRom3eguD25qtWYFhnDTJKWrd
rsfVUlCK/+VNRANO+iYjswe+M5FAXy02YlVqEfblV9b7eN1CFeFeVf74d7ePvD2kAWwotVUlD2b9AF
M+yYisofTzooi9/Pl2UqM+jqv3qBmVIQGn48LxsG7OmN6tK/HBDUAIAWHycpCUQFgangDwfqMj/K
SlTkdRURUPwY21GqlVSJmpxbikd1FKLU6/iDYS4FPbwzGN7eG4ZZu3lC5CHKkgi
h6LJdjxjXVb5YeE92FXSNHCFBROSpGNhQqxzKM+CR0Q6162z
qPe07h6d9ExFZY2BDLWISJSw/Wopbt+fjj3L53f
u6/tMTkcREVnj4mGyW6ZewNzt+Th02XYtWyQCu+NCUb/4Jbt+eKubDfps28BMcC0byKi5mBgQ80
mSRI+P6PD345qUWG2DWr+GuuL54cGeOgnfRETNwakoapZig4gZ3xbgzt8KbYKa7u2G3q7/sDe
LA5fO3gp8MT4E/7ouiEFNA8TQ1qV8A7Vp33VxOoqISI6BDTXp5xwDRn/ P2S4QvrmbFnsnd8b
4rtzivzG2Izb2BzYA076JiJrCwIYaVGWWkPRLCSZ9m285hLGGVgm8OjIQn98Uhtu6eUPVQbagN98UjDBvLhBuimi9xqY
wF5DsT9e2TvtOzTYw7ZuIqA4GNlSvjBIjJnxzGa+1+viJ239EZf4nxa7fDKjs8Hz9I3rX
nTQlGI4TSIrsfY532rTeDad9ERHUwsCEZSZLw8akKjNl6P9hKtpqO
EFkxHMe2biKhxDGxcXJVZQrFBhNSCaQt7FejNePDHQjy5rxiVVtMbXXwU+PrmECwdFggN/9W0iCN
SvgHYrGfiOhsiolrME3Vhb58sxz//W4IqsXpNS4SPEl18lAj3ViLCR4kKh3hJBYSQGPrvKRxhDzNUKijoVp4QRFt9hvTlN6WCVlAjeqtAGrSXjW8e/kVd+TV5LwZVlNqw/L5/bw1fX9lfTeS3LeuFpa36TH26VbVnZ9Un7pKgX5nZBpZ3Pq4yE4V5qB/6W2ZmWnaC3/DfuoQV9b77N2O6D/4Ipf8SXq3Lmwzv5zrF9h+GqBjKZYJbNvCAwSG14wGJb1H0RbaHeTYf3L2d/SlpuG9Io0DLAO/DdR2fC1+RaKpOMn6Lv/BO3+lz6+vSuZ0P3Ke2Lu/J4PTh0V9xsGWfpdWRCVkkmAhQy3/MtoQ0e7+fVL7eNN7hvmLvWIm2b1MO9m5NkKgTe/TGW4+AV9kwZ+mzVX3aBZXOlYOOC8O3yXY76uJoUuBl/tWybThd5tS5I9I01+oVjZU1SY+5yuwUhMYSt5Q7h3vZ2t7I6Ni8oPU6U8fN3PnxH7zE6LnXvrmhBkcM0eFzf3r5VBIbvd6aGZGxnKVTb9+b/PP+PWbNmAQBCQlzrxIoLjlGK6M0GsH2O+e+8UMdCSqNnXddTj1PqIXVvG8tR7m7fwiY/3J5pmbc+ad/aXGR0Bp+Y8M0iKfyxJOrd0ySpXUgr5ZfkwTkgWQTYX1WR3QXFo4JWBlWIyiB+7RdX6gCwiaKWvJiVAtYMktAe3kTDw9jPYKDfwWPqePc5MRzcK3AwKZxLM2CO3Ga92Pl4QM1ulIVY/HpK5kB4gIHHsM/qJjyrpdlR/+QPuHxTs6eZ5/Hd4mI/1JpTMFUoRBtPsmdUSs+0JHRoUMaOxQ6v7m0MOGWIb3+XGHu4oiBZv/nmRjGbO6yhRjbOgFXDcvFybOUNgEwSaCRf+s/cEmxGbhRJFgm2e5J5mVGnnbo0gNVxDjLZ4IxK//Y5aMCJx6E9Qwf1SDt5ZfPFXyLtEErt0/a0Ur5c7VqDQhPDP5mPuD+wOPeL7sWTVGQpjeFI7n+93mqDhd6U/0WFDUWZTW0VFbPuBYnQHe//a3y65Vi4B3p0TWB6bq95+7B/s0W/aIbP3W3YYzTQbKI5Q0Mc8XsCsF2A8tLh7R8fLTDNrDmhtCJ4H71n9x+nM/B+W3lgXwi8jWpLbBfS9NpoXKq4iLLS/OB3x5rG6PTp+yC5y2JgOiHS+Ad2u3q8uq+pe5S5TnRHIEzMEPTKRaWxkrRaixqdvfeXDNtEZfQNc9LVVLZbUUzWrTt4uo1qtbq3XPS0h1uUFb9xZxJFs0NTo+7K8grk1Iu0eGttGvBjwoLZ80Kz0DwhBEFbZ6fbVbB6A8eHuOxgmL4r4QWffFzvRhuF9vqaxX4dn9kyEVEKe6cCvkh3ZU1lqWTqt9AUv4kW/3dNgE2rzuMg7+QWQmlK5Z7Gw/cl+HBDiFdPl1xxo4d3GOo57p6eqVdRCS6eh/dsNBsd6B5Dx9gxz6ENI/A
```

```
o0fRQyiU95uwpQo5OtHntjh5avHF9EIJd9BTujsJ6xEaRnwNzA9c2Jr6etO+zpSb0Yto3EREDG1f
18akKLDpYYinrzcD5MjPOlzX+o9BHJSDCuzrIqf5PgS7e1X8P91Giy5UAyF9dPexiMEtI+qUUq34
rt3mWr0rASyMC8WBfH66lcQDJASnfQMOnfc/iad9ERAxsXNE3f+jw1P7iFt1baZJwtsyMs00EQH4
qARE+ShhECRfLba8dEqrGu/HB6B3IfyKOYpPybeexCnWNj5IHNrsY2BARAeAaG5dzINeAv+wuhNj
GS2rKTRJOl5psghqFAPxtoD923B7GoMbBxJDOsrJQYP/uwzVsTvvOYdo3ERHAERuXklZkxLRdBdD
XiTUUAvBJQjBGd/FCbqUZ2ZUicnRm5FSakV1pRu6VcnZldZ2+JYs2rujmp8Q7N3TC9RFeTV9MdrO
eimrpJn1Abdp3zf5CNWnf1gEPEZGnYWDjIjLLTbhnZwGKq+S/db8+Mgi39/AGAARqFOgX1PAzJEl
CSZVUJ/ARkXMl4KmuE6uDIZ0ZBqsA6E+9vPHKdUEI8uIgXluRAjpBUqkhmKrT6IXKckBXAdTZ36a
5atK+6x6K+X2mnoENEXk8pwY2WVlZWLp0KfLy8qBSqbBw4UIMHz7c5rpTp05h2bJlqKiogK+vLxY
vXozo6GgAwMaNG/Hll1/iwoULmD17NqZPn97e3Wi1IoOIe74vwKVKebTxzCB/TI9u/g89QRAQ5CU
gyEuBmEb2mZEkCcVV0pURHzM6eytxNU/jbnsKBaSQzhByL9VW5edC7NarRY8bz9O+iYhsOPXX8+X
Ll2Ps2LFITk7GkiVL8Oyzz0Kvl39zliQJiYmJmDlzJpKTkzFjxgwkJiZa9nW55ppr8PLLL2PChAn
O6EKr6UwS7ttVgPRik6z+kWhf/GOgf5u8pyAI6OSlQFwnNRKitAxq2pHNJn2tWGdTk/Zdoybtm4j
IkzktsCkuLsaxY8cwefJkAEBcXBx69OiBffv2ya5LT0+HXq9HfHw8ACAhIQE6nQ7p6ekAgNjYWFx
11VVQKDreFIpJlPDIT4U4kFclq5/YXYtXrgtkirUbsk35bnlmFE/7JiKy5bRoICcnB0FBQdBqa9c
EREZGIjs72+a6iAj5D4MuXbogJ6flv+m6AkmSsHB/Mb69KP9BNDJcg/fHBEOpYFDjjsRgeWZUaxY
QA/Wc9s3Ahog8nMctHs7IyHCJ577zhxofX5RPAfX2EbGsZzEunmvZHjbtqa0+R1flqP4Gm4Eedco
V50/jfCue3dcsAPC2dIMEZAAAHD5JREFUlFOz9fhfegZau0k0v77ujf11b+7e3759+zb6utMCm4i
ICBQVFcFgMMDLq/q3zqysLIwZM0Z2XXh4uM3oTHZ2ts0oTnM19YG0REZGhl3PXZtegfcvyoOXrr5
KbLs9ApG+rn9sgb397egc2V+lqRzYVlsOMFS06tl9JAldf8+1pH0bRAG5ft1alR3Fr697Y3/dm6f
1tz5Om4oKCgrCoEGDsGXLFgBAWloazp8/j5EjR8qui42jQWpqKgAgJSUFWq0WMTEx7d5mR9h
6XoeFVrsKd/ISkDwhpEMENdQ6touHWzcVJQgCxnflOhsiohpOXXG7aNEi/Pjjj5gyZQqee+45JCU
lwdvbG8nJyXjnnXcAVH/jXrp0Kd5//31IkEyfihx9+wEcffYSJEyd
i//79zuxWg/bmGPBoaiHq7lTjrRSwcVwIohtJzyb3IQWHQaqzKFxRXAAYqxq5o2njouSjMwxsiMi
TOXWNTVRUlCWAqWvKlCmycmxsLNatW1fvM6ZOnqU6e2Sfsc6bdCI+77oUC2MZ5SAD5M6IThnbn
Tr8dQqSEFhkAozrdUCYWXIYVHtfiR1qd9ny3jad9E5Lk6Xo50B3Sh3IR7vs9HqdWuwitGBeGWbt4
N3EXuSgq1PgyzddNRTPsmIqrFwKaNFerNmLKzANmVoqw+cXAAHuxr/1b61PHZrLNpZco3wLRvIqI
aDGzaUKVJxNRdBcgoke8G+2isLxZc4+ekVpGz2Y7YtH5PJp72TURUjYFNGzGJEmb8VIT/XjbK6id
fpcVLw7mrsCcTQ6x2H3bAiE3Nad81ak77JiLyNAxs2oAkSZi/rxjfWe0qPDpCg3du4K7Cns56xEY
obPmxCpZn1JP2vZPTUUTkgRjYtIEXjpTi04xWV3/YDXW3xQCrYpBjaeTQqyPVXDM8SDWad9cZ0N
EnoiBjYO9e7Icr54ol9V191Ni8/gQBGr4cVM9U1GFlwFRbODq5rM+7bsm7ZuIyJPwJ60DbTmnw9M
HS2R1IV4KfDkhBBE+3FWYrvD2geTrbykKJiOEksJWP5Zp30REDGwcJjXbgL9a7SrsoxKcwXwI+gR
yV2GSc/TRCjWY9k1Eno6BjQP8Xi7gwR8Ki7ah1Nkz7JiIPx8C
mlf4oM+GJ37QoNcp/eLw1uhPGt+KEZXJvNutsClqfGQXUn/b9M9O+iciDMLBphfwruwoXGOWZTs8
PDcB9fXyc1CrqCKwzoxw1FcXTvonI0/GUvFbIrhRRXCXPZpkT54sn+nBxBxx2BxSXs2H
uczXEPnEw94oFvFt25Ma4KC0+PFW73QDX2RCRJ2Fg0woDgtX47Rz3fGSdwnuHDHK3G/rRgRawBab09MYy7ipMzSB
ZT0UZ9FCdOAjViYPVrwsCxKieEPtcDXOfOJj7XA0pohvQjH9bPO2biDwZv9O1Uu9AFT4Yc3G/rRgRawBab09MYy7ipMzSB
YOiwQCgY11Axi994QO4VCUZRf7+uCJEGZeRbKzLNQ/7QNACD5BlQHOb3jqgOeXrGAt+2UZ03ad2p
27dqahfuLkRDphb6BKvQLVKOHvxIq7oBNRG6IgY0DhGqAf10X5OxmUEeiUkO/4GVokj+A8tRxCLq
KJm8RKkqhOn4AquMHAACSoIDYtWf11FWfq6tHdcK7AoKA8VHywCYly4CUrNqgRdYrI+8tnRb
oRr9AFfoE8lsCEXVs/C5G5CRi997QP/UiIIpQZP0BxenfoLzynyL7QpP3C5II5cUzUF48VHdGyJ
dvwCYe8dhetcY/FgUiUP+vVGhss3OM4pAerEJ6cW2OxOHabSIPZN/JeBRWf6M8lVympWIXB4DGyJ
nU1wZeenaE6axE6vrykuhPJMG5ZnfqgOeM2kQ9JWNPweAUF49qhN5/AC+B2CGgDSfKKT5RiHNJwo
nr/yZ4ROBKkX9G0derlLgcrZBNuIDAL4qAX3qBDr9AtXoG6hCd38l/FQCguXmKMbIhckV8AzAN
HwDxwRHVZNENxqc6ozpnfoMi+2ORjlJDQvzIT/SszZfUmKHDaO9wm4DnlEwm9sv5NJStMEo4XGHG
8wGjzmo9KQGdvBcL/v717j46iPB84/n1nNpcl5AIGc1MQqMtkNAWDSgKHkFEYEf9MBBLJcCgja
1aotVjk0UOF5o5dA0sYocT+UOpaVFsSoQEKqorSAIBRUQkqAxyeaesDPz+2M3k91cIIFcl+dzzp6
Z97Lvvvu9IVghRIuSwEaIkDTTMa/thXlL9wj7vvXklRS1uHHn3s+Cr3aJsHnZ+ywhcHxztmmlsXvqjj8HnMKc+hTsP9fGZmyi+Cr3aDni+CIv3bDsl1HtLq1qZ2+JkscH
JYuOinx0ZrDwBkE/gUzsAinHqRIdqMsFZCNFkEtgI0VF1jsAYdDPGoJs9MS7cvaDMtQpWOISTfsGo3tpDdAO3MS7cvquTpH0HI
vflXHl4bFDyrO8YOKc9z7/ad+ZadCrvK5wnONN+CJxxXUtO/bcVVZuKrc/M914XoKuCpUo0uIRni
QIjxIo3OQ8uwH1+SFByk6e7c1+TX1O8ltMiGuKBLYCBEoNN3zGHn33rhHjPPklRZ7JiafPenZZp/
03NLKb/oSDj0qv6dH5ffclX/QL79YD8WlO3E5Ol1Hk6ITL4cSld/KmnRR5y1zesiK9k1/apXfC0PQ
6n2cBeRUmeRVmnbKm0BS1giDvfrAnIDJLg+heUkSENyiKDNaICFJEBGtEBCsigjQigjVCdSRAEqI
DkMBGiEAWo55/QDM6wf9gQ6PoGCaqbstFWU1bNDPcqCaqoJL7mapFkK4y9Y
ER45OuPROVGoO3ErHVBpupWEoDQMNt9I9+z75bqVjUE+e/R7Nfk+V0vnOW+/MER230jmv/LdupXF
e8+yj6ziDgnCGOugUEkSnYJ2wkCAiQnS/QMgTGHkCp4ggDadDEaxBsK4I0I0jz7QZoiyJvnUBIwCdG
cJLAR4krkDMPs3Rezd1///MoKznz0Adfppn/Ac+4syrq8KycXEmZWVSRyFLfYZLcVNTfDjVhr
nlcO7rc7z5LuVTqXSMHzS1WWmpmMqHVPz7ms6lvdlajromjftAF0DzRNoKU1HaRq6Al2BGAFJr
yBEyaAl1T3nKFpqCstIQT+/d66yt0pXze70n7vlcpz6KCSmko5fnya0150p58T52ab8VWfpu6+fU
```

EcZYJllXPy/RcuvMpV/Xk2S8sML3nqekpTygsIHh/BMo0fdq07PLaaWWnDTB9yi3TL61qpf36anr
74leHmratxr1PYWHpDnA4QA/CcjjAEVRv2tI92+5l5YR07YrlCALdp9wnbb9PdzTq28wvlRnfA/O
6G1qs/YZIYCOEqBESSnlsd9zXX++fX1WJdu5sza2s7JOos6fQzn2DMi4+YTiQOTBxmCZQ92kx0fa
uvniVdq2pYcdVLdKLS5M94v+IeFACGyFEexQcYj+V5cc0oLwMVV6KKiuF8lJUeQmqvAzKvPtlpZ5
vVi4vRZWVePfL7DLKy1r0apAQom0cKTjPzW3wuRLYCCEunaZDWDhWWDhNm5njw7Kgorwm0Kkoqwm
AykpR7vPeS/MmGIZnaxoo06jJsy5Q5s3DNL35vmUG5SUldAoJttMYbjAMz5Uow5s2PfuW27OvvHm
aKQGZEA3R2uhz2zSwyc7OJi0tjW+//RaHw8GvfvUrhgwZUqfesWPHeO655ygtLSUsLIzf/va33Hj
jjQC4XC7S0tL4+uuvAZgzZw533XVXq45DCHEZlAJnJyxnJ6yurf/xJ44f5/rat94ayzRrAic7IHL
XDZK8gRG+W9NEGQam243bbWAYBkb1vtvAdBsYhhvD7cY0TEy3G9MwPC+3G8swMQ03pmlhWmBYYFg
WlmV59k0wvfu+2/KKKhzBwZimZb/H9JZVt2PXNy3P9A/Ls1WWhQlgWZ4pLXi2NbNovOGt5Z+u2Xr
zrZp8U2lYgIkGyvNdShYKU9Xaomrq+uRZSjX4HguFoRQmGqa3nqE0TKV526tptybtX9e6QLmlFIY
3bfm0Z9Vq29Nn7QJlnnZ965neuS8OyyTYdBNsuevZGgRbboIuUu6bH2SnDYJNd83PrAWY3XpdvFI
LaNPAZunSpdx+++1MmTKFI0eOkJKSwt///ndCQ2u+CMyyLJ5++mkWLlzI8OHD2blzJ08//TTr169
HKUV6ejoJCQm8+OKLZGdnM2PGDDJKSkoiJiWnDkQkhrgia5nnhAEIA6v01cbFfHbr31RqOX04g1wD
foMjw2be8aU+wRK1AqjoQAxPsvOq2rNp5fmmwsPzSZq20BRimRW5uLnFxsShvWFXfXNna85xrTXe
uv0x55/5CzfhMTz8N775h96P2cfANImvKPMesZt+wLDR8J3B7J27jnbRNTbp6W5D/PdHR0TX1vZ1
WSuEGTAVV3vpQ9+fi23ffILe6jmH6v8eo/X6fsd7bw3npJ9VlaLPAprCwkP/+97+88MILAPTr148
ePXqwb98+Ro4cadc7evQoFRUVDB8+HIARI0awfPlyYwdvvvvkmAPHx8QwdOpR3332
XadOmtf6ghBDiCqR5n7jy/EJpX4+uHzcMru/Zqa270WqOHz/H9deHt3U32lSbBTa5ublEX5XZ2
Jj48nJyenTr3Y2Fi/vLi4OHJzc4mPj6e8vNyyvvL42fB0/fryZRtA67bZXMt7AJuMNbDLewBbo473
YFccrbvJwc1+ChZa5tNueyXgDm4w3sMl4A9uVNt76NWkZWJjYykoKKCystLOy87OJi4uzq9eXFkc
Mubm5fnk5OTnExsYSGRmJ0+n0K6+vDSGdossImKiiIpKYmtW7cC8MUXX3Dy5EmSk5P96vX
t25fg4GCysrIA2LlzJ6GhofTp0welNGsWHDBsAT1Ozbt4877rijFUcihBBCiPaiTW9F/eY3vyE
1NZUNGzbgcDhITU3SyefNm8vLympDt3LkopLtLSWLp0KS+//DJhYWGkpqqbKWnnqbXXnqYRmzpq
CBJRS/PKXv6wzJ0cIIYQQV4Y2DWx69+7NmjVr6pSnpqY6+dnZWHJuAuq4k4bmt3+bLNmze3
CiI5FFRYWtty38wghhBBCtKI2m2JhBBCCNHhcJLARQggghBBCCCCNHcJLARQgghRMCQwEYIIYQQAUMCGyGEEEIEjCvum4
vVXOsRN5RVFZW8tRTT/HV8RGhqK0+kkJSfvx4HA6HVOQDmt3bmt3+bLNmze
PnJwcOnfuDHiexHvqaqfq1GvsedCeGYbAw88KcIy+LEiRO89NJL3HrX09HTee+HTee+8
9zpw5Q1paGqNKZcuW8Z///////AeAISZMmNbimnMvlIi0tja+//hqpjGnJnvHtZwAOXPmcNddd7XOAJqoofG
mp6eze/dudN2zvOa0adMYO3ZZsvW387ne/48MPP6RLly4AdOvWjT/84Q+tM4Amami8r7zyijZs3buT
qq6+261avI1hbU86FttbQeFNTUzl27JhJd75tvvmHChAmkpKaMpxyYQSGDTSM2xEnlHMn78eIY
NG4ZSit27d/PEE0+wfus2eus+88wz/PCHBQsW2P9F7p+u6339qH33EYsXL67
z5ZjVGnNc2pNbbrmF8ePHk5aW5ydzpPhm6v8+6mp6eTkJDAiy++2BAwfWe+6mp6eTkJDAiy++SHZ
2NjNmzCApKYmtW7cC8MUXX3Dy5EmSk5P96vXt25fg4GCysrIA2LlzJ6GhofTp0welNGsWHDBsAT1
0NF6AkSNHsmjRou20ZRzoa01NN7Fixfb+yUlJYwdO5a77767dOY
E7qu+lcjLy8s5evRoq/f5coSEhGBhYa7z+noXvWthp7HnQ0W7du5Z577sHCIy
/cwYOHEhCQkKd/B07djBp0iRERERDBmzBjefvvvtetvYsWMHkydPdPBjwL6w4dOpR33323RRt9qRo
a77Bhw+yAOy4ujujo6DrL03REDY23Kpyl3xoz37bffpnfv3tzxwbww2t1Kv2LTD+J2thzbESed+
+fVulry1hzZo1JCcnN3hVGYsR3SmSJQDN6NGDefPmcd1117Vi7Vi75pPPMcqNawaNavOX2+NPQ8
6koKCArKysli3bl2DdS52XXe5Ix4fa8nAlu3i3atB9AsrbjCWbVisV+vaMNuwijaqHuXk1
bM5P6t0HFYoanpnVWZwVSF7ygUKFrOKWzsWlTMwwfhwwmQ+bRzfLVu0H
xCQQS2IgL2rJlCzt37iQjI6Pe8oyMjOy8ZqlHrD6oyfOOFMGRp4saGM3Cnft+YpEBzi8ixV0MhTbK
Ys2cPjz76GZvlRtjlCdgiI6tW2qFfmqIPpv9c+Mg8HnFMWs3yKXLM3Ld63MsUsVtqIjQnLNkuiJNm+5YcEu
vs9Ri4Tov+bLNm+Ae8mRcFwo+EQDqmmhFeOC8yx7W9zkQ7qWo9WH+ty0HhKTxWbs3dc5x99Q3u0ec
mJOtBR0duzA0cTiMNqvw+rSxkoQEyeSmxrY0gICrUpNhxd1hyJn8G0kqpLQJ4pLYcG9SdLYHlEqWxzm
uJ8A6kpUc0LYTVCQX0yK6UMhkbL0MrKzcmQtAsyGYKE0CtYWq8KdNXZ8VQkP2G5e1+j4P9+ZV0hi+
vYEZFuCCdKsfQoO/bYmRqsvFfNrK/xGFuM2DcFvn00NjqQxkYqWjwLvrKSAYJFqpOBCOlX2KL7Kn/uCq
Ho8HVnmYjPmfC0kzMYFRE4z9T9uQYewU8MVfTpvXS6rzzS9NKVQeDT66K9CnXCSq9ZgoKv1L+ejUmw
Y8wFbvUXpuPG1AOHXHVmOkbzqW+Ge3w2lsEaYqLcmwjw2PvFhhsMgdnvW8gr5fqG9pyD/9BApr+ZeJ
dwqA+efQWRQcgZZBLCmHNAhq32zZp7ILfAkMcnFVEb0ZJ95cWUvtTHwuvjVgHlQ9mcswxIeGsBYUTz
Hnsu2d0KXnkX1WAnJ8kiZ2ut9yF9nQcFXA6dpQw7o/yr/pA3a5ORE2ljMTCN45IczQBWVGOeuvF6Hg
ZkhTQrbZ5d/YNeOY4kTp/dnKtqqZG0vN7MyXw0OnpEW1dzYUB/6ONDB7o++n4o+LKhBRr05Vk1Ts
YvNsGzqLC8CbFkUZ9BcmUEx9M9/PVn5Qbp3/F9eFxUYgDHxHPcxIMo3BDY/h9d1xTVaYzYpNC8J7XBNo
cQJmhY1fdGcuZw8JLKu5+Ls67AZP/DCEdgxF/DfN8/Ogcl6SgCxbRdtYhkBDoExNqYKE58sNcSyG6sYv
uFYPmF0Phac/HOp+kVy7RXSm3KfOCkgw8lUEIIl2/cwzY8yg8DWvyyJHQkWSBJ2vJbbBMR1Cnp7jAL
5I7M6odA1lGaVpf/MTAUTXBsArVOcZoJcpSMgPWqbv+R6ZzXgxMChPR5DrQVxTO0GuEFdqz+qvAD3PSt
lyIpDx2egPuXyS6vrp2urqB3GrZrGiFrwLc+W9uKNu1i9s9+MFbe/zwkRYfDq1ZMsaNbmv+GLnsUPtkdvk
s6qMsHAJE1b7FEtaoQSkwJbO3pqF1NrWD4fRsa19bO6SaBgEMm6PqwE4FNMPa5u5n1fTzxeGVU4eVxe
Jd+iebuNQdh4ECOS8UeHAUsXljGezW8GAjTmJ/0rAtIOP7XESHzRo2qsjrjf/eNFiUjZd7kBpU53kUKD5eq
y5p5NQGOuduzNmZCY2dsQvg2nVsOFmIC4ugVwi+t+wVPRhyg9l8CdXMcNFjOeY9ZvkppW+pD0j7u6YV4
0LuEvwoBUl5Q2TnFFHbHEq5I2nSQVPiAoh5CxXjlDY9DNNmZQqfq4JqiMRKHf5Z6dwBnFTYnQbMzEeQZS
lV9GA7v7GQAEdhkXbJoSm4AiZDECR3QQe33L9ZiQx9+LbpMsPo4yKe4DNLCBHr/f6R2xJPzLBNtANUyZu
qqAvrsGLL8bNHqMH2vrqJjbZGn2c+SrPwXUSNPyjc7WgkMCf/PRKmJ8XQOhjfFEGLbMxfjW0TrzwuElKB
KLlptpbyUjDFvM6UB9L/eGb+TNAw4nLnGJIjLcGz5YVMU+R+Ak3tX2rf0JwppdZy0RnwcBcMl3M6dJq
4bjUZp4FoAd6yVMgt6RaYNEYiFEVqVmjqMUq9I/5zCULAbVexVdQP2wEHEDWApXGmYvQ6uLA1cOLdXu
CsIcxh+uGRTLtXvW0aCjhwDGBMIyqYyaV7Sz7RdGhoI/lOW1Q8NewnSIGddwX3wN+v6rS4+EXVSgZsyMi7l
XEwYvKUyC4pUZVnFPYMYn5KtpgFX77UG0z3LmHhwY6Sb1dmpv/Pw6Fv+vEowHjW6vb2HuL4pbbv8YRcc2
/A8rHGomG7R8vJI0+zCpcW6CA1EGjw4bXhGDmYaGa8f8zRlwxPqWSbhfZJxIB2tv0hSOqt8LIvXYdjIUy
x+1Cm+2Qz6YV2G8+U+/XwR5nJEEqsPaaUcpC4ivbWiXbk3TWkdMWC1NaFrUoHoLN8zYCeUwNBB2RsBWPF
gC5BAxCdQ3A9kEJmoCuBxLl4ElxB3y4S/5N2l+G+z2C6jJxBsMdz5HNqh5ZrPxT8WCj8vz7crL+jLE2Bfm
sBx8sGhg8BIKdLNMvmJQwGqlrdH6gGPbiRwqT/YOPgDwcPKfDR9kGrEqEdeGGFEFQuu+BMuBXUrqxgZeXe
JgWQ8JOqhQxNexgJuHIBJaW2ePEEPHvSR9SQbgHjqcMF1M9K9nj9QlWjRnPzZrkDYO1tpWKyWeRk2DrmBHo1zgqJvgE7Q0R+WWL1hUtpkmCTz4vYNPvqNPKCV3qXWpgLUmtkjWMjdNNvZvPTZYI7B6VIvj3xjcKuajAbVgf7ggRhQsoVl2Ct4o6Jrn3ZE/XnjtHntDnxzuUMa7Ow1lhyJHgQJRhjOiNWJn8FaDLEJBAg/pIjQVBVFmeHrPz8y2xcBWrdZjWRKbGbfm==

```
Hn//8Zz744AMKCgqIjo5myJAhpKSk+K10ff78eX7/+9/z1ltvUVFRwU033cSTTz5Jly5d7DpHjx4
lIyODI0eOUFZWRteuXUlKSmLRokUXXIBQCNG6JLARQrSo/Px8Zs6ciWmaTJo0iS5dunDgwAGef/5
5XC4Xs2bNsuu+8cYbGIbBtGnTKC4uZv369Tz00EO8+eabREZGAvDxxx/z8MMPEx8fz5w5c6iqqmL
jxo3Mnj2b1atX06NHD8AT1Dz44IO4XC7uu+8+evXqxXfffceuXbtwuVx+gc3y5cuJiIhg9uzZ5OT
ksG7dOl544QWWLFkCQEFBAQsXLiQqKorp06cTHh7Ot99+S1ZWFhUVFRLYCNGOSGAjhGhRGRkZVFV
VsXbtWvsKyMSJE3nuuedYvXo1kydPtuvm5+ezceNGgULWLNmDfPnzwfg5ZdfpnP
nzrz22mtERUUBcOedd/LTn/6UP/3pTyxbtgyAP/7xj+Tl5fHqq68yYMAA+zN+/vOfRka
ycuVKlFIAmKbJhg0bKCkpoXPnzhw8eJCioiJWrFhBv379/NoSQrQvMsdGCNFiLMvi/fffZ9iwYSi
lKCwstF833XQlZWVfP7553b9u++2w5qAH7yk5/Qq1cv9u7dC3iuwhw7doyxY8faQ1A9+7dGT5
8OPv378cwDEzTZNeuXSQnJ/sFNdWqA5hq48aN89sOGBhq48aN88tLTEzEMAxycnIA7FXu9+7di9voYjI4RoKXL
FRgjRYgoKCigqKmLbtm1s27atwToxMTEAXHvttXXXKu3fvzscfwxgBxrVt5t8XXfddbz//
CUFpaWmfOTUNiY2P90hEREQAUFxcDMHjwYEaOHMmrr77K2rVrSUpkYvjw4YwePRn09mozxBCtA4
JbIQQLcY0TQBGjRrFuHHj6q3Tq1cvTp061rdqkPT6r94XX3LSinFsmXLOHz4MHv37uXXDDz9kyZI
lrF69mlWrVtG1a9fW7K4Q4gIksBFCtJguXboQFhaGYRgMGTLkkuqtmMCUUz3cz333333xTp+z06dPEx
6AqGTJ0/idDqJiopCKUVYWBhffvllcwzD1r9/f/f/r378/cuXPZt28fKSkpbN26Zzbr5wghLp3
MsRFCtBhd1xk5eiSS7d+/mf//7X53ygoICoIIcv/T27dvt2z8ABw4c4Kuvvm4oy1v7kvv7
L5bLrnTlzhj179pCnIyu62iaxu23387+/+fv95vBUqz15+GKKiorqvOfGG28Eo0KSkpEltCSFally
xEUK0qAULFvDJJ58wa9Ys+7HRmyDvgJZ12vRKd6+FWrMVbf4NAr+1ZCgIVkQM5ZvS2UbQrM/aT+9p
16zzyyCM8/PDDDdnuVlZVs2rSJ4OBg+8mp6s/96KOPmD9z3AhRc9PKsKS/o0CTA2q0x6Pm+Mj5ThL+J6sUvwGrNuc
oMfzzn/9k06ZN3HbbbVxxzzTVULaybds2O3ATQrQfEtgIIVpUXFwc2rVuzevVqtm/fTlFREdHR0fTs2ZN77rnHr+5jjz1KTz/9NG+99RZlZWUkJCRw//33ExcXZ9f52+uuY/
oMfzzn/9k06ZN3HbbbVxzzTVULaybds2O3ATQrQfEtgIIVpU165def3113nttdfYvXs3mzdvJiI
igp49e/KLX/zCr+4DDzzzAqVOn+Mtf/kJYJJSQmJiw+EgMTG
RBQsW+E0qjo6OZtWqVWRmZvLOO0+9QXFxMdHQ0N998s197jTF48GCOHDnCe++CO++Xs/9x/fff09YWBg33HA
Djz32WL1PXQkh2o4qLCxs2jVZIYRoZtXzfp5//vnzvkLnVRmzf28VRfsCxVDhnCe++AqGA5BATIzaeqGz3GoqQGghhAyEfgIIYQQImB
IYCOEEEKIgCFzbIQQQggRMOSKjRBBCCCEChgQ2QgghhAgYEtgIIYQQImBIYCOEEEKIgCGBjRBCCCE
ChgQ2QgghhAgY/w/EqPJxd69HfgAAAABJRU5ErkJggg==\n"
          },
          "metadata": {}
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "aa=[x for x in range(180)]\n",
        "plt.figure(figsize=(8,4))\n",
        "plt.plot(aa, Y_test[0][:180], marker='.', label=\"actual\")\n",
        "plt.plot(aa, test_predict[:,0][:180], 'r',
label=\"prediction\")\n",
        "plt.tight_layout()\n",
        "sns.despine(top=True)\n",
        "plt.subplots_adjust(left=0.07)\n",
        "plt.ylabel('Price', size=15)\n",
        "plt.xlabel('Time step', size=15)\n",
        "plt.legend(fontsize=15)\n",
        "plt.show();\n"
      ],
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 308
        },
        "id": "DIennULJPwOY",
        "outputId": "8b6f9ecd-5c37-438f-a0c1-ed1f57966a54"
      },
      "execution_count": 25,
      "outputs": [
        {
          "output_type": "display_data",
          "data": {
            "text/plain": [
          "<Figure size 576x288 with 1 Axes>"
            ],
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAk4AAAEjCAYAAADE247sAAAABHNCSVQICAgIfAhkiAAAAAlwSFl
zAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh
0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nOydd3gU5fbHv7M92c2mkkoJoZeEGoo
giAoIohAgIiAgKFUEFTs/leK9erHnXpBmCRpQRJqhBZAuvSMgIaGn1012k63z+2PJZme2ZDeNJJz
P8/iYeWfmnXdedme+e855z2EKCgpYEARBEBUieNADIAiCIAiCqC+QcCIIgiAIgnAREk4EQRBEBUie
EQRAuQsKJIAiCIAjCRUg4EQRBEBQRBEARBuAgJJ4IgCIIgCIIgCBch4VRD5ObmGh1AloHszQPPJRDc2GG5sE
MzYMZmgcz9WEeSDjVEHl5eQ96CHUCmgczNA/l0FyYoXkwQ/NghubBTH2YBxJOBEEEQBEEQLkLCiSA
IgiAIwkVIOBEEEQRAEQbgICSeCIAiCIAgXIeFEEEARBEEAThIiScCIIgCIIgXET0oAdAEARBELWNwWC
AWq1+0MOwIJPJUFhY+KCH8cCprrXmQ+UQiSongUg4EQRBEARBEVWxbLLavyeqkZrbzE+6+UDLwkZtIm
6icFgQFFREXx8fMAwzIMeDgBAKpVCJpM86GE8cGpnjHliWDsolar65RoJRhBElTmRpcP8E4U
4k2PALYkl6Pp7JrRG9kEPiyDsolar65RoImoXhmHg4+NTaYsjCSeCIKrMuusazqZ2qQkpBpQ1Cwc
BVAyJpoebqvz7k3AiCKLK3C4y9Qo7rSFK4QPYCQEQRA1CwkqCqjIf9tdbSpSGtGIGqsmwvhRBuS2
sGUaNGoWePXsiSnJ0r579255MmDABffgffflGJzXkJCAmJiYhATE4OFCxdCp9ZZ7Z9lWcxSE
mJgYjRoxIs7n6vvDakXx8dLIQar3JrXPTNbYi6USWtrqGRhBEDZCXl4kuLS2
txq6xcuVKvuKDBw4sFAVDUun7cv4uLioXWEd5sNmxAfHw8Nm7
ciOLiYqxbbt85u/0lJSTh37h6ys7OxYMECXFxc3NzcoOrR/JrOQ0ZOtqWtSREd2pbs
4vDtFQHcKyM4CAqbzAFVjovGs3yZRqsvu0L/HmnaZNSsnYzHr3VO3pgx+/UYO/+c6XRVkbH7bqWDAYs
9HhJFFpWhBbiFVFyUZ4XzUmxs3GiHrCTHe9ACxEWSFH0VA3UNKJknYuszcFb8HhmdSsYmsbcd6XCo
2CSdUdF5W9F6LsvJBHThECFRtnkeLzKYgwlH5L2s4iPzoycB2oxstfDTHrdl44y4Ay5oqpN3NXQw2ncF
4WeGjzpcDfOBXgXhHWWLULEPIocgz+23Q6qq+rUmprFZ2YYEHDmqL5Y2evMB6D4bhTBWF4qe5+BReX
WrUtXDlKaVmlOwhEiTRBgdJX67LdyqF2YDHsaLr8vuurTZIGCH8CXSHjfYiHoWlBjhU0ye9KQmlhZgIt9
dAyrq7xmNtUa4awXCBm8RxvD1VWqBKjlQXbfLl82jLmbvAKX3THLrsfljE0cqpA1DWmWoZNO08EWVYPaAOVw
ZbxM5JoT7mmzJ2ROBBMiFMgmCqLEpaehA2q6ihVFS/PzivGd94W6ekStKPyE4UmdBZXNiMO4EuqIh
fHm8g+bKt39RV8vDDiKCydDjbFSLDdPi8d8wfLFL09pYTr49gbQVO4qqBF9J4bZxbh8zIpVg9DWQI5
FpjwS1gIGsTnBZ2k3vxWm6XQDnyxAECVfUyfTfwjYTwr7QK53rz3pw40yHwP4Mg2uNRm8mw2x0A75ywk
KqZGHgqeosM49vwsb4Ojx8/D7qqYZh58mICkl2TqwbLXgHJmI1SDIPcxPGDSZ4BhEa+LVXixKDNE2Hi
nrEmA2oHj0hXo8H0T2OxLh3VTOJNMuT2DiFvJ5zWTtaCTamUobANC/mSiq2CR68KYX51LW1mqE5GBvD4a
C0Z1OUgSr1A8H+W1nR79Jme7d0mb+SxU1A16QNwX4/41DABDoLrsjoFA8oPDjLDZdsvdZdpD/Tiv
aKzDAOz3RQSK1Gtq1WUdawtT5tY3m/MbhV/Nzk8JSGIfvbPPyvzyfU3KLB2q5Q1JeW0w1MwDuEWE3Ue
5ixDiSQ8AgSOxeCVq5R8nc0lRybNtg5CRq2SpQk1HTiwDCgb7xhfaSGSxyo8cm2yiVQ75N9D4v4pwfbwf6
/PkO2N6Jt8DTSe2/XvULSGYvS8d+V7jcrlhYNhvCWMrB8aEy4kNdz3mBCt84+cZWTUz4kVEdEJKb2Mr
lkmgEDBALIFu1hYeqhDmadLmCqC2wKqPp9Cq/fJgXxjmLfTrnlSa68XaqT/0VGKD2wMgZRpKB8d0xTKnIu
```

```
LmWXKUcLNy/kGxF9T27QbTSxOZ+soFxRB1AHWr1+PHj16WLZPnz6NHj16ICUlxeU+Nm3ahP3799u
0Dx8+HN988011DLPGIeFEEA8peTzh5HNfOMVGeKKRrOIq77/xVsR5iRm0UJqDyx1anJzUxnv/RCG
0VrXxWJZFTFIunkjMRpcNGdhzl9x5BFGXaNOmDb777juEhYW5fM6mTZtw4MABm/YlS5ZgzJgx1Tm
8GoOEE0E8pDiKcZIKGUxpq6jwfL716O3OXpAK77vq+NnDNUYUaE24Vug4oWapEUixSn9wNkePg+n
m1XtaEzBpXx5HWBEE4R6lpdX740OhUCAyMrJaasu1adPGUny4rkPCiWiwsKw5gPn7q2q8cSQf/z6
jQomBEjGWUcAXTpLyx8GUNnKI3Xw6dA0oT57pLxNwzlfpWBzK0HKOb+0tgohn2Ar0KD/pdjHXPac
xsBxhRRAPMwsXLsTEiROxf/9+xMbGom/fvpg6dSpSU1Mtx/To0QMJCQn48ssvMWjQIIwbNw4AoNV
qERcXh2HDhqFPnz4YN24cjhw5wulfp9Phs88+w+OPP44nn3wSX375JQwG7vfPnqvOaDTixx9/xKh
Ro9CnTx8MGzYMCxcuBADMmDEDV69exbZt29CjRw/06NEDiYmJAOy76nbv3o2xY8da+lm2bBlnDIm
JiejRoweuX7+O2bNno1+/foiNjcW+ffuqYYYYdI6r4EIKof5zJ1mHGoTxcK+S+fH/4R41LzwVbLCN
VxWhi8Z9zKuy4U4phTT3wdmcvMEz19F3T5PPKrfhapSEI8hQiprkH1qeU8E9zSEe/8uSZAoZBiKe
QI34Sb3H76hEogd7EcrKTZ2hMCJCZrVU5pbZxTRFeQps2gnhYycjIwNdff40ZM2ZAKpVi5cqVmDN
nDn7//XdIpVIAwM8//4wuXbpg4cKFMJnMP5beffdXL58GVOnTkXjxo2xZ88evPnmm4iPj0fr1q0
BAP/73/+wZcsWzJw5E82bN8fmzZvx559/VjimTz75BNu3b8eEDRNu3b8eECRPQtWtXqFQqy3lvv/023n33XYS
FhWHKlCkAgMaNG9vt59vt59ixY5g/fz6GDh2KOXPmIDk5GSteePyYSHee+89zrEffPABRowYgRdeeAH
r16/H/PnzsWnTJgQFBVVuYiuAhBPRIHntr3wb0QQAuaUmpKgMaM+ryVZZtt4qwZL83/CAi3lFaO0
jQkxzz2rpu6axl8fJmpntFS4Lp+ZeQigl3PNDbYQT103QvZEEd9VGjnC6VWSwCDB+mRYWgNrAQkZ
PLaKG8PbxeSDXLSwoqNR5BQUF+PzzzzxEVFQUAaNu2LUaOHInExESMGjUKABAQEIB///vflnNOnDi
BI0eOYPny5ejatFevXrh9+za+/f/za+//57fPrppygoKMCmTZswdepUjB8/3nJMRTFIN2/exNzF
v3jzOsQMHDgQAREREQCaTwcfHB5GRkU77Wrly64g0AWLZsGaZMmcIRRWPHjsW
zzz5rmYMhQ4bg8OHDljmobshVRzQ41HoTLuQ5cOkwsAQwVwcrLxdztuccrj/JHB2tqiujS4AEjwa
Xu98EDOAvtW9Ni/K3FaL8AHHr1AUA0K2RBOEK7jG3rIRWTqmtW/VOMa2uI4gy/Pz8LKIJAEJCQtC
2bVtcvnzZ0vbII48wtkPgbbB3n1MDR8vWh0VkJlR0NsNsIuzxSlnF+Hj6xfzFPZCh0WPhrlIjlGAcJ6SzOkNhvAc7LdoNpvu1QXjs7d/ZbdhRCQyF9ch8/9Fi0zAmq2Pe5Svk1laZ5mPJYmsOE2Uzy9Cqy0NZyWjaYeTS68SY8BDMJ2EaWb4kKrYkKEcbc+JCmM5+e1ytUY4TJWcRzixoZbPwmCUnd3Ka7y0NYK8HdeTUhDhzm+sk4c/EaNubfP1WU0HDNSDE7RyfCbnYSBf8Ij+D3SBjKBoFOC0R8zhx7qB8NrnN77WkBOG9Ux+2x4NSDnUS2wHsQ2U7MI9Nh+Bv0pGm20IoaRHxFyZPwCcCLy+VpW0xjwqwRa0n0m98Ij/4FD+3
```

```
d1KpwKkvxfvHiRSxZsgTDhg1Damoqxo0bB5lMhpEjR2LixIkYPXo0J5C8LCh86NChNn2+9tprlpw
VO3bswPPPP4/x48djwoQJUCgUePXVV2vt/ogHT4qKK2ZaeIkgETJo6c198fbYmInFZ4rw7WU1umz
IrFQNNH6CRmuOZVbfUtg/bpYgw8rC4yEEjsUE4Y1O3Fi/FVc0yNS4JgL5S/2dBYfz6R/C/TXYwtu
xcPKI/xE95k6BiGcyN7Zvj+KdO6H56SdoontwYpZuFRstsWP5ch+8EfMuot7eiIMR3Th9MEYjPOb
Ng7iB/eImCKJuU6sxTnPmzHEYb1SWHdQezZs3x6FDh+zu+/rrry1/T5o0CZMmTarSGIn6DX9FXdl
Lva2P2LISDQCsDS7FBhaX8/To0si9RQT2AsPLOJ6pBeDlcL87rOJZm8a1kiPQQ4gpbeT46oIK2aX
lom/XnVJMbCOvsM+KsoY7Y3ZHBb67WozsUhMay4V2czgBgHjNGni8/rpNe/qkSfD8/HNAbI47a6a
wzeWUw1u9dzWoBR6f9T0W7lyK+XtWcvZ5zJ0LeHpCb7X6hiAaOh999NGDHsJDS51w1RFEdcHP4dR
SWSacnP9GkFQi7ZK9zNZlHMvSQlcJKxafvfdKcZRnvZraziyMZCIGPYO4okXo4jeaX6fO1VV1gDk
9wdnRwTjwbCCOxQTZTYMg/vVXs6CxghUKofn6a9ybPdsimgCgmZdt9vBcO25Qo1CE/3t6LnZ9+QN
Yq5WyDMvCY/p0CO2sxCUIgqhuSDgRDQp+KoIyi1MbH+e16dI0jkWQI5xZnIr0wMdnCt3us4xzOTo
8syMbo5K4cUGPBkvQ1upeGsu5gtDZmKypqNxKRZTldrInmkRbtsBj5kwwbLk4YyUSaBISoH/xRZv
j7VmcnN3HkW5PQvPDD2Ctsr8zRiM8p08HU0GCPoIgiKpCwoloUPBdda5anG5WolZKRSLlv5fUOJq
pdbvfm0UGDN6WjUMZtnFSQ5ty44v4uZLSXIxx4udxcsdV5wzRjh3wfOklMKby/lmRCJr4eBieesr
uOXyL051iI7KczO09tRGGp59GyYoVYK1iowQZGWYrF1vrC4UJgniIIOFENBhYlrW1ON0XThEVFPa
96aS+nSP4q+re6aSAtQGGBTBlX57D1XeO+OGqGvbizhUiBi/y4peCecIpw0XLmTuZw11FtG8fPCd
NAmMo/zdgBQJoVq+GYcgQh+cpxAIEyMqvb2CBi3mOVyXeVZv7148eDe28eZx94j/+gJhKLhEuwJL
Afqipyr8/CSeiwZCuMXFKoCgljOWFLBY4zzRdGYsTP8apra8Y657gFqVMLzEh+vdM6Fz0BOpNLNZ
e13DapALgy95KXB8bbFOTLtiDL5ycizS13gSWZTkpG4CqCydRYiI8x44FY1VKgWUYlCxbBsOIERW
e30zBvY8z2Y5XJd4tLr9H7TvvwNClC2e/x7vvgrl3z9WhEw8hcrkcBQUFJJ4eUliWRUFBAeTyihf
S2KPeZA4nHj5uFhmQXWJCt0ZiCFwosWHPTWdd4HJgYyl237XvOrtRGVedTZyQEP1DpZjeTo4VV8p
XwhXoWNwuYdDBhT533SlFtpUgk4uAS88Fw1dqP3o9VM4VPOlOhNO/z6rw5fkieEsY5PKDw93I48R
Hsno1ZG+/zXHPAUDJ119D//zzLvXRWCHEaavcVyq94xfaXetCv2IxSlatgqJfPzAas+BkioshW7Q
IJStWuHEXxMOESCSCl5cXVCrXc5/VNCqV6oEXr60L1NY8eHl5QSSqnAQi4UTUSbbcLMGUfXkwAhj
cRIpfnwyo8BxHK+rKeCPSyyKchAxgvejtVpERLMvaVBJ3Bj/Gqcy69V4XJUc4sQCCpa79sv3pGjf
1wJgWcoeiCbDjqiuxfx+3igz47FwRWMBGNHkIGXiIKlH7jWUh/fhjyOyUQSr59FPo3UgN4iNx/fo
qPYtCnQne98WeqWVLlC5cCA+rSgOSX3+FbsYMGHnWKIIoQyQS1ans4VlZWWjSpMmDHsYDpz7MA7n
qiDrJsr+LUGZX2HVHiwu5FSeUPJHFtSbxg457B0txblQQvurtjStjgqEQcTNvO0svYA/+8WXCyUc
qQBOe6+leacXCIE1txO573HuY2NrTwdFmvMQCzn1ojbbxSwBwLldvUxy4DHdX1AEATCbI3n7bRjS
xQiE0cXHQzZjhVnd9g52XWENabQkAACAASURBVFDyhNU9NdeyppsyBcb27Tltsvffp0BxgiCqHRJ
ORJ2EH6x9KJ0rKO4UGzA6KRt9Nmdi771S5JQa8fsNbk2zQA/bj3e4UoTJbRUI9BDaCCt3AsSNJtZ
2Sb9VgHMHXgLH6+qKv2o/J6thsnrPd/QTo5O/8zQKgK3VKd1OgHiqyrEr0kfqprXJYIDHK69Aumo
Vp5n19IRm3TroJ050rz8A3Z0kH2UAdPDlWg9v8O9HKETp/errZYiOHoVoa+UKChMEQTiChBNRJ1H
xoqmtg75LDSxG7MrBnns6/J1vwKikXMw7WsAJwBYxwLiWzgP/wr24L2NOgHhRESTffw+PV16BfMA
AeLVvb66NtnIlmLQ05OtMHAuOt4ThBKDzX/TJGsdftRIDiw9OFtoU8R3XwsMl12GwJ7fvDDu19/j
xX9a4FRhuNMJj1ixI1q3jNJv8/KBOTIThfuFudwn3EtpYlazH58vLUHraTvC4YcAA6HnXly1aBBg
rV4uQIAjCHiSciDqHxmAC/91/x8o1859zKpuadFtucmuhfdBNCbnY+cfbkXASHj8Or0cegccbb0C
SkADR2bMQpKVBdOgQPN5+G14dOkDx5psaIR7dk4r+Xim1cad1cLAE
TamNxshUK/Pgva1wWTiYTPF5/HZL167nNQUFQb9sGY9eurVjB4ZhEOln37rmLxMgimd54wvrMko
XL+YkxhSmpEC8eXOlx0UQBMGHhBNR5+AXoAXKXU1nc3SIu1Ts9HylmMFkF+q1hfNLfaj0kH72GeR
Dh0Jw547D8xiWRdDPPyBp+TT4qgsAAAEybl/teSIg2YFwirtQhOsqW6ETKBOgc4BrwsmVXE5Vtji
xLGTvvAMJr6CuqXFjqHfsgKldO5fG6gxHbskAmQBteAlMbxXbtyKZ2rSxWckn/eILinUiCKLaIOF
E1Dn4OYYA4IbKCJ2RxcxDeaioBNykNnIoXVhez7c4DV71KWT/+hcYF107j6WcxLFvxqF5zh348yx
OLZUiWA8hV8/YTYR5Po/rcgqUMfiytzcuPhdst5yJPWxjnLjXKdCanAa+l1Y0oUYjZK+/bhPTZAo
NRXFiIkwRES6NsyI6+dsXin5SgSWRaRnOhKD29dfBCsonX3j5MkQ7d1bLGAmCIEg4EXUOe8Lpnsa
IH/4pxtUCrijgu8gAYHIFK9HKsLY4jTq3Cy/8scrmGO1LL0G9aROKDh9GyaJFMDVtytnfOvsWklZ
MRXgJt0aaSMDY1Mf7O8/2Za82cEXLqn5+mNJW4bJoAoAQXowTXzjxs6nz4Sef5GAwwGPmTEh//JH
TbGrUCOotW8CGh7s8zopwZnHiZ36/XWx0WETZ1LIl9Lykm2R1IgiiuiDhRNQ5HNWAW3Fm1F7cGM
p1j/pD+vXvpABXM0q0FQhAgOgbWYKfvjl/zj7TP7+UK9fj9IvvoBhwACYO0naEbs4cFO/bB0Pv3px
jW+bcwTuLXgKKuS5EfoD45XzbMiK5pdyXeRMv91Or2brq3BNO/HgsCyoVPMePt41pCgiAevNmmFq
1cnuszmilFMHDjmD0lwmgEAs4AtHIAreKnVudrBGdOgXhwYPVN1iCIB5aSDgRdQ5RwmVC
gayMJfhnoB0+RWTSFewlt3DqOkAoZtBDpsPH71+ClLRdlrFgMzS+/2F0hxvr7Q715M072H8Zpb3r
tAjwnTQK05WkT+ILkb55wYlkW2Tz3XSM7KRQqgl/oly+c+G6tWe3lCJAJIGCAIA8BhjT1sOmTuXk
TisGDId61i9NuCgmBets2mDq4kgfdPYQCBh39bP/tyvJM2bjrnAS8myIjoR88mNMmXbq0GkZJEMT
DDgknos7BL2ViDwGA6EBzTMzAxh64MS4Uh4YH4q8RQW65ud46tAbtslI5bX++thCZkU5WiEml+Pz
VL7CrzSOcZvHevfAcNw64X/qjAy9AnG9xUulZaK00jkwITjJLV+HXq8ssMcFolRAqNV8H7xIV/Iv
zEViUg44yPf5+LhiHhwfiQiwvlkqvh2TVKigGDIDwyhVOv6amTVG8YwdMbdq4PUZXsRfnVBZ4z88
EX5ElzaYAcFISBNeuVXGEBEE87FDJFaLOYW9VHZ8OfmJ4WhW8lQoZtHfkcnIAk5ODCby4pjXdn8W
kkGcRtiULx0cGQeEgpUGWUYjRL36NfUsno/vdvy3t4r17IR81CupffkF7XwXnnCv5BhhNLIT38z1
ll/CtTUK3Sr6UIRMx8JUyyL9fSsXIAjn5aoQd+RPixER8l7gTSg23Jhcrl8M/MBCmiAiYWrcG6+0
```

```
NJjMTogMHIExNtbmGoXNnaNauBRsa6vb43IGfdgCAJfCeb3GqSDgZe/SAoXt3iE6dsrRJVqxAqZ0
SMQRBEK5CFieizuGKxalbgHsiyR7SJUvgUVIel5Tr6Y05I98DGAb3NCb8nqpxeG5OqQnFMjmenrY
MVwObc/aJjh6F4sknEXrlHPytlvqXGFlOdvJsnkAMtBPo7iohVlanwVcOodkj0ZBPnAjJ+vU2ogk
AGLUawhs3IN67F9Jvv4Xs008h/eEHu6JJN2IE1Nu317hoAoAoO7mcyhYAtPDmC6eKVz/qZs3ibEv
WrQOTn1+FERIE8bBDwomocziKcbKmq4vJIR0hSE2F5PvvOW0fD5yOQo/yqtzrU0r4p1nIvR+blOU
VgH6z41HSviNnvzA5GYpBg7Bkz//gVVouzs5a1dzLKuHVuvNwsrqtAkLkQihK1Vjx60fYuXIGpJn
ple6rDNbTEyULFqDk++8BT9dWKlaVdnashp73XYk2rjonMU5l6J95BqawMMs2o9FAHB9fxVESBPE
wU+vCadmyZRg1ahR69uyJpKQkS7tOp8OiRYsQExODmJgYJCQkWPadPn0affv2xfjx4y3/Xb582eE
1EhISLP0sXLgQOl3FBWKJuoMrFidntc1cQbp4MRhD+Yv3hl8YlvUdyznmZLYORXrbsbAstyBwtpc
/CrduhSE6mnMcYzRiysaluLPwCXyx+T9onXUDe+6WZzjn53WqisUpqvAuTnz1PKYd22B3v0YsQ66
nN/K9fMGKnVvrWIEA2hdfRNGZM9C99hogqL3HhFTIoG9Q+b8tA0B/P41AuJcIVlVtcE9jhMZQwWd
FLIZ22jTuNVauBPS2KxwJgiBcodZjnPr27Yvhw4dj8eLFnPa1a9dCo9Fg48aNKCoqwoQJExAVFYX
IyEgAQEhICEdMOeLChQvYsGED4uPj4eXlhbfffhvr1q3DpEmTauR+iOonvwKLk1zEoI135T+6gtR
UmzIcpo8+wB9PheLl/bm4ozG/qHUmYPedUoyM4FpbivQsrPWUTAB4BPhBvXkzPN57zya7tndpMd4
4sAZvHFiDK03bQTr5OehiYpBV4sc5zl5RYlcQ/vUXFs8bD7mK64JiBQJcHDEer4Y8gSPNu8AoFGF
0hAdW9/MFCgshuHsXwuRkc8C0Vgs2KAim4GAYo6NrxS3niKWP+uLRLVlQG1iEyYVo5W0WehIhg6Y
KIcfdmaoyoqOf83nTT5wI2X/+A+Z+0L4gLQ3irVuhHzWq5m6CIIgGS61bnKKiohBmZTovY9euXRg
9ejQYhoFSqcSQIUOwsxLZfnft2oUhQ4ZAqVSCYRjExsZWqh/iwcCyLHK1zmNXOgeILQHWlUHy7bd
grJIhGjt0QMCE59AzSIoREdxSLVtvlfJPt8nCbWABrZEF5HKUxMVBvW4dTI0a2b12u9tXIFu4EMr
OnTFt+jN45VACZDrzNfhlWyqEZSFZtQryESNsRFNa4wiok5KweubHONgyGkahWWhGKEUAwwA+PjB
17Ah9TAy077wD7YcfQjd9OgzDhz9Q0QQAzbxESB4bgkPDA3FyJHeVpLsr6wCA9fWFbizXmij59tv
qGSxBEA8ddSbGKSMjAyEhIZbt0NBQZGRkWLYzMzMxceJETJgwAcuXL3fofsvIyEBwcLBlOywsjNM
PUbfRGFjYqUzCobuLNdzsUlAACc9yqZ0zx+KOeqaZjLNv991SlPCye99TcwdoYrkvcMOQISg+fhy
l77wDU0CAw6FEXDuP/238N1L+9RRmHV6LYKHr7iMmLw+e48bB4623wPC+C2u7DsXLn2yBsXt3G2H
BFx51lbJVkvzUEvwM4q4IJwDQzZjB2RadOgXhiRNVGyRBEA8l9eIp2qZNGyQmJkKpVKKwsBALFiz
A8uXLMWfOnGrpPzk5uVr6cWaGFWYdK4cd4xYoun1ezWUE9x84ZAqVSCYRjExsZWqh/iwcCyLHK1zmNXOgeILQHWlUHy7bd
vj9GbBRpJZMjWmYWU2sAi4dQN9PcvF0unM4UApJZtmYAFm3UTyTm8i40eDeaZZ3Di57lovXc7Hrt
+EkLW1g0ZqsrG0t//BfW+5Sh4LhbZI0fC4Otrd/yC4mIE/vYbghMSICostNn/4VOvYPGgmRDnAn/
/k4yLmTJY/z5i8tORnOxiavUapLLfDW+tCEC5cD5zJw/JHq79MGrZpw98jhyxbOuWLEHqJ59Uahz
VBT0jzNA8mKF5MFMX5qGVk8oIdUY4BQUFIT093eLGS0tLs1iOFIryfDje3t4YOXIkvvvvuO7v9BAc
HcyxM9+7d41ig7OFsgipLcnJyjfRb33B3HtQ5OgDZTo8ZFtUMYfJKrEDT6+G1cSOnyTRrFlq2b89
pG5FXgFVX1Jbt33IVGNHZ3+JK06tVAIos+19sq0CHNo0dXvbKzBZ4ss1oBKmyMfp8EiZf2oVu107
bHCfPy4V8+XKEfvcdjNHRMPTvb66NJxKBycqC6NgxiA4eBGNHMJkUCjw/ciF+6/KUeYwsklogjv
aXM5xzZs1RqtGUpvza5OqfDd6e5YCqeX3lMl6oFWrZi6dK3zrLcBKOPnu24fWMhnYJk0qNZaqQs8
IMzQPZmgezNSHeagzrrrBgwdjw4YNYFkWKpUKO3bswOD7JROysLA3o9J0el0SEpKQrt27vdnBMlqWOk4lSgqYv2ySKqaZtD5+xjNMDDWrVDhXoRzvOeGsh3C7t/6vXQ6L8N1v5G3yKCfG/fwdzE5wk3Fpe8OmHRKqU+gwaoSGuxqyJcIkv6L/eqKiPdcxfEsmlGMX4o5GMk9p7SS7rR8sEJqVGtvNBPCDw5v5kaurUfeIEeDddAAU+jwHjLg9pe8mmWPOUOG3J8YJSiUdrJ/FhtWc9yX+UwAwZL6y1GJW2WpAwbXMlQqLJYZ6TTgyKpWzcLlEABZgXbA1M2X73fEfcd9ALuAVeTmFPYkkfg5hN5iEdEhE++OMNzMO+6
```

```
MduBeKeb9le96riiCIBo8JJyIOkORnoXeSjd5ihh4iqrwEVWpINq9m9OkHzXK7W6cWZzC5O6JsN5
B9mOQGlWTqw4wZyq3Rm9yvRhufaErz1V3KU+PUoOb6lAohHb6dE7TrCO/QGLQ4WS2Hkl3SjA8KRf
f/aPB4O05OF8JcUYQRMODhBNRZ3CUiqCyiHfsAKMttxQYw8Nh7NLF7X5aOhFOTdyIcQIcu+sCq8l
VBwBT28nRVG6eOwbmlYEtnAS410cCPYScudebgEv57sU5AYDuhRegl5fXwgwuysXzZ7ZDY2Ax/WA
+59i4S+S+IwiChBNRh7CJb6qqm45X0FcfEwO4kDqAj1IiQIin/bGEeroneB5xYHEKqCZXHQB4SwQ
4MzoYe54OwMFnG+FojPuZ1usD3XlWp2OVcacplbj87POcptcPrAFYFvk6rgXreJYWLMuCZVncLTa
gUGebL4sgiIYPCSeizsCPb6pSYHhBAUR//slp0sfEVLo7e3FOgR4CyETuCZJwLyEEdk4R22usAiI
Bg+6BUkT6SxqkaAKArrw4p4/PqFBqcF/M7Bk+GUam/LPWOe0fPHtpn81xd9UmXMo34OUD+ej4WyY
i16fjcHqpzXEEQTRsSDgRdQZ+8ksfaeVf+OJt2yyZoQHA2LIlTJGRle7PXpyTO4HhZTAM4zQRJuE
6/UK41rtSI7D8strB0Y75WxmKTZFPcNr+ve1rCEy22d7nHMnH7zfMmeRVemDMnjxoG1rkPUEQTiH
hRNQZDmVwXS1/3tNW+qVks5qukm66MuzlcqqMcAKAfqG27jp6+bpPJ38JBjXmzuXKy8XQuGl1SlM
b8dFTs2Gy+nx0yEzBC6f+sDn2bA43jkptcD/5JkEQ9RsSTkSd4GqBHuuuazhtmsq+lAoKINq/n9N
UFTcd4MDi5GZgeBmvdlTA2sMnYhreqrfa4pMePpy5TCsx4cvzRY5PsMM9jQmXQ1piTfdnOe2Ldv4
PEkPFK+mae1VfYD9BEHUfEk5EneDj0yqYrIwuVVkNJt6zB4yhXIgYW7WCqX37Ko3PvsWpci63pgo
RvnvMF55Cs2hq5tXwVr3VFi28RZjRXsFp++JCMW4Xlf/7F2hNMJocW/Tuqc3HLnjqFWiF5XFTzfL
T8crhtWhRgTBS6chaSBAPEySciAdCqsqAPpsz0Sj+Hh7dkonE29wg239FK3FkROVWg4m2b+ds659
+ukpjBYBQuRD8BXRBVci9NDzcEzfGh+Lg8ED8Vcn7JMy81dkLPla161gAX10sQomBxXO7cxC+Nh3
dN2biQq6t9UhjMCFfaxY+t/zC8G2fMZz9i3b8D7HCLKfXv6M24nyuDk9vz0L/rZk4mEYB4wTRkCH
hRDwQ/nW2EH/nG6A3ARfzuG6qbgFizOygqJyY0Goh5iW9NFSDcBIwDMK9uFYhURVXwkmFDNr7ikk
0VRFviQBvdvLitCXeKsEX54uQdNccN3ejyIgBf2TjKi/XU5qaGwD+w/BZMCjKLVgKXQneWvEeGJP
juKk7xQa8diQfRzL1OJ9rwMikXKj1lKqAIBoqJJyIB8LhdMexI+93VYKpZCC36PBhm9p0xm7dKtU
Xn9kdy1+oMiEwuLH9nExE7TOpjRwKK12bXcri8wvcWCcjCwzflYM7xeVC/Z6aK3A8gxtB9/HHnDa
/Y0fw1qnfHF7773wDzuaW92lggR23yepEEA0VEk5ErZNTakRmif1f5EIGCHEzqaQ1om3bONv6IUM
AQfV8zMe1kuPPYY3wRW9vXB8bAllVysEQ1YqXWIBxreQVHpdZYsLIXTmW8ixl8U1lhMqF0E+aBP1
jj3HaP978OTqmJwMwx99Zs++erUi6VUzB/gTRUKEnP1HrXMi1LY3hI2EgFphXKFU6UNpkgpgX32Q
YOrRyfTmgayMJXmqrgEJMX526xtR2FQsnAEhWGZGQbM73lKbhCvgwuRBgGJTExYH1Knf/iUs0OPv
jdBzsWIzlj/pwzjmTY/t5PpZJde0IoqFS60//ZcuWYdSoUejZsyeSkpIs7TqdDosWLUJMTAxiYmK
QkJBg2bd7925MmDABY8aMwZgxYxAXFwej0TY5HQAkJiZiwIABGD9+vOW/7OzsGr8vwnXO8YTTxNa
e+Of5EBx4NrDSAeEAIDx7FoKMDMs2q1DA0K9flcZK1B9aeYvxuJ0cWeFeQvQL5pZn+fpiMUwsa2t
xum/tZJs2RcnixZx9oqxM9HkpFu003OeJvTV1RzN10DtZyUcQRP2l1oVT3759ERcXhy68Yqtr166
FRqPBxo0bER8fj/Xr1+PixYsAgMDAQHz11Vf49ddfER8fjwsXLmDLli0OrxEZGYmEhATLf40aNar
ReyLc4zxvdVMnf3G1BErzV9MZnngCkMkq3R9R/5jW3tbq9F5nL3z9CNdKdEdtxPbbpTbB4WFWSU3
1kyZBO3UqZ7/g9m30jh2MJ//5y+k4ig0sztuxrBIEUf+pdeEUFRWFsLAwm/Zdu3Zh9OjRYBgGSqU
SQ4YMwc6dOwEAnTp1QkBAAABJpOhTZs2SE9Pr9VxE9UH/4XSyV/i4Ej3EO/Zw9nWDxlSLf0S9Ye
BYTJOSRsBgA5+YkR4izE6giui/3upGHedCCcwDEr/8x/oxnBTFIgyM7F7+VR8vfETNMlPcziWPXd
LseBUIYZuz8bWmxqHxxEEUb+oM4EaGRkZCAkJsWyHhoYiw8rtUkZ2djb27t27t2L/v37O+zr8uXeOG
FFzBp0iSsXbsWLEsm87pCgdaEm0XlLyshA3TwtS2g6y5MVhaE589btlmGgeHJJ6vcL1G/EAoY/Pq
kH3wkDESM2U1XVqD5tUgl59jjWTr8nc911YXxy+gIBChZuhT6YcNsrjX30M+4uXgQkr59GdP+Wo+
meVwR9em5Inx9sRh/ZeowcV8+rhWYfzAUaE3441YJ7pZQGgqCqI/Uq3TFBQUFmDt3LiZNmoSOHTv
aPaZv37544okn4OHhgaysLLz11llsQi8WIjY112G9ycnKNjLem+q1vWM/DqQIBgPJf/s09TLh743q
Vr+G3fTusX4uatm1xLT8fyM+vct/VBX0eyqnpudjWHbhdwqCpB4vbqea0BFIAvX2lOJpvf9WmkGF
RcDcVRXb0DDN/PkICAxHyww9grH6ICVgWA68dxcBrRwEAqX6NkRLQBLd8Q3HbN/j+/0OQI/fFL7v
u4dn2Xoi9oEC2TgAxI8MyXQo6e1POJ/pumKF5MFMX5qFVq1YO99UZ4RQUFIT09HSLGy8tLQ3BwcG
W/fn5+XjllVfw9NNPY+zYsQ778fEpj2UJjjwTmZeLAKg
s29GhCrRq1aTK1/H4+2/OtvDpp+vU/NPnoZzamosOdtrelWsxfFeO3eND5SK0be1kXF9+CfWoUfC
cPh2Cu3ftHhKRdxcRefb3AYBBIsWm0HY4Ft4JR5t1whdFnXFgVpeHOgkqfTfM0DyYqQ/zUGdcdYM
HD8aGDRvAsixUKhV27NiBwYMHAwBycnIwY8YMPPPPMxg/frzTfjIzM2H1Q/q9Vq7N+/H+3atavRsRO
ucz6PH99UdTcdTCaI9u7lNBmeeKLq/RINjn4hEvQLsR9TF+JCCR1jnz4oOnYMW+cswskm9qSZc0Q
6LfrcPId5++OxIf4NXPngcXh37ADZ/PkQpKS43R9BELWP28IpOTkZ77//IJmPT4v8fhVVRkrmR4lRlC
NwF9/OV9pAgBxcXEYNmwYLl68iVVLlmDYsGFITU3lFFQUgsAtIPcLF9u9k0mCncCJaVH4/+vcvdn
IS0tDYmKiJcXAsmXLxjkxx+Y2Yt6NkowZg/Hjx2PKlCf/MLxzNQgn4fnzEYOTmWrZ
ZpRLG7t2r3C/R8GAYBj885odwhe2j73KBAVqjC/GQCgXuTHgJPd5Yjw5vb8F7T7+GgxHdYBBULnG
rND0N0qVL4dWtGzxjYiA8fbpS/RAEUfbpS/RAEUTu45ar766+/MG/ePRFRWHo0KFYvXq1ZZ9iXOgz1QQ9EIsH69evxyCO
POO1jzpw5mDNnjt19QRVCdAAAIABJREFUCxYssNsNs+f/58zJ8/32Gf1jmf3nrrLafXJx4cRXoTrhd
yg3E7+lVdONlYm/r3B8TVYMkiGiT+MiF+GxSA6I3c4r2lBhYpKgPau7BYoYnc/Oi8HNISl0Na4tM
np0KuVWNtRBF+3nMFgTlpaJafjqb56QbzIS/uhCBxbnw1Q67Ve8bx/E+/ZBN24cSj/8EKxVVuVAJ
BEHUDt4TT0qVLMWzYMWzYMfPx8oNGhsdFPD0nWaBYAIirWCgXsBVOelpNR1R
AK28xtj4VgNikHGjvx2aHyl3PWt9UYWtdUkvlaNO3Bd6MjsLQHdko0rNgAEv/ABBWkIFeN8+j160
L6HXrPLrdvQwPwPvdamL8natRBv2wb1zz/D/OijjlJfLoVKp7J1
GEACAfwq41iYTC5d/4TuksBDCEyc4TYbHH698f8RDQ78QKY7GBGHv3VIEy4UY1FjmcpB2qFwIBty
```

s4UoxgyAPAYI9hbg8JgQpKgP+d7EIa1NKLMfc8wnG752D8Xtnc/ymVK/FHt1h9PojAaLjxznXYAo
LIR81CiXLlkE/enRVb5cgiGrCrRgnX19f3Lt3z+6+1NRUBAUFVcugiIZJuoabbFApYSpfl+4+osO
HwViV3zG2bg22SdVX6REPBxFKEaa2V+CZZh5urWyTCs0iyZrWPiLLD8qyTPiRFSR31Yql2NsvBup
du1C8eTOMbdty9jM6HTxffhkSq5hOgiAeLG4Jp4EDB2LFihU4d+6cpY1hGNy6dQtr1qzBU089Ve0
DJBoOGTzh9HYnryovwxYdOsTZNvCq2hNETdGE564rS7RpjSvW1BSV2RJrfOwxFB8+jNL337c5xuP
99yHiZcYnCOLB4NbP/RkzZuDGjRuYMWMG/P39AQBvvvkmcnNz0bNnT0yePLlGBkk0DPjCqbGi6mn
ERAcPcrYNFA9C1BKhciGQXb5KtIWXbdxTe9+KP+OpRVYubJEI2rffhql5c3jMmgVGX96/x/TpKD5
8GKxVhQWCIGoft95cEokEX331FU6cOIFTp06hoKAASqUS0dHR6NmzZZ02NkWggpGu4GZJDPCu3fLs
MJicHwsuXLdssw8DYt2+V+iQIV2nnI8YWlFq2W3rbPk4beQjRSCZAdqnj7OA3VAabNn1sLFh/f3i
OHg3GZD5XkJsLz5dfhnrLFkBUZ3IXE8RDR6W+fT169ECPHj2qeyxEAyezhGtx4seIuIvwyBHOtik
yEqyvb5X6JAhXmd1RgVVXipGvY9FIJsCQph52j2vvK8aBdNuVc2WkaUzQGEzwFHG/D4bHH4f2/fc
h+/hjS5voyBFIv/oKWkq7QhAPDLfeXElJSfjpp5/s7vv555+xe/fuahkU0fAwmFhklXB/dQd5VM3
iZBPfRG46ohZRiAW4PCYEh4cH4kJssMN4PVfcdTdURrvt2tdfh54Xtyf96iswdgqgEwRRO7glnOL
j4yGR2F8lIpPJEB8fXy2DIhoeWSUmztLtAJkAkqoGhvPjm/r1q1J/BOEuZavnnC1ysBcgzj+aE+d
kjVCIkpUrYQoIKD9Xo4H0P/+pzHAJgqgG3BJOd+7cQYsWLezuCw8Px507d6plUETDgx8YHlzV+Ka
MDAivXbNss0IhDL17V6lPgqgJOtgRTr2DuG324pzKYAMDoX33XU6bZM0aCOpABXmCeBhxSzjJZDJ
kZWXZ3ZeZmQkxlbkgHMDP4eRKQVVniA4f5mwbO3cGlMoq9UkQNUEbHxFEPBNT1wCu5T7ViXACAN2
kSTBGRFi2GaMRssWLq22MBEG4jltvrx49euC7775DXl4epz0/Px8//PADevXqVa2DIxoOGSXVa3E
iNx1RX5CLBZjeTm7Z9pcK0COQK5wu5eux804JDqVrwbJ2Cg2LxdB+8AG3aetWKghMEA8At1bVzZ4
9G1OmTMHIkSPRq1cvBAQEICcnB8ePH4dCocrr75aU+Mk6jn8VARVFU5CXmA41fMi6jL/6umDEc0
9kK4xYXATGW7yYppOZevx/B7zD9L3OnvhnS621lP9iBEw/Pe/EJ05Y2mT/O9/KPnhh5odPEEQHNy
yOAUHByMhIQGxsbHIysrC0aNHkZWVhdjYWPz0009UcoVwSKZNjFPlXXXM3bsQ3rhh2WbFYhgojxh
Rx4kOlOLZcHNpl3AvERjYsSwBWHK+CFqjnX0MA+3//R+nSbx1KxiKLSWIWsXtPE6+vr545ZVXamI
sRAPGJji8CqkI+GkIjN26AXK5g6MJou4hFTIIkrDI0NmuxjOywPlcHXoESm32GQYMgLFdOwivXAF
gjnWSrl6N0oULa3zMBEGYqVqELkG4SHpJ9WUNpzIrREMgRGbf4gQADoxRZqvTjBmcJnF8PKBWV9/
ACIJwSoUWpxdffBEffvghIiIiMGnSJEv1b0f8+OOP1TogogFRbekIWJYSXxINgmlN9Zh1SWhXI91
VG+GoNoM+NhamBQsgyM8HAAgKCiD59VfopkypsbESFFOhcIpIiICUqnU8v/SVpAMdFwokg+OiMLHKsanU
xAAIrmY6AuXULgrt3LdusVAojlf8h6iHdfUy4OiYYJ7N12H+vFKv/0Vj2pRbZzyQOAPD0hG7yZMi
+/NLSJFm+HLrJkwF6PhNEjVOhcPrwww8tf3/00Uc1OhiiYcKvURfoIYBIULkHPN9NZ+zRA5DJKj0
2gniQBHkKMayZh02eswrzOr30EqTffAPGaD5PeO0ahIcP0+pSgqgFXP7Zr9Vq0adPH+zfv78Gh0M
0RDKqMRUBuemIhkiEkvsb9oajEiz3YcPCoH/mGU6bhEpeEUSt4LJwkkql8w6y5Ytw6h
Ro9CzZ08kJSVZ2nU6HREbjExMQgISAD8FC6HT6ez2z7Is4uLiEBMTgxEjRiA
uLs5+Qjmi1rBJflnZrOEU30Q0UCK8uMIppQKLEwCza84K8datYHJzq3VcBFHXKNKbsP9eKS7n6+2
n7agF3HqDxcTEYP369TAYKv5SO6Jv376Ii4tDly5dy5dO1r166FRqPBxo0bER8fj/Xr1+PixYsAgAs
XLmDDhg2Ij4/Hxo0bUVxcjHr4ntntPykpVcCefOncOvv/6K9evX4e3elx0tUneoKDBckJ0N
gVRWe9fQ0pyIgiHpOE4WQU5Ylq8SEIr3J8QkwJ301Nm9u2WZ0OogdPBcJor7Dsiw23yhB9O8ZiEn
KxeN/ZKHvlqwHIp7cyuNUXFyMlJQUDB8+HN27d4d4e/vz9nP8MwFWYPj4qKstu+a9cuzJs3DwzDQKl
UYsiQIdi5cyciIyOxa9Pc1f+zd97xUVRbHP/N7Gz/wG7GzNpjCCCQkhQEAIIiCCCqg
UBaUX4SEo+MQWsMPjqdg16hMfKkqxF0CKjQ7yQAhVERApAQIhvWez2b4b749NdndmtqZukvv9fPj
ovVP27mR9txD7NmzXXZ5nwoQJkMlsQ3Gjx+IUEhSRJj3GBmdMUjhi0mDDlSM3uSHx27xUVRbHP/N7GzNpjdCCQkhQEAIIiCCCqg
UBaUX4SEo+MQWsMPjqdg16hMfKkqxF0CKjQ7yQAhVERApAQIhvWez2b4z749NdndmtqZukvv9fPj
ovVP27mR25txzzv2d997D7NmzXZ5nwoQJkMlsJQ3Gjx+P7du3Y++TIkf58VUID0lCGk8jbNGgQIJO
52ZtAaDkwNIXOwRJc0jh+K1kaC/pEeri/aRqm2bOhXLrU3iX74guYHn2UJIm3IE6WmrDyby1MWin
e7GRFpKJ+UZ3WytOHK7HmnEN2w2AFcrRWXNJY0NNFIe3GxC/D6ddff7UX8v3zX9F23zwzNxRUFC
AuLg4+/bFfXV37t0qzmQVnVcbTlCz1WlGkEtQjjdfiPQGL3JX74D1X9kZR4vzsLqjEvPt6V1AGE7
OVVb0ifRwAADzjBlQVPOoqqJoogCQzE5KMDFiHDGnMoRIaiEoTi6k7S1Bo4ABIsWtjIc5Pj4NcQgx
fZ3KrrTyjqZYIOYWkEL91vOuNT59oMBiQkZGNBqVOnIjIyEgMgMGDDBB5m1oymZmZLeq8ME3LJVVqY3LJVUAHMY
SW1GAzEzPYQgRLIvUGkO6lqyEBFS3oGtM7gcH5FrYcL4OEawUgGPmfORyAXqavdFdBk6FBF79tj
bxv/+F1kxMQ06KDx4FOIrTCx+PXXZoGo6ABOK+e5jAs0dVtOy9+PK/dzUj+/L
FRhlDcnKy221eDafc3Fw8+uitPZ/Pz3Bss/aF3Z4uf8nyjM/Pt/ep1Wq89t7/eeenmuMUxhmMIWkEPJlQHxGCBQ5Uj+oCN5+u4k6/+TcgDFRnxsJ0pjVwsHUlDx79Lm2s
AgAsORvu77waYpp9t1AVyPzggl8KG8Dpcb9ZiXX6lva2RhSEO5OdzeZjHHgOcDKeIvXshDQsDFx3
dsANuJNry/fBGbhkAvVMPheQundE1tGlDT4GEq/vh+EUdAMfzX83QWDemc7N55rwmhy9fvhw0TWP
lypXpbXXv38/12z95o0IGGzZMGzAmdG5Pqnu3/O/u3BGwnNpWfuvJitE5p/BDEUTxmFwjAzmknV4WIZ0hTWL3
tztKoUaOwefNmxbv5XEIIIyxj7ov9TfNHhwizGaCARfeQZvlTZjcnhIqE9nFAc9xA04lGgzK1YAOF
lbcpkguzLLxtOIt0bISGx2TlsDvXIOo/WWpuhtEENkKds5nJymnYNZ3o1nE6fPPo358+cjYU
JWLRoEQoLC1FSLUilyPs6smEVXmFlpWd0Yj+0r1tWrQqVlujYbrFc7ecYgGqKgCwz4d9jYWzbJ41Dkr1y7bSKs+HUxAwl1FzcX/S3nh+Ahbm+o
kffr0wcSJ79mxM5ZKyZF7mmxbMmAKUKpmD1p5yZKC4uBHtwXsD
QbuS7nUOk/t/wIsPp5pvrNS4CIdAQShJ403KyQ9Oiciuyzz4DrB7UxwnNzqFCIzQmcUjuWLFrqZ2
2TF41/17upG7eSbPXTy8pKbGHz2rp2LEjO5I5DaWkpoqi/i/PrAtLQ0pkWludy21Gl1iJBZs2Zh1qx
ZLrc5awaz2wRD0eFk/GEgXgitaKnuF+KGFYrWAOHOB1WW69tQFGRyAEDp3UEk
ooHZ1db6ORbWZRZDU+2/F9I9/QPHaa6D0trAPnZaPwcBEtmtibxMAHCsihpOPQI

HqX09RJQbAp/eXqQ+HaGu5Br4945CIg5JeIM+fRpUpSP3gw0LA9u7d4OMj0AIFGQSCp3U/BdClqe
adc6EhcE8eTL/fKtXN9TQCA0Mx3HYlu3acDpVZobB0raTw4UIQ3VxQc1rOPn0BktLS3OpGP7oo4+
K+nfs2NEwIyO0CnINfNt8YqL/sWlhmM46ZAhA11F9nEAIYJJCGFxxMpbOV5hxXYRvicLGuXMh++o
re1u6dy/ozEywbTTxOpA5V2HBVa1ro9jMAqfKTBgYI3e5vS0iDNU1t8fJq+E0b968phgHoZWSo+c
bSXVZLULCdIS2QpdgBntgtLefO1yJsZ19m2ywffvCMmAAmGPH7H2yDz+EYdmyRhkrwTNmlkO1mUO
YXDzJ2+4mTFfL0SJiONViZjkU6RtGC7Ch8Go4Pfjgg00xDkIrRehxSgj284a3WMBkZPC7iPAloZW
SKAhjl5tYv5SRTQ89xDecvvsOxsWLwcXGNug4CZ45WWrCpB0lKDFyeKC7Cu/eFGZPeakwslhzTsv
bv1uoBBcqHV6V48VkZV0tBTornAOX0QoasmYWCCXxDkKjkitIDk8I9i+/SXLiBKiqKnubjYoCm5L
SIGMjEAKNLiH8iYWM9i8n0Dx+PNj4eHubMhoh+/jjBhsfwTeeO1yBkprVxJ+e12Fnjs3DxHEc0g6
WI6ea70GZ0y2I1yYJ4g5E+U3N7G0CiOFEaERYjkOeoX6GE7N3L69tufVWUoeL0GoRGknhMtq/nEC
GgVFQ9kq+Zg3gtLiC0LgU6Kw4XMT3GL11QgMA+PR8NX66yg/ThckozOoWBBnl8Kvk6qzIrSZyEgC
QJzAy2zdzYjhADCdCI1KoZ2FkHQ/9UBmFcBfxfk+IDKdhwxpiaARCQBKvZkA72Ul5ehZ6P1dYmWb
OBOtUEovSaCD7/PMGGiHBG1uy9aK+P0ot+PhvLf51lG/AdgtlcGpKOwTLaCQH8Q2EjAIjCIEnRQA
Qw4nQiFwRCPj5621CVRUkTvkaADGcCK0buYRCR8GMWvg7csUPWXoM/akQk3aWoBBymB56iH/ejz4
C9OIXOqHh+emK68TvRUcqYXSyASgAXw6PQIjM9hrurOQbTj9cIX8vQByqIx4nQqsmSyM0nPy74Zm
MDHvVdwCwJiWBc8rfIBBaI10E4bpLXkqvFOqseGh/GU6WWrAn14h+GwtR9cA8cEGOvBm6sNCmJk5
oVMoMVhzw0VMkpQFnU2lENN9AyCgwwsISPSehFEGcqvnNluYfAaHVckWgU5JY3/ym4cPrPSYCIdA
R5jkJJyBCDhWaYHR6A1dbOPxUqYBp7lzefvJlywCdrsHGSRCz9ZrBrvzuCYYCOgczvL/1oDAWIU6
LJ8tNHA6TJHESqiMEFr/mGvBkRjlW/l0Fg4X1foCf1DdUx/zvf7w2CdMR2gKJAs/sZS+hujPl4qX
r31yshjEtje91KiqCbM2ahhkkwSU/C8JrfVyIl45PUGDfPdE4MC6Gl/jP0MDoeCVv3y1X9aAzMyF
//XUoH34YqqlTofrHPyD97juAbfhndiAiEr8koTpCc3G0yIgpO0vx6Xkdnj2iQY91BQ1uPF3R8G9
4f0J1VH4+JOfO2dscTZPCvoQ2gTBUd1njeXXVuQqx4bQ3z4RD1mAYBTp88vffB6qr6z/IVgbHcWC
5+oXFNCYWe/P4YbrlN4fhpliH8RQspbDilnD0ipC5XC05xslwUpgMSHn/DahvugmK9HTIvvsO0p0
7If3lF6gefhiqyZNB5eXVa8yBDsdxKNATOQJCgPDhGS2cb8cKE4eHf6uo98PDmSvaunuchN4m6w0
3AGFhDTEsAiGg8TfH6Wy56+2v/qGB6fHHwanV9j66pASyVavqP8hWxJFCI/pvKkT3tfn4/lLdQ5m
7cwwwOc09pRTQPZTB+hFR+PDmMLw9KASZ09tBybh/7d7RQQ6FBBh49RROvT0Bj2/9hJfn6Yz011+
hHjwYkn376jzmpiZfZ0WZwXeZhTIjy0uoD5ZS9mT65qT5R0BockoNVmx1UWBy8xU9nsyoaJDPqDa
zPJl8CQXRaiFPEBkCQlslQc3A2ReRW211W/TVYOHchvIOFJhw0KyGUbjCbtkyoKJhfuetgScPVeC
SxopiA4eH9pf79WJ3plbkshYOwOUqK4KlNGYmB+HBlGAoPBhNABAkpfHauV+wf/l9SC7J9vqZdEU
FgmbNahGep+Wnq9BzXQGSvisQKae7Q6hlFQjeJoAYTm2S9Zf0MLuJyn1+Qec1GdUXrgiqundSS8D
QPgr5WSxgdu/mdxHDidBGUDAUOjhNMjgAV7Wuf5MXKs3wtPDq56t6mB57DFxIiL2PrqiAfPnyhhp
ui6bCyOKMk8eOBbDNSx05V7Achz25/DBdOxXtl+o7ZbFA8cwzePKj5yC38sOvbLt20L/5Jqq/+AK
WIUP4x2k0UD7xBNCA0YKGpsrM4pU/NOBgu5+fPlSJCqP31JB8XeCJXwLEcGpzcByHrzI95zgcLaq
/8JowMdyfFXWSjAzQZWX2NhsWBuvAgfUeE4HQUhDnObk2nM5WeJ7klBtZcOHhMKal8frlK1aAKiy
s3yBbAafLxPlhhTr/PU6nSs0oNjhe8ioGODwh1mfVd6q4GN0eeQRyF2HUj4ZMw6X/HYJp/nxYxo1
D9U8/wfDkk7x9pDt22BLGA5TDhSZeGJMD8N1F77l2YikCYjgRmoGTpWb87TTDoikgJYx/M6qk9S9
pIpQi8CcxXPrLL7y2ZfRoQOpbkVMCoTXQRfB7cZfndFawok64iqs2sdw4fz7Y6Gh7P6XTQf7OOw0
x1BaNK8PppIs+b+wWeJtua6+AWurb65X+80+ohw9H8IkTvH6dVIHps97Go5NfwHGTwrFBIoHx3/+
G5aabePsrFy0K2JDdgXzxZPx4sXepBbEUQWCYLIExCkKjc6HCjOV/VWGxQPJ/RAc5ekXIeH3VDVC
Y+1Il/yQ+5zdxHKRbtvC6zGPH1n9ABEILQuhxyqpy7QURGk7jEvjL2S9UmsFxHKBWw/jMM7xtss8
+A3XlSv0H24JxZTj9Xuz/A3BPLj+8d0dHhZs9nbBaIf/Pf6AeMQJ0Tg5vU15UBwxe8DXW9bsLABD
ECCazNA39Bx+AUzg+h9JooHj9db/H3hT85kIUdGeOEUYvoleBqBoOEMOpTbDlqh4DNxfh+WMaHCr
kW/lTklQIF6xSKPch9uyNvwUPdIWPLmvJiROgc3PtbU6phOW22+o9HgKhJeFrqO5vQaju9g5yKJ1
+axUmDqU1v2fTnDlgnZT3KYslYF+0TYUrwymn2ooCP8J1FUYWRwVClbd3kHs8hrpyBUF33QXFyy+
DMvPHYLn5Zix8cxNOdkix92nNYgODTUqC4fnneX3SdetA5ef7PPamoNLE4s9S8XWuMnPYm+c5nyx
HkNsXpQgMkyUwRkFoVN4/XeV2W3IIg1BB4d0KU/0NJ+FMYVCs5wdJLYwwTHf77YBKVe/xEAgtCV8
Mpyozi2tOIXGaAnqESZEUyj/2Qq1xJZPB8K9/8bZJv/8e9F9/NdCoWxYmK4fzLjSwAN/CSLXsyzf
y1MJ7hDHopHaT08lxkH75JYJvvhnMkSOizcZ//hPVmzdDEhPN6y91M5k1zZ8Pa2KivU2ZzZCvWOH
z2JuCw4UmtwsY3NX1A2xG00HBRD9QKtAElOF0+PBhzJ49GzNmzMB9992HP/74AwDw8ssvY+bMmfZ
/t956K5YtW+byHCtXrsSIESN4+7dlrCyHky6sfQCIVdLoES5FuLxhPU4mK4dr1fxz9AjzLTlcmN9
kvvvueo2FQGiJCHMCs7VWmARhjfMCb1NSCAMFQ6GbwHDKrHTsZ54yBdaePe1tiuOgeOWVhhp2i+J
shdnt6uLf/TCcdglkCG7v4DpMRxUVQXXvvVClpYHS8pfjsxERuPjWWzCkpwNSKSIUPj6TJRKYHnu
M1yX77DOgstL1/s3Aby7ym2rZkq0X3de1vH2yivf3YShgVEeFbfWgsf4LmOqDn+XqG4+qqiosWbI
EH3zwAVJSUpCVlYWFCxdi/fr1eOGFF+z7abVajBzfddZfbc912221YvHhxUww74MnUWHh1rGg
AH94ShvYqCQbFyiGXUAiT8cNo9fU4XdVaeDODOBWNIB8SJenz5yG5cMHe5hgG5lGj6jUWAqElomJ
oxCkp5OttPyQOwF9lZvSLduQjCsPhKTWTk64eDCdIJDA8/zyC7r3X3iXdsQOSw4dhHTSogb9F4GF
lOWQUmhCrpF2G6Wrx1ePEchx25/DLrAxrLxPtx2zdCmVaGuiSEtE288iR0C9fjgqNBrV+pkg533A
uM7h/JptmzID8jTfs56aqqiD7/HOYFizw6Ts0BhVGFj9d1aNnuNRlflMtlSYOv+YZMLoTPzfvssa
CrzP5YqRvDQqFimKheGYR6Kws6NatA5jmMWECxuN07do1qFQqpKTY4rqJiYlQqVTIyMjg7bd9+3Y
kJSWhW7duzTHMFofQ2zS0vQz3dg3C0PYK+1JZocfJF30NTwhXAPmqZSLdtInXttxyC1ELJ7RZhEr

76y/zXyTCxPAe4bYVdWKPkyCHZvRoWATyHoqXXgpoHaCGwGjlMH5HCe7eXoKbNhfhhaPuvTJ/lpp
h9SEudKDAhAI9f78oZ6OnogLKRx5B0IwZIqOJU6mgf+896NatAxcby9sWIXgmuwvVAQCUSpiEIqc
rVjSbV8Zg4TDkhyKkHazAHb8Ui95Bd8XzPXIP7S8XJYm/eULDC392CZZgdjsrVNOnQ756NaR79kD
x3HPNds8GjMcpPj4eer0eR48excCBA3H69GlkZ2cjX5Do9sMPP2DatGkez7V//36cPn0aSqUSkyZ
N8uidAoDMzMx6j78pz+sP+y5LATiWKHemq5GZyVcN1lbSABw3c36lrl5jP5LLAHDMuqI4H87Hsuj
95Ze8rtxBg1ASANewoQiE+yFQINfChqfrMFjN4FCR43e076oGmRHF9vYfeXIAjpd0uKEEmZmFkGs
pAI4Z/N8letHnqOfORY+jR+1t5tAhFHzzDapuvLEe36buNMX9sPyKFL8V2J6FVgBlJvcv3Sozh12
nLiEpyPOLecV5GZxfoyoJB1nZVWRWAKH79qHzm29C5sLLpO3dG1lLl8IYHw9cvGjvr70OxnIJAEd
e6LVSDTIzxeepRTJ8OPr85z+Q6G3eL7qgACWrV6N85EiP428M9pRIkKtzndPaRcXitqBKbHX6bho
Ti19PX0LXIA5WDvg2l8H3V3SAk37+QslFKG9fAOmlS/Y++Zo1KAwPR5EXe6CuJCcnu90WMIaTWq1
Geno6Vq1aheXLlyMpKQmpqamQSBwPhjNnziAvLw8jRoxwe56JEydizpw5kMlkuHr1KtLS0hAcHIx
bbrnF7TGeLlBdyczMbJTz+svVi8UAHG7n4cmxSBYsWbaUm4HTRfa2gZYjOTkedUVTUgHAIW7Wr1M
EkpODPR4j2bcPcicjmVMoED5/PsJDQ+s8jkAiUO6HQIBcCxversO8Dla8e7nA3j5bLUFQ+y5oHyS
BmeXw9xG+Zs+Qbh2RHClDnJkF/nT8lnKNNOK7dOWLMSYnw7x+PaR79ti7kr76CtUzZwJU/XXc/KE
p7odjRSZ8nVvscZ9uoQwuOIU1f6mKwPR2QRgcKwPt4ppUmlj871ABbIFUG1/cFonrlDoonnsOsg0
bRMdwDAPjc8/B+sQTiBeEmZyvQ4HaCJxzGEpGRoXk5M4ex2+ZNQuSlSvt7fjffkPUo496PKaulBt
Z/HxVj64hDAa34xtJP+mqAGhcHjeskxr3pIbg6bOO+5oDhSE9u6DawuH+vaU4XMT3UA3JPoVHXk+
DpJj/92M7dkSTTMIaAAAgAElEQVT4+PEIbYZnScCE6gCgX79+WLFiBb766issXboUxcXF6NKli33
75s2bceedd0KhcK+RERUVBZnMNkvr3Lkzhg8fjhMCYbG2AstxOCVwk6ZGioUkwxo4VHexkh+qE64
QcoXsm294bfPddwOtxGgiEOpCnEqC/tH83+vWbJtHIaPAhCpBik7tr1gtpXm6aSznelWeUZAHyhw
5IqoR2RrQWVg8/Fu5xxVZIVIKExL5E8qvL+oxdlsJRvxSBHPNwWvOaTHkh0LM+rUUH/5VBb1TPKl
jkASjj++A+sYbXRpN1h49UL1rl01Py0tujjBUV+bDM9k0YwavzezeDaq01Otx/mK0chi9pRhpByt
w17YSfHWBrwCe5aZ2IgAkhzKIUUpEun6ZlRbMcWE0jT77G3Z9MEdkNFn69YN2zx6w111Xz29TNwL
KcCpxcmlu2LABarUa/fv3B2BLCt+1axxcmTJjg8RyFTmUEysrKcOjQIXveVFsjS2NFlZP+R6iMQme
1WEAszIWOE1eP2LHfOU6VlZD+/DOvy9TGV0MSCAAwJp7/Mv+lpjh3rQFVS7CUQnKYw8jymCBeg7V
/f5gFoRz5G2+0qlynEoMVs34tw0Uv9Td7hElxU6w4qRsAfi+x4PtLOvxZYsJThypxptyCn68akH7
SsTJOadJj449LoZ4zR5zLJJHA8PTT0O7bB+v11/s0buGqOl8MJzY1Fdbu3e1tymKB9IcffPo8f9i
ZY8B5p/vpiYwK6JyWv7kznJQSYE73IADA9VH8CcGac9U4IjCaBmT/hQ2fPwGlmZ+rZR43DtW//CL
KC2tKAspwWr16NSZPnoxJkybh6NGjSE9PB1XjIq1NCu/atSvvmOLiYp7kwp06vp06dj5syZeOy
xxzBu3DiPob3WzMlS/sqQ1EiZ/Xo6o2QoKJzsKQsHVLupxu6KyxoLZu8txYQdxThaaORVtKbAea1
TJ928GZTe8SJgO3aE9dZbff58AqG1MrYz37t+IN+ICiMrKkT73yFhvFBcsg+GEwAYFy3itZljx0Q
FtlsqBwuMuOXHIlHx3SGxYq97nwgGt8bJcXM718bTtmsGu9EqJKXgIo7/ZyoGblkr2mbt3RvaX3+
F8d//BuS+adkBrj1OrDeDlqJgnjqV1yVdv97nz/SV/Xn862nhgG+c6s5d0fA1/PaMjcY3t0cga0Y
clIzte/WL4l/ndZf4Cx+6lefi4FePIcjEnyAYnnwSus8+a3Ztv4DJcQKARYIfsTOTJ0/G5MmTRf3
R0dH4xinMk56e3ihja4kIVzO4CtPVEi6neZWoy42sz7WW0g6W40CBzUjbm8efbcXKOSiE5QIECMN
0pnvvBeiAsukJhGYhOVTKy72xcMA7JzXIdhK+lEuAkYISH8KVdRcqXS+9t/brB/Po0ZBu3+443zv
vwHLHHU2e69SQnC4zY8KOEgiVVSgASweE4v1TWp4h1CWUAU1R+Gl0FH7NNWLtpWpsuOzYfqbMhGK
92OtzQ/Zf2PHJPxGp46/Q46RSGJ97DsYFC+pUZ1MuoaBmKGhrJrAsB2hMHMLknv8mpsmTebczJE
joK5cAZeQ4PcY3LHPhS7TzhwjHkwJht7CIVfHF2W9LkKKGyR8Q6mfwOPkvIIu2KBFxpePQlrCD8/
p33wTpvnzG+Ab1B/ydmrFCItVejKchOG6CsGKE6q8HMz27ZC/8QaUc+dC8eyzYHbuhE6jtRtNrui
k8DxLkhw8CObYMV4fCdMRCA6EXqcPzvBzSobFyUU6aUKP0wU3CtkAYBB6nY4cgeTgwboMNWDYcEk
nMppo2IRF+0TIsGxImL3+W4iUwv01ISSaonBHRwXeHxwO56SGy1Usjgm0nQZf/gN7VswVGU3W5GR
o9+yB8emn61WcvC7hOq5zZ1HxX1kDep3ydVZeAn0te3KNyNFacFVQIqVDkIS/KKGG1EjXnj0ASP/
pXURe4a+yND7+eMAYTQAxnFotHMe5CNV5MJzcqYdzHGQffYTgXr0QNH06FG+9BdnGjZCvXImgqVM
Rk9wVn6x7ERHVFS7OCsQrPfzYOQ6KV1/ldZmHDWvQ2RGB0NIR5jkJucvF9uRQ/m/9ZKkFeovr3yL
bty/MgnQG+bvv+jnKwOKEwNtOA/h0WDgOT4iFXEIhSiHB5RlxyBgfg8x7HSGkWoKkNK6Pdu8VufH
KSez45J8INfAVwE3Tp0O7dy/YPn3q/R1EWk4eRDCdEYXrvv++3mOpZb8bFXCWA77M1CFLkEvmLk0
jTE4jKUScbzss8wjmH+Ibeqbx42F46aU6jrhxIIZTKyVba0W50fFLD5J4TtJ2KYJpMkGZlgblv/4
FSqdzeZzEaMA/D2/A2TfvxozffxEllsYr3XucmN27wRw6xOsT5lwQCG2dflFSDI1zP0Mf1Um8yri
9iobK6efOAvg6s1q0Xy3Gp57itaV790JSU/KqpcFxHP4UTBo7qSW4M17J837IJRR6hktdekQAYIi
b+pqxmmJs+2oh1C7yb/QrVgBqdT2/gY26rKwDAPP48eCcPF2SzEzQDaSTtS/Pvajml+e1ohXVicF
i46gWYZ6TyqjD6nUv8vqsXbvarmmApW4E1mgIDcaJEv6Mywq4rcsEiEN12kotgiZMgOyrr3z6vBh
tGb75+jls/+SfSCy5Zu/v5M7jxLKiGlnmkSPbRNkHAsEfKIrCF8Mj0SNU/BK6PpJBO5W4n6IokUL
zjx4KqloHDYJl8GBeX0v1OmVVWaFxSjUIYoDDE2LcGkjuEOoTAQBjNWP9F08hvKyI129YsgTGF15
o0LywyDqE6gCACw+H5eabeX2MUw5bXeE4zq3HCQAK9Bw+Pcf3wHlaGHS9wHB6ZdtyJJU63h0cRUG
/fDmg9OxxbQ6I4dRK2SJYrmxhxTIBzgg9Tjf893UwgjwHLiQExrlzoX/3XRjnzgXboYPoPKPOZ+C
v9PF4Zs8aMFaz2xwn6dq1kJw6xeszLFni8TsRCG2VMDmNzaOj0U7J/50OjHG/UuuxXnzR2QMFJo/
12YxPP81rS7dsAX32bB1G27z8WcL3NvvWLkolCcb4wKFYGoRn09k/v4NbLv/P6jAsW2LSZGhjhM7n
UYHWzpxjLnXfy2tJt2+o9nqwqK3KcVkwrJMCojvz7L0vLN+4SPUQ5nBPEk4uuIO03wSKhBx+EVZC
vFSgQw6kVordw2CZYOttORXsM1TkX+h149RT6b/qct92amAjtrl0wvPsuTHPnwvDuu6g6cQIfTXk

```
CBkbgcjUbkP7Lf3DsP9PQ7fJfos+qLXjpjGnCBLCpqb5+RQKhzRGnkmDzyCjU5oHLaODpVPeK/H2
jZBgs0CZacUbrZm/AMnw4LH378vrk771X9wE3E8L8JqFnw1dCZTR6Rzhe7iPOHcTC/V/z9rEMHQr
D88/X6fzeEIbqyv0QJjaPHs1rSw4fBlVWVq/xCMN0N8bIMTPZsyyAp1Bdn0gpap2AL+74CAzrMMr
Y+HgYXnih7oNtZIjh1ArZlWOwL2MFgEg5hWMTYz26qmtnN4zVjFXrXgTtlKtkTUhA9Z49YJ3E1QD
bktt/DXsQfZ7ZjL1dB4jO2TfvPHrPfQDKBx4As2UL6DNnIFu5EqrZs0FZHN4vjmFEKsYEAkFMSoQ
Ul+6Nw/o7IpE1Iw7RSvcvJgB4pBc/32b9JR2uuVN2pihxrtOGDaCuXKnPkJscocepr4dFMd4YXKP
rpDLq8Mn3/ARltmNH6Nas8aoCXleEoTpfk8MBgIuPh7VnT3ubYlkwO3fWazxCGYJb4+QY3kEBT3e
gJ4+TiqHRLVSCXvmZuPfEVt42w9KlDZYr1hgQw6kV8sMVfphucheVV1d17aq6p/Z+jj75F3jb9Mu
WgYuIEB1TZmRRaeKQGZOA2x75DPdPfxVlqhDePhTLQrZpE4JmzkTwkCFQPvssKLNjRshRFPTvvw+
2Wze/viOB0FYJkdEY2UkhkiBwxZ2dFEhwmvVbOGDYz0WiavT27WPGwNqjh71NsSzky5bVf9BNBMt
xIv26unqcAGBITZ7Tizs+QmJZrr2foyjoPvsMXFRUnc/tjbomh9diFhS3r2+e09FCvuF0U6wMwVI
aN7pRXI9S0Aj2co+O6qTE0u0f8ibquq5dYR4/vl5jbWyI4dTCMFg4rDhThRePVaLcKI556ywstgt
UhScmek+uC5fTaFdZjBd2fszrN02bBuuwYS6P4akRUxR2DJ+Em1/cgm/6jfH+RWrQv/8+zES3iUB
oFCQ0hfk9+TP3ciPnPt+RpmF84glel+zbb0Hl5bneP8DI0lihEZSZSvAQLvLG4FgZ+uacxZP7vuT
1mx56CNYBYi97Q1LX5PBaLIJwnXTPHsDkXnPPE0YrhzyBAKiqRgfrtg6ua8d6CtPVspjKwuRTu3h
9eQ89FHCr6IQE9ugIIh45UIbFRzV4/y8teqwtEBlPO68ZoXMK03VQSTAgxvuMK0xG4+n/fQaV2WF
0sRERMLz2mttjhPWfBsbI8MW0Hlj44DsYO/8TXIlxX82bo2no33sP5vvu8zo2AoFQd26N4yfw0pR
naRLzpElgOzt+u5TJBPkHHzTa+BqSE6XCMJ3rMlO+Eimn8e2Pr/Dyb4ztOzTJQhZhcniZH6E6wKY
Kz8bE2NtUVZVowY+vOJfRAgAJBaSE20Kgt7V3vUDBW6ktAAh//z+8tqVvX1QMHVqnMTYlxHBqQRi
tHH7Ichg2RhaYv7+cV5B38xW+3tL4RCVoHx4c0doyzM/gC48ZFy3y6Iq+JNDs6BrCoFuYFGemtsO
S5yZCdeYPnFu1CsYHH4Q1MRFs+/YwjxhhK3h58CBM99/vdVwEAqF+CAt7sxzgcWU+w8C4cCGvS/b
pp6AKChphdA3LnwIZlvrkNwEA88svSLl4ktenf+cdINh9Un5DUd9QHWgallGjeF3M1q1udvZMtkA
R/IYoh/5VaqQU4S5KwSR4Ke5O5eSA2bKF12f8179aRKkfYji1IC5pLBD+dHbkGLHqrE3YrkhvxXb
Barox8a7dqEI6ffYxr6Bifmg0TF68QcLCoV1r1IrtwnJSCbR9+8Lw9tvQnjiBqr//hu7772H897/
BpqT4NC4CgVA/gqQ0YpxkDFiAt6zcFaZ77wUbF2dvUwZDi9B1Enqc6pPfBKsVitdf53UZ7xoD6q4
73RzQsLgK1XHeCv0KEK6uY3btEokU+4JzbUQASHDyJkloCkPjxO+ZLl48TrLPPwdldZzX2rMnLAI
F+0CFGE4tiPNu6k0tPlKJ/XkGrDijhfOkhIItxu8NqrwcoZ+u4fWlD7sfVpnnat7CPImuXmYYBAK
heRB6na66W1lXi0Ih0nWSff45qOzshh5ag8FyHE4JEsP7RtXd4yTduBESJx0rjqJgev7fdT6fv6g
YGgqnP5uJBW+1tC9Yhg4FJ3MYj5IrV0BfvOj3WLKr+IZTvJr/rL+tg/hd4THHyWiE7PPPeV2mefN
ahLcJIIZTi+JcheuHnRXAxJ2lWHWWr9ESpaDtXiBPyD7+GJS2yt4uUkfgk8FTUSmskun8mSyHy1V
CjxMxnAiEQCRBMPu/UuVdTNE0axbY+Hh7mzKboUhPb/CxNRTZWiuqnBLDaUAkGOozZjPkAm+Tecq
UJveUR8r5xoe/eU5Qq2EZMoTXVRdZAmGoLl5gFA1zkefUIci94ST98UfQJSX2NhcSApOgxl4gQwy
nFsR5N4YTYFtm7Hxvh8koHJ3oQ5kBjQbyj/kr6d4dNht6mRIVJvezm2vVVjjnpUfKaVEyI4FACAw
6iwwnLx4nAJDJYHjuOV6X9Lvv6uSxaAryXIQfL/tgILpC+u23kDjpVzWX1ly4ou4imLVYRo7ktaV
1MpyEHieJoM2gX6TjHqMBjxNv2apVvLZpxoyA1m0SQt50LQhhqK5flHsPT1rvYITLvS8Hla9aBaq
y0t4uVYXioyH3AvD8IxWF6Yi3iUAIWIRL8n3xOAGAedo0WJOT7W3KaoWikZSy60u+jv+dlAzlcfW
gW8xmKAT5XKZZs8AmJtZneHVCmCBe2gCGkyQjA6iqcrO3a4Qep05B4uu6emgEQmUUJJStqLK7aAf
9559gjh3j9ZnmzfNrPM0NMZxaCGaWEy3/X3tHFHqGi2/gYCkwt0eQ95NqtZB9+CGva9nQWdAqbMd
WeJgxCMOGxHAiEAIXUahO64PHCbCtsBN4WqTbtoHZvbuhhtZgCA2naV2Ufhf2BQDpunWgnXK5OJl
MlO/VVETWU5IAANikJFiTkuxtymwG87//+Xy80cohX8f/3I5q8aS8S6gUF6bH4bdxMTjqoVKF7Eu
+Jpb5ttvAdu3q83gCAWI4tRCyNBaYne7ddkoaMUoJPh0WAWGEbGKiCqEy739a2WefgXaqX1StUmP
5LQ4xSncep8OFRrz5RyWvL8HFD4lAIAQGwt+nT6G6GswTJsAycCCvT7F4cZ3FFIWYWOD1Pypx19Y
ibLis836AG0Qvdx90hERYLJD/h68tZJo1C5yLguZNQYSw7EodPE4ARKvVpLt2udlTTG61Fc5JG3E
q2q1RZF9R7c5g1ekg27CB19USZWmI4dRYsCykX34JGI3e9/UBoYene5jNDdojTIo3B4Xa+xkKWNT
XB40RvR7y5ct5Xb/ePRuVSkfJFFeG0748AybsKIHwuVs7HgKBEHjEqSRwnkuVGzlU+PoSpijo09P
BOa14kmRmQvbJJw0yth8KGKSf1CKj0Ix5+8pxsqRuBlmBwONUl8Rw6caNkFy+bG9zDAPjggV1Gk9
DIBLBrKvhJAjX+SNLcE2YGK6ue3RB+tNPoDQae5uNjhYpnLcEAspwOnz4MGbPno0ZM2bgvvvuwx9
//AEAWLlyJUaMGIGZM2fa/7nDZDLh5ZdfxoQJEzBhwgR88803TTV8HlE//ABVWhrUw4ZBUvM96sO
FSqHh5Lh57++uxi+jI7GobzBOTIpFnIv4sxDZp5+CLiqyt7mgIBydyo8zOz9YOY7DJ39rMXFHKfS
C9IgIOYVRnXzTiyIQCE2PhKZEL7yrvobrALB9+4pU/hVvvQU6K6veY/sqhz+ufx+rdLOnZ/IEhlN
7D6u6XGK1irSqzDNmgHNaWdjUCEN15XUI1QGAZcgQcCqVvU3n54M+fdqnY696SQz3B9lXX/Ha5un
TAWnLm3QHTGJKVVUVlixZgg8++AApKSnIysrCwoULsX69Tc36tttuw2IfVjV8++230l02LRpE6q
qqjBr1iz06dMHvXv3buyvYIe6dg2d/vtfAIDk7FkE3XEHTI8/DsOiRYDSe904VwgTw3sIPDw3xyl
wswsRMpdoNKIHhGnuXMhiooBsx2ygvCbHSW/h8NiBMmzM4otrAsCkRAU+vDm8TrkEBAKh6UgIlvD
yJK9UWZEa6fvxhuefB7VpM6RVtmcEpdVC+cADqN6xA5DVXWiywMQ3Dn4raCCPk8q/F7z0xx8hueA
ocM5JJDA8+WSdxtJQCEN1RYa6rRKEXA7L0KGQbttm75Lu2gVjnz5eD/W2os5X6MuXRSVfTLNm1el
czU3AeJyuXbsGlUqFlBqdjMTERKhUKmRkZPh1nh07dmDy5MmgKAohISG4nh48847sb2eVaH9RfHKK5B
UV9vbFMtC/v77UN96KyRHjtTpnOJQXd1tXvmHH/Jymzi1Gsa0NIQJZjcVRpsr9+lD5S6NpigFjY9
uiYCCCZjbiEAguEGYIO5VBFOANTISL9zDNySYEyegWLq0vkMTUeKngcBxHAoEOU5x/hhOLAv5O+/
wusxTpoBLSPBrHA2N0O085oBRqv/yt8AYBaWX/FRlkCk4VTHUJ1UEP2x3Hgj2G7d6nSu5iZg3nj
```

x8fHQ6/U4evQoAOD06dPIzs5Gfn4+AGD//v2YMWMG5s6di60e6u0UFBQgzqlUQPv27VHQxDWW9On
pKBkzRtQvycxE0J132lay+SF7b2U5ZFYKPU51u3mp4mLIBSvpjI89Bi4qCmGChPISgxUcx+HHK3p
ev1ICvDEwBGemtiOeJgKhhdC5jpIEtezPN+LNGyZjU+87eP3yjz4C4+TJ8AedxXXoaW+uf7mhlSY
OeieDQsVQCJH6/mxitmyB5O+/7W2OpmF86im/xtAYCL07eqtYCsZXLHfw/26SY8dAOU2g3XGtITx
OFgtk337L6zJ5SLkJdAImVKdWq5Geno5Vq1Zh+fLlSEpKQmpqKiQSCSZOnIg5c+ZAJpPh6tWrSEt
LQ3BwMG655ZYG+ezMzMwGOQ+PpUtRNmIEEl5/HTKnXCKKZaFcsgS6337DlSVLwCm8h9dy9BQMVke
IL1zKoezaZXi/5cV0evddhGgdCuPmsDD8PXo02MxMVFfQABzj2ZdnwK+nLkFrcXw2BQ5rehuQLNM
h+zJ8olGubwuEXAcH5FrYaMrrIK+SAHAoPP9dqEFmZrHPxy8/KwMoBnOnv4wbcs6gc3m+49zz5iH
r889h8NNDc0VHARCnL2w+W4y+rO8hu0vV/PNEMVZc9FWok+OQ8uqrvK6yESOQBQBNfJ8K7wcrC9B
QgoXDCNTmXUFmifBI3+jZtStUNdeFYlmUfPstygSeKCGXyhVw9rGwpTnI1Pnn9Qo9cACh+Y77xap
U4lyfPmDdXN9AeD4kO+mXCQkYwwkA+vXrhxUrVtjbU6ZMQZcuXRAVFWXv69y5M4YPH44TJ064NJx
iY2ORn5+PDjXLR/Py8tCuXTuPn+vpAtWVzMxMxM6ZA/2ECcALL0D2xRe87ZHbtyO0sBC6tWvBxcZ
6PNfFbD3gZCb1jJQjObmj32OiL16EeuNGXp/l2WeR1LcvAMBYZgb+chh5JpbCJWksAEfe04BoGe6
63vfPzszMbJTr29Ig18EBuRY2mvo6GCLNwDnH77uYlSE5mZ/4XG1m8fbJKpwqNeORXmrc0dE2kSr
UWbHvoM1zX6EKxb2z3sb+D2aDYW3eCKa6GimLFkG7Zw8QFubzmHLzDABKRf3HtTIkdY0H7WPtsmu
5/PN0ClMgObmTT8cy27cj6Px5e5ujKMiWLm3ye9Td/ZB0ppBXUF0a2xnJdSxeTN99N/Dee/Z2x1O
nEPnYY273N1k5FB/I4/Xd3CvJ70iD6qWXeG3rpEn2946QlvB8CJhQHQCUONWu2bBhA9RqNfr374/
CwkJ7f1lZGQ4dOmTPhRIyatQobNiwARzHQaPRYNu2bRjlxaJuLM5XmLHgNIsHp76Ia198Cy4khLe
dOXEC6hEjQDv9aF2fh++aFSaG+wTHQfHcc6DMjpAf27EjTA88YG8nhTBwdsJyAO8HCwC9I+tRbZx
AIDQbwkK/2VorrCzfc7DsdBWWndbi1zwjpuwqxdky2/Pim4s6ONeXPZR4PZ6+hy8KKbl0Car77wf
MrouRu0IYBqqlSM/idJnv5xGKX7b3Nb+JZSF/801el+Wee5q8Jp0nhMXThSus/UGo58Ts3g1Y3Yd
s/dFwcgdVVARGkGfcUpPCawkow2n16tWYPHkyJk2ahKNHjyI9PR0UReHdd9/F9OnTMXPmTDz22GM
YN24cRtTcAMXFxTx5ghkzZkChUGDixIm47777MHnyZPTxYeVAQ2NigRm7S/FFpg5fXtDjuqq+KNu
1B9bu3Xn70dnZUI8c6TFp/GQZ32XdxVPVaTcwW7dCumcPr8+wdCngRy6QfCpUMhR6R/KNsu8v8/Obrgt
veUtHCQQCECKjeSU8zCwQ+UUebtxUiIP5tpyiH50WgXAAnjlcAZbj8MX5auHp8P6ts/DL0Em8Pun
evVA+/LDHl7EzuS7qy9Xyqx95TkLxS19X1EnXrwfz55+8PkMzqYS7o5ugKoNwMusP1oEDwYU6dP/
osjJIfv/d7f4NkRguXbcOlMVxHmu3brAKBFVbGgEVqlu0aJHL/nQPFbmjo6N5Wk1yuRxLG2GVh7+
c0NC45JR8qTFzOB/dGb127YLqgQcgdSpZQFVWImjKFGh/+QWsCyPvSCHfcIpU+Gnv6vVQCqQcLEO
GwDxpkmjXG6Jl+LPUMdMTFvrtFRFQtwyBQPCDeDUtElE8X2nB+J0luDojDhcEiccHCk348C+tSMs
HAEBRWDjtRYzWXwNTs6gHgE0ZWi6HfvlygPb8rPJkOO3M0eOJPj6I+UIsReDTijqdDoqXX+Z1mca
PB9uE0jW+kBwmNJx898SJYBiYb78dsk2bHF07d7o1ZIR/907+JoZznEi7yTRrFuBjCDZQCSiPU2s
io0x8g4XKaCAkBLrvvoNp9mzeNkqjQdDEiaLK48V6K/IEs6nRnfzTgpK/8w6/9pJEAn16usubt1+
UZ49ST+JxIhBaLO2Url98ZhbYlKV3ue354xqX/QBw1SRB1Rdfgu3cmdcv++YbKJ580qvnyZPhdKj
QjLxq37wrQvHLOJX3V5t8+XLQeY78HU4ms3nhA4xkQagus6LuHifAv/IrQhX39j5cV2ckR4/ytbE
YxiZ62cIhhlMjcahC/ICyW+9SKfTLlsHw7LO87XRJCYLGjwd17Zq973gx/8ZNjWREekuekBw8CLl
TMiAAmB58EGyvXi737x/tPocpMVgCtZTcMgRCS2VB72C4m+sfKPC/PJSFAwpDo6H98Uewgnpu8s8
/h2rmTKBaHOarJceD4QQAX5z3rXadvx4nKj8f8vff5/WZHn642XWbXJEsCNVdqrKIctP8wXLHHfz
yOSdPgnIh2aOzsFh3iX/9oxX+eZxkX3/N/+zRo8FFR/t1jkCEvAUbgWtaC7J04kvL09+gKBgXL4b
x0Ud5+9A5OQiaMAFUjYSB0HAaGCOHz1RUQPXQQ6BYh8eKjY21KZi7ITmUQbAb/ZPrIoi3iUBoydz
UTo5Tk2Px/uBQ9BGE3bdki0VuhTzZW42e4fzjcqut4BISUP3jj2AFK4Sl27cjaOxYULm5onNxHCf
yOP2zh4rXPlDgfUyAn6rhHAfl00+D0jmMAjYystlVwt0RoZAgyik9w2gFrnkxOD3BRUfD2q8fr49
x4XXacHf4pusAACAASURBVFkP5xQnGsB93YJ8/yCtFtLNm3ldLT0pvBZiODUCe9wkNYqS+igKhld
fFd1MkosXETRxIlBRgWPF/Hj2AA8eIR4cB+UTT4DOyeF16z/+2ONyYZqicL2bpa7EcCIQWj6dghn
M7q7GvBQ1r7/K7NmLcWcnOZ6/IQSd1GLDCQDYrl1txpPA88ScOIHgwYMhdcqrAWyildVOS/WUEgo
P9eTnNGUUmkVGkRAry6FQ77tquHTDBki3bOH1GRcvBpySpgMNodfpQn3DdYKiv1KBijjHcVh9lu8
pfKRXEEL9iHZIN28G5aQZyMbFwXL77XUYbeBBDKdGYFeO61mSy9UR/P65b89RdUU6smUwjR/P65b
gXE4Fr99Xw0m+bBlkAmvf+PjjsAwf7vXY/tGuDaReJL+JQGg19PFjIkQBWNg7GBRFoaOgeO41J7c
E26MHtLt3wypIsKYqK6F64UbsHChSRC7n7JPFV
QAMUzz/D6LP37w3T//R4/o7kRGk6ZdVQPr0VoODH/+x9gckQ3jhebccpJDoICRIa2N4RhOtOMGQD
TOhYXEcOpgTFZOezPd+1xciuVL5FAv3IlzAJJfOmxY/hqZRpkFtsNHSmnkeCDFIF040YohIJjffr
A8PzzPnwDoB/xOBEIrZ6UcCk8pSxuHBkJlQSgKVu5lr41z4UOAsMpt9pWmulqlQUcx4GLi4N261a
YBS9nwLbiTj1wIKTffYdcwVL32vNOSuQvfnGXtF6LUMOpnbsEZqsVygULQFc4JqOcXA79Rx8Bkro
Vrm0qRIZTRT1W1gGwpqaCjYmxt6mqKkgOH7a3V53T8vYf2UkhqnXoCfr8eTACiR1zCy6xIoQYTg3
MkSKTW5d3lsYCs7ukPpkMui+/hOWmm3jdIy9k4Lsvn4HEakH/GBkoL8s4JQcP2nRUnOBCQqBbs8b
nCuY3uPBqBUupOlfFJhAIgYdcQrkV0+0YJMHtHRTImtkeB8bF4MiEWLsXR+hxOldhwaDNhUjdUIj
U7wuwO8cABAdDt3Yt9K+9Bk7w3KHLyqB6+GEMnjUO/bNPOz6z5vkyXmA4HSky4ZKHJfg+iV+yLJQ
LFkC6Ywev27BkSYsoNCsK1dVDywkAQNOi2nW14brcais2C/T7Znfj5555Q5QUPmQI2C5d6jDQwIQ
YTg3Mnlz3yYwWDsj2VFhTpUL12rWwCrScJp7ejR/XPI4hQZ5rNzFbtiBo8mRQTi5XTipF9ddfg/V
Dwj5OJUEHwcOnV7jU5/IHBAKhZZAa6dpwSqnRDpJLKPQMl/JCX0KP0948I85X2p5r2dUsJu8qxbR
dJRi7oxTJURNx70sboespXsXb8eRRHHtvOr7+6ll0L7xsP2+8msFAweRt5JYSmErLwezeDXl6OhQ

```
vvAD5K69A/tZbUP/yI2KqHFUnRInhHAfFs8+KX+YDBsAkWJwTqHQL5f+dLtYzVAdA5BGs/HkHDBY
Wr5/QwHnuT8HPwr5mM6Rr1/K6WktSeC2tI+AYQMzuFoRvM3UoMVjBgQIFwDltMVNjRlKoh8seGor
qTZsQdNddPP2LMWf345YFE4EN68DF8+tLgeMg++QTKBYvBsXxPVr65cthvfVWv79HahSD3GyHkZc
SRm4VAqG1kRopxdcu6qmmeMhnFBpOrtiR40hXWKfogh3//BbXSjci6O10UAb+5HLmH1tw74mtuHR
8LKRT74a1WzfMoziEnL+ILqU5uPHqKdx05SSiH3FdVfxuAIUAzsZ0wbaUmxF0951A78EATYM5fBj
yN98EI1DHZjt0gG716oAP0dUSr5ZARtsqUgC2kjQVRtYvaRohluHDwUokoGu0tmKuZqL3B8dxJpS
f3B+toJEc6nuaBrN1K+hiR/FoLiQE5nvuqfM4AxHyNmxg9FYOVWYOLCgoJcCw9gpsu+Z4UGRWWtA
30ooyI+tWTJKLikL+uo0w3z4aiWWOZbwhF86CGzwYxvnzYXroIXByOSR//AHFK6+IHgwAYHjxxTq
LjaVGyLA12/Hw604MJwKh1eHW4+TBcGqvkoAC4I+SUAXH4Ph9j+GG8eOgWLRIJLpIcxySd/8M7P4
ZAPBgzT9/SCm6jJSiy8C+LwEPVVPY2FhU//QTOIFoZyDD0BS6hDA457SaLrPSggEx9agdGhqKy9c
NQNeTjtym4Sf34q9b/2FvJ6glODg+xq/6dPJVq3ht0+TJgMq/UF+gQ0J1DUxSCIOOagkUNIeOagY
DY/gPoGWntOi1rgCDfyjC/P1lLs/BcRyeyg7CkLSvcawT38VNabVQvPMOQpKTERofD/X48SKjiZN
IoHv/fRifeKLO3yOttxqhMpvHLFxGYU53P/Q7CARCi6BXuNSlIKYnD7NMQiFG6f+rI1dnBZuUBN3
336N682ac7dD0uUVsVJRNMiEpqck/u750CeF7x371kBbiKwdTh/Ha//j9Z1779RtDEeSH6DF99iy
YAwd4faa5c+s8vkCFGE4NjFxC4cC4GHyWasCBcTGimVupkUVtAGztJT0u1KyOuFhpxreZ1diXZ8S
Kv6ux/rIe+aExGPrYF9jY5w74CqdWQ7d2LcyCki7+omRoXJgeh4PjY3BuehwUDLlVCITWRpCUFhW
RpQB08+JhFiaI17LguiDEKmlIaUAlOIVz0V7TsGG4/ukNmDL7PzjZ3jcDykrRKEhKgXHOHBhefBG
GJUtgfPhhnOjcGxbae8jNPGYMtLt3g+3Rw6fPCzSSQ/jvkk/PV8NSDwVxAPgudRSvPTD7L/QovAQ
AuDFGijs7KVwd5hbZmjW8tmXwYLdVKloyJP7SCMglFLoGcbb/hni+xPvyDTCxwO0/F0FQexMAoJc
pMXX2e3jx+Hf4966VoEtKxDvVYB47FoYXX/QrEdwTtYmhBAKh9ZIaKcV5p1VaCWoaKi8TpQ5BEvx
eIl7pNi5BhX/1C8UljQVZGgtm/urwqu+4ZkCR3opXjlfiYKEJRkiwoe8obEgdiQkXD+I75gzoCxd
s9TotFnBxcfhLGY0N8s44lNAXR+N7gw4OwplpcQiu8YLkVVvRr1sBgg1ajDifgbvP/A//yD0GSVU
VwLLgpFJYb7kFhqeeAtu3bwNdsebhkV5q/PcvrT1EWqhnsSXbgDHxCvxZakZCsARRfpRE4TgOGfJ
Y/Np1IG676CjSPPvoj1h895OY20PtdRU3D40GMmFS+IP+BlxbBsRwamQ6BzNgKNuKOlcYLMDXF6p
dGk0AoKKBNbdH4bY5z6DK8Chkq1dD9vnntqK9SiU4pRLWAQNgfOopWG+4ofG+CIFAaJX0iZRivdP
y8yIDB6OV85jX4ipBPFpBo2+UbfVtz3ApEoMZKCUU9DXqlLk6K7qtFddEA0VhS/ebUfmPqaLPDDe
yWP59PjS1NpoZ2JtrwD0JtpyZ2pJUVQo1NqWOxNXb78KEu/llX1oLsSoJ/tFVia8uOv5W756swut
/aHC+0oIgBtg0Mgo3xvpWlqvEwKLKzOGLAeN4htOs33/GsklPYFyCf8XkZWvX8pXC27WDeexYv87
RUiDxl0ZGSlMehcOyq634s9S9zMCifiG4M15pe6AEBcG0YAG0J05AU1oKTU4OqjIzofv2W2I0EQi
EOjG2s5L3IjCznHux3hpcGU53dFTwJEuUDIWh7X17iVs41wLBYXIat3Xgh4syCh3Py98FtTwH+FP
LswUyvxe/JM2pMrPdW1htASbvKoXR6lv47nLN9d6YOgJamcNI6lBZhHOx5/1KCAfLQiZMCp8zB5C
2zogFMZyagGgPiZRZGouo0GUtkXIaD/X0T+aeQCAQ/CEhmMHGkZEIkVKQ0kBntQRJXlIMOgaJt4/
sKDZafMuR4dBOSbv9TKGmU7He8bw8XiIwnHyt5dlC6RUhxZB27r+j1uzd6K3lco2mYLU8CBtS+Zp
OqnVrXR3iFmbrVkgyHboWHMPYDKdWCgnVNQE3RMlwqNC1V+nvcjPydfw43ZmpsSjSsegZIfXP6ic
QCIQ6MLyDApn3xuGSxoKkEMbrcyfORVmTwS5CRCM6ujac2qtovDwgFBaWg660ADP7d3H7mb0j+Yb
ChRqxTQvL4YQgz8pV1YPWxj9T1DhY4GZFNmyGry9cdjKwvhgwDnOO/WhvS3/+GYbSUnCRkd5PxHG
Qv/cer8s8cSK4du18GkdLhHicmoBRHmZdeTqWp4cSr5agQxCD66NlxGgiEAhNhiuVcHdcFyFFqNS
xH03ZVgwLaR8kQe8I8fz8y9siMbmLCtO7BuGWCNbjZ6aEC+q0VZphZTmcrbBA55Q8Gq2gfTYaWjJ
j4hWI8xDFKNC5SZgVkFXlMJz2JQ2AJq6jvU0ZDJAtX+7TeSQHDogkcYwLFvh0bEuFGE5NwC1xcrw
zKBQqBpDSgKfnknBpMIFAIAQaQVIaK24Nh1xie555Cu/d3I7viXqwhwr9/fAMRSkkiFI4XlUGK3C
lyirKb7oh2nstz9YAQ1P49vZIKCWuX+DnfCwA7Oxx4mgaOf/g6y3JV64EVVTk9TzyZct4bfOoUa1
SgsAZYjg1EfNS1Mia0R777onBDVHuE+aExRwJBAIhELkrXonsme3x27gYHHYqAixk8fUhCJY6xHR
f6h/q92f1EOhKna0w45gwMbwNhOlquT5ahisz2+PA+BjMERTg9bUA8GVBLhT10DywMTGOtk4nMoq
E0CdPQrpnD6+vPsLLLYWAeksfPnwYK1asgNlsBsMwWLhwIfr164ePPvoI+/btg6SmrtDMmTMxZsw
Yl+d46aWXcOTIEYSHhwMAoqOjsczLH7+pqHWFJ4VKcbTY9azAn5pABAKB0Jz4ovUWIqNx0Y/8KVe
khElxoMBhKJ2rsIg8Tv2j29azs/baC6+/Lx6nciOLCpMjzCmXAO0i1TA+8QSUixfb+2Wffgrj44+
Di4sTn8RqhfLZZ3ldlptugnXQID+/ScsjYAynqqoqLFmyBB988AFSUlKQlZWFhQsXYv369UhNTcU
DDzwAhUKB/Px8zJo1Cz179kRiYqLLc02bNg2z66mc3Zh0CXYfhyceJwKB0Nqor5huD0Ge09EiI85
X8D0m10e1HY+TM93D+NdVeF1cIfQ2JQYzoCkKpvvvh3z5ctB5eQBsuU6KJUugX7MGEIRBZR9/DOb
IEV6fceHCunyFFkfAhOquXbsGlUqFlJQUAEBiYiJUKhUyMjIwZMgQKBS2BOu4uDhERUWhoMCFkFo
LoYuHpb4kx4lAIBD4pAiMg505Rt6iGhmNNruYRliAPbPSApbzrOXkynACACgUMD71FG+bbNMmyN9
4g9dHX7oExSuv8PrMo0fDMpIva9BaCRjDKT4+Hnq9HkeP2hRMT58+jezsbOTn5/P2O3z4MCorK9G
nTx+359q4cSNmzJiBhx9+GBkZGY067rqQ6EYQM0Rat+KZBAKB0JoR1vwUYnUjoNkWiFXSCJU5jMZ
qC4ccJ23A3/KNSDtQhmWnquzimJer+NfKeTJvmjUL1u7dedsV6emQrV4NWK2gz5yBavZsUAZHkWE
uNBT6994TeaVaK1RFRUX9qgQ2IH/88QdWrVoFrVaLpKQkFBUVYejQoA4MyZM3j22Wfx+uu
vIzU11eU5CgsLEDtoLEAtx9Wifp7qa34vK/RxRE
EAoHQthl9RIlSs+sXc4yMxeb+Bsja6LzzgZNynK5ypIAs62nAkAgW2XoK035XwALbdYuWsfihvwG
vZcqwtdhhLD2XZMLkOIcxJb92DT3uvx/Sykre55jDw8FoNKCsfNHmrKVLUeom77ilkuyh5mtAxYX
69euHFStW2NtTpkxBly5dANg8SJEB8UIsXL8Yrr7zi1mgCgNhYR64Dtdffz2uuuoq5OXleXBc7KMHMqSmZ
mpsfzhp/IQ7mRb7P2jlUjOTm+wcfSnHi7Dm0Fch0ckGthg1wHG75eh+sulWBfvnhi2VktweEJcVB
```

```
6KUwc6NTnfuhbWI7TVTp7WxsUg+TkYGw8oYEFVfb+UjMNKiYBJZkVABzJ9YOS4pDsXNomORnG9ev
B3HMPKKPjmkvLy0WfbR41ChELFiCigbxNLeF3EVB3WklJif3/N2zYALVajf79++PkyZNYvHgxXn3
1VfTr18/jOQoLC+3/n5OTg1OnTqG7wO0YCHRxEa4jK+oIBALBNUJJglqeSg1u8UZTfRHmOdUmiAs
lG8ABCWqJKFSX6CLv1nrjjdB98gk4ifvFTOZRo6BbubLNhOhqCSiP0+rVq3H8+HFwHIekpCSkp6e
Doii8/PLLMJlMePvtt+37zps3D8OHD8fff/+NlStX2iUHFi1aBIPBAJqmQQdM00tLS0Lt37+b6Sm7
pEsLgd0G5ALKijkAgEFzjalVerJLGtCRx2kNbQ7iy7kJNgrjQcGIBHC8xo8TAVxePVrg2PC3jx0O
bkAD5ihWQbt0KqsrmvWI7d4b+jTdgufPONmc0AQFmOC1atMhl/8aNG90e07NnT55O02effdbg42o
MElx4nLq5mVERCARCW8eVx2lej6A2u5rOGaHH6VyFGRcqLdCYxCnML//Oz1uiAFzVWtEz3LXxxPb
tC/0nn0BvMIA5cAAwGGC54w5A4UsB59YJeVM3E0JJAgruV9sRCARCWyclXAqGApzK0+HWOHFh4bZ
IxyAJghgK1TUXp9LE4ZerBpf7HheIL4fJKbflcngoFDaDiRBYOU5tCaEIJkMmTQQCgeCWEBmNJ/u
o7e12Shp926jopRCaopAUwn+nfHZO69Ox626PJF47PyEujmaiX7QMsUoahXpbrJmDTYekPuq6BAK
B0Jr5V79QTE1SocTA4vooGXnhO5EYzOBUmSPpO1fHetjbRrxaggExxPj0F+JxaiakNIWtd0YhXEZ
BSgMJwe6rixMIBALBRtdQKQbFyonRJOCfKWrvOwkYl6AE1QaTu+sLeVM3I0mhUpybXr/ilwQCgUA
gDG4nQ2okg5Olviuoj0tQNuKIWi/E49TM1Ba/JEYTgUAgEOoKRVGY3zPY5baeLlYkdgyS4IYokhp
SF4jhRCAQCARCK2BCghKRcvFr/cEUNYRSTXd1kpMwXR0hhhOBQCAQCK0ABUNhTnexIGj/aCmuF3i
XbiBJ4XWGGE4EAoFAILQSHuihFr3YKQp4fWAYajNCwmQUxicQxfW6QpLDCQQCgUBoJXQIkuCl/F
4/rgGgE3vKjnUlkd7bWYcTpeZ0ZdIOdQLYjgRCAQCgdCKeLx3MKZ3VeGyxsIkRGjfGErX1+kI
MJwKBQCAQWhnRSgmilRLvOxL8huQ4EQgEAoFAIPgIMZwIBAKBQCAQfIQYTgQCgUAgEAg+QgwnAoF
AIBAIBB8hhhOBQCAQCASCjxDDiUAgEAgEAsFHqIqKCq65B0EgEAgEAoHQEiAeJwLh/+3de1BU5f/
A8TeCLKBcFJMETzAiPilQbxShcGJJVHUqUWaZqjIxUocFMjApgplEIYEAWgU1VaJFb3x0rHvXqqUe
LhA4VZBoLGIGGkCCyvz8cz7g/w/nbzsz7g/w/nbzsz7g/w/gWskqf1wwze55zds/nPDzz7Oc85+x5hBBCiFaUxSxEkIIYIIQQQQgghmqpU
kcRJCCCGEaCVJnIQQQggghWknmqmtjGo2GmJgrl27hpGREWW+88+88Qa2hoICIigKSkJAIDA5k0aRJt27ZVlhkzZgzjxo1Tlgs5bhoJ
MTDA1NSU0NJQhQ4aUOH6B0SoUgEH6Nmzp7L9TLc6Vs2TLKysrtpjMbbnqo2ZQlL9TLKysro2rKysrAE5OTkdtSt
5eXlotVo8fP8fcI2BjV5u+b4xFJDu6sa3yGSZ2NhBp3hPLcz8iXxL//WLAtcWtSsFc/GfxtGeqvQNvkqr3/UJZtTUmv/o6J+N
PUNtcQkICX3zxBb/99hw7sc+cAmDlzJvaePnz9e5c+cAmDlzJvaePnz9e5c+cAmDlzJvaePnz9e5c+cAmDlzJvaePnz9e5c+cA
hGBsbt/8BtJGW6iEhIYETJ05gaHh3rX58+czAzuc3/wG0kZb
q4e/6xZycHHbu3IbqYWmFnyE+wW7q1ev4ufnR2hoKiImW7q1ev4ufnR2hoKiImW7q1ev4ufnR2hoKiImW7q1ev4ufnR2hoKiImW7q1ev4ufnR2
0008r6wMCAhg/fny7x38/SZzaWGxsLBMnTmTmT27NlcvHiR0NBQPvnkE0xMTMTPQd2iPn6+uLh4cHBgY
GnDhxgrCwMLKysgDw9PQkPDxcxcpOfYsWOcP3+e1NRUU4MTQd2oPQe26wgMDQ11vgR
Onz5NVFQU7u7u5OTk4OLLiwpYtW/QY4aMzduxYfH19iYmJSlPTk6mvr6e9PR0bty4gb+/P0OHDsX
FxYXvv/+etLQ09u3b3h7m5OWFhYeRw8eJCoqivb2dxYsXV9nUbI0uSQ3qqvzbLqoaKlivC3sf1mJdqzOzRrPvmDfabYg+5O8YBtHFhTAqYGE6eIjD
IgK2tbYc6vjLpsr/f1ffkuRqOlqsnH2gfPfu2vCjFtcpBJoRL1mDFtX4vbxmqhmYQqFCVqbhEnGqQDMsPGXDtSiphKlwrqZcE9uUqNhpPqyNEPUp5gk8ZMzBfXJMpr
S1rlz58euPcilujZ0/fp1zp8+Oe/bGQszDh0F7VIqsaOqV7VFD+JX7JA65qRd1aHYrQ4Z4LLFLNDHV0FqHc2lqKxmRL6aGukXqOS
0HNnjIzc3l+nTp2NsbIyxsTHP0jdCZzYPr7QGpqgxzcPVVZsgt0qW2DKrKhnSKqJc8qxDsXWdtFVZWqtbuq4TxmBOLzAtyjj2zS4ObHtSOO
BgQEWFhZ4e3sr//c3FybUVVRNaWtpwJCcWvByIWy/PdtUfFeLJAVNmIytHwxG9Yq5d2yaC4e9hvcC0cObjtbDamtcMxjJAMUk3qD0p3HSOO63L
e5vPPP8fd3R1bW1sAZs+eeTW5u7qMIrj92Oph5ycnLo378/AwcObknsY3Gy3HjPHU46tbi6ipdALh+7w
s7CgrK9NjVPqRnJyAgamrKjBkzdM4dOxg0ozOqIdO3awd+9ebGxsWLJkCuwuFLXSYl6N
Wq5Vt7OzsOvQXxj3VdXk5+eTpkKilF28eJGDNm0DNvLy8mDmDSe3boe8ZHm/8vJy5Q7Tj5/d
4oawbNGiQs7se3v4/0Taa++eYbampqGDp0pqGWUVnWhvTEAtkS1jeiHbmyoxbr4ZFNe6J+X7k8nUz6bQ
VtVrNn3/+SU1NzRSGrY2m5VrkWcmFiataxO3btxk1+fDgBHQAGYr5KcWbSbS49PZ2
8vVDy2b98OgJ+fH4/+HfH4sWLcLY2Jhff+vbk5/ThGu+RuQC1kf2Vg+2T+PjY0NBgYGF+XVv7z++uscPHiQFZr
Q32HpzeffZqwYcPo1asXXC+tqnn8XcHfI/tlnn8XUI1JRr166xevenzd
uXQAYNGgQn332GRYWFtTU1LJRNMsmhrB+/Xq2tsTTl/evOHfkLBKyuvbetBgnpcyWi
1Nbdu3WLjxo2oNxaXGRxSGsra2yBStLSsMyPbmyHS9sXVbW1jvXj9sNMkGrWQ4ueFOQdZ5oC
/3hmjg4MDkyZNUm6Q7Y7DhsbZkNWoGRVVfqY9RqVRBsaVCdmOeMh4vtYsJMm5FU5kMGoaWuJ4cWBfkLQNf
KClNTUwB69uzJ5MmTJ5ZvMd6xjGhZSk2fVjdTs9cNqjxl0DhzVqDoltDlWR+lrMD9uncC9QU0D7Ldq6urUt6
1a1csLCwAsLS0M/Pr82O2jYf1i2q1WqetlJeJe8ePXtm7j5ZVNBLw+YPHlJTxoadFtFFHf3d8fnpFQT
8Ft8dkji1ISsrK4YPH05PmFVLVFhfK6qqqKr7/
+GicnJ32E+Mg1NDRRQXV2tLBcWjavq5WomUu1FdfrK2MlX+d4u2QWqeeSSlhGBTXSUOzDmvcsXFHh
TJihl97eHuro6vvzyeVFWloaWq2tOz8fLyUtVdvLuWxNH9Q3dW1qUtxj6rH2BSLkSxKTTl9KuFDHbtXLh
wgfDwcDZs2MCIES9Q27ltdwdzdD6o2H9YvPPfccJ0+VJKn1NtUDtsE4O5N4ce
PH9c5wQKorKykaqakJuPPtL3ezs7MeiPchcdW2stLsU6OhoKisrMTIyDqj2UitQ09D+ROFVUVPDiiy9ia2u
r/AwfYNOmTcTHx3PlyrBarX4+PgwmbqUe7PZ48PUb76FRVVREaGsrt27p27fp1kkTpWKpUuX4u7uTnN
zs/I4AoCJEycSHBxMp04d9/x9lI7S0Oo7NvbqJS99957fPvvvvsbPlu3kSCCuAmGSaq7ZAHCf7+pNu
yZQvHjh2juroaU1NTxNTTExDE0xAtSmZ0zLS1C/b29sZAHWNjZ/d6fed/+/JtG50isCeMKg5YzEKsSFJJo
rUalUejySf6eleli9ej3UtBtz9QbOobKAt9W7hBKVvr7g2PsRhSDBl2l09dVXm8C7TK8K8t6vQhgJ8MLwE
+/aH9YlZWFrt37QvFYyv4eOX3H8Oy3VQ79+/UhLSyMrK7rp/8f58+MP3J8JYwAMstNs1HN+
DmpubGGDTBgACEhIfdoMKUYKUPRrbU3YbG5LkCCCCHamCRO9qhRCCIIERBEENgghBCtSEI10ASE2i+t9V0VWRlpaGj48PdnZ
khBBCCNFKjJgJJIYQQQrSSPzKR1YPPDQctNHu3Nzc/nabxMRQCQJCvkQDCkk1B2i+t9VWVRlpaGj48P
2+g5HCNEO5HEEEQoh2V1BQoLxGLGxcVLYJoVLF68GA8XMhEgbAhHk9PAyMhIRJGjFhLNJ5lwSExM
```

```
ZOXKkvsMRQrQDGXESQrQ7FxcX5XV9fT0AvXr10ikHdCa+FUKIx4EkTkKIx5avr6/Opbq3336b4uJ
iXnvtNbZu3YpGo2HUqFGsX7+e2tpa3n33XQoLC+nbty+RkZEMGDBA+azm5mb279/P0aNHqaioQK1
W88orr+Dj4/PQGI4ePUpycjIajQYTExP69etHWFgYpqamzJ07F4Dly5cr258+fRqAmpoatm3bRn5
+Pjdv3mTQoEGsWrWKIUOGKNu6ubkREhJCeXk52dnZNDc34+3tTWhoKJ07d26zehRCtB1JnIQQT5S
KigqSkpJYtmyZMmN6bGwsGo2Gad0m4e/vz7Zt24iIiCAlJUWZcHnjxo1kZmayZMkSBg8ezKlTp9i
w0jXcxwAABCFJREFUYQOWlpYtzkh/9uxZ4uLiCAgIwMXFhbq6OgoKCqirq6N3795ER0cTFRVFWFi
YzuhYY2MjK1eu5MaNGwQFBdG9e3cOHz7MypUrSUtL05kyIjk5mSFDhhAdHU1JSQmJiYmoVCqCg4M
fbUUKIf4RSZyEEE+U2tpadu/eTa9evQD46aef+Oijj3rrbeYMmUKAFqtllWrVnH58mUcHR25evU
qhw8fJjIyUhlhcnNzo7Kykl27drWYOF28eJEBAwawawaNEipWz8+PHK63sjWo6OjjqXGbOzsykuLiY
lJYU+ffoAMHr0aGbNmkVycrJOUmRmZkZsbCydOnVizJgxNDY2snfvXUb1JgQoi3J4wi
EEE8UW1tbJWkC6N27NwCjRo1Syu6t//333wE4c+t+YMnTp1YuLEiTQ1NSl/o0eP5scff+TOnTt/ua+
BAwdSVFTEpk2bOHv2LLdv325VjGfOnGHw4MHY2dkp+wIYMWIEly5d0tl2/PjxOhM9T5o0iYaGBoq
Li1u1LyFE+5IRJyHEE6Vr1646y3+fuBTI3N3+grKGhAYDr169z584d/Pz89H7W7du3sbW1/cv6ZcxaVUwRgEEZUcyuGBkZ4ezTOTJk0lPT2f58uXMnz8fe3t7ampqKCwsxNramnnn
Hi58vhNAfSZyEEB2ehYUFhoaG7Nq1S7nn6X7/P3m5n4+PDz4+PlRXV3O5WUb1JgQoi3J4wi
fk5MTa9aseWCdsbGxznJVVZXXOcnV1NQDW1tYPPSYhhH5I4iSE6PBGjx5Nc3MzN2/e5JlnnvlHn9G
tWzf8/PzIy8vjl19+AcDI6G4X2tjY+MD+O4XBpQfn09gYKByuS4vLw+VSkX//v3UZx
CiEdLEichRIfn4OCAn58nuSkpKoqalh5MiRWFlZUVR
UxLlz5wgMDARArVajUqnIzMykS5cuGBkZ4ezszOTJk0lPT2f58uXMnz8fe3t7ampqKCwsxNramnn
z5in7qK+vJzw8nGnTplFSUsKePXXuYOXOm3BguxGNKEichxH9CWFgYffr0ISMjg6SkSkJLp06YKjoyN
Tp05t8T1OTk4cPHiQ48ePU19fj1qtZunSpcyZMwcAlUrFunXr2LlzJ8uWLaOpqYnTp0+jUqlITEx
kx44dJCUlUVVVRffu3Xf2dtb5VR7AvHnz0Gg0RERE0NzczNSpU1mxYsUjrQshxD8nU64IIYSeuLm
58eabb/LSy/pOxQhRCvJ4wiEEEIIIVpJEichhBBCiFaSS3VCCCGEEK0kI05CCCGEEK0kiZMQQgg
hRCtJ4iSEEEII0UqSOAkhhBBCtJIkTkIIIYQQrSSJkxBCCCFEK/0fV9T0dUkr8PwAAAAASUVORK5
CYII=\n"
          },
          "metadata": {}
        }
      ]
    }
  ]
}
```

Sprint 4:

LOGIN.html

```
&lt;!DOCTYPE html&gt;
&lt;html lan=&quot;en&quot; and dir=&quot;Itr&quot;&gt;
    &lt;head&gt;
        &lt;meta charset=&quot;utf-8&quot;&gt;
        &lt;title&gt;login form&lt;/title&gt;
        &lt;link rel=&quot;stylesheet&quot; href=&quot;style.css&quot;&gt;
         &lt;script src =&quot;login.js&quot;&gt;&lt;/script&gt;
    &lt;/head&gt;
    &lt;body&gt;
        &lt;form class=&quot;box&quot; action=&quot;login.html&quot;
method=&quot;POST&quot;&gt;
            &lt;h1&gt;CRUDE OIL PRICE PREDICTION&lt;/h1&gt;
            &lt;h2&gt;
                 LOGIN
            &lt;/h2&gt;
            &lt;input type=&quot;text&quot; name=&quot;&quot; placeholder=&quot;Enter
Username&quot;
id=&quot;username&quot;&gt;
            &lt;input type=&quot;password&quot; name=&quot;&quot; placeholder=&quot;Enter
Password&quot;
id=&quot;password&quot;&gt;
            &lt;input type=&quot;submit&quot; name=&quot;&quot;
value=&quot;Login&quot;  onclick=&quot;validate()&quot;&gt;
            &lt;h3&gt;&lt;a href=&quot;register.html&quot;&gt; New User ? Register
&lt;/a&gt;&lt;/h3&gt;
```

```
        &lt;/form&gt;
    &lt;/body&gt;
&lt;/html&gt;

LOGIN.JS
function validate()
{
    var username=document.getElementById(&quot;username&quot;).Value;
    var password=document.getElementById(&quot;password&quot;).Value;
    if(username==&quot;ibm&quot;&amp;&amp;password==&quot;ibm123&quot;)
    {
        alert(&quot;login succesfully&quot;);
        return true;
    }
    else
    {
        document.getElementById(&quot;username&quot;).disabled=true;
        document.getElementById(&quot;password&quot;).disabled=true;
        return false;
    }

}

{
    document.getElementByI
d(&quot;email&quot;).value=&quot;&quot;;
    document.getElementById(&quot;pwd1&quot;).value=&quot;&quot;;
}
```

predict.html

```
<!DOCTYPE html>
<head>
    <title>Crude Oil Price Prediction </title>
    <link href='https://fonts.googleapis.com/css?family=Roboto' rel='stylesheet'>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/predict.css') }}">
</head>
<body style="text-align:center;background-color: lightsteelblue;">
    <h1 style="color: white;font-size: 50px;font-family: roboto;">
    Crude Oil Price Prediction </h1>
    <h1 style="color: white;font-size: 50px;font-family: roboto;">
        Enter the Oil price for 10 days </h1>
        <form action="/predict" method="POST" enctype = "multipart/form-data">
            <div style="color:green;font-size:50px;font-family:roboto;">
                {{prediction}}
            </div>
        <input type="text" name="val" style="border-radius: 18px;padding: 20px;width:
300px;height: 15px;text-align: center; align:center;" >
         <br> <br> <br>
        <input type="submit"/ style="border-radius: 9px;;padding: 10px;width: 150px;
        height: 40px;text-align: center;background: #003d66;color: white;">
        </form>
        <br>
        <form action="/predict" method="GET" enctype = "multipart/form-data">
        <input type="submit"/ value="Reset" style="border-radius: 9px;;padding:
10px;width: 150px;
        height: 40px;text-align: center;background: #003d66;color: white;">
        </form>
```

```
</body>
```

REGISTER.CSS

```
body{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
    background: url(ppp.jpg);
    background-size: cover;
}
.box{
    width: 300px;
    padding: 30px;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%,-50%);
    background: rgb(14, 14, 14);
    text-align: center;
}
.box h1
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}
.box h2
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}

.box input[type="text"],.box input[type="password"] ,.box input[type="date"],.box
input[type="Number"],.box input[type="Email"]
{
    border: 0;
    background: white;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid #2af003;
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: #fff6ff(18, 18, 179);
    border-radius: 24px;
    transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
    width: 270px;
    border-color: rgb(238, 26, 203);

}
.box input[type="submit"]{
    border: 0;
    background: none;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid rgb(211, 15, 152);
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: rgb(73, 31, 224);
    border-radius: 24px;
    transition: 0.25px;
    cursor: pointer;


}
.box input[type="submit"]:hover{
    background: rgb(100, 182, 53);
}
h3{
    color: wheat;
}
```

REGISTER.HTML

```
<!DOCTYPE html>
<html lan="en" and dir="Itr">
    <head>
        <meta charset="utf-8">
        <title>login form</title>
        <link rel="stylesheet" href="register.css">
         <script src ="login.js"></script>
    </head>
    <body>
        <form class="box" action="login.html" method="POST">
            <h1>CRUDE OIL PRICE PREDICTION</h1>
            <h2>
                 Register
            </h2>
            <input type="text" name="" placeholder="Enter Username" id="username">
            <input type="email" name="" placeholder="Enter Your Email Id" id="Email">
            <input type="number" name="" placeholder="Enter Your Number" id="Number">
            <input type="password" name="" placeholder="Enter Password" id="password">
            <input type="submit" name="" value="Register"  onclick="validate()">
            <h3><a href="login.html"> Login </a></h3>
        </form>

    </body>
</html>
```

STYLE.CSS

```
body{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
    background: url(p2.jpg);
    background-size: cover;
}
.box{
    width: 300px;
    padding: 30px;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%,-50%);
    background: rgb(14, 14, 14);
    text-align: center;
}
.box h1
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}
.box h2
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}

.box input[type="text"],.box input[type="password"] ,.box input[type="date"],.box
input[type="Number"],.box input[type="Email"]
{
    border: 0;
    background: white;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid #2af003;
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: #fff6ff(18, 18, 179);
    border-radius: 24px;
    transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
    width: 270px;
    border-color: rgb(238, 26, 203);

}
```

```
.box input[type="submit"]{
    border: 0;
    background: none;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid rgb(211, 15, 152);
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: rgb(73, 31, 224);
    border-radius: 24px;
    transition: 0.25px;
    cursor: pointer;


}
.box input[type="submit"]:hover{
    background: rgb(100, 182, 53);
}
h3{
    color: wheat;
}
```

APP.PY

```
from flask import Flask,render_template,request,redirect
import numpy as np
import joblib
from keras.models import load_model

app = Flask(__name__)

@app.route('/',methods=["GET"])
def index():
    return render('login.html')

@app.route('/predict',methods=["POST","GET"])
def predict():
    if request.method == "POST":
        string = request.form['val']
        if(string ==""):
            return render_template('predict.html')
        string = string.split(',')
        x_input = [eval(i) for i in string]
        sc = joblib.load("scaler.save")
        x_input = sc.fit_transform(np.array(x_input).reshape(-1,1))
        x_input = np.array(x_input).reshape(1,-1)
        x_input = x_input.reshape(1,-1)
        x_input = x_input.reshape((1,10,1))
        model = load_model('model.h5')
        output = model.predict(x_input)
        val = sc.inverse_transform(output)
        return render_template('predict.html', prediction = "The predicted price is
{:.2f}".format(val[0][0]))
    if request.method == "GET":
        return render_template('predict.html')

if __name__=="__main__":
    model = load_model('model.h5')
    app.run(host='0.0.0.0', port=5000)
```

14. ADVANTAGES & DISADVANTAGES

- The model is able to forecasting the price accurately based on the past 10 days data.
- But whenever the commodity is affected by external factors which are caused naturally then the predictions are bad.
- This cannot be predicted by machine learning model

10.CONCLUSION:

Therefore the ml was deployed as a web app and the user interface is handy for stakeholders who do not have much knowledge in programming. The predicted value is displayed in the user interface.

11.FUTURE SCOPE

The machine learning model can be improved by converting it from univariate to multi variate model for better understanding the price value. This can result in a great boost in the prediction .

GitHub link:

GitHub: https://github.com/IBM-EPBL/IBM-Project-47385-1660798812

Demo Link: https://youtu.be/RE-tcRPhQgs