```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 3,
   "id": "12c2ab3f",
   "metadata": {},
   "outputs": [],
   "source": [
    "import numpy as np\n",
    "import pandas as pd\n",
    "import matplotlib.pyplot as plt\n",
    "import seaborn as sns                       \n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "id": "94f54145",
   "metadata": {},
   "outputs": [],
   "source": [
    "df = pd.read_csv(\"ACME-HappinessSurvey2020 (1).csv\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 5,
   "id": "a0e35bdd",
   "metadata": {
    "scrolled": true
   },
   "outputs": [
    {
     "data": {
      "text/html": [
       "<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>Y</th>\n",
       "      <th>X1</th>\n",
       "      <th>X2</th>\n",
       "      <th>X3</th>\n",
       "      <th>X4</th>\n",
 "      <th>X5</th>\n",
 "      <th>X6</th>\n",
 "    </tr>\n",
 "  </thead>\n",
 "  <tbody>\n",
 "    <tr>\n",
 "      <th>0</th>\n",
 "      <td>0</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>4</td>\n",
 "      <td>2</td>\n",
 "      <td>4</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>1</th>\n",
 "      <td>0</td>\n",
 "      <td>3</td>\n",
 "      <td>2</td>\n",
 "      <td>3</td>\n",
 "      <td>5</td>\n",
 "      <td>4</td>\n",
 "      <td>3</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>2</th>\n",
 "      <td>1</td>\n",
 "      <td>5</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>5</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>3</th>\n",
 "      <td>0</td>\n",
 "      <td>5</td>\n",
 "      <td>4</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>5</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>4</th>\n",
 "      <td>0</td>\n",
 "      <td>5</td>\n",
 "      <td>4</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>3</td>\n",
 "      <td>5</td>\n",
 "    </tr>\n",
 "  </tbody>\n",
 "</table>\n",
 "</div>"
],
      "text/plain": [
       "   Y  X1  X2  X3  X4  X5  X6\n",
       "0  0   3   3   3   4   2   4\n",
       "1  0   3   2   3   5   4   3\n",
       "2  1   5   3   3   3   3   5\n",
       "3  0   5   4   3   3   3   5\n",
       "4  0   5   4   3   3   3   5"
      ]
     },
     "execution_count": 5,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df.head()\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 6,
   "id": "fa9c3882",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "<class 'pandas.core.frame.DataFrame'>\n",
      "RangeIndex: 126 entries, 0 to 125\n",
      "Data columns (total 7 columns):\n",
      " #   Column  Non-Null Count  Dtype\n",
      "---  ------  --------------  -----\n",
      " 0   Y       126 non-null    int64\n",
      " 1   X1      126 non-null    int64\n",
      " 2   X2      126 non-null    int64\n",
      " 3   X3      126 non-null    int64\n",
      " 4   X4      126 non-null    int64\n",
      " 5   X5      126 non-null    int64\n",
      " 6   X6      126 non-null    int64\n",
      "dtypes: int64(7)\n",
      "memory usage: 7.0 KB\n"
     ]
    }
   ],
   "source": [
    "df.info() #no null values noted\n",
    "#Description of Data\n",
    "#Y = target attribute (Y) with values indicating 0 (unhappy) and 1 (happy)
customers\n",
    "#X1 = my order was delivered on time\n",
    "#X2 = contents of my order was as I expected\n",
    "#X3 = I ordered everything I wanted to order\n",
    "#X4 = I paid a good price for my order\n",
    "#X5 = I am satisfied with my courier\n",
    "#X6 = the app makes ordering easy for me"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 7,
   "id": "3c1558ed",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array([4, 3, 5, 2, 1], dtype=int64)"
      ]
     },
     "execution_count": 7,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df.X6.unique() #all of the features have ratings. Categorical variables."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 8,
   "id": "f06adf2f",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "(57, 69)"
      ]
     },
     "execution_count": 8,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "#creating happy =1 and sad = 0. This lets us seperate between happy and sad.
This is a balanced class.\n",
    "df_happy = df[df['Y'] == 1]\n",
    "df_sad = df[df['Y'] == 0]\n",
    "# around 55% are happy while aroun 45% are unhappy. \n",
    "df[df['Y']==0]['Y'].count(), df[df['Y']==1]['Y'].count()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 9,
   "id": "7b7ca9d4",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "image/png":
```

"iVBORw0KGgoAAAANSUhEUgAABBwAAAJNCAYAAACSmWZtAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAABNMklEQVR4nO39fbhlVX0n+n5/
QsVCAQUp7RIsCxRpiChICdiAQRAblRYwgHJueLElFXPFiDEkdM6JornetvE1j+YYi4AFaTWYKAHR2KFp0MM

TFQubULwYaZVgBbogIFFzBUHG/
WMvKgVUFbuKOdfce63P53n2s9Yc82X95i7Wb05+e8wxqrUWAAAAgC49aegAAAAAgMmj4AAAAAB0TsEBAAAA
6JyCAwAAANA5BQcAAACgcwoOAAAAQOe2HjqA2dhpp53a0qVLhw4D4BGuvfbaf2qtLRo6jnGQh4G5aq7n
amORrSZ6cmXvvv2ytvbuqdkxyUZKlSW5NckJr7UebOpZcDMxFm8rD86LgsHTp0qxatWroMAAeoar+YegYxk
UeBuaqeZCL709yWGvtp1W1IMnVVfXXSV6f5IrW2vur6qwkZyX5vU0dSC4G5qJN5WGPVAAAQE/
ajJ+OFheMflqSo5NcMGq/
IMkx448OoF8KDgAA0KOq2qqqrktyZ5LLW2vfTPKs1todSTJ6feaAIQL0QsEBAAB61Fr7RWttnyS7Jnm/
ql44232ranlVraqqVXfddVdvMQL0YV6M4cD88cADD2TNmjW57777hg5lXlq4cGF22WWXLFiwYOhQAOiYa+Q
TMwnXyNbavVV1VZIjk6ytqsWttTuqanFmej9saJ8VSVYkybJly9rYgoUJJRdvuS3JwwoOdGrNmjXZbrvtsn
Tp0lTV0OHMK6213H333VmzZk123XXXocMBoGOukVtuPl8jq2pRkgdGxYZtkrwyyX9JcmmSU5K8f/
R6yXBRwvSQi7fMluZhj1TQqfvuuy/PeMYzfHm3QFXlGc94hm0QFXlGc94hmorwIRyjdxy8/
wauTjJlVV1fZJvZWYMh8syU2g4oqpuSXLEaBnomVy8ZbY0D+vhQOd8+vzN7/Mr5UrV2bVqlX5+Mc/
d211q5Psu8G2u9Ocvj4IwLmaz4Z2pb83vRwAAAAADqnhwO92u/
MCzs93rUfOPlxt7n11ltz1FFH5YYbbbljXdvbZZ2fbbbfN7/zO72xwn5UrV2bVqlX5+Mc/
3lmsALAprpEAw5OL+6WHAwAAANA5PRyYKoceemgOOOCAXHnllbn33ntz3nnn5ZBDDkmS3H777TnyyCPzve9
9L8cee2zOOeecJMlv/uZV5lvf+lZ+9rOf5bjjst73vOeJMnSpUvzhje8IVdeeWWS5DOf+Uye//
zn59RTT83ChQtz4403Zu3atfnwhz+co446Kocckg+9rGPZZ999kmSHHTQQfnEJz6RF73oReP/
RcxDt71370E+d8m7Vg/yuTDJ+v4++95uGddImFu6/sv7o83mL/
GM36TlYj0cmDoPPvhgrrnmmnz0ox9VMkuuuuy4XXXRRVq9enYsuuig//OEPkyTve9/7smrVqlx//
fX56le/muuvv37dPttvd9666356le/
mi996Ut5y1vekvvuuy+nnXZaVq5cmST57ne/m/vvv9+NFABzimskwPAmKRcrODBxNjZ66sPtr3/965MkL+
+23X2699dZ16w8//PA87WlPy8KFC7PXXnvlH/7hH5Ikn/vc5/KSl7wk++67b7b2688cbcdNNN6/
Y58cQT171+/etfX9d+wgkn5ElPelJ233337LbbbvnOd76OCBnH/+
+Tn11FO7PG0AeFyukQDDm6Zc7JEKJs4znvGM/
OhHP3pE2z333JNnPPnhu///S5kJz85SbLVVlvlwQcfXLfNw+3rr/vBD36QD36Qc9SjiGH3rr/vBD36QD37wg/
nWt76VHXbYIaeeeuoj5p5dP1ls7P3Dy095ylNyxBFH5JJLsnnPve5rFq1qoOzBYDZc40EGN405WI9HJg42
267bRYvXpwrrrgiycyX9ytf+UoOPvjgzT7Wj3/84zz1qU/N0572tKxduzZ//dd//
Yj1F1100brXl73s/Zeva/+Iv/iIPPfRQvve97+X73/9+9thjjyTJaaedlt/6rd/
KS1/60uy4445beooAsEVcIwGGN025WA8HejXUYDQXXnh3vrWt+ad73xnkuTd7353nve85232cV784hdn33
33zS//8i9nt912y0EHfSI9ffff38OOCAPPTQ/nsZz+7rn2PrX5kz/
5kyxcuDDJTLeo7bffPm9605t2m4mqtdXKgPi1btqzpWjc/
3HzzzdlzlzzzzZ2HDmMsli5dmlWrVmVmWnWnnXZ6RPupp56Kscdd9xj9t1///1z+U6e9KQNdzCapt/
h5piLs1RU1bWttWVjDGcw8Mcy6YNBMAlcI10jgeHJxf3m+Ux2ZZJqZtqrWrqZKKqYpv4/
M2rVrkyRnn312nn3nP7KEzOmLDfEwM4MDBwM6lq00gxe4a57rrr1i0//ExUkqxZsyZveMMMbcscdd+TnP/
3bg5wTAHTBNRJgeJOUjBvrTBNRJge4JOWiz1SwcTrhqa7/qWYDTPOGrSEXPLygwcGd4iw/w4cNlAi2t0Z8y79/
dOcc845+dCHPPpQkuf766/ONJgwJ//IV/
5J99903tTbb5BWxzX5vSsk1sxhcE+2+1113XbHX300wUuHk8drSHIHaHv/
TnP/
95dt1113XbHX300dlmm22yzTbb5BWveEUuaaPP3pT8/+++
+f3XbbLUly4okn5uqrr85Ww47LJdddln23HPPHPPPPDAA9l7726f/
wKArrlGAgxvmnKxRyqYKm9729ty+umnZ/Xq1fnkJz/Xq1fnkJz+Z++67b926qnrEtg8vqVK/
OpT30qb3rTm3qOHAD65RoJMLxJy8Uy8UKDkyVf/7nf87OO+
+cJLnggggsnyqsAbA7XSIDhTVMuZnBg4myzzzg6zu7Tv//
zmc98Jieeiie0J4TwQAOuYaCTC8ScvFHqmgV0MNRrNo0aKcd955r/+xHUjX7779+xJk+fXt+/PqHPrgnlvPvk
Xvva1+a2227LH/
zBH+aajAZZ352vvvd7+ZlLHHzZ09n9EVuudjbpYJlPkbnvwrdgCwPO+GEE3LhX5pSE9bEB8rDCA4n/Lz79/
ENRKYRnJxvxQcmDjrz2mbPPBBDN//PfXUJDPPPR199EbbNVbc2zpzKbnooos2uM/
VV1/9iNFkwTAATN3Q9VKlkcJ3O62bvjlQEwG71OXbQXBsX4dHk4i2zJXm470cqPprkd5M8tF7Fhz1q8qD1cuDB33323m4It0FrL
3XffnYULFbNeYULFw4dCgA9cI3ccq6RQFfHfkQEJjtba9dW1aW/
yJMuTZMmSJd0GR2922WWZXm270WWXrFmzJnfddffdfQcxLDxcuzC677DJ0GADAGM+5LcnDfT5ScVCS11
XVa5IsTLJ9Vf3f3XJGuranFr7Y6qWpzkzg3t3FpbkWBB{dt1116HDAJJU1XOSXJjk32Smp
9mK1tofVdWOSS5KsjTJrUlOaK39aKg4YVq4RgIMTy4er94eqWit/

```
afW2i6ttaVJ3pjkf7TWfi3JpUlOGW12SpJL+ooBYMo9mOSdrbU9kxyY5K1VtVf+dfDe3ZNcMVoGAIBO9T4t
5ga8P8kRVXVLkiNGywB0rLV2R2vt26P3P8nMjEE7Z2bw3gtGm12Q5JhBAgSYAlX1nKq6sqpurqobq+rto/
azq+ofq+q60c9rho4VoGvjmBYzrbWrMjMbRVprdycxKTPAGFXV0iT7JvlmHjV4b1UZvBegPw/
3Nvt2VW2X5Nqquny07iOttQ8OGBtAr8ZScABgOFW1bZLPJzmjtfbj2U4DZfBegCduVOB9uMj7k6p6uLcZwM
Qb4pEKAMakqhZkptjw6dbaF0bNa0eD9ubxBu9trS1rrS1btGjReAIGmGCP6m2WJKdX1fVVdX5V7bCRfZZX1
aqqWmVUfWC+UXAAmFA105XhvCQ3t9Y+vN4qg/
cCjNmje5sl+USS5yXZJzM9ID60of0Uf4H5zCMVAJProCQnJVldVdeN2n4/
M4P1fq6q3pzktiTHDxMewHTYUG+z1tra9dafm+SygcID6I2CA8CEaq1dnWRjAzYYvBdgDDbW26yqFj88gG+
SY5PcMER8AH1ScAAAgP5srLfZiVW1T5KW5NYkvzFEcAB9UnAAAICebKK32ZfHHQvAuBk0EgAAAOicggMAAA
DQOQUHAAAAoHPGcAAAHmO/My/
s7dgXb9fboQGAOUQPBwAAAKBzCg4AAABA5xQcAAAAgM4pOAAAAACdU3AAAAAAOqfgAAAAAHROwQEAAADonI
IDAAAA0DkFBwAAAKBzCg4AAABA5xQcAAAAgM4pOAAAAACdU3AAAAAAOqfgAAAAAHROwQEAAADo3NZDBwAAA
AB9ue29e/
d27CXvWt3bsSeBHg4AAABA5xQcAAAAgM4pOAAAAACdU3AAAAAAOqfgAAAAAHROwQEAAADonIIDAAAA0DkFB
wAAAKBzCg4AAABA5xQcAAAAgM4pOAAAAACdU3AAAICeVNVzqurKqrq5qm6sqreP2nesqsur6pbR6w5DxwrQ
NQUHAADoz4NJ3tla2zPJgUneWlV7JTkryRWttd2TXDFaBpgoCg4AANCT1todrbVvj97/
JMnNSXZOcnSSC0abXZDkmEECBOiRggMAAIxBVS1Nsm+SbyZ5VmvtjmSmKJHkmQOGBtALBQcAAOhZVW2b5PN
Jzmit/
Xgz9lteVauqatVdd93VX4AAPVBwAACAHlXVgswUGz7dWvvCqHltVS0erV+c5M4N7dtaW9FaW9ZaW7Zo0aLx
BAzQkd4KDlW1sKquqaq/
G43I+55RuxF5AQCYClVVSc5LcnNr7cPrrbo0ySmj9uWTcsQH0rc8eDvcnOay19uIk+yQ5sqoOjBF5AQC
YHgclOSnJYVV13ejnNUnen+SIqrolyRGjZYCJsnVfB26ttSQ/HS0uGP20zIzIe+io/io/
YIkVyX5vb7iAACAobTWrk5SG1l9+DhjARi3XsdwqKqtquq6zDbDyTdnlrbdYj8hogB+CJq6rzq+rOqrphvbaz
q+ofH/WXNgAA6FSvBYfW2i9aa/sk2SXJ/lX1ws3Y1wA5AE/cyiRHbqD9I621fUY/
Xx5zTAAATIGxzFLRWrs3M49OHJlZjrwLAADDQuT57OCxOcmVVXZ/
kW5kZw+GyGJEXYGifSPK8zMwgdEeSD21oI2PpAADwRPQ5S8X1Sfbd
QPvdMSIvIvwGBaa2sffl9V5ya5bCPbrU
iyIkmWLVvWxhMdAACTYixjOAAwdzw8js7IsUlu2Ni2AACwpXrr4QDA8Krqs0kOkOTbJTVa1J8u4kh1bVPklak
luT/
MZQ8QEAMLkUHAAmWGvtxA00nzf2QAAAmDoeqQAAAAA6p+AAAAAAdE7BAQAAAOicggMAAADQOQUHAAAAoHNm
qQBgzrrtvXv3evwl71rd6/
EBAKaZHg4AAABA5xQcAAAAgM4pOAAAAACdU3AAAAAAOqfgAAAAAHROwQEAAADonIIDAAAA0LlZFRyq6orZt
AHQD3kYYHhyMcDm2XpTK6tqYZJtkuxUVTskqdGqZyRZ3HNsAFNPHgYYnlwMsGU2WXBI8htJzshMIr02jybX
Gty/XGSP+4vLABG5nQe3u/MC3s9/
sXb9Xr4zTJN5wo8xpzOxQBz1SYLDq21P0ryR1X1ttbax8YUEwAj8jDA8ORigC3zeD0ckiSttY9V1b9LsnT9
fVpr/
f65B4Ak8jDAXCAXA2yeWRUcqurPkjwvyXJfjFqbkkkV4AxkIcBhicXA2yeWRUckixLsldrrfUZDAAbJQ8D
DE8uBtgMs5oWM8kNSn5Nn4EAsEnyMMDw5GKAzTDbHg47Jbmpmpq5Jcv/Dja211/
USFQCPJg8DDE8uBtgMsy04nN1nEAA8rrOHDgAAuRhgc8x2loqv9h0IzAX7nTnMmE/
XfuDkQT6X+UMeBhjeluTiqjo/yVFJ7mytvXDUdnaSX09y12iz32+tfbmrOAHmitnOUvGTzIzAmyS/
lGRBkn9prW3fV3Ct9avbbW9y/yvfdmmUpm6s9vIOFmy2ahz/
HwEB8FjyMMDwtiQXt9a+VlVLey+6cvzbnkk1fqOnv16ecVtnoqy4P0Sw1gy8uUK/xvmz02kV2S/
lGRBkn9prW3fV2AA/Ct5GGB4W5iLVyb5eB47k8VHHWmsf7DxIgDlktj0ctlt/uaqOSbJ/HwEB8FjyMMDwtiQXt9a+VlVLewwLYM6a7SwVj9Ba+6+97CgCzJQ8DDO8J5uLuaq+5LTq+r6qjq/
qnboMCyAOWO2j1S8/EnkzvhJ2MwOTDDtSvEoOSnFrbW1D7+vqnOTrkQnTFDS+/Tq0LvMDLMzDwOMiTwMMLwOc/EnkvzhaN8/
TPKhJP9pxI5+5PMnyJFmyZMkWfBTAcGY7S8V/WO/
9g0luTXJ059EAsEHyMMDwOsnFrbW1D7+vqnOTXLaJbVckWZEky5YtU2gG5pXZjuHwa30HAsDGycMAw5XWVi6
tqcWvtjtHisUlu6OK4AHPNrMZwqKoXVNXFVXVVVd1QVZ+sql4sL2gWbJGpl7zaNy1e9pZUmtMXTX9dkkbjw7Uy1Wtqao3J
zmnqlZX1fVJXpHkHWMIH2DsZjto5KeS6R2d02UwbD2eGhiP9+ZP0isTLxKMS3E1tri1tqC1tourbXzWmsn
tdb2bq29qLX2uvV6OwBMlNkWHBa11j7VWntw9LMyeyyeRhgOHJxQCbYbYFh3+qql+rqq1GP7+W5O4
+AwPgEeRhgOHJxQCbYbYFh/+Y5IQk/zvJHUmOS2IAM4DxkYcBhicXA2yG2U6L+YdJTmmt/ShJqmrHB
MRuYLBqBz8jDA8ORigM0w2x4OL3o4sSZJa+2eJPv2ExIAGyAPAwxPLgbYDLMtODypqnZ4eGFUYJ9ttANlt7wgAnj
h5GGB4cjHAZphtgvxQkr+tqr9M0jL7Nr7eosKgEeThwGGJxcDbIZZFRxaaxdW1aokJK9vrd3Ua2QAr
CMPAwxPLgbYPLPuAjZKprvk3SR5KsqK19kejrmcXJVma5NYkJ6z/
LBwAG7a5eRiA7snFALM32zEctSDSd7ZWtycyYFJ3lpVeyU5K8kVrbXdk1wxWu7d1kwxWgYAAAmSG8Fh9baHa21b4/
e/yTJzUl2TnJ0kgtGm12Q5Ji+YgCYdlV1flXdWVU3rNd2dlXdXlXXjX5eM2SMAJNOLgbYPLPu
AjZKprvk3SR5KsqK19kejrmcXJVma5NYkJ6z/
LBwAG7a5eRiA7snFALM32zEctSDSd7ZWtszyYFJ3lpVeyU5K8kVrbXdk1wxWgYAAAmSG8Fh9baHa21b4/
e/yTJzUl2TnJ0kgtGm12Q5Ji+YgCYdlV1flXdWVU3rNe2Y1VdXlW3jF532NQxAABgS/
TZw2GdqlqamSmDvpnkWa21O5KZokSSZ44jBoAptTLJkY9q09MMAIDe9T6NT1Vtm+TzSc5orf24qma73/
Iky5NkyZIl/QUIbLb9zrxw7J958XZj/8iJ0Fr72qjou76jkxw6en9BkquS/
N74ogIAYBr02sOhqhZkptjw6dbaF0bNa6tq8Wj94iR3bmjf1tqK1tqy1tqyRYsW9RkmwLTR0wwAgN711sOh
ZroynJfk5tbah9dbdWmSU5K8f/
R6SV8xALDl9DRjWvTZa+vaD5zc27EBYK7rs4fDQUlOSnJYVV03+ZM+nlNZgoNR1TVLUmOGC0DMD56mgEA0Lvee
```

```
ji01q5OsrEBGw7v63MBeFx6mgEA0LuxzFIBwDCq6rNJvp5kj6paU1Vvjp5mAACMQe+zVAAwnNbaiRtZpacZ
AAC90sMBAAAA6JyCAwAAANA5BQcAAACgcwoOAAAAQOcUHAAAAIDOKTgAAAAAnVNwAAAAADqn4AAAAAB0buu
hAwAAAGB67Xfmhb0e/+Ltej38nHLbe/fu7dhL3rV6s/
fRwwEAAADonIIDAAAA0DkFBwAAAKBzCg4AANCTqjq/
qu6sqhvWa9uxqi6vqltGrzsMGSNAXxQcAACgPyuTHPmotrOSXNFa2z3JFaNlgIljlgqYA/
ocTXZjtmSUWQBg87TWvlZVSx/
VfHSSQ0fvL0hyVZLfG19UAOOhhwMAAIzXs1prdyTJ6PWZA8cD0AsFBwAAmKOqanlVraqqVXfdddfQ4QBsFg
UHAAAYr7VVtThJRq93bmzD1tqK1tqy1tqyRYsWjS1AgC4oOAAAwHhdmuSU0ftTklwyYCwAvVFwAACAnlTVZ
5N8PckeVbWmqt6c5P1JjqiqW5IcMVoGmDhmqQAAgJ601k7cyKrDxxoIwAD0cAAAAAA6p+AAAAAAdE7BAQAA
AOicggMAAADQOQUHAAAAoHNmqQAA6Mlt79271+MvedfqXo8PAE+EHg4AAABA5xQcAAAAgM4pOAAAAACdU3A
AAAAAOqfgAAAAAHROwQEAAADonIIDAAAA0DkFBwAAAKBzWw8dAOOx35kXDvK5137g5EE+FwAAgGHp4QAAAA
B0TsEBAAAA6JyCAwAAANA5BQcAAACgcwaNBAAA5oW+B0K/
eLsP9HbsJe9a3duxYa7qrYdDVZ1fVXdW1Q3rte1YVZdX1S2j1x36+nwAAABgOH0+UrEyyZGPajsryRWttd2
TXDFaBmAAVXVrVa2uquuqatXQ8QAAMFl6Kzi01r6W5J5HNR+d5ILR+wuSHNPX5wMwK69ore3TWls2dCAAAE
yWcQ8a+azW2h1JMnp95pg/
HwAAABiDOTtLRVUtr6pVVbXqrrvuGjocgEnUkvxNVV1bVcuHDgYAgMky7oLD2qpanCSj1zs3tmFrbUVrbVl
rbdmiRYvGFiDAFDmotfaSJK9O8taqevn6KxV+AQB4IsZdcLg0ySmj96cku/WTMnw/ASGvt9tHrnUkuTrL/
o9Yr/
AIAsMX6nBbzs0m+nmSPqlpTVW9O8v4kR1TVLUmOGC0DMGZV9dSq2u7h7/9FAACzt3VfB26tnbiR
VYf39ZkAzNqzklpcVcnMteAzrbW/GnYoABxDfUaVXVf9eGhGqXhvM/xTlnQ5AvTsTbbHUiI9Dr2epvxZFb2/
MC3s9/
sXb9Xr4zdZbwQGAuau19v0kLx46DgAAJtecnaUCAAAAmL8UHAAAAIDOeaQCAAAGUFW3JvlJkl8kebC1tmzY
iAC6peAAAADDeUVr7Z+GDgKgDwoO9Oq29+499s9c8q7VY/9MAAAAYHknBAQCAJ6zvPzJM6B8UWpK/
qaqW5JOttRVDBwTQJQUHAAAAYxkGttdur6plJLq+q77Wvrb+wVCivrb+BlW1PMnyJFmyZMkQMQJsMbNUAADAAFprt49
e70xycZL9N7DNitbastbaskWLFo07RIBuqSAAAGMApcd0Ki6Fbwb+yauS3DBsVADAd8kgFAACM37OSXFxVyc
w9+Wdaa18ZNiSAbik4AADAmLXWvp/
kxUPHAdAnj1QAAAAAnVNwAAAAADqn4AAAAAB0zhgOAAAwj+135oW9HfvaD5zc27GByaeHAwAAANA5BQcAAA
CgcwoOAAAAQOcUHAAAAIDOKTgAAAAAnVNwAAAAADqn4AAAAAB0buuhAwAAAOam2967d6/HX/
Ku1b0eHxiWHg4AAABA5xQcAAAAgM4pOAAAAACdM4YDAMCU2O/MC3s79sX
b9XZoAOYpPRwAAACAzik4AAAAAJ1TcAAAAAA6p+AAAAAAdE7BAQAAAOicggMAAADQOQUHAAAAoHMKDgAAAE
DnFBwAAACAzik4AAAAAJ1TcAAAAAA6p+AAAAAAdE7BAQAAAOicggMAAADQua2HDuCJ2u/MCwf53Gs/
cPIgnwsAAADzgR4OAAAAQOcGKThU1ZFV9fdV9b+q6qwhYgCYdnIxwLDkYWDSjb3gUFVbJfnjJK9OsleSE6t
qr3HHATDN5GKAYcnDwDQYoofD/
kn+V2vt+621nyf5XK0vyvIOctHlqWwFKxHsVWDGztFfqyUL0PEPy5c/
jstz/uWoU=
```

ZP8UpLvDREfwKTbWB6uquck+VKS5yc5U7GBPhjDgUn36iR3JHnh0IEATLEN5uKqWpzkz5K8qbX20BCBAUyJ
x+Th1toPW2svykzB4ZSqetZQwTG5FByYWFW1T5IjkhyY5B2jG1sAxmhjubiqts/
MX9b+r9baN4aLEGCyPd498ahnw41JDhl/
dEw6BQcmUlVVkk8kOaO1dluSDyT54LBRAUyXjeXiqvqlJBcnubC19hdDxggwyTaRh3epqm1G2+yQ5KAkfz9
cpEwqg0YykapqeZLDW2tvGC1vleSaJL+d5P+T5N8m2TbJ3Une3Fr7b0PFCjCpNpGLL03yf2XmL2oPO7W1dt
3YgwSYYI+Th381SUtSST7eWlsxWKBMLAUHAAAAoHMeqQAAAAA6p+AAAAAAdE7BAQAAAOicggMAAADQOQUHA
AAAoHMKDgAAAEDnFBwAAACAzik4AAAAAJ1TcAAAAAA6p+AAAAAAdE7BAQAAAOicggMAAADQOQUHAAAAoHMK
DgAAAEDnFBwAAACAzik4AAAAAJ1TcAAAAAA6p+AAAAAAdE7BAQAAAOjc1kMHMBs77bRTW7p06dBhADzCtdd
e+0+ttUVDxzEO8jAwV8nFAMPaVB6eFwWHpUuXZtWqVUOHAfAIVfUPQ8cwLvIwMFfJxQDD2lQe9kgFAAAA0D
kFBwAAAKBzCg4AAABA5+bFGA7MHw888EDWrFmT++67b+hQ5qWFCxdml112yYIFC4YOBR6X7/sT4/
sOAOPn/
mXLbcm9i4IDnVqzZk222267LF26NFU1dDjzSmstd999asWZNdd9116HDgcfm+bznfdwAYhvuXLbOl9y4e
qaBT9913X57xjGf48m6BqsoznvEM1VbmDd/
3Lef7DgDDcP+yZbb03qW3gkNVLayqa6rq76rqxqp6z6h9x6q6vKpuGb3u0FcMDOXd8v53THf+G92y/
ndwXTYxD3x2VX1j1V13ejnNUPHCtPCNXjLbMnvrc8eDvcnOay19uIk+yQ5sqoOTHJWkitaa7snuWK0DJ259
dZb88IXvvARbWeffXY++MEPbnSflStX5vTTT+87NKBjvu/APLCxe+Ik+UhrbZ/
Rz5cHixAYq2m6f+ltDIfWWkvy09HigtFPS3J0kkNH7RckuaqkSrJ7/
UVB8Pa78wLOz3etR84udPjAd3xfQd4rE3cEwNzgPuXfvU6aGRVbZXk2iTPT/
LHrbVvVtWzWmt3JElr7Y6qeuZG6qeuZG9l2eZHmSLFmypM8wmSKHHnpoDjjggFx55ZW59957c9555+WQQw5Jktx++
+058sgj873vfS/HHntszjnnnCTJb/7mb/pWfvazn+W4447Le97znq9ze97Ce+97zniTJ0qVL84Y3vCFXXnllkuQzn/
lMnv/85+fUU0/NwoULc+ONN2bt2rX58Ic/
nKOOOiqHHHJIPvaxj2Wffffe9ne9t7927t2Mvedfq3o4NXZi27ztzU595OJGL54u
N3BO/OsnpVXVyklVJ3tla+9EG9nVPDFPkid6/HPPvX553/c5MT4gXHPCqHP+6I/PVv70mSXLBx8/
J83ZdktPO+D+z8Mm/lJu++73cedfdOefdZ+Y1Rxyaw449OR/5w9/Pi1/4b2diOfrX8rH//
AfZe689kiRPfvYvb/b59DpoZGvtF621fZLskmT/qnrh/
r4rWmvLWmvLFi1a1FuMTJ8HH3ww11ximIyTT760Y+u+5+JJLnuuuty0UUXZfXq1bnooovywx/
+MEnyvve9L6tWrcr111n+fr371q7n++uvX7bP99tvnmmuuyemnn5rzzjhjXfutt96ar371q/
nSl76Ut7zlLbnvvvty2mmnZeXKlUmS7373u7u7n//vv9zwf0zPcdmAs2ck/
8iSTPy8xjFnck+dBG9nVPDFPmidy//D/fWJXVN/39un222/apufpLf563nHpifufd71/X/g9rbs9///zK/
NWF/3fedtYf5r777s+bTvzV/Nnn/ipJcsv3bs3Pf/7myaa/
dm5tGJI5OsrarFSTJ6vXMcMTA9NjaYcPtr3/965/Mk++23k++23X2699dZ16w8//PA87wlPy8KFC7PXXnlH/
7hH5Ikn/vc5/KSl7k++67b2688cbcdNNN6/Y58cQYQT171+/
etfX9d+wgkn5ElPelJ233Lbbbvn0d76T448/PpdddEnH/++Tn11FO7PG2YSr7vwHyy/
j1xa23tqBDxUJJzk+w/ZGzA+PR5/3Lz338vN9/yvXX7nHDMzHi0bzjmNfnfnmtX+3rv1X/8O/
z5Oe9KQ8f7fnZulzd8nf/68f5Ff/
w6vy5Su+mgceeCAXXHRxTjrhmCd8rr09UlFVi5I80Fq7t6q2TnJYkvcnuSTJaCSfq7t6q2SfLKJP8lyaVJTkny/
tHrJX3FwHR6xjOekR/96JE9Eu+5555188U++clPTpJstdVBB9dt83D7+ut+8IMf5IMMf/
GC+9a1vZYcddnig1k8WG3v/
8PJTnvKUHHHEEbnkkkvyuc1c99LqtWrergbGG6+b4Dc93G7omravHDjxknOTbJDYMFCYxVn/
cvJ51wTO677+frtpv9/UvylG22yeGHvCxf/+N/G9X5i+/+N/yt1++6Amfa589HoZGHvYpWv/
yrSSXt9Yuy0yh4YiquiXJEaNl6My2226bxsX54orrkgy8+X9yle+koMPPnizj/XjH/
O0pz0ta9euzV//9V8/Yv1FF1207rvL3vZuva/+Iu/yEMPPZTvfe97f73v5899pjpinTaaaflt37rt/
LSl74OO+6445aeIjDi+w7MAxu7Jz6nqlaP2l+R5B1DBgmMT5/3L39z5f/ziPV/eelXkiR/
celXcsB+L17X/oXL/mOu+/xsP7fnzoU3nnmnDnJJwCo8yUHd7Bw/ZFMh7bmTjJ/HPKdiK3vmn9Av/
b0lNcp89ZKq5Psu8G2u9OcnhfnwtJcuGFF++atb31r3vZ3veZh/nxS9+cfbdd9/88i//
cnbbbbcccdNBBj1h/3354ADDshDDz00Uz372s/va99hjj/zKr/xK/5kz/JwoULk8x0i9p++
+3zpje96QmcHbA+33dgLtvEPfFJA4QDzBF93b+87KWPTDf3//znOeSoE/PQQw/lwj/+wj/+wLr2F+y2NK/
81VNz511352Pv/4MsXLjVE+XP/4MsXDjTe+IlL/rlbLfttjn5Dcc+gbP7VzUzU8/
ctmzZsqY76vxw8803Z8899xw6jCdk6dKlWb16dXbaaaenH+Hr7nHrc0Z5/
bbb8+hhx6a6a73znO3nSkzbcwajL36FZKvpVVde21pYNHcc4bCgP+77Pre87c5NZKvo37bkY2DyTfu29//
Yb171/wQGvyt/+9UXZaccdHrHNaHaWf8n3nNK38lrz/qVY/Z//b/fWdeddybcv3XvviY+5cnP/uXN/
j721QeHsugkcBMFfOAAw7I+973vo3+zwcwGXzfAYD55r/
+xSU55KgT857fPS1bbqlPGuVPBVYdxqDP7sAYL6ZpvsXs1TQq67nHzffOMxdvu8AwHz23//
mgAMOyL777ptvXvKxXvVbt2bZLk7LPPPPzknnZTDjX7LPPPzknnZTDDjssu+++e84999999wkyVVXXXZWXv/
zlOfbYY7PXnXnvLW95Sx566a16+7d955ecc73rHuu0eee25+6+d7/e5BzAjbM9x0AmG+6b/
3njz00EP51Gc/nPz/nzPf/V/WHfe8T/9lfvfsc8Z+Pno4MHF+9r0fZZ999M899lm3fM+R1r3tdkwjWw/
ON77xjVRV/vRP/zTnnHHNOPvShDy/++nzjG9/Iv/

zLv2TffffNa1/72iTJNddck5tuuinPfe5zc+SRR+YLX/
hC3vjGN+ZFL3pRzjnnnCxYsCCf+tSn8slPfnLs5wrTzvcdAJhv+rx/edUrDs5fffm/54SjX51lH3t9/r//
129nwYIF+bOLLs7H/8u7x36uCg5MnG222SbXXXfduuWVK1dm1apVSZI1a9bkDW94Q+644478/Oc/
z6677rpuu6OPPjrbbLNNttlmm7ziFa/INddck6c//enZf//
9s9tuuyVJTjzxxFx99dU57rjjcthhh+Wyyy7LnnvumQceeCB7791tdyzg8fm+AwDzTZ/
3Lycc85r87TXfzuuPelUOPeiAfPm/fzX/
dvfd8sCDD+aFe75grOeZeKSCKfO2t70tp59+elavXp1PfvKTue++
+9atq6pHbPvw8sbaTzvttKxcuTKf+tSn8qY3vannnyIHN5fsOAMw3Xd6/
vOnEX82ffe6SXHjRX+XkE47tOfINU3BgqvzzP/9zdt555yTJBRdc8Ih1l1xySe67777777cfffdueqqq/
LSl740yUwXpR/84Ad56KGHctFFF+Xggw9OkhxwwAH54Q9/mM985jM58cQTx3siwOPyfQcA5psnev/
yl5d+Jf9u/5ckSfZ/yYuy5vb/nT+/+Es54ZhXj/dERhQcmCpnn312jj/++BxyyCHZaaedHrFu/
33z2tf+9oceOCB+YM/
+IM8+9nPTpK87GUvy1lnnZUXvvCF2XXXXXPssf9aHTzhhBNy0EEHZYcddhjreQCPz/
cdAJhvnuj9y9Ln7JyjX334un1+9T/8+/y7l+6bHZ7+tLGex8OM4UCvhpgW5qc//
ekjlk899dSceuqpSWaeezr66KcKM3uN8LXvCCrFix4jHtT3nKU3LRRRdtcJ+rr776EaPXwzTzfQcA5ptJu3+5
//YbH7Hub6/5dt726yd3EPWW0cMBtsC9996bF7zgBdlmm21y+OGHP/
4OMICqek5VXVlVN1fVjVX19lH72VX1j1V13ejnNUPHOpf5vgMA8829//
zjvPDg12abhQtz2CEHDhaHHg6Qma5LG3LooYfm0EMPfUz705/+9Hz3u9/
tNyh44h5M8s7W2rerarsk11bV5aN1H2mtfXDA2Abj+w4AzDebff/
ytO1zw9Vf6jeoWVBwAJhQrbU7ktwxev+nVw/u5/Xd6pqaZJ9k3xz1HR6VV
1fVedX1RaPgui/
2S3ndwcAw3AN3jJb8ntTcKBTCxcuzN133+1LvAVaa7n77ruzcOHCoUNhwlTVtkk+n+SM1tqPk3wiyfOS7JO
ZHhAf2sh+y6tqVVWtuuuuux6z3vd9y/m+A8Aw3L9smS29d/
FIBZ3aZddsmbNmzof054fAsXLswuwuydBhMkKpakJkK+n+wydBhMkKpakJliw6dba19Iktba2vXWn5vksg3t21pbkWRFkixbt
uwxV2Xf9yfG9x0Axm/S718evPd/93Tklm3/
zVM3+95FwYFOLViwILGqqSnJfk5tbah9drXzwa3yFJjk1yw5Yc3/
cdAJhvJv3+5bb3ntDbsbdkClEFB4DJdVCSk5KsrqrrRm2/
n+TEqtonSUtya5LfGCI4AAAmm4IDwIRqrV2dpDaw6svjjgVgWlXVwiRfS/
LkzNx7/2Vr7d1VtWOSOi5IszUzx94TW2o+GihOgDwaNBACA/
tyf5LDW2oszM1jvkVV1YJKzklzRWWts9yRWjZYCJoua9aTN+OlpcMPppSY5OcsGo/YIkx4w/
OoB+KTgAAECPqmqr0Vg6dya5vLXLX2zSTPengA39HrMwcMEaAXCg4AANCj1tovMmv7JNklyf5V9cLYZ7ltV
qVWtmtRp/
IDJpeAAAABj0Fq7N8lVSY5MsraqFicz0xVnpvfDhvZZ0Vpb1lpbtmjRonGFCtCJ3goOVfWcqrqyqm6uqhur
6u2j9rOr6h+r6rrRz2v6igEAAIZUVYuq6umj99skeWWS7yS5NMkpo81ck81OSXLJIAEC9KjPaTEfTPlOtq3q2q
7JNdW1eWjdR9prX2wx88GAIC5YHGSC6pqq8z8sHbx1lrl1Vd9s1rJZW0a3e9ibYCY/sE9bFt
+q6uYkO/
f1eQAAMNe01q5Psu8G2u9OcuT4I4Kt9L6vVk8keV6SfTLTA+JDG9nP6LwAAAAwT/
VacKiqBZ7kpNny6tfaFJGmtrR1NDfRQknOT7L+hfVtrK1pry1pryxZBL5Jpbvs2CQ39BUDA
AAAMIw+Z6k4KMlJSVZX1Xwjtt9PcmJV7ZOkJbk1yW/
0GAMAAAAwgD5nHLg6SW1g1Zf7OiYAAAAwGb0gFAAAAAE6vPR7487rVpCg4woD6fG06m69lhAABgb
vFIBQAAANA5BQcAAACgcwoOAAAAQOcUHAAAAIDOKTgAAAAAnVNwAAAAADqn4AAAAAB0TsEBAAAA6JyCA8CE
qqrnVNWVVXVzVd1YVW8fte9YVZdX1S2j1x2GjhVgUm0iF59dVf9YVdeNfl4zdKwAXVNwAJhcDyZ5Z2ttzyQ
HJnlrVe2V5KwkV7TWdk9yxWgZH5sLBcnyUdaa/uMfr48XIgA/VBwAJhQrbU7Wmvf
SZKbk+yc5OgkF4w2uyDJMYMECDAFNpGLASaeggPAFKiqpUn2TfLNJM9qrd2RzNwIJ3nmgKEBTI1H5eIkOb2
qrq+q8z3eBkwiBQeACVdV2yb5fJIzWms/3oz9llfVqpadddd/UXIMAU2EAu/
kSS5yXZJ8kdST60kf3kYmDeUnAAmGBVtSAzN7ifbq19YdS8YdS8tqoWj9YvTnLnhvZZtra1orS1rrS1btGjReAIG
mEAbysWttbWttV+01h5Kcm6S/Te0r1wMzGcKDgDATqqoqyXlJbm6tfXi9VVZcmOeSnOOX0/pQkl4w7NoBpsbFc/
HDhd+TYJDeMOzaAvm09dAAwbre9d+/ejr3kXat7OzZsgYOSnJJb/STvT/
K5qnpzktuSHD9MYANYYWO5+MSrjrkt8YIjiAPik4AEyo1trVSWojqw8fZywA02ojufhApi1fGoMBPYSwA02oTudg0mMDE80gFA
AAA0DkFBwAAAKBzvRUcquo5VXVlVd1dcvTdW1dtH7OeVnZMcneSC0WnZMcneSC0WYXJDmmrxgAAACAYYxldIeqWpp
k3yTfTPTPKs1todyUzlZ69UrzqkzxErIskzZxxAAAAMD69T4tZE9e2/k3e9T4tZVdsm+XRScsneSC0PSk4AEYqXUtyJvsm9T4tzavyx6mv
MC3s9/
sXb9Xp4AAAA0nMPh6pakJliw6dba18YNa+tqsWj9YuT3LmhfVtrK1pry1pryxYtWtRnmAAAAEDH+pylopKc
l+Tm1tqH11t1aZJTAAAAAw+jzkYqDkpyUZHVVXTdq+/0k70tRu9PSXJJJXzEAAAAw+jzkYqDkpyUZHVVXTdq+/0k70/
yuap6c5LbkhzfYwwAAAHorOLTWrk6/6ysQEbDu/

rcwEAAIDhjWWWCgAAAGC6KDgAAAAAnVNwAAAAADqn4AAAAAB0rs9ZKgAAAICR/
c68sNfjX7xdr4ffbHo4AAAAAJ1TcAAAAAA6p+AAAAAAdE7BAQAAAOicggMAAADQOQUHAAAAoHMKDgAAAEDn
FBwAAACAzik4AAAAAJ1TcAAAgJ5U1XOq6sqqurmqbqyqt4/
ad6yqy6vqltHrDkPHCtA1BQcAAOjPg0ne2VrbM8mBSd5aVXslOSvJFa213ZNcMVoGmCgKDgAA0JPW2h2ttW
+P3v8kyc1Jdk5ydJILRptdkOSYQQIE6NHWs9moqq5orR3+eG0A9EMeBhjeE83FVbU0yb5JvpnkWa21O5KZo
kRVPXMj+yxPsjxJlixZ8gSih7lrvzMv7PX4F2/
X6+HZhE0WHKpqYZKnJNlp9FxZjVZtn+TZPccGMPXkYYDhdZGLq2rbJJ9PckZr7cdV9Xi7JElaayuSrEiSZc
uWtc0MHWBQj9fD4TeSnJGZRHpt/jW5/jjJH/
cXFgAj8jDA8J5QLq6qBZkpNny6tfaFUfPaqlo86t2wOMmdnUcNMLBNFhxaa3+U5I+q6m2ttY+NKSYARuRhg
OE9kVxcM10Zzktyc2vtw+utujTJKUneP3q9pKt4AeaKWY3h0Fr7WFX9uyRL19+ntdbvwzYAJJGHAeaCLczF
ByU5Kcnqqrpu1Pb7mSk0fK6q3pzktiTH9xEzwJBmO2jknyV5XpLrkvxi1NySuNEFGIMtzcNVdX6So5Lc2Vp
74ajt7CS/nuSu0Wa/
31r7cvdRA0yWLcnFrbWr86+PYDyagX+BiTargkOSZUn2aq0ZqAZgGFuah1cm+XgeezP8kdbaB7sIDGCKuCc
G2AxPmuV2NyT5N30GAsAmbVEebq19Lck93YcDMJXcEwNshtn2cNgpyU1VdU2S+x9ubK29rpeoAHi0rvPw6V
V1cpJVSd7ZWvtRBzECTDr3xACbYbYFh7P7DAKAx3V2h8f6RJI/zMxzx3+Y5ENJ/
uOjN6qq5UmWJ8mSJUs6/HiAeevsoQMAmE9mO0vFV/
sOBICN6zIPt9bWPvy+qs5NctlGtluRZEWSLFu2zPPKwNRzTwyweWY7S8VPMvOXsCT5pSQLkvxLa237vgID4
F91mYeranFr7Y7R4rGZeSYZgfMfhnhhg88y2h8N26y9X1TFJ9u8jIAAea0vzcFV9NsmhSXaqqjVJ3p3k0Kra
JzM3zbcm+Y1uowWYTO6JATbPbMdweITW2l9V1Vmb2sbc7wD9mU0eHm134gaaz+shJICpM9tcDDCtZvtIxev
XW3xSZuYgfrzneVfG3O8AndjCPAxAh+RigM0z2x4O/
2G99w9mpgvu0ZvaobX2tapausXmVhAfAom52HAeicXAywGWY7hsObOvxMc78DbKaO8zAAW0AuBtg8s32kYpck
H0tyUGa6jV2d5O2O2ttTWb+Xmzmvt99Jnmfwdm7bb37t3bsZe8a3Vvx56tDvMwAFtILgbYPE+a5XafSnJpkmc
n2TnJF0dtm6vai19ovW2kNJzs0mRvVtra1orS1rrS1btGjR5n4UwKTpJA8D8ITIxQCbYbYFh0WttU+11h
4c/axMsmhlvt+XpcOkU8oYmtiffVagUzTdugQdxewRIDTvsFIxbfv6zGhmgC5kVUVl1tEqtmWAEEcgrqTmGs
73wD1WjkUm2tvHSzS8ijwB8AAEM0MlZC3NNIdtaGAWzaa8ukQFJMCMCBeaIzy2Mue+NUPPS4awKDH+z
fvBvmsDDQOi2AlygMwHOzrKVmCOrUwExLGsDMQ8yRzSeDQVDxq6+Jh9xPnldH9v3T7HT06xw5fm
wvMw/Vk1upDWtQGIE6oqaR1p+MK5RqXMazDtZhAyREDyBtywe1tUwPUv1fbnCzqCWPV7cfUVyJqe4CaN2y
ZaGh3UmMNY2oAzRpFQzXwAXiyPkwVAZdY0eFMx9NZAD5LUNXXlq68BzrVQ7Klcp83uGOzD13pL/
yx0Lq8mRTh/uD3Y7jhkcZJNLBGswgUJ7JRIROtLoVg2KEmU0ONYkS+UjQbvm2AH/Sw6+W7Z/
B9h5ogkL7Lvr1zh2kmGaqbAYpCtDSFGtgMTb/+pm1WNN/

```
9wMm9Hh8AgEeazcxtAPOZHg4AADBmZm4DpoEeDgAAMH5mbgMmnoIDAACMmZnbgGngkQoAAACgcwoOAAAAQO
cUHAAAAIDOKTgAAAAAnVNwAAAAADqn4AAAAAB0TsEBAAAA6JyCAwAAANA5BQcAAACgcwoOAAAAQOcUHAAAA
IDObT10AACwMbe9d+9ej7/
kXat7PT4AwDRTcNgIN7kAAACw5TxSAQAAAHROwQEAAADonIIDAAAA0DkFBwAAAKBzCg4AAABA5xQcAAAAgM
4pOAAAAACd23roAJgbbnvv3r0de8m7Vvd2bAAAAOYmBQcAAGCD+vyjVLL5f5ja78wLe4pkxrUfOLnX48O08
UgFAAAA0DkFBwAAAKBzgxQcqurIqvr7qvpfVXXWEDEATDu5GGBY8jAw6cZecKiqrZL8cZJXJ9kryYlVtde4
4wCYZnIxwLDkYWAaDNHDYf8k/6u19v3W2s+T/HmSoweIA2CaycUAw5KHgYk3RMFh5yQ/
XG95zagNgPGRiwGGJQ8DE69aa+P9wKrjk/z71tppo+WTkuzfWnvbo7ZbnmT5aHGPJH8/1kCTnZL805g/
c0jTdL7OdTINca7Pba0tGvNndmI2uXgO5OHEf8OTyrlOpqHOdV7mYvfEc9Y0na9znUxz6p546zEHksxUb5+
z3vIuSW5/9EattRVJVowrqEerqlWttWVDff64TdP5OtfJNE3n2pHHzcVD5+Fkuv5dnetkcq5sgnviOWiazt
e5Tqa5dq5DPFLxrSS7V9WuVfVLSd6Y5NIB4gCYZnIxwLDkYWDijb2HQ2vtwao6Pcl/
S7JVkvNbazeOOw6AaSYXAwxLHgamwRCPVKS19uUkXx7iszfDoN2IBzBN5+tcJ9M0nWsn5OI5x7lOJufKRsn
Dc9I0na9znUxz6lzHPmgkAAAAMPmGGMMBAAAAmHAKDo9SVedX1Z1VdcPQsfStqp5TVVdW1c1VdWNVvX3omP
pSVQur6pqq+rvRub5n6Jj6VlVbVdX/
rKrLho6lT1V1a1WtrqrrqmrV0PHQDbl4Mk1bLp6WPJzIxZNIHp5M05aHk+nJxXM1D3uk4lGq6uVJfprkwtb
aC4eOp09VtTjJ4tbat6tquyTXJjmmtXbTwKF1rqoqyVNbaz+tqgVJrk7y9tbaNwYOrTdV9dtJliXZvrV21N
Dx9KWqbk2yrLU2LXMrTwW5WC6eBNOShxO5eBLJw/LwpJiWXDxX87AeDo/SWvtaknuGjmMcWmt3tNa+PXr/
kyQ3J9l52Kj60Wb8dLS4YPQzsdW2qtolyWuT/
OnQscCWkIvl4vlOHma+k4fl4UkgFw9PwYEkSVUtTbJvkm8OHEpvRt2prktyZ5LLW2sTe65JPprkd5M8NHAc
49CS/
E1VXVtVy4cOBp4IuXiifDTTk4cTuZgJIQ9PnI9menLxnMzDCg6kqrZN8vkkZ7TWfjx0PH1prf2itbZPkl2S
7F9VE9k9sKqOSnJna+3aoWMZk4Naay9J8uokbx11AYV5Ry6eHFOYhxO5mAkgD0+WKczFczIPKzhMudGzW59
P8unW2heGjmccWmv3JrkqyZHDRtKbg5K8bvQc158nOayq/uuwIfWntXb76PXOJBcn2X/
YiGDzycUTZ6rycCIXM//
JwxNpqnLxXM3DCg5TbDRozHlJbm6tfXjoePpUYuq6umj99skeWWS7wwaVE9aa/
+ptbZLa21pkjcm+R+ttV8bOKxeVNVTR4M7paqemuRVSSZ+NG0mi1w8ebl4mvJwIhcz/
8nDk5eHk+nKxXM5Dys4PEpVfTbJ15PsUVVrqurNQ8fUo4OSnJSZat91o5/XDB1UTxYnubKqrk/
yrcw8rzbRU+NMiWclubqq/
i7JNUm+1Fr7ysAx0QG5WC5mXpGLJ5A8LA8zr8zZPGxaTAAAAKBzejgAAAAAnVNwAAAAADqn4AAAAAB0TsEB
AAAA6JyCAwAAANA5BQcmUlU9p6p+UFU7jpZ3GC0/
d7S8fVX9Y1V9fNhIASbXpnJxVf1ivennLh06VoBJ9Dh5eElV/U1V3VxVN1XV0oHDZQKZFpOJVVW/
m+T5rbXlVfXJJLe21v7zaN0fJVmU5J7W2ulDxgkwyTaWi6vqp621bYeOD2DSbSIPX5Xkfa21y6tq2yQPtdb
+f4MGy8RRcGBiVdWCJNcmOT/JryfZt7X286raL8mZSb6SZJmCA0B/
NpGLFRwAxmBDeTjJ85OsaK0dPGRsTL6thw4A+tJae6CqHi4svGp0g/
ukJB9KclKSuwcNEGABEKlKSwwcNEGAKbCgXj1YtrKpVSR5M8v7W2l8NFSPAJNvIPfELktxbVV9IsmuS/
57krNbaL4aMlcljDAcm3auT3JHkhaPPyX5t9G5OEmWtNaWJfk/
kny0qp43SGQA88v6uXiCJyEtB4mMiabgwMSqqn2SvDLJAZmZ1mdxkgOTnFpVtyQ5J8nJVXX2cFECTLaN5OIb2SHJHkwCTvqKrFSV6W5PSqujXJB5OcXFX
vHyxIgAm3kVyc1trto9fvJ7kqM118AejYRvLwmiT/s7X2/
dbag0n+KslLBguSiaXgwESqqkryiSRntNZuS/KBJB9srf2/
WmtLWmtLM1PRvbC1dtaAoQJMrI3l4tEo6U8ebbNTkoOS3DRcpACTaWN5OMm3kuxQVYtGmx4eZgeKDgwqX4
9yW2ttctHy/93kn9bVb8yYEwA02aDuTjJi5Ksqqq/S3JlZsZwcKML0L2N5eGDM/
PHtyuqanWSSnLuMCEyycxSAQAAAHRODwcAAACgcwoOAAAAQOcUHAAAAIDOKTgAAAAAnVNwAAAAADqn4AAAA
AB0TsEBAAAA6JyCAwAAANC5/z/wGN8IUM/rNgAAAABJRU5ErkJggg==\n",
      "text/plain": [
       "<Figure size 1296x720 with 6 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "fig, axes = plt.subplots(2,3,figsize=(18,10))\n",
    "axes = [plot for row in axes for plot in row]\n",
    "#plt.legend(labels=['Unhappy', 'Happy'])\n",
    "for ind, feature in enumerate (['X1', 'X2', 'X3', 'X4', 'X5', 'X6']):\n",
    "    sns.countplot(x=df[feature], hue=df['Y'], ax=axes[ind])\n",
    "    axes[ind].legend(labels=['Unhappy', 'Happy'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 10,
   "id": "1f03dd50",
   "metadata": {
    "scrolled": true
   },
   "outputs": [
    {
     "data": {
      "text/html": [
       "<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>Y</th>\n",
       "      <th>X1</th>\n",
       "      <th>X2</th>\n",
       "      <th>X3</th>\n",
       "      <th>X4</th>\n",
       "      <th>X5</th>\n",
       "      <th>X6</th>\n",
       "    </tr>\n",
       "  </thead>\n",
       "  <tbody>\n",
       "    <tr>\n",
       "      <th>count</th>\n",
       "      <td>126.000000</td>\n",
       "      <td>126.000000</td>\n",
       "      <td>126.000000</td>\n",
       "      <td>126.000000</td>\n",
       "      <td>126.000000</td>\n",
       "      <td>126.000000</td>\n",
       "      <td>126.000000</td>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th>mean</th>\n",
       "      <td>0.547619</td>\n",
       "      <td>4.333333</td>\n",
       "      <td>2.531746</td>\n",
       "      <td>3.309524</td>\n",
       "      <td>3.746032</td>\n",
"      <td>3.650794</td>\n",
"      <td>4.253968</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>std</th>\n",
"      <td>0.499714</td>\n",
"      <td>0.800000</td>\n",
"      <td>1.114892</td>\n",
"      <td>1.023440</td>\n",
"      <td>0.875776</td>\n",
"      <td>1.147641</td>\n",
"      <td>0.809311</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>min</th>\n",
"      <td>0.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"      <td>1.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>25%</th>\n",
"      <td>0.000000</td>\n",
"      <td>4.000000</td>\n",
"      <td>2.000000</td>\n",
"      <td>3.000000</td>\n",
"      <td>3.000000</td>\n",
"      <td>3.000000</td>\n",
"      <td>4.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>50%</th>\n",
"      <td>1.000000</td>\n",
"      <td>5.000000</td>\n",
"      <td>3.000000</td>\n",
"      <td>3.000000</td>\n",
"      <td>4.000000</td>\n",
"      <td>4.000000</td>\n",
"      <td>4.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>75%</th>\n",
"      <td>1.000000</td>\n",
"      <td>5.000000</td>\n",
"      <td>3.000000</td>\n",
"      <td>4.000000</td>\n",
"      <td>4.000000</td>\n",
"      <td>4.000000</td>\n",
"      <td>5.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>max</th>\n",
"      <td>1.000000</td>\n",
"      <td>5.000000</td>\n",
"      <td>5.000000</td>\n",
"      <td>5.000000</td>\n",
       "      <td>5.000000</td>\n",
       "      <td>5.000000</td>\n",
       "      <td>5.000000</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "</div>"
      ],
      "text/plain": [
       "                Y          X1          X2          X3          X4  
X5  \\\n",
       "count  126.000000  126.000000  126.000000  126.000000  126.000000  
126.000000   \n",
       "mean     0.547619    4.333333    2.531746    3.309524    3.746032  
3.650794   \n",
       "std      0.499714    0.800000    1.114892    1.023440    0.875776  
1.147641   \n",
       "min      0.000000    1.000000    1.000000    1.000000    1.000000  
1.000000   \n",
       "25%      0.000000    4.000000    2.000000    3.000000    3.000000  
3.000000   \n",
       "50%      1.000000    5.000000    3.000000    3.000000    4.000000  
4.000000   \n",
       "75%      1.000000    5.000000    3.000000    4.000000    4.000000  
4.000000   \n",
       "max      1.000000    5.000000    5.000000    5.000000    5.000000  
5.000000   \n",
       "\n",
       "               X6  \n",
       "count  126.000000  \n",
       "mean     4.253968  \n",
       "std      0.809311  \n",
       "min      1.000000  \n",
       "25%      4.000000  \n",
       "50%      4.000000  \n",
       "75%      5.000000  \n",
       "max      5.000000  "
      ]
     },
     "execution_count": 10,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "# Description of statistical information\n",
    "# y mean = 54.7% , this means that slightly more customers are happy than sad\
n",
    "\n",
    "# x1 mean = 4.3   x2 mean = 2.5   x3 mean = 3.3  x5 mean = 3.6    x6 mean = 
4.2\n",
    "\n",
    "# x1 top 75% = 5.0, 25% = 4.0   This means that x1 scores are mostly between 4 
and 5\n",
    "df.describe()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 11,
   "id": "78813cca",
   "metadata": {},
   "outputs": [],
   "source": [
    "df = df.rename(columns={\"Score\": \"Y\"})\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 12,
   "id": "edf0ad9a",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<AxesSubplot:xlabel='X1'>"
      ]
     },
     "execution_count": 12,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAWAAAAAEGCAYAAABbzE8LAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAKnklEQVR4nO3df6jd913H8dc7ud3c7CBrUrSkc/
kjfwwnOEsplcoc2s5siv7jH/tDW/zDMZE0q6Cgfyj+N/
DX2gtOShUSdYqiEylZMPUH+o+TpNal0jEuM92aFdvdsa2y0HmTj3/
cM4i3N8k5yT3nnZM8HnDpOfd8cz5vPpzz7Peek3tSY4wAsHi7ugcAuFUJMEATAQZoIsAATQYoMnKLAfv27
dvHDhwYE6jANycTp8+/
ZUxxp1bvz9TgA8cOJBTpQ07t3FQAt4CqenG773sJAqCJAAM0EWCAJgIM0EATAQZoIsAATQYoIkA
AzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYz/
ZtwwPJbXV3N2tpa9xhxL49y5c9mzZ0+eenuqpHb9vvAYZbzNraWp57/
oVceOsd3aMshd2vref8+fNzuW8BhlvQhbfekfPv+mD3GEvh9mf/
eG737TVggCYCDNBEgAGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQ
RYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJg
IM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAszSW11dzerqavcY3KwuXsjrr78+l7j
8DNbIxcvHhxLnftDBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEAT
AQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCL
AAE0EGKCJAAM0EWCAJgIM0ESAAZoIMECThQR4fX09jz76aNbX1xexHMBSWEiA2mjNTYsWcWOLWA5gKc
w9wOvr6zlx4kTGGDlx4oSzYICJuQf42LFjuXjxYpLkwoULzoIBJuYe4GeeeSYbGxtJko2NjZw8eXLeSwIshbkH
+MEHH8zKysYnT1dWVvLQQw/NewmApTD3AD/yyCPZtWtjmd27d+fhhx+e95IAS2HuAd67d28OHjyYqsrBgwez
d+/eeS8JsBQW8hNPp9Ohi+8+OHT2TwTJcRDmdjR4tcJB2tx3OMnnJzn7wnO/qXYlG/uyp/fiM0PzZyTjH2uVM
mUoQsADJaSa8UJgWAVFacBkzFtjDaqqVMdEN4gFnAExqWHrzFv/+wIT/
o7aMWaAr3PCTSpTydLOcx9SfezXfBHPN3LqxnZvzc9HGLRDnrMxiDewmC/y7p/HUtjf9rqGzRD3nNTdI/Gx+Jbs6L9a/n1
DfrxFYvSyUu7v
neudY4w7t35zpgBfj6o6Nca4dyGLzcBcszHXbMw1m1ttLi9BADQRYIAmiwzwkwtcaxbmmo25ZmOu2dxScy3
sNWAA/
j8vQQA0EWCAJjsa4Kr6o6p6paqev8ztVVVPVNVaVX22qu7ZyfWvY673VdXXq+q5ydevL2iud1TVP1bVC1X1
n1V1ZJtjFr5nU8618D2rqu+oqn+rqv+YzPWb2xzTsV/TzNXyGJusvbuq/
r2qnt7mtpbn5BRzdT0nz1bVmcmap7a5fWf3a4yxY19J3pvkniTPX+b2Dyb5dJJKcn+Sz+zk+tcx1/
uSPL2IWbase1eSeyaX35bk80m+t3vPppxr4Xs22YPbJ5dvS/KZJPffAPs1zVwtj7HJ2r+U5JPbrd/
1nJxirq7n5Nkk+65w+47u146eAY8x/jnJV69wyE8lOTY2/
WuSPVV1107OcI1ztRhjvDzGeHZy+bUkLyTZv+Wwhe/ZlHMt3GQP/
mdy9bbJ19Z3kTv2a5q5WlTV3Ul+PMlTlzmk5Tk5xVw3qh3dr0W/Brw/yZcuuf5SboAn9sQPTn6E/
HRVvXvRi1fVgSQ/
kM2zp0u17tkV5koa9mzyY+tzSV5JcnKMcUPs1xRzJT2PsY8n+ZUkFy9ze9XVaer6sP
b3L6j+7XoANc237sRzhSezebvan9/ktUkf7PIxavq9iR/
leSjY4xvbL15mz+D27ylwtezbGuDDGeE+Su5PcV1Xft+WQlv2aYq6q9W27f0f1adIBfSvKOS67fneTLC57hDcYY3/
USSV8YYp6902Dbfm+t+TTlX13PygTHGPUk+kOQXq+q9W27f0f1adIBfSvKOS67fneTLC57hDcYY3/
j2j5BjjONJbquqfYtYu6puy2bk/nSM8dfbHNKyZ1ebq3PPJmt+Lck/JTm05abWx9jl5mrarweS/
GRVnU3y50l+pKr+ZMsxHft11bm/vtKkk8luW/
LITu6X4sO8N8meXjyTuL9Sb4+xnh5wTO8QVV9d1XV5PJ92dyX9QWsW0n+Mk
Oq9kwuvyXJg0k+t+Wwjv266lwd+zXG+NUxxt1jjANJPpTkH8YYP7PlsIXv1zRzNT2+vrOq3vbty0nen2Tr3
5za0f1aueZpt1FVf5bNdy/3VdLSX4jm29IZIzxB0mOZ/NdxLUk30zyczu5/nXM9dNJfqGqNpKcT/
KhMXnLc84eSPKzSZ6Sc5MXj9Mkl9L8j2XzNaxZ9PM1bFndyU5W1W7s/mE/
IsxxtNV9ZFL5urYr2nm2nm6nqMvcENsF/
TzNWxX9+V5FOT7q8k+eQY48Q898uvIgM08ZtwAE0EGKCJAAM0EWCAJgIM0ESAWSq1+Ult/1VVd0yuv31y/
Z1VdaKqvlbbfLoW3IgEmKUyxvhSkk8k+djkSx9Lsj7GeDHJ72Q7/nvCSk8k+djkWx9L8uQY48Ukv5XNv78MS0GAWUa/
l+T+qvpokh9K8jtJMsb4+ySvNc4FM9nR34SDRRhj/G9V/HK/G9V/
XKSE0neP8b4VvdMcC2cAbOsPpDk5SRbP3UMloYAs3Sq6j3Z/w6xhxlEWE8I0Hq0jwWC3gA8RhHgSYpTL5hKxPZPMzir+Y
zTfefrt3Krg2Asyy+fkkXxxjnJxc//
0k76qqH66qf0nyl0l+tKpeqqofa5sSpuDT0ACaOAMGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZr8HzomlrLzr
2A6AAAAAElFTkSuQmCC\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "#X1 boxplot showing a score leaning towards 4.0 and 5.0\n",
    "sns.boxplot(x=df['X1'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 13,
   "id": "303558d7",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<AxesSubplot:xlabel='X2'>"
      ]
     },
     "execution_count": 13,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png": 
"iVBORw0KGgoAAAANSUhEUgAAAWAAAAEGCAYAAABbzE8LAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAKjElEQVR4nO3df6jd913H8de7ScsyO4klRWs6lz/
yh7qBWymlUpjBH8PNMf8ZLBAV90dLh4So4EAZin8rsnr/
MLZ14GZmUeZk1G4oqPiPbiS12zo6tqN22qzYbsNtssBs9vGPcwLJ7b0356Qn3/e9y+MBl5xzz/
fe75sP5zz53u+595saYwSA6d3UPQDAjUqAAZoIMEATAQZoIsAATfavsvGhQ4fGkSNHrtMoAN+dzp0795Uxx
u2bP79SgI8cOcZ8+ubyqAG0BVfWmrzzsFAdBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoI
MEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQJOV/k84prGxsZHZbNY9xp5x/vz5HDx4MI8+
+mj3KLASAd6FZrNZnnr6mVx89Wy4cKF7DFiZAO9SF199Wy788Nu6x9gTbn3yQ90jwDVxDhi
giQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAz
QRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJ
gIM0ESAAZoIMEATAQZoIsAATQQYoMkkAd7Y2MjGxsYUuwKuwutx99g/
xU5ms9kUuwGW4PW4ezgFAdBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQ
YoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAA
M0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAA+zgzJkzOXbsWB577LG1f28BBtjBI488kiQ5ffr02r+3A
ANs48yZM1fcX/dR/PcX/dR8P61frdtnD9/PhcuXMipU6em2N2eN5vNctO3R/
cYe8d3Lnp+rWA2m+XAgQPdY+wJl45+Lzl9+nSOHz+
+tu9/1SPgqnqqs5W1dkXX3xbTsGuNFd9Qh4jPFwkoeT5O67776mw7LDhw8nSR566KFr+fIbzqlTp3Lu3/
+7e4y946Z9OfCqWzy/luRd33oWCAbdx///1X3H/wwQfX+v0d1/1X3H/wwQfX+v0FGGAbJ06uOL+Os//
JgIMsKNLR8HrPvpNJvotCIC96sSJEl4XR8AATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBi
giQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAz
QRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGa7J9iJ0ePHp1iN0ASvB53i8AS
vB53j0kCfPLky
Sl2AyzB63H3cAoCoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEG
aCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwAB
NBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE32dw/
A1vZ962s58PknusfYGy6+lOSW7ilgZQK8Cx09erR7hD3l/
PmXcvDgwe4xYGUCvAudPHmyewRgAs4BAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBE
gAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZrUGGP5jateTPKla9zXoSRfucavvZ7MtRpzcZcq/
lunet1Y4zbN39ypQC/
ElV1doxx9yQ7W4G5VmOu1ZhrNTfaXE5BADQRYIAmUwb44Qn3tQpzrcZcqzHXam6ouSY7BwzAlZyCAGiwAB
N1hrgqvpAVb1QVU9v83hV1R9W1ayqPlNVd61z/
69grmNV9fWqemrx8dsTzfXaqvq7qnq2qp5XVae22GbyNVtyrsnXrKpeVVWfqqpPL+b6/S22mfy5
9r2vqv61qh7f4rXGW1+QSc3W9Jp+tqs8u9nl2i8fXu15jjLV9JHlzkruSPPL3N429Zsv8j/aLa7iqlmRE
Hkjw+xSyb9ntHkrsWt1+T5AtJfHGE8ubn8zyTNJDm/qUU+q/D4jXLVlqy5aDfXzw/TqUO
+S5GBV3bHOGa5xrhZGzTNyyRKkq/JGbfRXa6wT/vf3kh9ks+RoSQXXk6sBw4hSx/SgSIs/ndTunjM9eaXgK
Dhw+bhFSpYYkfmN6y7ekAm/H2aOWtqkNP3RFB+ahOfeIwzjWbuGTc3/
Lhrfbf9ZpcYq6u1+SzSQ7t8Pha12utR8BjjH9MMvrI3JB8cc/
+S5GBV3bHOGa5xrhZjjOfHGE8ubn8zyTNJDm/
abPI1W3KuyS3W4H8Xd29efGx+F7ljvZaZq0VV3Znk55I8mLa/JJebarda6XlOfAz6c5L8uu/
9cdsELe+HHFz9CfryqXj/1zqvqSJI3ZX70dLnWXHqrhzRY/
tj6V5IUkfzfG2BXrtcRcSc/z631JfiPbv5N26Xo7+k3SPJXiwRSMcbPRtc5XJzKm/3+Z
Z+d9q/1iSjSR/
PeXOq+rWJmJB9J8qtjjG9sfniLL5lkza4yV8uajTEujjHeleTOJPdU1Rs2bdKyXkvMNfl6VdXPJ3lhjHFup
82tn/NHvc/lNOsZXXA+O+1YymPHVJNEyb4m1aeJ5l8u9e/xQxcM1XWx+uOwRVSbnE4kLmHDpy6ZqzoBp+L98
k7wx8rLBDmEmqkLNT7L
uqbs48cmfGGH+1Ljw8+rcmxSYta3a1uPb8w8uMMb5x2eZcX3P0vJzyKRO/PEtg+eyMsFmlVjEQOi/
7kryjqp5N8N8liSn6yqk/e6n7
e4cYxxJcJzJ348xfmHTZ2pZ+NuAx+ZQPV1iU+vqeqeNt5nDpeGzUh+qVwf/ZqT11ZfrswDf9epG+SPheMC/
JQVT2X5HcyfjIC3Ovf/CuZ6zu6r5bTXuZ6bqeqZVxm4NpLlFFDGGrx6r/5Lnq8QlTLvf/AJ5WYq/
XPNr0b2map6V/escDV+DY09p6rem+ToGOOBqvrjzK9g9g9ZHMr4vzxxar6wSTnkvzI4i/
TYFcSYPacxZ9Jn0vygjD0vygST3JnTGOPbm7b5dJJ3jjG+2DAiL7GWtfwkHUxhj/F9V/
UaSTyR5yxbxvSfJLUn+rWM+rWJZzfijJu8cYOl3qENoJMHtOz0bTbzfJrpjpW82uzfijJu8cYO13qENoJMHtOVb0xyc9k/j8S/
Noiuqmq703yN0net7hYNuxqAsyesrhC2C8kea37hjbNu7+v6pu
yfzygR8cY/xl54ywLG/Csacs/
peCnxpjvGtxf1+ST2V+mcD3JfncZZv/
8hjjqcmHhCUJMEATAQZoIMEATpyAAmggwQBMBBmgiwABNBBigiQADNBFggCb/D4c+njvxkJB0AAAAAElFTkSuQmCC\
n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "#x2 boxplot showing scores between 2.0 and 3.0\n",
    "sns.boxplot(x=df['X2'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 14,
   "id": "abc7d5b9",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<AxesSubplot:xlabel='X3'>"
      ]
     },
     "execution_count": 14,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAWAAAAEGCAYAAABbzE8LAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAKjElEQVR4nO3df6jd913H8dd7SXFhm2ySEsNNXZD7h6i4WU
rpVh3BH8Oton8oOKhj+2Njf/lqNoDa1YLBTBJ1I7YaCjv3lRjKr62gnF+xcu0g7S9bVZr8/8/+
+N8o+ntTXJucs9535M8HnDJOfd8k++bD9/z5Hu/
556TGmMEgOV7RfcAANcqAQZoIsAATQYoIkAAzTZu52N9+/fPw4fPrygUQCuTqdOnfrrSGOP6zd/
fVoAPHz6ckydP7P7txUANeAqvr8Vt93CQKgiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJg
IM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAm2/o/4WA3uuuu3LmzJmsra11j8DEEcvXc/
To0e4xrnkCzMo7ffp0PiPfX3M4z2PPi891j8DEEcvVYc/
enP2Bt3dPsRL2PfFI9whMXAMGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYI
AmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0
ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwLvQsWPHcuzYse4xe4xgCz2+bh3If8q
V2RjY6N7BGCyyOejM2CAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwAB
NBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoI
kAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNlhLg48eP58uwQeXsTuAlbCUAD/
00ENJkhMnTixjdwArYeEBPn78+EvuOwsGmnm76B2cO/
s958SJE7nzzjsXvduV9vTTT+fs2bO59957u0dDZCWfPnk1G9xSr4xVffT4bG19xfM1pY2Mj+/btW8i/
fckz4Kq6p6OVtXt9XiL2A8k/fLcV7
ePcbK+PYrvzvr33/A8TWnRf6ksPBrwLfffvLt7t9xxxz2iL3iXASlh4gO+++++wsDjdAWbZNM/
+6X3Hf9F2BmKb+Gdu4s2NkvwP9b+G9BJLOz4M1nwdDXOm9FBmgiwABNBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0GRv9wC83
cB3oWOHj3aPcIwWQYsIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoI
sAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0E
GKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzTZ2z0A7IhvfTP7nnike4xe4qVsOfF55Ic6B6DCDB
```

```
XgYMHD+bMmTNZWxOV+RzI+vp69xBEgLkKPPDAA90jwGVxDRigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE
0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQpMYY829c9WySz1/
mvvYn+dJl/t1FMtf2mGt7zLU9V+tcrx9jXL/
5m9sK8JWoqpNjjJuWsrNtMNf2mGt7zLU919pcLkEANBFggCbLDPD9S9zXdphre8y1PebanmtqrqVdAwbgpV
yCAGgiwABNdjTAVfVgVT1TVY9d4PGqqj+oqo2qvequnEn938Fcx2pqi9X1aPT1/
uXNNcNVfXPVfV4VX22qu7dYpulr9mccy19zarqlVX1qar6t2muD26xTXyzE27XtPVf1rVT28xWMtz
8k55up6Tj5ZVZ+Z9nlyi8d3dr3GGDv2leQtSW5M8tgFHn97ko8mqSS3JPnkTu7/
CuY6kuThZcyyab8Hk9w43X5Nkv9I8oPdazbnXEtfs2kNXj3dvi7JJ5PcsgvWa565Wo6xad+/
leTDW+2/6zk5x1xdz8knk+y/
yOM7ul47egY8xvhEkucussnPJzkxZv4lyWur6uBOznCZc7z7UYY5weY3x6uv2VJI8nWdu02dLXbM65lm5agxe
mu9dNX5tfRe5Yr3nmalFVh5LcluSBC2zS8pycY67dakfXa9nXgNeSfOG8+09lFzJ2+afoT8aFX90LJ3Xl
WHk/xoZmdP52tds4vMlTSs2fRj66NJnknyj2OMXbFec8yV9BxjH0ryniTfvsDjXcfXh3LxuZKe9RpJ/
qGqTlXVPVs8vqPrtewA1xbf2w1nCC/
O7L3ab0hyLMnfLnPnVfXqJH+d5DfHGP+z+eEt/rnqzFX1nKml1irpY1G2N8a4zxxiSHktxcVT+8aZOW9ZpjrqWvvV1
X9bJJnxhinLrbZFt9b6HrNOVfXc/LWMcaNSd6Deq6i2bHt/
R9Vp2gJ9KcsN59w8l+eKSZ3iZMcbz8kuS6qtq/
jH1X1XWZRe6hMcbfbLFJy5pdaq7ONZv2z8zKaHWo+xC83VtF63Jvm5qnoyyyV8m+Ymq+vNN23Ss1yX
n6jq+xhhfnP58JslHkty8aZMdXa9lB/jvktwFOL3mGl6mq762qmm7fnNm6/M8E3L3/M8S9ltJ/
jTJ42OM37vAZktfs3nm6lizqlqtqtdOt78ryU8l+dymzTrWa5ulLFps471uuRcHes1xnjvGOPQGONwknck+acxxi9v2mzp
6zXPXE3H16uq6jXn3s7sDyZvjnlXV3Vnbid5a5LNvzm1o+u/197Kn3UJV/
UVmr17ur6qnknwgsxckMsb44yPZPYq4kaSF5P82k7u/
wrm+sUkv15V30xyNsk7xvRS64Ldmss3C1VXcXtyewJ+VdjjIer6p3n
zdWxXvPM1XWMvcwuWK955upYrwNJPjJ1f2+SD48x/n4Z6+X3HSD48xPrbI9fJWZIAm3gkH0ESAAZoIMEATAQZoIsAATQSYlVK
zT2r7z6r6nun+66b7r5/
ePvpozT6R7J3ds8Kl+DU0Vk5VvSfJ+hjjnqr6k8w+wep3Mzuevza9hfqxJG8+984m2I129/
n+RUVb0ryY8lOTrG+Pp5j39X/
HTHChBgVs4Y4xtV9e4kH0vy1nPxraobkvkx9kvUk73b2y27nLIFV9bYkp5P836eOjTG+MMb4kcwC/
KtVdaBrOJiHALNyquqHM/sN7gJ+S1X9YHP/+R4Ddr0wdiT2e+n03y48ufDuYnwKyU6ROy/iizzyj+ryS/
neR3qurQ9ElkqarXZfaBQp/rmxQuzW9Bs FKm/
6XgJ8cYvzTd35PkU5l9TOAvZPbh2JXkD8cY97cNCnMQYIAmLkEANBFggCYCDNBEgAGaCLAAE2+
A9hGlLNSDEtbAAAAAElFTkSuQmCC\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "#x3 boxplot showing score between 3.0 and 4.0\n",
    "sns.boxplot(x=df['X3'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 15,
   "id": "e936a25c",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<AxesSubplot:xlabel='X4'>"
      ]
     },
     "execution_count": 15,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAWAAAAEGCAYAAABbzE8LAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAKmUlEQVR4nO3dW6il91nH8d+TmYROk8oYJmiY1O6LfVFUsM
YhpERr8VBsI3rjRUEremHR6jBFUFSGirgveqM2EyDSVvsIR2om
f692G9kurNnZq2ZvdYza+bzgc2swzv7ffiz3i/
vftc+1BgjAKzeTd0DANyoBBigiQADNBFggCYCDNDk4CIbHzlyZGxsbCxpFIDr0+nTpz89xz89xrhj9+MLBXhjYy
OnTp3av6kAbgBV9cxej7EdBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAAT
QQYoIkAAzQRYIAmAgzQRIABmggwQJOF/iYcXIu2t7czm826x1HDh/
PQQw91j3LDE2DW3mzE3mw2y5NPPtl/cOAL/Ktvz7nXv617jLVw2xN/
1D0CE9eAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNB
EgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmg
gwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCPA1aHt7O9vb291jAFnu8XhwKZ+VqzKbKbzXhtrb
QM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmiwABNBBigiQADNBFg
gCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgz
QRIABmggwQBMBBmiykgDPZrPcf//9mq9gdwFpYSC3trby8owSM6jWXQm0gEPZrD3lGhvWS8XgCMvM8Vj6S8Kr
wgB5zLCj8q+5gLBU6hq8k78yk7e7+j9zvcbPUjgVWbp69Ouwx0KvN9mJvW8I3fdxCBsU5j7enpv66o/0kpE6S/
LZK7FmCsDq+8idNo0qk0hKZgnGzD02DF8Jwu2KRajVYhHw8e7M1WXgdwQ0HbhCa1/PDLijz0pnbnUZfS0lr/
OZdzAdB5Ue3gvE1GZ2HfDSLKzkmN/pf/gzi+uR7KFujdpmA8UDpAdFxf9yFxm0UB5/YlGLHpHpH/K5bxTRcr/
LLpM5kL66IGLBp7QfWFDE4PdQv7OKmMkbm84S3XX8QlWNm0pu8XjH2ZE7nxb/0swfDHFtGBr0lbGKr3WQw4/
L5yZ3A4kYeDfk+XOVbDI0I2iDMdJ04bfsd7YkbugmZX6nXLkKrJvvOEcJh9E5iJB8tHiaz8WsNd96jN4/fGBEt
nxT/3m/dwWQ/xTmzyqPAEJgsdU8CgbSo47XwBZ8gXWu+4Xcfu8mIo3R1jGz/HqEQ+0Gsm2r7hGFOHHnhKzfwvXK
q0+MhxgmySJjDGYmbtLbTtxVn6VNQ/MQjKGYdXXYFHr+z/v9ev79ifdI1zpvB9z4H/5H8/81Byrq9Kq/
tdkw+eNTDTbSqmuNk3qAnSqg2dNbqugX6n5Duex3fr0OZNdM+tPvs8ck4nf2/nQtivIu7cbXSqaGDnXUzSTy/
ueVe/2zIrj9nbx+cwFenmN9fJLqWH0YiMjPbUa6/4b1WyBmlbS+kmcGtS4aVbXx1gm5FHV7wH2v/
NeOg/t0O9NQ7Y9DLRKB5SB5dqnjeZwz01NRs8BHnoZhULiI4h5i0vyGznfoMrzC/rR0Iw81fV7xGEqgcdb2dF7S
lEeasyZsr0lN0xIc2qEOU6CbbSuafePUX4htdoNfozPQlM4Up/ELnV+RXSlMnEORAP
PG9W0mNcuc8gu66q5r1vbUXAvKz6LtJ8rN/Pmta66lSYcfpaHc52SjPD61C3ddVdZiaqg+5DiDuCZfGRfz9pX/
O1esDl9+ADQ/YFDH57Bz1Tt/+k0djrt0MTKbWKWUmqi6jt5wapyXuOuhmPhTmkSo1EXXGCJ0TW+9d8+Nt5gd
HVh+nJX5R3V6z/cNEd9Rz1GjaQrwEPTJ0P5OQ8PeP7rIEwnrKRJCmgEWcwlbXN2Ozd5PT5N3KdLHWqQBjsyWoq
zpHcmTAzq/ZGnWvP/YPXfbg4GwLp4hZvlttyrFQ7ycQnvBGvdvD5fPvqVFcb/3Y0cHcLtAfW1D13PrqrqRuC0LW
uywoUYi1JDvRelULxrYY/J61Uw9W4bJOMBXUwqsF+/0fhe+ypLm+uMmKAsaKs1pN1ts3/Lr7nZp/LUeJVMv7zdbaTM/
TcBw1o1G8ZIYYxH6q5NQMlPXJtdcpSRxqTZQMbyPqLKaMJ2LM2ADmZoidKFppbXNTdJNNzQdjuTKiXE7dpYoXqx
q7reCRjAvqN/mz+3ECt3hiRwA8TqU1Y2nnZNKoXEWPfzSW05XbqHCZbOcTiVhmcnaTttbXiDGqEqdlQ9Cmo+6m8
Wg8+5/hrlC+yZGOmqzjw3X3xMMbjdGb3ulXbU0L8a0qMdYYhrJnDpT/4tXc9tzSrjaPqSaBZXfe8LzsU+cl7Ml
PtQLCDZYBEaYAxJnQg9RFbZFRfmpk+rjXNxcMMY1YIHwMR2XDdWIB3xMUsnTJD+zjR7TWjFNN4V1kv/e1FGuv
2tMgsnJm2RmxfuSAe4f5dKHO4pfaj+JZyexyfajnGXvHVZkldV8vpRxawrIW5Wi7sN4zFqkeyxpDgCXnh2FXZz
X3HO2/Xa8vAu9/n1OAh6w0z6vF8p+bF7KY81jP4bXtIJ5TW/pXF0qdGjHpj1xMrYeaeZhA3zfKeSbQMXUcGgkhd
7uGB8K5LVhUaXBcfnUgL7BDVVdfPzaZ2TAp4HJvvDgkWEhjYvI5hnvbUJHDahQmbSsVqFRP7IsJ0J9smpF00
1IbyxSJx8DOZbG03C0PfJZbg1WfdgDx5IUaWfXXBgSVjXOhMzVqE0KQ8POP0yRfdnvrUcHZ4n/DLq00bOdTL1
e/+x4Vbgya4vVvNMdhMEYOt3T/ZYR5+ZmStfOBLPLnW+y47vb/bv6GNx/bTgzn2/YL4YzMO9rmSAIh50UcRjTZw
/sTGV3fcdJF9iVr6gA7Z1CaBlobANbdAQoeQ0bHJcN4DFJd5vO5H+2ybE0ZLnmJ+jq6ttTiT73JN9fRXJo54
mRTzyT8wazjv2wEy6FhVO5rLhnFnOt7tmzX9Bb05vY/HH4WU3eP8kHu+4ZbRunPTbcUVcaY6PYW3wi5twB+YvP
OgVRzhiVCQwjxVr7pUTBwiJsNLnhY0Tmy31VqYr3/xTyXyC/Ip4KKk6/vYD6nvEs8/tbqWcbmdXZJmhSo2OKf
rn+mcg8GjyTYr8hNXadbKv+Emr/EqZF5RDjGLDzTKTq/UK/nV4IoMkIzdJOecOfDe8Q4dpUTFL9tbM/vTmGOCD
lZmuIhJjy1e3/y8gMK1GONcxFW4aS87sT3zLlvrjzQtq+Y6JhXy1n9C/eTYzRs/ypv4qYkaMQZ7Ynf9DtXi
JlD9yVzG6dW0Av/9J2fHLDiNnXmGK6sROLujQlqlBfP4vE33cEt6a4s70sZTQuz5HF9aL/ZrlTdX/jP28hw
FZjJ5AQjIv6mF/++ZTIvsfR5nKqtxb2b5t1SbqjMvhYgSOcU3ZbOWS3wnaAh6MnAwKhrcXlyQ5ZJ99yjyMfDl9
I7ZYxxYHgRgrB/nyS8dCp/7mNLxnEE3pg6bgMXbMzxvSO+jP/bz0O0OEAlLNlB3O6xoZOX8bhTdCoRS/LbKfHT
LkXSGnKWO0BLYb1rxs9pc1jrbUBf3Q/TeqkeahB/P8X8xvzsl8DnLD/1CP3ZL7PoWZeUTqO1wPO75/4M3S4X
R6J5Dzd7ZCX4YeTGoHNuzQZY/Cnz97UeS0vm7rgaY+9OTyGO4fAl5nIQDnaK6JZ3PwOE8yJYRi07LDxuzHm/Y
gvR5XjYekqgnCsnuZWeyx8tFVvH0Xxs9H9wF8kaxV3i8N/zM77nUWkCxehsPSe8Z0pWOO6NaNRNrhv2oYB
/yRCpSiTgmvDV7lvvdNbL/oVaRY2/bfrfLRxJ/5FX+2AHMgKyzRMj8N8Gn6yQd+cxXCjyzGAc9BKWoa+xz
Lkt+RDufmfXt9tkabKcP+bDWQEN7eQnJZyXU5BlfZsPwAp8BafFJr8uoGhKoJtxlaPUoQgQtLHUnZ+wHxKjw
Wpqo4s52d8t5QvvKZ8dv9xpFBEYKl+pQWZVIjuTZ9jfRm33EDt0fVaDa6XqsCNMmRYnyKXGtHj0Q5/
NQyDPVJ67l/H5OhnsbO1cRCtGq4HgJORWyVJbUoXI8o+gs+Y/XtqX9eNRmLNvLp9Y86AFBZ7gAiLnG8x3M
cTeTSKdU3/XdpoLe7Mw48CTcPTLUCmuxSaYyDQ1NGQL/MWfO0qPBjSfRRyNnXNtv1pwSC7F1sWpvVcx4Tmux
k+S1m3Oeh7wIwRVsT2nwpiZ+9O7tG0zBqdm4Ig2HqzNkbvbUM+b8BvzKUpNM3bu5mh4Q3MQ+cpIoWjCMJ1oa5
ShjCYQmZTAUs+IcCBiiqiwiK9yZ8Me4iJmgb/1ekvh2V9HVgXm5HAtwvV+5OUlq4PDmLCx8QY9t2+8tEKWoz
/iGIeu2ow/tQYChbX+n1ghiiJM+Fv8WGbeAnIrZuvBJ6Hh+ihSfzUi2uf3SBGtcCdtBoh2OYa6NLHmhsVNZ+
ka8aShJHAa3BEpDzc2LD65nPcUgbOMG9bRs2JqJkpz2PMV+a2XaP0Eirnxw8kJQqQt9D3NqEGsv8fhOnufWx
VRtkIRTe0p57p3fhDa9jK4Vjch0Fp2UGYpNNdvKfMuzqd5Kq1Ve6kc0/PGjR4v+hbYOmvs1q1ZX4aN3b8Z4cNp
5gSO2nrltVKwjVymg3i3i3zF5t7Wq0imKL9ZPYqYtGtBODw7WfCWpgIYN7jLD1UAOUtPk/4vP1GL7/9e9B4BAdgY
U1LJ4pYlcZH6yZyAb9VxDAffb15+Oqa6u1HQLRXxWBcL9+WFIbLovomdNkZf2wdpVeJGlZpXOC5/Rlc9N0a3Yy
... 
```

```
iHJe8YYj1fV8ez8+Z8fvmCbH0tybIzxM01jwlxcgmDd/ESS/xpjPD7d/
+0kr6+q72ycCa6IM2CAJs6AAZoIMEATAQZoIsAATQQYoIkAAzQRYIAm/
wfhfZynSdcKpwAAAABJRU5ErkJggg==\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "#x4 boxplot showing scores between 3.0 and 4.0\n",
    "sns.boxplot(x=df['X4'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 16,
   "id": "a04b7d91",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<AxesSubplot:xlabel='X5'>"
      ]
     },
     "execution_count": 16,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAWAAAAEGCAYAAABbzE8LAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAKgklEQVR4nO3df6jd913H8de7SWczO4glZSvptiD5Y+jQtZ
TSURhBdLgp+o9/DOoKK6RMJEQFB/
qH4l8bKLJ6wR9NDRjtFEE3RumGg038y42kdrauQ66zm02L7TbcDxbomn3843zD0tub3HOTe877nuTxgEvOj
2/yffPhnCff+z0/
UmOMALB8N3QPAHC9EmCAJgIM0ESAAZoIMECTvdvZ+MCBA+MCBA+PQoUMLGgXg2nTmzJmvjzFu3Xj7tgJ86NChnD5
9euemArgOVNVN7vdKQiAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwA
BNBBigiQADNBFggCYCDNBEgAGabOv/hIPdaG1tLevr691jrIyzZ89m//79OfHiJewesNvw/+t5Lv9TLo7Zjv
x19whMnAMGaCLAAE0EGKCJAAM0EWCAJgIM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQ
BMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZo
IsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwLvQ2tpa1tbWuscAstjn496e4RgG4KCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkA
AzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWC
AJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzRZSoBPnDiRI0eO5OTJk8vcHQAYBKWEqAH3300STJqVOnlrE7Jw
Ww8ACfOHHiVdcdBQPM7F30Di4c/V5w6tSpPPDAA4ve7UwOe/
Zszp07l+PHjy9lf0tbtrW1tY+uH33/79FBSmpDq+bRidw4ozYUD6RUyTnFRC/p2sbMV/yDV6zAYH4tjDIdo3m3VZs
h+/HjOfOV/u8dYHTfsyb6bXufxNadF/qaw8HPA991336u33///
YveJcBKWHiAjx49mj5l7ZCzq8bLZvA5g9LdBUOUr49+qqrrzqv8CzCzlbWgXjoId/
QL80MLfBZHMjoI3HgkDXO98FBmgiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESA
```

```
AZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDB
AEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMECTvd0D8FqHDx/
uHgGYLPL5KMC70LFjx7pHACaLfD46BQHQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEw
EGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiw
ABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJnu7B4CdsOd738y+Lz/
ePcZqOP9Kktd1T0EEmGvA4cOHu0dYKWfPvpL9+/
d3j0EEmGvAsWPHukeAK+IcMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADN
BFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKBJjTHm37jqpSRfvcJ9HUjy9Sv8u4tkru0x1/
aYa3uu1bneOsa4deON2wrw1aiq02OMu5ays20w1/
aYa3vMtT3X21xOQQA0EWCAJssM8MNL3Nd2mGt7zLU95tqe62qupZ0DBuDVnIIAaCLAAE12NMBVdbKqXqyqp
y9xf1XVn1TVelX9e1XduZP7v4q5jlTVt6rqyenn95Y015ur6nNV9UxV/
UdVHd9km6Wv2ZxzLX3NquqmqvpCVX1xmusPNtmmY73mmavlMTbte09V/
VtVPbbJfS3PyTnm6npOPltVT037PL3J/
Tu7XmOMHftJ8q4kdyZ5+hL3vzfJp5JUknuSfH4n938Vcx1J8tgyZtmw39uS3DldfkOS/
0zyE91rNudcS1+zaQ1uni7fmOTzSe7ZBes1z1wtj7Fp37+V5GOb7b/rOTnHXF3PyWeTHLjM/
Tu6Xjt6BDzG+Jck37zMJr+c5NSY+dck+6vqtp2c4QrnajHGeGGM8cR0+TtJnklycMNmS1+zOedaumkNvjtd
vXH62fgqcsd6zTNXi6q6PckvJHnkEpu0PCfnmGu32tH1WvY54INJ/uei689lFzyxJ+
+cfoX8VFX95LJ3XlWHktyR2dHTxVrX7DJzJQ1rNv3a+mSSF5N8ZoyxK9ZrjrmSnsfYR5N8KMkPLnF/
1+Pro7n8XEnPeo0k/
1RVZ6rqwU3u39H1WnaAa5PbdsORwhOZfVb7p5OsJfnEMndeVTcn+YckvzHG+PbGuzf5K0tZsy3malmzMcb5
McY7ktye5O6qevuGTVrWa465dvz5WLSV4cY5y53Gab3LbQ9Zpzrq7n5L1jjDuTvCfJr1fVuzbcv6Prtew
AP5fkzRddvz3J80ue4TXGGN++8CvkGOPxJDdW1YFl7Luqbswqbswsco+OMf5xk01a1myruTrXbNrn/yX55yQ/
v+Gu1sfYpeZqWq97k/xSVT2b5O+S/ExV/
c2GbTrWa8u5uh5fY4znpz9fTPLxJHdv2GRH12vZAf5kkvunVxLvSfKtMcYLS57hNarqTVVV0+Xqr+W7M1uXbyxh
v5XkL5M8M8b440tstvQ1m2euLurav90eV+Sn03y5Q2bdazXlnN1rNcY43fGGLePMQ4leV+Sz44xfnX
DZktfr3nmanp8/WhVveHC5STvTrLxnVM7ul57r3jaTVTV32b26uBqnouye9n9oJExhh/nuTxzF5FXE/
yvSQf2Mn9X8Vcv5Lk16rqlSTfS/K+sUw1eCZ1GKGGB3yWY/g4acFjFhu81crTeF8e5XKD+N7xxW2tHZd8ur1o/
1XtJs8k2kcBUVq+bQ7Nnr0YK2ub9OGxm2R4Sd1QqqO13Ne2wb1XSXvvk7e+oTWqZd+5WhBfV/UV0+ek7/
Ivx9jPFZVH7xoro71mmeursfYa+yC9Zpnro71emOSj0/
d35vkY2OMTy9yvXwUGaCJT8IBNBFggCYCDNBEgAGaCDBAEwFmpdTsm9r+u6puma7/7/2HT9rVV1vn747Vmf7J
4VtuJtaKycqvpQksNjjAer6i+SPDvG+HBVfXeMcXP3fDAvAWblTB+TPJ3kQ0k+TPPpkZJKjSe4YY7wswKyaHf0kHCzDG
OP7VfXbST6d5N1jjJenu26q2Zdov5LkI2OMT3TNCPQYFbVe5K8kOTtST4z3faWMcX3bWMcbzVfXjST5bVU+NMf6r
bULYghfhWDlV9Y4kP5fZ/
0jwmzV9IfZF32T1lcy+keyOphFhLgLMSpm+IevPMvuO4q8l+cMkfzS9G+JHpm0OZPaFQl/
qmxS2JsCsmqNJvjbGuHDa4U+TvC3JTyU5XVVfTPK5zM4BCzC7mndBADRxBAzQRIABmggwQBMBBmgiwABNBB
igiQADNPl/O5yjepH0kHsAAAAASUVORK5CYII=\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "#x5 boxplot showing scores between 3.0 and 4.0\n",
    "sns.boxplot(x=df['X5'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 17,
   "id": "043d7cb9",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<AxesSubplot:xlabel='X6'>"
      ]
     },
     "execution_count": 17,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAWAAAAEGCAYAAABbzE8LAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAALGElEQVR4nO3df6jd913H8de7uZUFN9lMSg2326JcRFRwll
oq1TFmO7JV9B9h+0NbhVoUSWtFh/qH4h/5gbf9Qa0k6BTaRkYS06/
MvNZJtmrkWKZtos0i4lqzV3djf5+Mc9hXp7k5yT3HPeubmPB4Tde8+35/
vmvXOePTmn59waYwSAxbuhewCAnUqAAZoIMEATAQZoIsAATZZmOXjv3r1j//79cxoF4Pp0/
Pjxr44xbtr485kCvH///
hw7dmzrpgLYAarqy5qy5v93FMQAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQ
RIABmggwQBMBBmgiwABNBBigiQADNJnpd8IB29/999+fae+fs2fPZnl5NZUX7c+us29/usVYNhhTp8+nVf
+51z+63/d/aex69xLc7tu/w/
ATrRrKavf9YHuKbaF3c8emdt1ew4YoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDN
BEgAGaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABm
ggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCDBAEwEGaCLAAE0EGKDJUvcAcLUee+yxJMnBgwebJ+F6
dMPXX86pU2tzuW4W8Ztt77rnnukfgOlYXvpHV1dW5XLenIACaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ES
AAZoIMEATAQZoIsAATQQYoIkAAzQRYIAmAgzQRIABmggwQBMBBmgiwABNBBigiQADNBFggCYCDNBEgAGaCD
BAEwEGaCLAAE0EGKCJAAM0EWCAJgIM0ESAAZoIMEATAQZoIsAATQQYoIkAAzRZSIDPnDmTBx98MGfOnFnE6
bY9+4KdYSEBPnToUd8v4nTbnn3BzjD3AD/33HMZjd/edjjd56Dd2yOd5ab3lO3+mcvVSKNNMiHZcXdEKk6S1W8Xz
zPP0QtOLTn9uHAHO6cMeZ37jWQEdJkpwZ3uZpIbXzrwAzbzCo+tUOMVwZOWyHvCTVcqZCmSnnZv/ThGxOm
ucdxbXbnTE0iv37hp8mq1OE3PnV01ahZwv1Z8nGOFyzbRu3XVzRQv+f5uexc/fXjn3uVrKE3FvPp7RqxpdP
yS/7O6KxKL96/RHkv3zdOWZpw1xtddfUuNoO4sTE+1FFz5h3Ucp9c9fXyxUGqscrPHZ6xuhoScajJE5DoyyfhyU
mnXmKQdfLZFbSk2ZF7v9k4/qcyv+BR7L1ifkeMbEfPt/lJJv9yntPQRIxi8k3PIIuIXB5/0tbN6PdtaMlFAyIUA3
PeNTJ6rmsqcGZEs9WOnsSPuaIvaIA2nw/FlhhfGUXLH3Cz8Wrdd20cqr7VMZ7ZmlGH8NGPgd96XH2rbWVm/Ba3Ay
uhDELgHNXwb4LKwD2zHaTyrZsjeQqJHYxWEC/HzOQtxcAWXvuNQ4CabJ2/mX85zJ4Cq7hk8DU/Ir8UNZbGWBtDO
jXtXq6rCI9bmV5e3bzy3ne/nkbx2+uu/zMTZ2r8n+I6X9bvLHXfpu/jcPkjRHO/ZsFNjOqvFHa3FWpVz19mIXTRA4
BgXFk4N4DCyxsTdc/mqrbrRkvSKC2HSZmqNaXZL/8ywf/fbe9u+GjXcTl2O3fbyh4HMecvhH44UqOrnkJmirvQq
l7Zw/g6uMSzhMWe3BTJTXXz2jvhOVYdCRNNZb4XgIoaGuTBsk4wMGFJ/W2mUbNkL2jT2AyRTNYu5uT67Pw6jsiz
SlbPWj+WczC64iKPXl/qRmkP4J9rfo9FNgXawXxh7XqETwzrRVfEBCPHt4v21Ddq8uhiCxh00XCnAb1KRTKmxS
cz4Txcj5AZDcgzUcMXWoP2XO3QuoGSCawa66Nt5EaXN/T0fhAZkNcXcWlDI/Hyxe0mjyaYKmXKXqY1gCFLOfW2W1W
PVZ+NPG8vazbZMnxq60qOsm7ZXWzP1fULtVBWVqYT2WvYtR7QYDQqz0u/XNhuW3xtRa0x1a0DZfmOg2/EbAKbMg
bZ6HLhPG2A4tHvd+Zw9u/+5NyKswgNHLqDYxTEE3wdp94/Ko5C/UgWHtEtXHf6vtqaxpmpLH53wr8Dd7Nbdnqz3H
zPPn4LPd8PqBEkSjzH/kssL/vetHuv0kNLAwNaWVcKzSWmOnlpNDD5xvx76sbfyqiJ/1BhpSXtHwBjNYH4RsfszVQ
2Sf2+Ja1jgMoNgegIP8oXaLtJ05N/pQrt5aZtWfNX3OmlPJu4pONT0R0Vt4ruObJ41ikQaQFdh1UMDvGOQbG+JU7N
yv8J0b3hSn/wjaZ7FQbr/HgoYprwBI5cqgU/lmQBh0vHa9JezvBdfcwVdpnv+N1frl7p4tDm4cVbjGuoEf2HHh4sU9
XXn3c2QzBPxf+xfKxXYyvkr6sgd4ZKXu9JJHsyLSA5EAhZG4A9sy9iJKjMvt3SeXtRHfZxc2GqYPf+vqdkxy4E6z
Lg+xGx+7ioEPmLeEzbzuD0qxxKMG1cYe5Eo1VxfKrVIbnpeXBxqjRMAFD/9vQMQ4WhtYHg9Y91JxHdM4Vn2AETaMd
3QOhn9IffVXmhz4tNjEtBqkLWFn+7FBbs1a4q3OKXQwQzukKcaHS2jaz/YuoYg8dAnmBE3G/Db7v8I1O+D1O2tLJpR
xOEf7Ib1r3jz8ZZLsKccfG/eWKQ+9uxVcj5YDmxFCZPPq5I3QLevsfw4jsbZd7EX6xUHAwlZGeQe0x2zGhhN+QPN
vR5PM3+M8v/QzLWdLAYxVQP0kFlfHr6SnP6ZVHWmbb+vz3hgD3S1tR+EkxxFt9vN8AnaG3TpXz2+ruLPJn94NHS
Xj01tQYXXn9D0GdTqYz08T/1DqSwtvaF1h94Jw1U8rVhf5lnXDxGz9cQ97wqNmMrjmtFz8KnUmkVz49sdPc2+zxdZ
6X00HbQ7fN8a+ljl1FkJOPqR0OnXZq9zVmNNLVc1BqqBJPfZf7lCbq/vI7cEH6gW6wR1xUM/7u1Gf69tq2H5Vt3mKX
QfpWo8W4W7TN1rK5jGyO27+/a9odnM0p8dG2DEsV/qhjh7ZLTSxcqbxd2eX1Vm9Ks6gW1Zz0YbIzNs0c3O9LEl9y
bxOfAeJSYGxz73JBIKL6yfq32GDRfwVD7fvULxRL2pAFTVR4+RSPb8Bm0hP4IaLw8a+bUN+d+8Ctbxmk4rKzKY6
ezMY6cD16N7A/Q5mMb5RxRntyTKbeTjmNvN7hZzt4gwzuXFXX3d7eQk8qD5dtpPvMWvhUdi9ZsRMkNlbLUttrFJ
JmgbLiTBRAnl6fQBXf/ZEDvLDUyYebF7KJxuHdUprC2LPdelZHp2Lj8LcGwq8NV6R2mrCMQW6uMQuQJgUzrOu
x5z7l9MT8hDEQgDTdJ2GLBjdkKZ0JaoTVcPJ+Ba4VV/uTtXbovN/fm3hTtY8dhmYnwKDdz6t+0Tz4tm3FXs8Pz2PC
0z96yBLuNj8HVk1Fr3Jwpq9aGlJUYsj+O1sL4r/KMlb2nnjjOk7xkPD5Iae2QsdmWjRM6EXd3P7xSL6BYVlyOmXS
TdYTgjqa8b3cPVFvdFXyv7T+u+CNH+vm51TOdlRhFlfTLkRdG9p1kfT16XE6DxfFmk7TDeVKE7I4KHjvxmhgbLBQ
0tB0y+r6OFFfNVhUsKs8wVjC+VXiV+CHjHwOSS1u/w0v4GGk0PmVK3LuWRnu7BCC8ltv9YgRNMJSTmGk2S/wLa
4fkDvMwxsHBdSq/hc9zEDUtU9qbBmBWb0YLpzGkyGcDH7VM4l6qVNNFOBjjnY+PXHgQHVqKvhvfXJEtS4lVwsVtOeN
vsDvpbMC9bdEF9FDWXMWwHjxFJ5+NTtA6qn+Pl0JVR5Wdkj8Wle2ZSH1LBBhRQnDb2Ol4TTHxtoM/+ZxKIPNmf12MeM
2k3V6JhrENFDnCbBgwx+bw3ZvzXlj7dKMltiojNZTeWbE3utH6dgvzs9Wwak/fYYw8dPtSNb2sm7y8MdK2jsxqcS2X
o+lhOH/5/c/yjfpKsD/ZBQHx3ZAd+X3g1P/3A4ZUyRBjhyMjFHXDL0n8JPsyBO99f23PMjM+TZ5OiVs3FFUa3a+fLr
BT7y6VDwM8SuXWgqsBgzHGJT0+zY33Pn+Vb/jzDR+mRXGnvwBNZ6dQqMxYPR/KMRBPmNsNXzpjuZbUtmXjPf8aC/K
xPcOmBS8+9t3LMp0DbYn3QyrQNafW1pW7hS86LhjsVkTm1eWNNDwk77Jp4jwPyX8/PuGkF3Gf8dTMtQH0a+/dHs+
Uk+lORvxxg/ueGwhe9rmrmabl/fXFVvee3rJO9LsvG/nNrSfS1d8bSbqKo/z/
```

```
qrl3ur6vkkv5n1FyQyxvijJEey/iric0nOJfmZrTz/
Vcz1E0l+vqrWkqwm+dCYvOQ5Z3cm+akkJybPHybJryd5x+tm69jZNHN17GxfkkNVtSvrd8i/HGM8WVU/
97q5OvY1zVxdt7E3uAb2Nc1cHfu6OcknJt1fSvKxMcbRee7LW5EBmngnHEATAQZoIsAATQQYoIkAAzQRYLa
VWv+ktn+vqm+dfP+2yffvrKp3VNWnav1T3L40eRs1XLP8Z2hsO1X14SQrY4wHquqPk5wcY/
xOVX06yW+PMZ6avI36whjjXOuwcAkCzLYzeZv08SQfTfKzWf+woJUkj48xfqhzNpjFlr4TDhZhjPGNqvqVJ
EeTvG+M8WpVfWeSs1X18STfnuTpJL86xjjfOStciueA2a7en+R0ktc+dWMCUB
ZtupqncluTvrv5Hg4Vr/QOznk3x+jPFvY4y1rH984cJ+vQ5cCQFmW5l8QtYfZv0ziv8jye8m+b0k/
5jkbVV10+TQ9yb5Us+UMB0vwrGtTH5LwY+MMT44+X5Xks9m/feLfVOS38/6h2Yft/
LAGOPVrlnhcgQYoImnIACaCDBAEwEGaCLAAE0EGKCJAAM0EWCAJv8H+Hz8L6me2toAAAAASUVORK5CYII=\
n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "#x6 boxplot showing scores between 4.0 and 5.0. \n",
    "sns.boxplot(x=df['X6'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 18,
   "id": "a0c28781",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<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>Y</th>\n",
       "      <th>X1</th>\n",
       "      <th>X2</th>\n",
       "      <th>X3</th>\n",
       "      <th>X4</th>\n",
       "      <th>X5</th>\n",
       "      <th>X6</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>Y</th>\n",
"      <td>1.000000</td>\n",
"      <td>0.280160</td>\n",
"      <td>-0.024274</td>\n",
"      <td>0.150838</td>\n",
"      <td>0.064415</td>\n",
"      <td>0.224522</td>\n",
"      <td>0.167669</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>X1</th>\n",
"      <td>0.280160</td>\n",
"      <td>1.000000</td>\n",
"      <td>0.059797</td>\n",
"      <td>0.283358</td>\n",
"      <td>0.087541</td>\n",
"      <td>0.432772</td>\n",
"      <td>0.411873</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>X2</th>\n",
"      <td>-0.024274</td>\n",
"      <td>0.059797</td>\n",
"      <td>1.000000</td>\n",
"      <td>0.184129</td>\n",
"      <td>0.114838</td>\n",
"      <td>0.039996</td>\n",
"      <td>-0.062205</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>X3</th>\n",
"      <td>0.150838</td>\n",
"      <td>0.283358</td>\n",
"      <td>0.184129</td>\n",
"      <td>1.000000</td>\n",
"      <td>0.302618</td>\n",
"      <td>0.358397</td>\n",
"      <td>0.203750</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>X4</th>\n",
"      <td>0.064415</td>\n",
"      <td>0.087541</td>\n",
"      <td>0.114838</td>\n",
"      <td>0.302618</td>\n",
"      <td>1.000000</td>\n",
"      <td>0.293115</td>\n",
"      <td>0.215888</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>X5</th>\n",
"      <td>0.224522</td>\n",
"      <td>0.432772</td>\n",
"      <td>0.039996</td>\n",
"      <td>0.358397</td>\n",
       "      <td>0.293115</td>\n",
       "      <td>1.000000</td>\n",
       "      <td>0.320195</td>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th>X6</th>\n",
       "      <td>0.167669</td>\n",
       "      <td>0.411873</td>\n",
       "      <td>-0.062205</td>\n",
       "      <td>0.203750</td>\n",
       "      <td>0.215888</td>\n",
       "      <td>0.320195</td>\n",
       "      <td>1.000000</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "</div>"
      ],
      "text/plain": [
       "           Y        X1        X2        X3        X4        X5        X6\
n",
       "Y   1.000000  0.280160 -0.024274  0.150838  0.064415  0.224522  0.167669\
n",
       "X1  0.280160  1.000000  0.059797  0.283358  0.087541  0.432772  0.411873\
n",
       "X2 -0.024274  0.059797  1.000000  0.184129  0.114838  0.039996 -0.062205\
n",
       "X3  0.150838  0.283358  0.184129  1.000000  0.302618  0.358397  0.203750\
n",
       "X4  0.064415  0.087541  0.114838  0.302618  1.000000  0.293115  0.215888\
n",
       "X5  0.224522  0.432772  0.039996  0.358397  0.293115  1.000000  0.320195\
n",
       "X6  0.167669  0.411873 -0.062205  0.203750  0.215888  0.320195  1.000000"
      ]
     },
     "execution_count": 18,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAh4AAAEzCAYAAAB+GQo/AAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIH
ZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/
MnkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAABowklEQVR4nO3dd3gUVffA8e/
dTSGN9J4AgdA7oUmvP0AQ9FXsoqBgV+z9xS5F7CKiICKig2sABSRToIoYaekEJ6QkLK7v39sXkDSwoEsr
ssns/
z7ENm58zsudlhcvbeO7NKa40QQgghhD0YHJ2AEEIIIf49pPAQQgghhN1I4SGEEEIIu5HCQwghhBB2I4WHEE
IIIexGCg8hhBBC2I0UHkIIIYSoQCk1SymVppTaWcV6pZTR6TymVoJT6RynV4K4WHEEIIISozGxhczfohQ
OOyxzjgo/
PZqRQeQgghhKhAa70KyKwmZAQwR1v8DfgppcLPtV8pPIQQQghxISKBY2csJl5Y0Vy0Xm6VTRvVs/6+
+J3viW7c6OgWHevXR5xydgkPdMLCJo1NwmF17jp076DLW9/
qRjk7BoTIT4h2dgkN1f3ytsufrXdDf2jXb7sYyRPI/
M7TWM2ryspU8ds7aWCGEEMLGDDUfwCgrMmhmpSaSaU8d848bF54+CCGEEMLZXcBQy7
lorW86x3oN3F/T/UqhIYQQQQji72hk6sQpPIQQQgjnLcKrWi7bwmPm0xMY1a1aFmUp
wtnJUIsQggh7EauaJQiPhxBCCCHsRgoPIYQQQtiNFB5CCCGEsBsAl5wZMr/ItL03UTjIPQ9IM7n3HnjIYEDJoWn6rl1iEbjuDE6s+IeTEXOP
w6bEZzZE1NkbnQVLt4cO1XttzpRsdy7+0CIYngl1gI/B1gfKzj0RWTgPIIhfxRPMI7Nm0kpTzpgwpsxXV+mnyUdVXQbw7vYCQRs3t1MlgL98Qi9AVBt47I0NP4BSSxL/zNGrgG+nA4ExYbCEh2KOakrSSrMbPV
```

```
bj/HBF39hNmuu7NOMm4e3s1q/dO1+vvp5OwB16rjyyB09aFQ/EIAFS/
5h8Yq9KAUxUQE8Na43bm7OcShc//
BUWnYdTHFRAXNeH8uxfdsqxASGN+DOF+fg5RPA0X1bmf3qGEylJXQaeCP/d8tjABQV5DN/
6kMkHdhRvp0yGHjmk7/ITj/
OtKf+Y68mXZCARt2JHfQUymAgeet3HF07y2q9Z2ADmo54BZ+w5hz6832Orfu8fF3Xh5ZQWlQA2oQ2m9j8ab
V3Rb4kRLfpT49Rb2AwGNn15xds/emdCjE9Rk2kfruBlBYXsmz6faQf/
gcAN8+69B37HgHRzUFrls94kNT9G8u3azf0Abrd8gqz7m7EqbxMezXpgm3Zm8mnPyZg1pqBncK5tm89q/
Urt6by3YpjANRxM3LPNY2JifDmRPYp3v16D9l5JSgF/
9clnKt6RDmiCRfFr0EXGvYbD8pA6o6fSNow12q9R0A9Ygc/h3dIE46smcHxTfMtz/
vXo8lVL5fH1fGN4OjaT0ne8o0907cd6fFwvNlLfuKD775mznOvODqVWmcym3n38zVMeXoowQFe3Pvf7+kWV
58Gkf7lMWHBPrz9/FX4eLmzfvtRps5axbSXruFE5km+/z2ezyaNxN3NhZfeW8ryvw8wuFdTB7bo/
LTsOoiQqFgm3NSSmBaduemx95h8d68Kcdfc8yrLv3mfTcsWcNNj79N92B2sWvQJGcmHefuBgRTkZ9Oyy/
9xy5MfWm3fb+QDpBzZSx0vH3s2q+aUgcZDnmX73HEU5aYSd9d80veuoCD9YHlISWEuCb9OJKhpv0p3sX3On
ZQUZtsp4YujlIFeo6fw0xvXkJ9xnOteXc7hLUvIStpbHlOv3UB8wxrx5aNxhMZ2pPeYqSz870DAUpAc3b6M
3969A4PRFRd3j/
LtvAMiiWrdh7wTx+zergthMms+XrSfl+5qQ6CvO098sIXOLQKJDvUqjwn1r8Nrd7fF29OVzXsymPbdPqY80
AGjQTF6WCMaRfpQWFTKY+9toV1jf6ttL3nKQMMBjxG/YDzFeWm0vfVTMg+soTDjcHlI6alcDi1/
m4BY63NDYdZRts+5o3w/
ne5ZRGbCSvvlbmtOVHhUmalSarFSqoEdc6lVq7dvITM3x9Fp2MSeAyeIDPUlIqQuri5G+nVtxF+bD1vFtGo
Sho+XOwAtYkM5kXmyfJ3JZKaouLT830B/5zjxtO1xFX//
+iUAh3ZtwNPbj7qBYRXimnbow5YV3wHw969zadtzOAAHd/
5NQX62Zfv4DfgHR5Zv4xccSasrhrD2589s3IqLVzeyFYVZRzmVnYQ2l5IW/
ytBTftaxZQUZJJ3PB5tLnVQlrUnJDaOnNSD5KYdwWwqIWHdd8TEXWkVExN3JXtXfwVAasIm3Dx98fQLxdXD
h4hm3di94gsAzKYSigtyy7frfttrrJv3IhptvwZdhP3HcgkP9CAs0ANXFwM92oawfleGVUyzBr54e7oC0LR
eXTJyigAIqOtOo0hLUe3h7kJUiGf5OmfhE9acU1mJFOUcR5tLObFnGQGNelrFlBRkk5+yp9pj369eR05lJ1
GUm2rrlO2nsq+9P9fDQarr8ZgN/
K6U+hyYrLUusU9K4lzSs04SEnC6WAgK8GL3gbQq4xev2EOXNtEABAd4cf2Vbbjx4Xm4u7nQsXUUnVo7R3er
X3AEWWmJ5ctZJ5LwC4ogNyOl/Dkv30AK8nMwm0wAZJfFnK3bsDuIX/97+fLIh6bw/
bRncfe8xHs7AHefUIpyTp8wi3JTqRvZ+ry31xra3PoxaM3xLQtI3rLQFmnWGi//
cPIzksqX8zOPExobVzEm83TMyczjePmHYzaXUpiXTr+7PySwfitOHNrGmjnPUFpUQIMOQziZlUzG0Z12a8v
FyswpJsjPvXw50Ned/
Udzq4xfujGFDk0DKjyfmnmKg0n5NKlX1yZ52oqbTzDFeafPdcX5afiEt6zxfoKa9efEnqW1mZrjXQ49Hlrr
b4D2QF1gk1LqcaXUo/97VLdTpdQ4pdQmpdQmUtJrOWWhK/lwpqh8YtHWXcdZsnIvY2/
sAkDeySLWbjnCvLdvYsH7t3KqqIQ/
1uy3Zbq1p7LJU2f9MiqbYKXPimnSvjfdht7B9x89B0CrbkPIyzrB0X1bay9XO6vJJ/
atn41i8yc38M+8+4jseCO+9eLOvZEDnc97WvnEOo3B4EJwg7bsXDqLBc/
2prSogA7Dx+Pi5kHc1Y+yYcEbNsraNip9l6uYU7jjQBZLN6YwakhDq+cLi0xMmhvPncMb4VnH2Ubbz30snH
MPBhcCGvUgY+/
y2krq0mBQNX84yLmOuhLgJOAO+ADm89mp1noGMANA9WzvHH2YTiQ4wIu0M4ZO0jNPEuTvWSHuwNEM3vx0JR
OfGIKvTx0ANu9MIjzYB7+6lnHunh1jiN+fysAeje2TfA31vuZuul81BoAjezbjH3K6d8Y/
OJLsjGSr+PzsdDy9fTEYjZhNJvyCI8k5IyayUStufeojPnhiOCdzLRMJG7XuRpvuQ2nVdTAubu54eNXljhc
+Y/Yro+3QwporykvF3Te0fNm9bijFeSfOe/vifEtsSUEm6XuXUzeyFTlHN9d6nrUlP/
M43oGnh8W8AyIoyEqpGBNwOsYrIIKTWSlorcnPPE7aAUv7Dqz/kQ7Dx1M3NAaf4PpcP3F1+T5HvraSb1/
oT2FO1b2Hjhbo60Z69unhkYycIgLquleIO5yczwff7uO/Y1pT18u1/
PlSk5lJX8TTu10IV7QKtkvOtak4Lw03n5DyZTfvEIrza/bh1j+mK/
lp+ygpyKrt9BzLiW4gVt0cj8HANsAT6KC1nqC1ful/
D3slKCpq1jCYpJQcktNyKSk1sfzvA1zRob5VTGp6PhPe+YNn7ulLdLhf+fOhgd7sSkjjVFEpWmu2xCdRL9K
PS9XK7z/m9TFdeH1MF7av/
pGug28BIKZFZwrzc6yGWf5n79aVdOhjuSql6+Bb2b76JwD8Q6IZ9+rXzH51DGnHEsrjf/j4BZ69Npbnr2/
KzBdHsXfLiku26ADIS4rHI6A+dfwiUQYXQloOJn3fivPa1uDqgdHNs/xn/
4ZXcDIt4RxbOVbagS34hjXCJ7geBqMrsVf8h0Obl1jFHN68hKY9bwQgNLYjxYW5FGSnUpiTRn5GEn7hsQBE
tepFZtJeMo/tYva9TZj7cFvmPtyW/
MzjLHiu9yVddAA0jqpLckYhqZmFlJSaWbM9jc7NA61iTmSdYuIX8TxyQzMig09/
INFa88G3+4gK8WREr2h7p14r8lL24OEfhbtvOMrgQnCz/
mQeWFOjfQQ1H0j6nj9slKEDGQw1fzhIdT0ezwEjtdbx9kqmNs2b8AZ92scR5OvHsYW/
MmHWdGb9ssjRadUKo9HAg7d356nJSzCZzQzp3ZSYqAB+XLYLgOH9W/DF95vJzT/
Fu7PXlm2jmP7Kf2geG0LvzjHc/fxCjEYDsfUDGda3uSObc952rvuVVl0H8/JXuyg+VcCcN8aVr7t/
8iLmTrqXnIxkFn30PHe+OIer7nqRY/u38dcvswEYOvpZvH0DuPHRdwEwm0qZOLa7I5pyUbQ2sX/
J67S55SOUMpK8bREFJw4QEWe5bPz45gW4eQUSN/
```

```
YrjO5eoM1EdbmVDdOuxtXTj1bXvwOAMhhJ3bmEzANrHdiac9NmE6tnP8lVTy9EGYzsWfElWUl7aNnfUhzGL
/uMI9t+p167gdzy9hZKiwpZ/vH95duv/vxJBtw/A6OLGzlph/
nzjHXOxmhUjB0Ry0szd2AyawZ0CqNemBe//n0cgMFdI/h62RHyCkqZvsgyhGo0KKY+FMfuw7ms2JJK/
TAvxr+zCYBbB8fQsVlgla93ydEmDi57m5bXvgUGI2k7fqYw4xBhba8GIGX7Ilw9A2h720yMbpZjPyLuerZ+
dgum4gIMLu741e/Egd8nO7YdtuBEczxUTcfHavwC//KhlsS3bnV0Cg716qPPOToFh7phYBNHp+Awu/
Y4xyWqttL3+svv/kE1kZnglJ9Za033x9fadRKFYfRNNf5ba/
5svkMmejjbzCIhhBBCnM2Bk0VrSgoPIYQQwtk50VCL82QqhBBCCKcnPR5CCCGEs3Oiy2ml8BBCCCGcnHKio
RYpPIQQQggnZ5DCQwghhBD2IoWHEEIIIexGCg8hhBBC2I0UHkIIIYSwGyk8hBBCCGE3BiV3LhVCCCGEnUiP
hxBCCCHsRgoPIYQQQtiNFB5CCCGEsBspPM6Q+Nattn6JS1rUo3MdnYJDvR9Vx9EpOFSToWMcnYLDHE2c5Og
UHKp+vwmOTsGhGvQudHQK/ypSeAghhBDCbqTwEEIIIYTdGKXwEEIIIYS9OFOPh/
NkKoQQQginJz0eQgghhJNzph4PKTyEEEIIJyeFhxBCCCHsxpkKD+fJVAghhBCVMhgMNX6cD6XUYKXUXqVUg
lLq6UrW+yqlflJKbVdKxSulRp9rn9LjIYQQQjg5W3w7rVLKCHwIDAQSgY1KqR+11rvOCLsf2f7KW1vkzKopFQzs
VUp9qbUurmq/UnHo9qbUurmq/
UngIIYQQTs5GQy2dgQSt9UEApdRXwAjgzU7lcJDCCGEcHI2KjwigWNnLCcCXc6K++QD
4ETgO+AA3aK3N1e1U5ngIIYQQTu5C5ngopcYppTad8Rh31m4rG7/
RZy0PArYBEUA74AOlVN3qcpUeDyGEEMLJXUiPh9Z6BjCjmpBEIPqM5SgsPRtnGg1M1FprIEEpdQhoBmyoaq
dOXXhs2H6MD774C7NZc2WfZtw8vJ3V+qVr9/
PVz9sBqFPHlUfu6EGj+oEALFjyD4tX7EUpiIkK4KlxvXFzc+pfh5WZT09gWLdepGVl0vr2kY5Op9bVa9ufn
qMmogxGdv05hy0/vlMhpuftk6jfbiClxYUs++g+Thy2HAtunr70G/cegVHN0WiWf/
wAKfs32rkFF2fD9qN8MGcNJrNmaN/m3Dy8g9X6P9bs46uftgLgUceV8WN6EVs/
CIAFi7fzy5+7UQoaRgfy1N19L/ljP6JlXzrd8DLKYCRhzTx2/
vpBhZhON7xCZOv+mIoLWTt7PJlHdwDQfMA4Gve4Ga012Um7WTv7EcylRfhHtaTrrZMwurpjNplYP+9pMg5v
s3PLam7tXxuYMvUDzGYzV4+4kjF33FxpXHz8HkaNeYCJr7/AwP69KSoq5s5xD1NcUoKp1MSA/
r259+477Jt8LVi7bhOT356B2WzmmuH/
x5hR11cat3PXPkbd9RiTXn2Kgf16ADDh1XdYtXYDAf5+LJw3zZ5p25yNhlo2Ao2VUjFAEnAjcPYBdxToD6x
WSoUCTYGD1e30gjJVSg28kO1qk8ls5t3P1zDxySF8Nnkky/9O4HBSllVMWLAPbz9/FZ+
+cR23Xd2eqbNWAXAi8yTf/x7P9FeuYdbEkZjNmuV/H3BEM2xm9pKfGPz4/
Y5OwyaUMtB79Jv8NOk65j3ehSbdrsM/sqlVTP12A/ELa8jcRzrw5ycP0/vOqeXret0+kaPbl/
Ll45356qkeZCbts3cTLorJbObdz1Yz8clhzJ5yI8v+SuBwYqZVTHhIXd554WpmTrqB266B266JY+qnKwE4kZnPd
7/t4OPXruOzyTdiMmuWr0twRDPOm1IGutz8Osveu4UfJ/
SmQaer8Q1vYhUT2aofdUMbsuj5bqz74gm63DIRAA+/MJr1u5NfXhvMTy/1RRmMxHQaAUDcdS+w/
ee3+PmVgWz/cTJx175g97bVlMlkYuLkd/ng3Yks/
OYzfv19OQcOHq407t33P6Fb147l69zcXPnkoze
f3fyP+PX3VRw4dLTSuHc//IwrulgX5MOHDmDa2y/bK127ssXltFrrUuAB4DdgN/
CN1jpeKXWPUuoqesrBXgG5KqR3AMuAprXV6tbleYBtnXuB2tWbPgRNEhNcjIrQuAT4BzS/
Nh61iWjUJw8fLHYAWsaGcyDzJ5vs5kMlNUXFr+b6C/lz3TRsIjQ2jpyUg+SmHcFsKmH/
uoU07HilVUxM3JXs+RXDx8imnVj159fAGA2lVBc4Fy/
pz0JaUSE+hIRWnbsXxHL2sqOfe//HfthpDvxsR8Y0568tMPkpx/
FbCrh8MYfiG47yComut1gDqxbAED6oS24edTFwzcEAIPBiNG1DspgxMXNg4KcVMtGWuNWxxsAV4+6FGan2K
9RF2hn/B6ioyOJiorA1dWVQQP7sWLlXxXivlr6e/
r37UWAv3/5c0opPD09ACgtLaW0tBSFKl+nX2Jv/
f7gT4+1o9H9e+FXXr+tgrXbsyGgw1fpwPrfVirXUTrXUjrfVrZc9e9N11pPL/
v5uNb6/7TWrbXWrbTWrbTWc8+1zCgg8r4xtKD3rJCEBp0+YQQFe7D6QVmX84hV76NLGMlQVHODF9
Ve24caH5+Hu5kLH1lF0ah1l85xF7fDyDycvI6l8OT8zGa/AiAoxJzMr
iOKGxcVYx3gHh5J8Zk3kc74BwzCYThbnp9L9nGkH1W5F2cBur5zxNaVGB3fK/
WOlZJwkJPH3sB4f7UGHAuhmN/N53b/u/
Y9+b6oe244cEvyo79aDq1ia5y20uBp18YJzNPv5cF2ckExbSvEFOQdXrouSArGU+/
cDKObCf+9+lcO3ETppJTHN+1kuRdlt6fjV/
lwHj5xN33X9RysCSc2Pt06CLkHYindDQkPLl0NAgdu7cYxVz4kQ6f6xcy8zpbxC/a4nVOpPJxE23j+dYYhI3XjecK1q1sKzzsz0UGTvKM3e0Pxo/
aa7XOZDJx8233cCwiRtGXk3rVs3t6WzBfvWbrpdb1AhRWsaAAPvrvVYq8Ydgt0iR3gf8+u0+ekkxPkDN/
lXL2ZF+N8C4lLQXWZ9gQ+BqZW8si3fWrV02fPqwVUpRNwYeuu4yxZuExN1qu2Aso7WcTaLUeY9/ZNLHj
Vk4VlfDHmv22TFfUpvP5lFZJjNYag9FIcExbdv4xk6+f6UVpUQxZGt8EotX7GbcTVc
AkJdfxF+bDzH/3Vv59sNRZcf+pX1yrvRT+dm/
hCrebzdPX6LbDeK7Z7uw4Ml2uLh7EtPlWgCa9B7Fxm8msODJ9mxc8BLdRj4LF/Pl9aYR+XnPN486e89SEPPzgOo9F
YIdRoNPL1vE/47Zdv2Bm/h4SEQ7bK1CZ0Je0/+52f8s4MHr5/
dKXtv5zZqsfDFqqbUfY3UKC1Xnn2CqXU3kriz1w/DhgHMPGZ67j1mq4XlWRlggO8SP2SyRB/
p4V4g4czeDNT1cy8Ykh+PrUAWDzziTCg+viV9cDgN6dGhK//zhXdm9ik5xF7fDyDycvI6l8OT/
jOKGxcVYx3gHh5J8Zk3kc74BwzCYThbnp9L9nGkH1W5F2cBur5zxNaVGB3fK/
WOlZJwkJPH3sBwd4sTuhumN/N53b/u/
Y9+b6oe244cEvyo79aDq1ia5y20uBp18YJzNPv5cF2ckExbSvEFOQdXrouSArGU+/
cDKObCf+9+lcO3ETppJTHN+1kuRdlt6fjV//
lwHj5xN33X9RysCSScPt06CLkHYindDQkPLl0NAgdu7cbR2TdoLlK9Yw46OpxO/
aa7XOZDJx8233cCwxiRtGXk3rVs3tkndtSTuRQVhIUPlyaEgQO3fusfjEgQO+Kt25ials6fK9cx44PXid/
lXL2ZF+N8C4lLQXWZ9gQ+BqZW8si3fWrV02fPqwVUpRNwYeuu4yxZuZexN1quAso7WcTaLUeY9/ZNLHj/
Vk4VlfDHmv22TFfUpvP5lFZJjNYag9FIcExbdv4xk6+f6UVpUQFxwx+xQZK2U+mxX8WvZGt8EotX7GbcTVc
AkJdfxF+bDzH/3Vv59sNRZcf+pX1yrvRT+dm/
hCrebzdPX6LbDeK7Z7uw4Ml2uLh7EtPlWgCa9B7Fxm8msPDpjmz8ZgLdbp9aYR+XnPN486e89SEPPzgOo9F
YIdRoNPL1vE/47Zdv2Bm/h4SEQ7bK1CZ0Je0/+52f8s4MHr5/
dKXtv5zZqsfDFqqbUfY3UKC1Xnn2CqXU3kriz1w/DhgHMPGZ67j1mq4XlWRlggO8SDuj+zg98yRB/
p4V4g4czeDNT1cy8Ykh+PrUAWDzziTCg33wq2vpduzZMYb4/
akM7NG41vMUte9k5nF8AiPLl70DIziZlWwVk59xHO8zJmVAlqTn3mc1AObAUhY/wNxI8bbJe/
aEhzgRVrG6WP/
```

RObJSodLDhzN4M1PVjDxqaFnHPuJhIXUPX3sd2rIzn0pDOzRpML2l4qTWcl4BZx+Lz39winITrWKKchKxtM
/4nSMfziFOSmEN+9JfvpRivIzADi6ZTEhjTpyaP1CGnW7no1fW+Z1HNn8E1eMuvQLj5CQYFJTT/
dupaamExwUZBWza/c+nn7uFQCys3NY89d6XIxG+vbpUR7j4+NNx7i2/
LVuA7GxMfZJvhaEhgSRknZ6+kBqWjrBwdYd8Lt2J/DU85MAyM7JZc26TRiNRvr1vsKuudrb5dLjMU5r/
WcV656rbqda6xla645a6462KDoAmjUMJiklh+S0XEpKTSz/+wBXdKhvFZOans+Ed/7gmXv6Eh3uV/
58aKA3uxLSOFVUitaaLfFJ1Iv0QziH1ANb8A1rhE9wfQxGVxpfcS2HNi+xijm0ZQnNet4IQGhsR4oLcinIT
qUgJ438jET8wmMBiG7Vm8zEauvoS06zRiEkpWSfPvbXJdAtroFVTGp6Hv99+1eeua+/
1bEfEuTNrv2pnCoqKTv2E6kf6c+lLOPwNnxCYvAOjMZgdKVBpxEc2/6bVcyx7b/
R6ArL1VtBMR0oKcyjMCeNk5lJBDeMw+hmKbTCm/UgJ9nSu1mQnUpoE8sfo7BmPchLu/Q//
bds0YyjR5NISkqmpKSE3/5YTp9e1n9Qf/lhHot/nM/iH+czoF9vnnnqYfr26UFmVjZ5eZbO6lOnili/
YQsNGtRzRDMuWMvmTTh6LImk4yll7V9F757W97Na/
P0sliz6jCWLPmNA3+48+8R9l33RAWA0Gmr8cJTqejxWKqWmA2+VzWyl7FKZqVgul+lkh/
yqZDQaePD27jw1eQkms5khvZsSExXAj8sss7SH92/BF99vJjf/FO/OXlu2jWL6K/
+heWwIvTvHcPfzCzEaDcTWD2RYX+ca6zyXeRPeoE/7OIJ8/
Ti28FcmzJrOrF8WOTqtWqHNJlbNfoIRzyy0XE67Yi6ZiXtoOcDy3UTxSz/
jyNbfqd9uILe9s5XSogKWfXz6Cp9Vs5/i/
x74BIOLG7mph1n28X2OasoFMRoNPHRHT56c+DNms2ZIn2aWY39pPADDB7RkznebyM07xTufWa7kMhoMfPza
dbSIDaV3l4aMe/ZbjEZF4wbBDOvXwpHNOSdtNrFh/rMMGD/fcjnt2q/ISd5Hk16jANi3ag5JO5YR2ao/
17y2jtLiQv6abRk+Sz+0lSObf2bY879jNpWSeWwn+1Zb5r79/
cXjdLrhFZTBiKm0iHVfPOGwNp4vFxcjTz35IPc99BRmk4kRw4fQqFEMCxZapuSNvLbqeSrp6Rn898VJmM1m
zGYzAwf0oVdP5/qD7OJi5OnH7+Xeh1/AbDYzYthAYhvWZ8F3iwEY+Z8rq93+6RcmsWnLDrKzc/m/
q0Zx79hbuGb4oGq3EbVPVPVTZmBqCU8gcmAt2Ah4HWwKPAZOCjc90S9X+SNk6t/AX+JaIePecE38va+1GX/
qdIW/rP4y86OgWHWfrxJEen4FDXvbnJ0Sk4lDIVOjoFh/Lwj7XrJUODv/y0xn9rf73lLodc1lRlj4fWOgu
4Wyn1MLAUy93KumqtE+2VnBBCCCHOzZnmeFR3Oa0fMAnLF8IMBq4EliilHtZaL7dPekIIIYQ4l8ui8AC2AN
OA+8vmePyulGoHTFNKHdFa32SPBIUQQghRPRvdMt0mqis8ep09rKK13obl1qhjbZqVEEIIIc7bZdHjUd1cD
q31J7ZJRwghhBA1dVkUHkIIIYRwDlJ4CCGEEMJujEbn+cI/
KTyEEEIIJyc9HkIIIYSwGyk8hBBCCGE3UngIIYQQwm6k8BBCCCGE3UjhIYQQQgi7kcJDCCGEEHYjhYcQQgg
h7MaZCg/nyVQIIYQQTs/mPR6vPvcrV/
ikvZ+VB1Hp+BQDybGODoFh+q8fomjU3AYby8PR6fgUB+MauLoFBxq45ESR6fgUAu2Ftv19YxG5+lHkKEWIY
QQwsk501CLFB5CCCGEk5PCQwghhBB2I4WHEEIIIezGIIWHEEIIIezFqJSjUzhvUngIIYQQTs6opMdDCCGEE
HYiPR5CCCGEsBspPIQQQghhNwYZahFCCCGEvUiPhxBCCCHsRu7jjIYQQQgi7kR4PIYQQQtiNQQoP27n+4am0
7DqY4qIC5rw+lmP0WICQxvwJ0vzsHLJ4Cj+7Yy+9UxmEpL6DTwRv7vlscAKCrIZ/
7Uh0g6sKN8O2Uw8Mwnf5GdfpxpT/3HXk26IPXa9qfnqIkog5Fdf85hy4/
vVIjpefsk6rcbSGlxIcs+uo8Th7cD4ObpS79x7xEY1RyNZvnHD5Cyf6OdW2BbM5+ewLBuvUjLyqT17SMdnU
6t843uSL0e96AMRk7sWkLy1m+s1tfxi6Zhv0fxDI4lcf3npGz7tnxdaJtrCG4xBLSmMPMQB5dPRZuc65tEQ
5v1os01/0UpA4fXf8O+ZdOt1nuHNCTupsn4RbVk1y9T2b/i0/
J1HW6cRFiLvhTlZ7Bs8hB7p37RYtoPpP9dk1EGI//88Tnrv5taIab/
XVNoGDeIkqJClrx3N6kHt5WvUwYDo95cQ37GcRa+dp0dM784o598iw7dB1N0qpAPJ9zJoT3bKsSERDRg/
MS5ePv6c2j3Nt5//g5KSy3Hdou4Xox+YipGF1fystOCVWfgFhqG1maULP2Xx/A/
s3LLa4Uz38ag2U6VUXaVUo0qeb2O7lKrWsusgQqJimXBTS+ZNvp+bHnuv0rhr7nmV5d+8zd8zZ4Svl1GQl033YX
cAkJF8mLcfGMhrd3RiyedvcMuTH1pt12/kA6Qc2WvrZlw0pQz0Hv0mP026jnmPd6FJt+vwj2xqFVO/
3UD8whoy95EO/PnJw/S+8/TJqdftEzm6fSlfPt6Zr57qQWbSPns3weZmL/mJwY/f7+g0bEMZqN/
rfvb98jw75o8lsHFf6vjXswopLcrlyJqPSNm20Op5V69AwtpcTfyCB9j59d2gjATG9rFj8rVAGWh77UusnT
GaPyYNIqr9VfiExlqFlBTk8M93L7P/
z08rbH5kw7f8NWO0vbKtVcpgYMDdb7Hg5Wv58KHWNO8+kqCIeKuY2GZdCQhrzAdjm/HzjHsYeOdkqw/5v72P
Vxw+4nI/HSP8+dqX2PwYTXi+XBES34+NV7Gfts5cXBLQ+/zs9fvsdDI1qSn5dFv2ss77Onty9jn32fSeP/
w6PXtWPqEzcBYDKVMuetJ3nk2jY8O6oHg264l6iGze3WrtpkVKrGD0epsvBQSgUqpb5RSm1XSr2vlKpqcGkr4A68BXRRSlXVkrGqtewqaNZlIBt+/hKAhK3/4BMQgl9QWYW4+K6DWPvdlwCsW7SAZl0HAfD33z7i27bxi89gMB/
ksFkbBymlGfEJyQEY+NBfZtdQ9k7E8Y1kD/oCZJGXdFJgqRsKc3gQFRbZ7F80ht3k4ekw+UVDoQUGrUTJBHDvLZ8hZOO+WZqTWzeQ5Cbl+HGRTD4H7Vs6+gbd/CH2o6xSRerGF7Ajb9ITM/1QKUC2s48DWVzYnf83Lb3FMYFWmqv2mSs9Ew/zsGHG27uecGZHEKUMu8w9tmXwMUI0IfO0r1DUhnzL7X2w9pAyb++GM0ZBaLzp4iY6Nfuzx2XCqZOeZz6XiGx7IwlI99hU8E/pzqC2KsFD5RQdwjHyU+kzwxKHxAYTsEFgBY05fymxWgbx3yBg/ZmzunT5XsTSqUqrk29R8tnDcw8zH/Osbz0AEnAP7m4uufs/JNp5U+dOG3dGVfB90TG0e8fGxt7JLrCkuRXOaRJaCWf3nzBeX0tNEqCw07wlnNOaVHs9d7ORHMOL0Fz7YLwTgGzHBuhRSsrOYMaBpeYi+NQS+4dhvvGAzAdwD9TS6/TFqaZJ3Mthrf4mOpwfJlUQrT3BH72t9aXv+C+HIA7ib0O5y/zxdYWOdHbPaP+15+2IZwzL1NAr5Q/FYRo9kHSozEgu2VQV6GFSP/xk3gt/Sm+4GKqAlsYdGjgRDk3K2cFh2ORdXB84DcMU01IV+9FuYVkqSWvSLXjM+23z9k0+9TQ+2jI/KQSQ9V0Cz6PJUZHOdDwL/RZ6OqJnp/bnW+7mU/Pzk5bg/JYL+M0n73hsrCYv0X4/qhb9wEVqKmcK0UDHdTVUHxuDxCAEsT59/OWLMn7vkvJf/DOiwBx7/60v1oGyUt43pFVsuqI3g94QTVgLIfUw4zcKMbm6kjmlmQOCmZIq+6J+b0OBu4GxnqiVg2uxUWLqizurkOv8dqaMw1cw7mhuxKqImqUsiIgOSZzZ8ZDCNw1gwrmw80kH/91DTEP1ytWIG/ZgEV4x19Jd+3yk3Ez2E8CVVu6dsjLFzzKE0vFHQ0VEVuGpx5dgWt0TKy9SbLI/1FlJHvA1Q4Wz9WtKDsxXrcAlZHx4rp4xEufFpRs94i20TOlGb8flDwbFgZmP6iGQf6E9ej+IxFoiPlY3wb0bGEVr3GKdxBoW8mMA13FmWT0oH3XuJ64tVEyRZ3oKXTnU6QKO8gPw4fc1jgt+EqIXaEQD6Pt9tb1EDK6JyJiI7MSJ0A1Z/k/B0qATi00CPcPu/ngBNBH98ED3fFXAqOnm3Xx1pQLF4rwqXiTo80HZf8rXD1DqUjyM79Wqp/1fe1sNJS/iOcT+rr/ZPg1oTw/ko3kCJ4/CGKwP9n4l2wA7R4T6qt1jJo8aXYNPcLf8vYmxrxa1LhR7Mxa5jeGH3wRPm/8zDH+3jhFMeUe17ee7JHYSn7yk8cDXIxfJqZ7Ehqp+VmWZs3E+YDPE1OmwBBXe3LN6Y7yGtxqUn1vO51PHwfzvhm7dnPDQZKSwUEJQxiqZf+3XL4eAUQJd0UZn1yYD/XVWSeWY+/dNJhX4dP/OjOe3/ZTpGGWL84cDCE6Hm8MBYzlYJSbHmQ+NfsmJ6LqNwj0s3sKvEyOVtNyJ6VZpRqMFKJdP6zAYMLRwx46J4bTv1dvkYIoZnGgDbHXMXMAB3CZ7nRPlqGQgrqmWuKIdQJ9E3WfpwAGR8P/YqbZCc84l4YHoHDYOKa8sDiDkWlNGnUuAsWH0BVZfcxcMQ2zCM1lpW1WJNaIA3X/ksyQYcIPvpUbk34BMFt1H7vhHWMn2bHMmJDE/SzfXF6FWZjmGw0T1GG4x3ZIxSLdThA1fWAQ/aeTBHT60Q9yoPTm1hd9kIVwpQ/qk2j/rS0e7rQh0FPPhEHIzX8jJBjlxHmh0jmD5Rk7FVvYcUKOw7iuTnn98EoV3sbgfV6kv6DvHrsLvUaTEx4b0m08f9p34IBFxYQzVYfDvd0EvTvHVTd2HHbwUWMAi2cL0eQrsTjrkn7PbOY/jXvR/ny94kmvG0fqoYDe05Tyd3ZT+x0zUW5EyVDZR3Q90UftGlqAWE8OnCc9sDaHZxWoqdmHvtRgQrBngw/ls/o1Fn5upk+LHJtbZ+3A1vLsoxNfAs/R6Iie+M4+VhBMhzVJuBAz+JPyfvpySXJnMDOwfuVHWBBXnd4X/0EmcR0xs4vqUdtPl3Jw8lyqxJCbOSCtKpE3kkXQPzDJ0l+zLQXGLjXz9NjEI/Ht1Ovv3IR2uuhncAmBCYjM42t9UMn+TX1L4z46x+HJ7lZCNDKfqmP3Gx7Li9mmCQcWS/6iJ9Ok3uk1QLphzfZ91IFuv4bX7NQQoutMPfrgDkwIZ7B/pW4J+kr3V+r2rnEWvJ2Ek+jK9AcE/A3FmAKH6wy4WEwOrvpSHwMEYQzkFWRJnEBHtZiLMRRrnxjKAOxVJlNbGh6NFQcBtXMHBFADEfCBfIwLtAGeEZHFu0dl3AyrNxSY7cwSJaqxCSBVJj/LXvjUBGMUIU2A74GK7U5OdtovxoI2P8fQqcJfmIeUHdQTEEzoVrTmOVTOu+xzTEcIOhvx45Td1HqOuFDd8ZXCxNqjeDkmVlXbpU4/bP4T09ZQz6BaAmHOpXsm4aLtAqm4zeV4Fj4QX0r+i/ZEOhFimvT3T6zhz6gRxDCh3DLhRmkHxWWJFgaiYTvqJGd8xPOP1IGbijYQeJ2OTe0dzrvHtfkyzBGq5OzEBgnM9mVeMwk6tqFqVtCGjq9EaPkjgCntGhaMyp0OMnFhZ30nF0B9U0+o5ZcfROCR0fCZ/vMnJuj/lPe6aoanY5e1uUYTMjK+wRmMcqBLCYbqHh5e6f2hDGXjTNX4JGOJjALFKUWnRmmWVyHjDThaOsjpZmdLeB2eYDCJOUOOfMvoWO6HnCGNOHCFnHYzwu+VElAcVqmlimnKuFaPXCqhx6X04vOXU4VyExMFFjhvuhkJSTlfGEA2zsGZZkFfcg9yLXMfLz1bNi2T5AKnBbmADzGf8CLJ71dqz4VRmxVS3XfE5nBzB54q7yNnSAM46dmzGdI4p27mQnr1ORmUgzH2aX8Dlsw/9PRQjSCfcThhTfp0q7JPYx+d7PMhlmuBpytDxdu4vY16rCPrQ7n0XxhIEObNM3TNmBaLAzI3M4W1WpNVfZHd9ba1BYpVMbfwJjQCSjuqnyaVVMnRJdV3IdHFBRStBWUltWDHYbHXjJmD8Q1AhcdeW8l4NCD3yu8P6TTV29tpnZPmMGxiudeEn5mYxOg3Ksbjp+46xc8XQdr1ToGnGXfY9GkdnQhK4Z8WVUT0VxhnFEX5IbKCMP3aKIGxQBuvTqnxDzQ3yPfKPeHJy0tmhkhw3a47WGLlgb3Or3ekFt3AN6eFUyCe9gCN5ry3fnk58J9sYc8AbNVK0vLlwaM/9PFT84pBffbzAvR0ljzq6p6nQAJ5Pfrj6OmiUmUKsdrovH8Ja3DqdERcf0g8HEEuJS0MdfLCR4gIBzymXNIBbKh9M2Tt89hcFg==

```
MInwPr/vk9ABLlWMcfLY/
rfOZkVnz+H1pf+ee5MASERZJQVDQAZqYkEhFif1338AinIyy4/92ekJhEQYvlwGVG/MV51/
Xjxkz+Y9OXf9Bp2a4XXCA6vT0zTtuzfucGGLRFQfeHhorVOBtBabwD6As8ppR4CtD2Sq6CyriGtzwqpGKPP
imnSvjfdht7B9x89B0CrbkPIyzrB0X1bay9XWzqfLrIqfg8Go5HgmLbs/GMmXz/
Ti9KiAuKGP2KDJIXNnMf/
g6oY3b3xb3AF27+4nW2f34zRpQ6BTfrVcoK2VfnR75hTkr1Ven47u+1VxDTqOJiCnBOkHthmo+xsp7J2n8+
5/38xRqMLDZt34I0HR/Dq/UO5buwzhNdrXB5Wx8OLx9/8ms/
efJzCk3m1mru9ONNQS3WTS3OVUo201gcAtNbJSqk+wCKgZXU7VUqNA8YB9Ip1oUWY8YIT7H3N3XS/
agwAR/
Zsxj8kqnydf3Ak2RnJVvH52el4evtiMBoxm0z4BUeSc0ZMZKNW3PrUR3zwxHBO5mYC0Kh1N9p0H0qrroNxc
XPHw6sud7zwGbNfuTTHgU9mHscn8PQwkXdgBCezzvo9ZBzH+8yYgAhOZqWA1uRnHif1wGYAEtb/
QNyI8XbJW9SOkvx03L2Dy5fdvIPOe7ikblR7ivJSKD1lGYbKPLQW77AWZOxbbpNcbaEwOwUPv9Of8j18wyn
MSXNgRvaTl5GET9Dpc6BPYCT5mSkVYuoGRZFUHhNBfmYKTa+4hthOQ2kYNwijax3cPX0YOn4mv7xzpx1bcP
4GXX8PA/5jyS0hfhOBYdHl6wJDo8g8YX3Oy81Kx9PHr/
zcHxgaSeaJ4wBkpCWRm51B0akCik4VsHvLGuo3aUPy0f0YXVx47M2vWb1kPhuWL7Jb+2qbM13VUl2Px1Oc9
eFCa50HDAZeq26nWusZWuuOWuuOF1N0AKz8/mNeH9OF18d0YfvqH+k6+BYAYlp0pjA/
h9yMlArb7N26kg59LKNCXQffyvbVPwHgHxLNuFe/ZvarY0g7llAe/8PHL/
DstbE8f31TZr44ir1bVlyyRQdA6oEt+IY1wie4PgajK42vuJZDm5dYxRzasoRmPW8EIDS2I8UFuRRkp1KQk
0Z+RiJ+4ZbJeNGtepPpZBPN/u3y0/bi7huJm08oyuBCYGwfsg/9fV7bFuel4RXa3DLHA/
CNbMeprKO2TLfWZR37B+/gBngGRKGMrkS1H0Zy/FJHp2UXyfs34x/
eCN+Q+hhcXGne4zoSNvxiFZOw4Rda9rkZgPAmnSg6mcvJrBRWzZ3AR3c14eNxLfhp6u0c/
WflJVt0APz2zXSeuLETT9zYiY1//kjvYZZzf+PWnSnIzyE7veK5P37TSroOuBaA3lfdxsYVlnP/xhU/
0bx9dwxGI251PIht1ZmkQ3sAuHfCDJIO7eHnue/
aqWW2YTQYavxwlOp6PD4HPlZKTdValwIopUKBqUBT4GU75Gdl57pfadV1MC9/
tYviUwXMeWNc+br7Jy9i7qR7yclIZtFHz3Pni3O46q4XObZ/
G3/9MhuAoaOfxds3gBsftRxgZlMpE8d2t3czLpo2m1g1+wlGPLPQcjntirlkJu6h5QBLsRS/
9DOObP2d+u0Gcts7WyktKmDZx6ev8Fg1+yn+74FPMLi4kZt6mGUf3+eoptjMvAlv0Kd9HEG+fhxb+CsTZk1
n1i+LHJ1W7dBmjqz+kGZXvQ7KwIk9v1OYdYTglkMBOBH/C64e/rQc+T5GN0+01oS1uZp/5o/
jZNpesg6spuXID9FmEwXpCaTFLznHC15atNnEtoUv0v3uz1EGA0fWLyAvZT8x3Sx/bA/
9NQ93nyD6PfoDLnW80VoT23s0f0wcRGlRPp1ue5fg2C64efkzZMJadv36LkfWf3OOV700aLOJpZ88xsgJP6
CMRnYsnUPGsd20G2QpILb9NpODm3+jYdwgxk7fQWnZ5bTObsuaJbTvMZj3f9xN8alCPnzxrvJ1z7z/
A9NfvoesE8nMffdZHpk4l5vue5FDe7ezfJHlYoGkQ3vY9tfvTP1mC2azmWXfz+LYgXiatetG72G3cmTfDqZ
8ZbmlwLwPXmDrml8d0s6LYauhE6XUYOBdwAh8qrWeWElMH+AdwBVI11r3rnafZ89/OGNH/
sAbQHfgYaA18CgwGfhIn+fspHt71vl3DL5WoWVUHUen4FAPJsY4OgWHWn9TiKNTcJjE/
fsdnYJDJRxKdXQKDrXxiHPdG6a2LdhabNexjx+P7q/
x39rh9RpXm6NSygjsAwYCicBG4Cat9a4zYvyAv4DBWuujSqkQrXXW1Y59V9nhorbOAe5RSDwNLgeNAV611Yl
XbCCGEEML+bNTj0RlI0FofBFBKfQWMAHadEXMz8J3W+ijAuYoOqP4+Hn5KqY+xXEo7GPgWWWKKUcq4p8EIII
cRlzqAMNX4opcYppTad8Rh31m4jgWNnLCeWPXemJoC/UmqFUmqzUmrUuXKtbo7HFmAacH/ZHI/
flVLtgGlKqSNa65vOtXMhhBBC2N6F9HhorWcAM6oJqWyJ1rPXPr8BWwD8gUmAFx/pIJOIl5JWrmPCOIDrRpBUDLwPgYOxQH
AA1inlPpba13lLbGrKzx6nT2sorXeBnRTSo2tZjshhBBC2JGNhloSgegzlqOwDTLs4OyZda30SOKmUWgW0xT
I3pFJVDrVUN5dDa/
3J+WQshBBCCNuz0eW0G4HGSqkYpZQbcPCPw41kxPwA9lVIVPA9lVIuSilPoAuwu7qdOt23OwohhBDCmi1uIKa1LlVKVKP
QD8huVy2lla63il1D1l66drrXcrpX4F/
gHMWC653VndfqXwEEIIZycrb70TWu9NpEI4OUd+90pNSeEhhBBBCODmDY740
/
oJI4SGEEEI4OYPz1B1SeAghhBDOTjlRj4fjvp5OCCGEEP860uMhhBBBCODlbXE5rK1J4CCGEEE7OmYYvpPAQ
QgghnJz0eJzhhoFNbP0Sl7QmQ8c4OgWH6rx+iaNTcKgu88/5DdGXrZRpzzg6BYcaHPzvPvft+2Oao1P4V5E
eDyGEEELYjTNd1SKFhxBCCOHkZKhFCCGEEHYjQy1CCCGEsBvp8RBCCCGE3ch3tQghhBDCbpyow0MKDyGEEM
LZSY+HEEIIIexGJpcKIYQQwm5kcqkQQggh7EaGWoQQQghhN07U4SGFhxBCCOHsnKnHw5nmowghhBDCyUmPh
xBCCOHkZHKpnQQ06k7soKdQBgPJW7/
j6NpZVus9AxvQdMQr+IQ159Cf73Ns3efl67o+tITSogLQJrTZxOZPb7J3+hdlw/
ajfDBnDSazZmjf5tw8vIPV+j/
W7OOrn7YC4FHHlfFjehFbPwiABYu388ufu1EKGkYH8tTdfXFzc65DwTe6I/V63IMyGDmxawnJW7+xWl/
HL5qG/R7FMziWxPWfk7Lt2/
J1oW2uIbjFENCawsxDHFw+FW0qsXcTbGbm0xMY1q0XaVmZtL59pKPTsbn1Ww/x3mfLMZs1Q/
u35tZrulitX70hgZlfrcFgUBgNBh4c3Zc2zaMclO3FW7t+J1Pe/xqz2czVQ3sw5pYhVuv/
XLONj2b+gDIojEYjTzxwPe3bNAYgL6+Al6bM4cChJBSKCU/dTttWjRzRjFrhE9GWyI53oJSBjITlpMX/
YLXeP6YHIS2HA2AuOcWxDTM5lXXEEananDMNX1T71OYpFQagtU5RSgUDPYG9Wut4eyRXLWWg8ZBn2T53HEW
5qcTdNZ/
```

```
0vSsoSD9YHlJSmEvCrxMJatqv0l1sn3MnJYXZdkq49pjMZt79bDVTnrmK4EAv7nl+Id06NKBBVEB5THhIXd
554Wp8vN1Zv+0IUz9dyUevXMuJzHy++20Hs6fciLubCy++
+zvL1yUwuHczB7aohpSB+r3uZ+9Pz1Ccn07L694n6/DfnMo6Wh5SWpTLkTUf4R/TzWpTV69AwtpczT/
zx6JNxTT6v+cIjO1D+t4/7N0Km5m95Cc+
+O5r5jz3iqNTsTmTyczbny7lrf+OJDjAh3FPz6VHx0Y0iA4qj4lrXY8enW5HKcWBwyeY8NZPzH1vjAOzvnA
mk5mJ78zjo6mPEBrszy13v07v7m1p1CCiPKZLh2b06d4WpRT7DiTy1Isf8/0XlmNh8vtf061zS958+R5KSk
o5darYUU25eEoR1XkMB5a+RklBBk2GvEFO4iaKcpLKQ4ry00j4/SVMxSfxiWhHdNex7F/
yvAOTtp3LYo6HUupuYB3wt1LqXuBnYBjwnVLqTjvlV6W6ka0ozDrKqewktLmUtPhfCWra1yqmpCCTvOPxaH
Opg7K0jT0JaUSE+hIRWhdXFyP9rohl7ebDVjGtmoTh4+0OQIvYMNIzT5avM5nMFBWXlv8b6O9lz/
QvmndIU4pyjlOUm4I2l5KRsAL/mCusYkoLcziZtq/
y995gxODiDsqA0cWd4oIMO2VuH6u3byEzN8fRadjF7oQUIsP8iQj1w9XVSP/
uzViz8YBVjKeHG6qsG7qwqAQnOj9XsHP3IaIjQ4iKCMbV1YVB/
TqxYs12qxhPzzqn21tYhCprcP7JQrZs38c1Q3sA4Orqgo+Pp30bUIs8A2MpykulOD8NbTaRdeQvfKM7WcUU
nNiHqdhy7itI34+rZ6AjUrULpWr+cJTqejweAFoCHsARILas58Mf+BOYaYf8quTuE0pRTmr5clFuKnUjW5/
39lpDm1s/
Bq05vmUByVsW2iJNm0jPOklI4OliITjAi90JaVXGL16xm85to8tivbl+aDtuePAL3N1c6Ng6mk5tom2ec21
y9QqkKP9E+XJxfjreoefXY1NyMoOUbd/
SbtQXmEuLyDm2hdxjW2yVqrCx9Mw8QoJ8ypeDA73ZtT+5Qtyq9fuZ8eVqsnILmPTMf+yZYq1KS88mNOR0z2
ZosB87dx+qELd81Vbe/+Q7MrPyeG/igwAkHU/H38+HCRNnsy8hkeZN6/Pkgzfg4eFut/
xrk6tnACUnT39oKDmZgWdQbJXxAbF9yUvaZofMHOOy6PEASrTWBVrrDOCA1joFQGudBWi7ZFdDugZpbf1sF
Js/uYF/5t1HZMcb8a0XZ8PMapeupJlVVa9b45NYvGI3426y9Ajk5Rfx1+ZDzH/
3Vr79cBSnikr4Y80+G2ZrA5U1trJfSiWM7t74N7iC7V/czrbPb8boUofAJpUPxYlLX+X/
FyoeH726NGbue2N47ckRzPxqjR0ys5FKj/OK7e3Xqz3ff/EKb712H9NmWeY9lJpM7Nl/
lJEjevPVzBfwqOPGrHm/2jhhWzr/P7TeoS0JjO3H8S1f2jAfxzIoVeOHw3KtZp1ZKeVa9vPQ/
z2plKpzju1QSo1TSm1SSm36aVNmLaRZUVFeKu6+oeXL7nVDKc47Uc0W1orLPjGXFGSSvnc5dSNb1XqOthIc
4EVaxumhkxOZJysdLjlwNIM3P1nBq48NwdenDgCbdyYSFlIXv7oeuLgY6dmpITv3pdgt99pQkp+Ou3dw+bK
bd9B5D5fUjWpPUV4Kpady0GYTmYfW4h3WwlapChsLDvQhLT2vfPlERj5B/t5VxrdrEU1SajbZuQX2SK/
WhQT7k5p2+pyaeiKb4CC/
KuPj2jYhMekEWdl5hAb7ExLsT+sWDQEY0DuOPfucd6JlSUEGrl6nh05cvQIpKcyqEFfHrx7RV4zj0J9TMBX
n2zNFu1LoGj8cpboC4hrKeja01olnPB8IfFvpFmW01jO01h211h2v6hhQXegFy0uKxyOgPnX8IlEGF0JaDi
Z934rz2tbg6oHRzbP8Z/+GV3AyLcEmedpCs0YhJKVkk5yWS0mpieXrEugW18AqJjU9j/++/SvP3Nef6HC/
8udDgrzZtT+VU0UlaK3ZEp9I/Uh/+zbgIuWn7cXdNxI3n1CUwYXA2D5kH/
r7vLYtzkvDK7S5ZY4H4BvZzmpSqnAuzWLDSEzO4nhqNiUlJpat3UP3TtZXaSQmZ6HLegr2HkyltNSMr4+HI
9K9aC2bNeBoYhpJyemUlJTy2/KN9One1irmaGJaeXt37ztCSakJP19vggJ9CQv25/BRyweNDVt20/
CMSanOpiDjAO4+Ybh5B6MMRvzrdyP32CarGFfPQGJ6P8aRtR9SlFdxCO6yos01fzhIdXM8VgLTlVJvaa1LA
ZRSocAkoCng0CnzWpvYv+R12tzyEUoZSd62iIITB4iIs1w+eHzzAty8Aokb+xVGdy/
QZqK63MqGaVfj6ulHq+vfAUAZjKTuXELmgbUObE3NGI0GHrqjJ09O/
BmzWTOkTzNiogL4canlYqPhA1oy57tN5Oad4p3PVlm2MRj4+LXraBEbSu8uDRn37LcYjYrGDYIZ1s/
JPvFrM0dWf0izq14HZeDEnt8pzDpCcEtLx9yJ+F9w9fCn5cj3Mbp5orUuu5JlHCfT9pJ1YDUtR36INpsoSE
8gLX6JgxtUu+ZNeIM+7eMI8vXj2MJfmTBrOrN+WeTotGzCxWhg/F39efzVhZjNZq7s15qY6CB+
+G0bACMGtWPl3/v4beUuXFwMliu5HhlW6XCMM3BxMfLU+Ju47/F3MJvNjLiyO41iIljww0oARo7ozbJVW/
j5t3W4uBhxd3Nj0oSx5e196uGbePbVmZSWlBIZEcRLT9/hwNZcJG0mccMsGvZ/
FqUMZCas4FROIoGNBwCQsX8pYW2uw+juTXRny/UQWpvYt/
hZR2ZtQ44rJGpK6SrGxssmkU4EugEPA62BR4HJwEdan1+5tOLlNpfkfBB7aTLUOS/
bqy2J6y+vP+o11WV+1ZN+L3cp0+5zdAoO5RPcxNEpONS+P6Y5OgWHanfb13atbgvys2r8t9bT298hFXiVPR
5lk0jvVko9DCwFjgNdzxp2EUIIIYTDOU+PR3X38fBTSn0MjAYGY5nXsUQpJZcACCGEEJeSy2SOxxZgGnB/
2RyP35VS7YBpSqkjWmvnuse4EEIIcdlynh6P6gqPXmcPwRUFJhZSqjvHEIIIYRwctLjIYQQQjg95+nxkMJDDCCGEcHZONNNQihYcQg
jh9Jyn8JA5HkIIIYSTU1rX+HFe+1VqsFJqr1IqQSn1dDVxnZRZRSJqXUdefapxQeQgghhNMzX8CjekopI/
AhMARoAdyklKrwraJlcZOA384nUyk8hBBCGdnm1umdwYStNYHtdbFwFfAiEriHgW+Qg/
Z7PIBI4NgZy4llz5VTSkUC1wDTzzdTKTyEEEIIZ3cBPR5KqXFKqU1nPMadtVdV2SudtfwO8JTW2nS+qdr8q
pZde46dO+gydjRxkqNTcChvLw9Hp+BQKdOecXQKDhN23zRHp+BQye/f5egUHMqlzr/
7/7791fyqFq31DGBGNSGJQPQZy1HA8bNiOgJfKaUAgoArlVKlWutFVe1ULqcVQgghnJ1t7uOxEWislIoBko
AbgZutXlbrmP/9rJSaDfxcXdEBUngIIYQQl4HaLzy01qVKqUewfFvFCMzUWu9SSj1Q9vp5z+s4kxQeQgghhJPTpvMefbGqt6WxkooSs5ZXWnBorcecb+bOfAlEIIIYSTq8GNQx1kadfTiuCg8hhBDCyWlT7fd41BQpPIQQQgjnZyzz9IeCGEEIKJ1eW9PGoKFJ4SGEEEI4uTq8ON2H5Kd4ASIn7dbf7+O2jcmDULemWVMRjqe9YiEIC5eP7rTmcLOdNBmGnwJ0u3z4hZIySrSCDxkYY1aGHxRyVqnJMJZ7XjvphqXLf5Vw2A2gNXu3kWZhdR4xkVoONVpaGZnVetvBJnIk0TDVtGpF6qfDD7FlIIWIXSJ9yWu2lrOwl8WOuaRW/WjB/iwFgA/kekzCy+JE61B9fg6eMbY3yYwcEQUEaJnlwWzJ98OHr/rdZvqlD6/VSU8pPLP0S1qJCkEEOyPmGqCGp73JjTd0jGjJU5+DXNCx1RmxE0E7x43A8z6BcFPV1K1vxtk0p31HOYWKJ9j0vQOO1xWv2bO+hC36D3evIuv8ELdCI2x4AiBvlmXBsLeP4xr7xKgXTqz1VSfpTrP2I7d71TyidoaQ7nc7JqMCY4bxJZ7grX3ZPuPd61tJxHSN8Pr4m03PwQxuVZqK6xQEvMiFkwEe+P5uXU1xfnkt3mzJN6lz/7R8w9R5l6eXAGZzW/O3q7PNPRrHWvSycXaeyQNjHR7WLNvNfEXCNW56HZaf+bFx9rrkIl6MMOXUNW0xzn92uT+LP8IdTl8wUOjEMmIbOWX13DKYsQUqu69hm5hZ1m7eGLHAhbPDmQ5nSc75bi+NcxFmfb+O2L1rfPY2Cx/R9r8dcN1V9aXCxxp4O4wvjtxkkNE9sL9DAWRtyxhLxpPkCeVH3MAZeI1hGEC19bnZHD/Ojr9OLhSWZ4P1AD7yTvmvO0tdbkZY8Q+EtjYr6CnRBF5rnvrbb0YRPAHdC87X8vsgnHMRHmmCw7Tmb37tDTmxZxajy7GPLQwEnXl2FHPlbkMdZCvR6tVF+Ml4ovvP44aSvOtqRnkN5gAsxB+q/yLF+eQQOaNDD0Sy0kbGR66V/Rb6TZ+y6jqNbGiF6n/u3ZtfXk1Dh5qoUWgmNYxtSN8+Fj7hWfqpBq7ViD6X6tr7YS1eR00GZHF9k0Mjbxx5TKJJ9wK+I6e3F7yG6kZyPmLhWb7ehHfA0ECQnWrMnOwvn1krf+6KO+mhcgO/ON0/mEKL+8V5qnTePj1fIXhbQkp9CQ/7N11i7nHE3m0nYlXR6czHSDhHhqnaUsr7C7mUJiNCWwMtd35HoyPTTvLjrcgQDyPzdJNmpRrPAtEzWd5gA4Z8zzW+fiWW8sCrDgH+frrz4Sf7v+tcPd4HNIb6L+w1xFGfMI9Rn/v6z6kR2+FVvG3zQxuf/7gUr5OiC9yRP73wzPGk/4nTzT7G+fwFr4yN/4CYqwaXd+3rNw35ll5GTdvW/7zx5AjBhPsP+GMu1uXl7LqXsqSB1SmaQnVK+rHyWv3QFTDDEF4OepCNQzH/pT/fxTx9A6pZOgFpkGmWLa4RnfTfAfz8L4yJaAtBmUXGN75dhrE2v5v7XyNNxQj1PZiA97sT6MmMhaPKZjNVqWWuTWfRfHRJP5iYkVPlqaKcF5c5JZhV2Gu/d2k66fK9rphy4RjwkIrkO91Wd6nPLRkh/GvLNg/uy+QOBCTncmXhmnKpQOCdoFIIN1r/pB+yg+PBMVn8Of/Lv1+yV+92DDoGxbM5A3+8vmoeNYONnBtBAR7TThUmiMTdMOjI4d8TjXEa0gzgGAUQHc8IdE3IEf2lT91E3vM8s9mGxuEBR3KA4B9jGQINmnFlqnhJvYkKmhSpqIFQE6IJ4Wj3z1LnDJXEQr5FN6HvFNTJ1X0ieW9C5gDFfdqRnOzhBWGJvVz6/3fhxwrs/fqwQ1cJCGUW/yQvbgA5F8dwH57k3GT1ucGiR4mH6Fb8EWUH5HXDa9YAOv/VCeFTuCQzPNjcFBsp3QnTgs1Ce1jTgdCwbKF9QK3x5Yq8e9lHtFHT6L6cl0I55Ibg8wjNDaOPnFLWXM+4EfW6jpB/iFWf2+TXNkZPCJXRKzPyT/Qn4ke3KMSjRfE55bs1ixNbP9N0TXdlOnpdBPELhBGGtINHCdCJ+I3cn5k8WIHSH+xVaXSfj5L8ZIEgl2p3WzmdOUnjWjoj7m4Pk1k0NMhsGHVqHdEDKkz/c2FI1jzPZj7H0q9VwSbU/aqU/1mn+DNoKd/m7zp6DEvPS/9JNMwu7XKIwvILzyDZZDuUGmSWX8K5yNuvKYxdXWvXe7Xt7v2XUjSRKVkpJc1tbWqJk2vrp1Vq5evZqSkoDx47ea1dL6i3fHrqFkFSfLH8ZhV0hczjzUF9mP92ln7xhaXrT7P0GwfmtfmYN8zfYPutRUqjJfV4ZkTG4gX4Hx+T/xuEiTUZk8QHqfxHh/xZ0P7HExjJWgpmBuJ9h13pBTIn72Zz92eXOvFHBOMaNiuuGyXkAsFtnp7A42zkUiCUkCHTbiVwGTaxNbwZ18NqYImjDFiRhnUNMn0JkDmMVIO2ez9Z3NL8zHPEkNZVRvNhVn3kcBKnPPVA5KvrW7YbmtfhPHTN6cRTszPzjWxhvDAWMSdbs/MGzS1DNSbtDvIGP0K+hbFoTrjb4zbHHnUK5Fa4xY2pK7e8qMAWZEAYQKTYNpuqI0Pjl7VKmY9sOfW8jEi/Z4EA47cRnwEpqdQpWmsuR1ZuCiaekn4mebxnmR4yhJ5VfBSHdwEP6rjH4K8Hb36OUutTTJCaYxZjsO6BZkJaZ6FjXZh9MjCNCBGpHFVNWQJbsXKEE2XXhGB0mCLnhQKWcPOCQ9rgUn4N/SEWHzAO3uyUHiBDGm1D9rP8MdWy3D6w3HR3qvn+TZKvJR9rjKU3GHXvLCdzG2gz3afuCbyWgu0tB2ys3lLEDBwyWUK+FS14xcfHvKl6AwFbYJ5RNfkYrdHBsgU9Q76XE+zGk4a2ShMjOs33wUFV5fE5Lnw/G0qr76Xy9FtcvAwYjpDu3nOv4ObFMeYttp4VA8n/Bz3PqYx3Y5S5GNNqz7b0Ld6T2yrwcj/uhvfzAnzVD77X8Ct+rKiJlETzzGqRvPpCHbFwBu5+/WTuZzOkY+8f3Pq2P9cDP3sAlhO0nYeqXrWNlwZ4GpwZxoGTKw9ky8Otaiu3L8Y6KLXqfkyBnPiSNKUUbFZC8mLwcGuelqM4YVOUl3HwPrJhhv+ffDR9XqBPvzDdpmdg+dsqpNl8+p0SRMpT4ZnhaZ+Kry8hLiO1TJhh5nIoEnRpQP49UxdXaOdFyqAw7RRhIm/DzW6ygQpg6kXwuKQXmC3W7uXJWLYtTK2Ha9NkF0HhJQiMbdTXjNa4d7qyMqe+QYD2Vwkx3tfqm1JgRt1Jn2+JOq1D0aYsQJNlH9Yf+7J+Ss4ESwqy62fKqEoTwW0nvwMPZIyM/zRt4qrDqsB5WVl9dn0CUzDuUwZ8sC0Fyvsuy9A9SBPl97Smsb7pjeF7iaWa3hMu1hw2UEKoEzypRkUlmg/5FqPBHgWxDrqP+sjbS2GEFD/UwkA3Jhu4SKMIq6K9jxjIDCrt60fBEgTC4F9WLpYfLywpb6Bp0mmQHL9wjMK5tGNAEpqBqgEA59BvP0LzI7AMOhfb/mPnIu3GH10BZyHXbDKGNmsK2Q6hp0QIQGV9YjH0FbDBImYxN45HI1ZXLKK8B3IjVbGFDc8EfZQpwn3qvIpmT3mVm2pgvI0b+X7Cl1EV3DpIL4lEsf+SbJDQ43G9jYHHOeKn4gCsy1qh5cPbkxnJJWKbCQz7gOmJz2LuT0MI3I8ofRLsrfmkVbCP0w9Vw2Gq0OyABhUuKaVrNSBGQRO+j2DKUEyQBFRgFFh4vNzrw5DzHlsfTSq0XfrWM5lNmwIHcwIW0jDMt7T5VhE5JhPoYVq3MZhbN2X2pIiSdnzXGHk3AL+0D5Hyi0pZJzT9DXu7nAWvqWLPUTXUr6bWIe49yG0vVpK/WvXo/GcZ9KmAdzsE5c8XPPnKFX8MHgTBQCx+NrO7vd0f9t8PgiNKW3V9rUeI3TnFQNaG5w0Y7r3Vwda7KZhnvB1yHfKf0gbuX8IxcAVz9W7ImdvEuELyIBt5m7Ty5rNYcxd1dyMpo5DTjeWJLuvMm+TWqOvjEs+VHDm3WeBgsZBbfU/95VWONPMWBRZ5yAAIVZ/mBMl+uNNjp9Rt2uyyKp8WbiwMtPzCm9tqcZpbCfHAHT99O4wTtJOy0hLzDbAQRrhLgz6YnWPFVKO19oD3k5wyWmZhmxL9WBsvDOIVXZ4Ur5RLG+e7qqS5L4nkzKHvyHAkB4eqNJ5UdSyBgVWOWYrSu7xZySkr8ADfD1RPmIJSGFO5Kyee44w+V5lP2/P+M6kPWHx0fcONUgTrBZcTkUpD5nw4ugtD+QTglsWCgzK9WyPqF3cA1s2fY8q3/Rbr6WAeWdLDTMcdqtEwhfyfKs7Ihvu30aswlNKFWc7u5wRHAW/+CsGJIQvc0xO3VNdeK+5tF1w+C2FcDP5q6qqbmPvoqXjYTt2UiNbgWVzlHaMGj5VPWtf9Stlln8TPu8x3Ui00WeUd5JVYHH7fqLlZFy/ycCA62MZLzpQw9XUlsE+/6iNXhy/1yVNXmQSvG6MFC4aiGlQpgSl8RcJM0UMNd/LMCCsAxmSnz4f4DSJz+N2eBHyQXpFqKnl/x8/OXbTS0xq0J5FqzvYM1ExK3lAPt4XR2+xF3jv/RkmbKHR6QjAJFdw8bHdKOBJSnUOrCSs8f7RrBNVGwEyMUPFkoqAsxu2uTBqzTtcOGMt0QKXCcQh8fbnljZpbVgGyy0IgOXcz+BI21gJwMjNvnbZF7N/WxY0qbC5V6gr5p36p6rxYD0AxUqRLBBXa2FxqMk1BLUFfEfr/bvlqvZjqfJbUXpmEMKzwEU1HpjR9fo8pZR3upXyBh4M+MRprzsFMtu7e1n7fZN48K4WVaA8fqd5c3zJLzG8vUXwRMgLdZhVzXQkJO4MqG19MByc+6C2LYGSsVe73O7ZM/kfp7NaLQg7nO9h7F6AhL37JdEgG4E0jGiSzr2q6KutUNKnZgRKXJqC7dg2+9n8WsKzRujqgCHsPNiFLxIfJ5TQdZvH9Zdw5OIXQuwFW2jtq3fPlNmh6eq2SzHbDQ9sYp5n9Hm3ndlmR+XPb1ZNOwc9y2JDMqXEe2c9kxP/0ZUaU6wd6w6ltsQz0IUiKM/bA7xDBLZGdzJvb4AY9KXzUHKyFfSHHYWC3Je+uu1MBAqfeyvCYYMM5ftUKMJ+A3HMJ7l5v2ezCpjAlvn/d20QBgjQtXgaGJQxDZH19XaGNyfGdg1kWG1pjRZIeBPW0JhfSAYlGyIyuuNQ4eHtZp8WbSH9U4rwp9uRy4x4aHkSbmyZ+uT3bCo0YpQDa7qnJEVYDgcuaAp9BOhLJLfEKmXrvvHRcUUhuvgz2/pvYd/YbwPpOptjIv72gUsaGTBlUnCYC8Qu9rW3VzVSd2Fqks3n5bPjtvYdMdnj8xvSTuAPv0K0kyYKQIaVtY72BDOMmE5Z6tZoIk80h4g7U5B+bI57HxFaWiZIPGnOEGV80jCswnCUx8RhsXJfwO4X/jlTzvlG8sjIZK2D2tFGrmDmgaaPA1Nfi+4UcoXpZc7FYkSRwDVJ7tbfQQN0D4gEDBQzVpVs5jGDpwL/0gsPRpP9gRP7O5k4UN9TQH3HTrKQkx2L/t/yixaiqmE7eGR/xJJk7lddRi3LDJDz4dr5d8KSVcHdlLh9+BIFaRbQWsiQH0gU/aW+ZvmYS40UOIzYAHiXYBkDb6BHGXsF4VXZfoaNqQyDi7aeKaeVGeoqMaFGITg2lc96p/r9uQ2NiJOhnQe9cR+zb1lkiGdDhgS5EP3cHRGZJeUVeaiRoHmfYEcsXT0A/G59ePddF/2i0Lh6GjTJnFhyvRCqf3DsmgNFNPMwmCM4g3Bh4L6yShV5kHZrpv/pRjc1Gk8UjjAA8ozZEakWxhB5GL+lI7k3Vg4G/jWP0yeq8c0ZpmHlkUILj/0KAcDRfJuE1lVe3xqt6WOgYfSNuszyZj86Rb4ZZmHz18ciCfVgvZ7pGSv2gGUY8WSL0iGtQ7SxPg6+2zdEpdI4GVmNjsiU5kDI0Yw/BwVTMb4fxDN+6zRr/DXX1v+tLf0vZnIhshrrc/7nqnCp3wVO99yX6kK0gYyIEW1WKuTwzKiwfZKeRNZEC8kDXuzJ2QqHG2ZPOy9PGMCm2SO2jb3DI4w5wu0yOELIRdOzOSfgY4Lpek2t/WQRZf8cEZeIAqGdPz3qaj7lXpUpVnQ6Z1KEgk7jVBbsZ72wgw8tBxEOmahc1rEuDAAHhVAp+eWAL56eYUW86UryhNeqYFW0Sox9gLgmE/lgbeYcrbrqRwGd6yzk9Vh3exHzQ0kMAHNtNZ8t3UHNRrxjwjBjzsJvvBa5IaKkd9ARkpqS5fJ7IWeZ2KNNqe9cqTK1DjZJsQdtz9bnxlLf+eNkdIcxhkvFqNb3N8nfAnS9m4FHMmO+XPvZp6wZ03qdzbHVI2V4H7CWyKQPbYoOnsBa6NOvl3P0zGT2xwVNSCbymA5qrxwd7PT4z/YQvfgW7VmNHOmPakrQLq0gyd7FRKDYNe+rlOKh91Imi+RvEDvMMfAmQHOVLhzvBUYtWI4YOuxG2KpqoK4x38o4aGS6p6LRo2BwSyUh3xbDTN2AF0IRqzqvcg4HZpY2oPHbFoGu1h3ByubhvnDhhlQSC/Ubql9Ym0GX7W4jsdvOT+4xMKkT2AlQGmgNhmw0QbU/YcMy2/Zd6UFpqGzpjpAGDQKxBVhq13qw4sKzt3Nc7cGfYd/gBZtGfbnStPJXcyqOAR7NzFw+z+PwT+2B27NZQO4LTlv9Em5Dki21Ph/NUZDAJsx7LqHPVdWV/fPV6PRF8QHUe+OGtZCEd6l99gkmmTxj+qkoU8vqV5qHPzXkn4dR3SpStvbXBmpvY8oJIS+h4KCLKQLsdwv/HuXeeSXdSpZ5f/O99R1xdO79/aHNpgyY2bG0LNgh0JyYlAQRAR4Xg6V9sFqDANfBmXbpFVvx4tpNvW7EK3F35tdxSkN2DLL8CjtqjgwKd6kUpPxHl7zjlfUDNcWwiJ7zh5aL5jdrLd5EtUcjwiaFgufd8/3xwWNHQgBOJtIy1n1dKc0lyQCqnj2/Fd0uV75nhsfEWjpdUt3pClkmOkgPvF3d8KOEjUvPnFnCJ1a3xU2nK4bCCH1YRiqY/rRmv+GqOWYEFkRcGM4jVJeQ5mUSIlEMnHBIbtQ1LRCovdSLsfWDunbVy8THbdyoPhVUzmqoJebCQdo8FJ2h4cYnAUE1IQPYGPZoVQBcTSmCYg9eJ8mbwZ2ydnnwH9yOnJ5yTK69q/E4F3oORB+oi5v60vTlVlcMkYYGf95xgAi01BELMuShWvZ/Rkjs/SJmlPFcMl8QKXtoWnlHP8/Co9eJ62RVqFYTuI6LIdupGb/kz1CYZTBdCQqEGyA2hlf10SK4nt+x+1K1wVmGXKr/Kc9VUI2xbzwXm5h+iI/k4uqG6g4bXYumZb3MXqp6Yhx5CtEEGWrkm/4Z+R1i1b6dfhC4FgGRCsyrNuxbXE2hqjhQz8U4u4o15a3PNmc1MD9hb6tXe77Gyh5Edd/xEKBg31ZC0A3LUpNXMrL3Wjm89ASD3qD1u/Pw98/9MvXQH6CzB6XNGwuq6XgZvYtE9/iXBkkDPcbxPFQ43mDwrpWlv2eoP4uD3j6GmRkEF5QAeRb6GgzPP6pPDjVjd55QMUqXsJfm2SV8W5qKpgB7xcimK1ypoI+DtYvBKcdDXJ6ZZ/lShXG+lv6uSf2lu1yVqK8Rs3gwubmHKsuVKlY3jvmxwtgq1Y1kOk36XBWiPiY5XRtDlT+yVJt3mjzM8Vi8tD53iqf0EwYoJmb/UhBYKr7sjkeg6SbvUozh9HS3fVBiWKZ0y6l5JITRcxXukZCX0pRBpPaGyvOQU7zFjbq9qBlHjv8v9W4GpMXmXh8W3YlTs1pqhUHsVk6TpWuYgxMtSvFzDJt8aaH6MKTn+AKezlaHiDFJJxEZ9hFxmI63Mu87P4qCLv8Dwe1lRpDXDB0l3DQRSgfwu/nmxSOFwkMjzlhM3MpQY3K7rh55Xz4ekwbEz99VTBq0skKZZXqxWUIJkbvxC5X1RmmbVVlTkOgFxExdfJOwbNTS+HE3g7m1mmIQUP7cGtYGLS9mIxldudjkSs1ia2pOJx7dmPDYcr+EJ8EBUOBvVz4ykNJ0gXzljEMj8MlXHUCbO3bzjZUCPwXnR4fSY3yZJ7ovNHmGYnkSf0dXHQ0U30x1ssqOuK1KDwjXZxjrIeAVd9BcRHzJRNaijsBZdPw3lhGmuUY7/XFaf8/1fNuW4wLHsGkDjdGxSSB5DDOqRKUrxH8Ui8D3G3uzcOBo02FCdq0GGUehF2l9bYAU0XRWp60MLvJWQZ2bzOlDmS0MEJ+rGqzLlBoB8IGtpW2cFvKKmdcxAm4zgbu1kaOV7Ig/ZnqqmM4N2OjtnW0ynMu2PVkX1RS5T9rnq3LdFlOkT36W3LGuukPIKwPaS1WGnhPivd0IETXp7FPnUyOn8jrjIqA6szxGSiDHW7/AxOoukzC8CsIJ/52/ZJA0nZyRZlMg/kmAeDmVRDjpW+Spi6rm5nJ+KoV7O+dUqQmSNdIGAlRPWU4TVzrr5y2Dzgm8CgIUmIJcM1uoLsq8+GDGUL4tqLLBvxkmm8e8wI2GYBAz2G9S8uoKLvpSo27wlbdEgyk5SKzaAqBGZ5M9Mjw0c4XtLGzCTVD3qvLOOFiZgMoB4yobUsJZVjx+3KTV5fvGFvBZ37CbzqbrK6sp7nhBY3YSIQ3RWODPKmGapWPIqr5hBMZkPKpcIxcVT3U4H/bzEm8W2dLX0UVVuPRiTtErxXjgXJJLqPpMsIgdbeFEe1Z2U9V/OYLhQGH2wFcEtVBYmA5B8ZUdSs6hM0Np2yHbqUg3ePsMXRqBT9pgQeWsaYIIltlyDGubiKcHFzQHAmJv/zXcSUTH7CvG2Ou6ZsRSmzYEzsSxhTwRuZx/dtE7RCQgxGnbx1tKmvSmVh3WHsyNWxzyxXJyzexYdFhM7Znn84Xb1Mh8zWRZhRDpuEsbSuUUiJ9YCVtbf+tyJsLPSfaKzdmm5RXkljfgu8e8xE2ssTZg6TLsbfsq8c3b5EyMZIJhkg4dZYgNoWOoo4bN7SxYsOEZzG0nWhODMvcXRxJLFJlmB1Va+LX0hoDJh19yHtunydRyoNJsLVgvHa3tyBj+OYrR2DE1lMZpRq2Y08Kx6ubYEwgfdfeupf5p3ywfJNSAyB56dlTdOHEwdpCqXOADs4z9pLHdXlbhXNV+gXa+BdEPzhSPGJb0YGgEIVtAlbFt0ESzmJ6cHxDPE2xqaMOp3TB/8L4bPsjN3xoWG1ybu1Xv1SvOnI2Vyp4KTI68UcXxh6F80+13BQzDeOzh/SGCE/NrqQfRLwNGPgWiRYB1Z30tbl7/zESyMAZLsnr6etMk96tR9PIr5NwvROGA9jt2IQQXeF3IjFcvfpM8EaIn2/n7H4+LR7PQ7JpGVUkLRXYKXuh5JwJO+w0wuh+ZNwVfm/DEtP1kvbBHsUTqDKj7fxj4Dh0OEYcY5Bt7MHd01rxchdYvgEHIJjsdmzkafnlfUfPs/YemoWsL6VqkoFv6EFYaAqaAdVDyRyZA8LBe3XSWaFDhFwvHyH4bTFzNqctJmfSkUaeHMtRhK7dIVkFmbXjcV/Nw9aKHzR1OlHWKRW0D11mDALYhjRQbhFxm6aTzy6iVMLdgb1lBDPmj4YxPi8NXC5f2nYgEdH5TPdMQbIiUxq8lVqQnULv/pFThBvaeqjHrxUBEXQDBhZDEZjhk6MR0JgLpzVfg8PHmAXprYJcqyumzO0q2yBzpGC1R1YCwRJ3WeWSeuFZuNwN7LPKNzj8H6CsXmcimOQkWEP7Tj3bmK6SzHCnqqqdEzRUa59URsJrUmeOvqZmFr6LNQIm+UQJrZ8aXPDI9YAPfkOBpzB/+2UYOdPbUX7JDoX9TcqMTNbmM3W80EmchLbeQsjNqcpgRuLq8LyEG8JwLD75oq1cjyoZZGhwzm6NYY5nN8S/sSWvKxl2IHcBEDW3y5mI4wakLwAG/s2AAfrmV4BGhd0kUFoEqkvZsEqZ5YfMJmbUEcQVTQ/5aVrbjdiOO+n/tHEWZgdp4gqwcePmlpBvfS5ZxLlxRBBgpfmXwN7oTWOv7AOeAvrnM5wmRTY1HJAQSnf+5aYxtiYgjn5PkiYmTGcfh0EAYt1ePsmN7G5zQFI8e1MfPI8gwWvW/UeYhKvuqwz0yvfzj2CIQMTlx9fjk6pNlyf0QpuK5g2mRBRJqsdPE5DvOzY2uNtpeQZCWZAanVW4MmuLyDH0hRp2sIjr/hCCzo/M+VeG3u9UfrFPhPgr6Zv3JvDbuxdUuEJaYcjxVJr9k2Kl2qq7EtTI0+fWrXq+WKAMOuOh3mDUkC2bNC3nBgcU3+5uEFvHcy8e2u8i+pVJ/PNJiTRXX2b3RkY2c4NFNtqY/NBNZSPH8iCbmd/KTeLMhMd6xNF54Ahb/NVN2OctJfGxcYsQbTpQPFPqEqL1OpdRT10lzA3pZNhnRfAnv/Vcl/e9oxwVuwZkV/ZpzB+vqG9A4cJePbSEYAPwhnMavB9h0nDKn4DAr6vaWNbo8eQjsbHOgIGXySDl/FCyoMiOeuP3rndaaOsNGhhQgbnCp66vv1+afVOyPcWbR3SKlFh6PH2VpaTfNgi5CwTV6qGmSoWMvuEzyUOsTO7i6rBvaKzcYIv7u3UG53Z3YzIqs5u3qgZqlwG7lWdzUeSL/hOgjxgDkQ4uPFd3ZfxKyHoMyYQ0IDaL7qQcaMUaiR6B4VwMcJ99v9yNI5hDLn7Dj0ab2ZqCyrYOfYrVxkIDG6vuGoUp9ibfF6uzyLAR2NS0RBYZJUVAfsDjOeBbJNN8rWCmHCbzhjiKzUybw+tvJp5khtAhNYZEwpyvtS4FEQZ7pgHAu5BHCIEZvMWfBhaD4Q/UUamd9DHOq/XWx91sPvk/5jGOjTtRVQ/Oi5Tr0EazX6O93UEf3znS6/JSAJ+v/0uVZPZWTEdhn9RgwUEK5XU2uc1eSxVBfPyuttyUrYAzSsxmzsrVHK0c6QZoxDTRhOS4KzfPbxC2BWxPkahkfSi09dAJEMAV11FyI6sdb4aXJ6I3zeyyoZrfLGZCqt3jF4WHZ3RuIZl1B8cEsJ9XvqWgOrW8d3Y6ZLI7VG6yF+dKQNdXukDRqROB2nRBvUy/6PZTpBtMGbs+3pnHZSaDh4mmGRBdBMdBZDbjJGxPXHcQ3RWXmeLuKSyIzaB8IyA1KWFsnlI5CdQtUItCQcbPUpkuLzJxUEphjhIGGLl1R1kfPnVCBPFuwcwW2vSQbPmn5xrUZ+u/GB/HXQUBxH6tBbvS+3LtDshcaxYrHjiA5qZWidf9J3CtAQnGjLOsPeKOBYvNuCcs2u4cqhvq1WgR/m2wQ/Sl/9W2lBN3yWNhOHfHmMwcP7HzuqXWlAnmq7B3dB3xt7/ox31XBMrYT8RUhsLFDvnVLKrcV8TjD+ODn8sJmAmpaPaakzPkhVA8E5xlmrYjf3xKS9wGo0vQx+WFHcp+UrYWRn+SMuaRvtZM0D+qYT0g8mnZXqljCSHz7szBdLdjtjzQi1B7CFUVpCW8u8X6+Bm9wbzz6gNbL7HLQ5mjROgVg8xeHN3/TUhP3xqK3bZxb/4q7phNTkBuG4y1wbcq8HcLE/fCxGoc2g5p4dqg2TR1z2u3cb4nnIVnIjYw7hy8TXfECD6dyhG6jmoE8i0x4SaOxAKBVm5TcjdtpgQxhpySUtSOFPrMmBJ1OB6KXJAkWs6Qu23CFtxDSHVIWxOTCf2yEe9gcc8M+cnsjLEOHSoBF9y6yA4YNZ1aaM1R5S7I6Ddpm0HRxXbP86fGwBuJq8Rw7bs+gf3lJbUyEmkWsvOsHjTLfoeJ1PWXvaFVUHb2fQubmtKQTmrcR/WXGkkINhvX4vzO2QkGF3s5fSGvuiDkSpVSD/b7hi8wxQr08VRwRfgx0W1lQDu91tB/S1jNBjVW4ysTC6rpVwzFb9AfqXQZgIZ4aiMcbahl69Yb/JuggYzWGRWFB7rK7dYo8XuHF4R2Nad9cjzM9q1hFGjm60DmhlSeeiWg/Q40J/v3S9WR9bvJAqo7N9eEvTnc5o5DcRqhVlqsv7EWtWCAuflnBumBqNRIXV3LDjNhg3wpcQ4m1DQF2h6VaVwKAHgtOI3U3r59kXb7vFmN3pNWvq34mjiiIWm0f8XDYJPNDg6lNkLrDLMUZMBkLVqNs2+QV1sYGZMuE/uOfs5MqtiENqTjE+MxXO2nZzK7hdeyHjGiBzpu8bbPT8Db6IGAmeDDjAWa/PibQMsMtY6wBE8npyvBz4SG9hBOiIaRVjYKfzmq2jhyuV/5bkpvx8UnHHyNhMJLcpwc/OfrmiTh+iSH81zGkyvRtwLfpEndwqVr+KdfPtnwMsjI2UIq1jUcRMmGwNNNpP5zd99gMcIPfNUO21q4ZfCLz3zcWoOavKCS+7Snny0PyRimXIL4Nz9bOG/SCbF2ZtQuY3IwLzFAY3iHE+lptWtZZMrsmEsq1ImzCrlzMzzdIbgVUEB1N7XcE6trdZJuXD4s+qbJPeOlaiPRIHnaOUhIY3Kzpa0vwBqF0vx7vkfaGBV05l4Ea1sBbWf6+nkX1wgQtfJtW2V08PVQNCSl6x8QfwoMxHt8cTZbYhfBoYcYL4vNagMRbdbOIgH5yM6tZIG8oNoiRsoVu2Bq4u9pJSS3mMoIO5OYBfvCsYlxl+OE6dgvNtxk4YXAu8EZdtkQTh2CAQ8CMQlKHdeFTDC16KCnK3R6EjmX0QX7eJMLYjH1/CsGfqHcjDUQsRWVJdkbRnKvfr5oDIX5RjdI9AQxwk3QNrxAfpG1KG7QVqIpcYlUv8NWUTTMqMVdSJQmqjzyW+6ulH5UC/PfwAOeP8dvaNtANdWOr0cWZhx0hEkrMdN97f7d8BUH1TdL5d8p3kI8jaR8e6eJLJOXmz/KJqGbL77WeV5KH0V3VfrTBtklbfb8nHuM18OIe+fKMjK4mgvR2hUiMI3mbBJ1FZ9kWsQd/lH8qYhvmc0IvEvUuc3zflKa+m0cb+MMlM6Bl8sEo8ZUMptRt5HCDgeoI6NYuU3Gmz7ga3U3VK0IDfK9YZPaBPJqEQprVdVeQVUbtqvtjuTSQ+H60qdAyRbJ1JtTylTcmcMZYmZIOzFGcKJ7DvAuQxFu5ic7NmpuNFVHG1rTYh1k7sRXrUK+DixvrVvF/Zc8VwJzxFZ9eQGLEBYG2wvobo0+RFfcI5p7mRBnbyHK9wzL6VmYlNaG0B1L/3qIJjbPI2L6NjmJWiQgfMcb73SD5u3eP5xxwmIL6P4XoJHxhsHcl1hq1gMYeeZ0nzHYqoP7zO2KgXaBs8gEjnsNc4ZeTJOQ02Vm6V2EyYKWmeGJKUzUD+G9oXUnJbgWWYQmo3THR9QzqIxYUe2a8NAjSD9LQWFqeLsLHVGd72JIOnk0xeeeCjLLpUh0nU1c8K6E0tn2EsLwQxVTzI3IeyRvlF8o2GnCdjvvRHovOR0mZKgTqPBNeZdjNH0LrwKUc9NlvY2Y6vIJI/tqxFYl7KvVeWaoAgvLXTVCvtFX16pqcTnrIrIHgxe2IIBaWWVzh3YuLBlxQmAJ6MRNQb9bQMlQLwgg31yuJKTCTfd7d4UUO5Rj9+HwOhI8o3cz3fUAAVpNY9XJ/MNyu7u/l1bPLN2U9oHTdffXdRiJJUxXGDsxlq88DNtlCHGGe2kbJ+Zq8SVIPtHdBzZpadxgG1grmx/YZs5DgTVP8ijJe0Dyo1GlHkvW3HxYw2PfA6ZCcJiYuO4r/QZJ2Zt/e9O1aHSbvWnFnHuMx7sUshVXxY3hPD3h+gSHsMoWFLSVGMmtoPFa6mrYsfeLPzn1Wp+Eb21rEzBhY34otfqY+vbp2QfKqRqR3SVdNuxVTzOG8Hbf93r/KDfEY9mS5iaRm5u8YP8VFD6tnMV6fQ2avNeemSC/SbxIXOM7qsOC+pvy6yM+IjKu9J40B42dDXGZ9YV3vWN/bNuMdVOU0ugHNjG1dRfIo8+KFoBRjO4hHYxqJ7lTYhdbdqWd+KM0Q8mqUb3WTxIbmpzNIoJA8jaGSsAtMlnx0Okx6o8rrlFPFvTyEZJgCvAaCVpI8n9Ro7QzKqmBlWRoH3hAJLP3phV6Kz+JuxlWOj4vVO8MKzcBuc9DfE9yMN9m8gyZQBj1sPUB0IATmnSAa5gTe8P+I5RzbwvLgrw0/kE54BEwyVe1Xxx82kLhsa8oD4JE0SZMTWFm1vAKoVnYH/EW8L9JFqb5b1xTI3ZqsyxuoXQBD+KVyeVlUsl94/3HY8vDL7n6nwKEZIIKxkdlD9tHP8g8DXKFrHaYCrVKi4xvpPTM5Pfq9e2B6v6Dqsm5sHYFE5dXv2uqrtc/GsM3ZJWBzH2pLgp/3jnJc6vNPmvrA/e1K76Pe2M1ePk1uvjQdLxUzaXBJKZ3HcGiB6ISTyr3hd+rgzpXsAx+nijIPbxUkwOv39Vxa5TB00OQufCuMlJkwdQG3Cj1CbP/9Vd/lzK+38pzwNoPVnQJpHw1VZ9Gw1hhmnmsUZaJz8XfB/kOPDT5AAsQvGK3cZ9uePMSsl8Vpr8ZymlXn0DXkxk3ySEX6QPHt9SK4kwHiePsuFS6yUaSdmfo6jp2JlxBCuOuGYhn9zdWiFm4YSEJ2+iQEYCuRlbDsxhrypspYD3DGkSsfy5PwAtbL3dHHeyBWqUQVKV/w/Jxj5sB/xQtS8qH4NYcCiYmsFPJxeiBxyAu4h8mFd+Jm5vqZ/Lp3rq5wyIdeB6H4xQzDnmhdUnKMcnR7Q0LqcC7aGPHsOXAJC3g0H5pJH4n5cFaI8PQ7Hcm2VyhxjRGGV86rGkXs7dgMkGK2WkOjjUz8Cb1YgrpCgejfdkg4wqWOlDx/kHhn89yFv2PLSJM7YP2ZSvfA/zdGkoeuAe/Gt29kIzJV5l5x5F+ft09Kmd3aiiL48bBRpGCd3FUdVhUH8oaAowbyI2c7u+LJxXdDbBPfnkVFBOqtRKqgt1ip1Iki9OK3TPEHhv4KYtMD7ylmlgUQKFl8Z8HFVP0VrUlObLBqkOOJNCeudCG9tFQgyx0k1x5U0VPxvo3R5HSC/9qXhN+rBOVXDxcD0/T17uLG4GeD8iapZ0rzfRVoxm1yMjOhoSc0sOQ9khvBOrDU/+EQntzzNRKBh3s3I8VQVebKqxKNN2Gh9gBnojnOh/NOOyvKapcM17dCUmQN2P1KMpz0fbpVlRm9EgRvTs5e1yhuIl9UKY2TJcT8Kvbgb3slWa7oVSM/D9EtOr4c7P/O/ZAC18EO2YqKffKBn5qGHbuJ8MjHjbU4a8Zq4AwT5mQUtNp4E8k+sGVtBsHIzJTBwvT47F5wMg7tc7MkQA9b97u8eN4pjZz8WRXxu1VLK7k2xANEYS+/7EPMu7Js5SXs0MK8z2dXQDa93r0E5jvQVjsgqJ2VNBXmcV0BxKnhEwQ1hWfPEmRkn3EMAHiUtj6JKgQo/cbfU9UA/NfNSQ5Ewwp9DfowfPtBoqI2ebspQm2mRAvT8DlmUIQzsw88ovOpiJSiXsGyvxjfLDWcRvbGmKhFYZ/6G/LrLnN0qcPBb7P4t8gQ9URzYlXxahFLZoVdLq7k4cUkpPaiJuzAjD/w9Dw0BxIqBO7ChU1TCCG9R8wtGA0lALSQB9BX0uiHWVkDMp7iAwB0HXEwjyABuwZZ+b6rZkdAUMpGVb3n2BYbiXZtyyNGLmbOUkZiKxuNe/o0ve3cd9z2dk19RpYDjTDehaiujAMlaeRzKawd03yAdl3lDD2Dxm0MDkk0n1jR8QuY8b4mhaYFjTq2OjOJlZu8tegFhiv6SpSWZpMEErYfyA+XP5vyb/ezvO5ymSX3jVbltPkrJfvxbfw7rsSWpMxctPaFFxBdqKT5ObhSmpxNXuudLFrYdnqRxKGpFsqBodcvJOX9vE/kaZTKqODpjKlTDyVxl0sGRcSlUV3/nM4DKWuaZgbeEEVb0mJM+GZoTqaE+VgLUS8W4I+cpchqTwpKEyVoTMFD8QXpEeTl5HTY9dlNlTOIlrC0gQYgZnjHa9YtPdfa3TWxezBPTbx4lI5nCZAR1VDZPQ9HBwdRrrpntIlpBN3dqcKfw5tkVpHz4rG9n1bELgoqgO0r8zjuLSlTM5UEg2I+bqQNE3JyQ8Axyp80/gkKAAh3jzwFUdqIQIrvgO3Q0vIuHXpOr7u3pH+S/ZUD7ox1bkCDiS7Ejbt3dFN9wsl7RbdR+DRtQ3Zq8ccmbfjefowJe5VMOGGHBpONMTA3zRHJoUE6CatUHnJ5JLoy+Ufm5mCcDRWGUG6zoxBKFAkqdP8Ki21trqy8IEsPmdFc+hbtNuldvCZXf9k6BNCOIaGj6R8UyD+wnBMGaHrzcUADxYTnWWYLTTzfjOc6qKTqkuKWrS4c4sDUOzBOebWl8xyjzjoFlpvw2Yw7+mwHGIpsvNgcwupbYixfdMh7XmeCYDeMvzG2vSBdjEHQD6BLCQRUYsCoLBEyVeWTmXlUAQZPpA7m+M2zzoSavj3vpSSNzxGmmi8RU6LvjkXvttSp1p7jfbtqyJCDRdDDIjzS4KvBTq2Q63tJcD3Om6q+0wT8t9bI/+BiAkp4+1x+OxqhgOBwjMJHGJ9mZAYwCZCGAw8Y8jM7p9ZuH0fh7zbt1ObrzsJv1pkcmHjpmbjN2W/EjLaD0glsKl0D0RMhKw5Qyyrdi1iPEmsbsFudW1tswsZ0pgDlGK2M/nLHz3kMaxk7DY3vbaBMX/O6l8zUy2IeRZK/AGf89u/u9U9Ktse1zBqqxg7DKvhpiHxBgXiQfCeujh/YTVrqtmJ2dpYCCjaJf7kNXDsRCjDDn1ZDOXWFRf+ZQP0ufLh1Rd/cK9X1G7Nz6PXXPZCNLzX13mz9QVN7xAr/aQxT5EL3oAW4Iv6a0IQ0v3C+qzvzdwS+6GXrzLyX9lj0RHxm7CaDzZ0N+kf8DMn7HVWDlEpQsNxyXkvP70UrpUL+fD6Z3ZTP4xHtbuMMRVJlvVeHbkmNbSy4jRTCtH8zQWrhsnu5+ClxrSWoXQLoVz+b1dG+qeL7tpoezGQh+xVbzf1o7wb8TxqX2bC1O9L+bRcIxq1kvbuzvHhkIOEoSOUvV/sDPU87zsu/RNKpIo2QVnC5PqdmWCrrkyT7mtcMtfmaF8EkpdXd87eMFA5xwRV3G80JvWNqZ4SsFNZJO8/Vx3bMXIFQiDwWv3IqjUBVtiF5PVDdTWBj/D4vOAgJEqOWsHfuYh8/E+Awk3CsvRKTgHf0VbBaW2o/u3fDnhmqtRGHXSwfh3xQIsBPVaWshGivZWYwBQj0a+H8/Gbi4Xi/L/xjThrXZY/pCrEvLqQpVTTCT/q0YbnEQtHPRVpyA77ndrBm1xAdfHMBi8ct3RMb8PEcZjoJK10vjcXk9vTl68Bh6Ox5NFxcpyIPWEQnxHYXmZ0aybyD4+W/YtmGhNSwyiE8V3W6Y0HHG9mrK9LVKSXhwy3PnIKx6mHkh4Qb1EjhqMRwi2MQpSvLiAv76wb8DnPp8wV5SH7pDkXVmqY92ghnKYE0UM8sBoRf03gYIqYPY5vTzaXqhvTFt5Dxmx8yGcgtSQPsm2m9mXpXjtiTQ8ZLyhr8zgLGlaxd4DEKnRq1/JKLlCchpOYCrtn0PGm+uXCfXg2gGW2NLwK3dHdKUfHX/R7WZSDvIC6b03D0PcSTd/RFyvN8fqvwafsLhbcSB3rOpC7ixpBMsv+FtSMDTD5VjwH6BnaPC3k/AYjRLpzSybLzCRZiE1yCRfk8yTP25iQlrDqwzx4ZCkbgqRlGpa2KbVLS9rVe0bmMeELYnOVrHQrTtTD6fTm1TzOcYNBxF9XCWRvDRWXpxfRwszH9B3qwwfH9bEiPvu8nmxPvXrNjcwSJd0w6SvQUzEdF0lRIihFMRqS9Xxh5kyUq5KYv/Mp/AUU64uWUc2dL9HUSJnXhS9DA/TBvVRuwi80cnavPhhgUaW8iqzQ2YXSdnW2Rwj9DdW8yCQUGG7ddHR5pudWbArVADTUfCq9pPGbV6Ggz9XQ+xq8wrRK60o8wDBZn6BizvEn0xKXr3qgONQxIi9HVS5aGMZEAy1sqpG5oXBB2UnG0GHPBY0ysJpH1kyg/uzJDTl+w8iDRNdHqPBt5FqnDFZwz9IcQr69KW5mPDfAxE1UvT0b1GJl2WDW4OFgR4KAg8i2uhG5pjk9+TQHZ6xGbVYRTUyEwPdcXYV+r3pOJXH/Pb5hmYAyf6DsvqX4ld+RnfwuQkQEVMlu7pC4yL2IRtTBD0Jz8s3NUaYMnxNFkfsLhjJCGXpZHn6oNRn5GUhGzAh+YXj4JKXAvMvIPEkUfP4ugJxYdGVpoUzlPLGT0GPMGg1Xj48J6NCd9JX2ULsxKyuIuqWGk1KKhVADgpHnE4Hmse44ItT5Y6F2y+Mf89QyHjqLj7D/uFgWx5spJbFuSjkYf30IU9JMxS5ZnvdB4jmTqoQfTsGQP/xuLTsVIHNuUkTxF7RAaOXsWtrmufBPIcnZw2+qp0D6g4lMjOJhrDj/BDkMdPB+1mGMJxK/5pLsoyebIfZz0qgpd1dngkzXA/aJSr43+5t6u6T4jTUy9lkyg4CU7ZtSxvOPOhXpWORaoHF6ELiSgJQ1VoW1E6hXjiyQu8Hf6L9HUh+59QGfEnfXb5LWqSrHUxTGXcoEjjeJTe6W6qMDN8xJD5QMw4S+WqCYjVfEfsCo/iPpMGdwE6H7h21D4lcdH+2KBxJ/+5HeK9VmOwUcWOAfr8S/xI1ttpZj1u03p+fPLLSGxUK8+v9k3O6q7RxQyHAkbbXmT9hOHhA4pQYnNHrfdoukbwfWyM6VJs1LnV9iQ0xbOjNkujCmFKNXWZ0xa8pmbZjjqfzzvfWd9+uNKsJcuqfz7eyRLWs6Q7YxoSeLsmHIYHmXizAXgPaG1Hh3tK9d4mO7Kq68iE5eAfZ3QBuCSA9sdaZhyg4ikfaG6cDNDjMsVD4QsKEWE7RaWz3oPFCpCKEVm9vJlpzlaW3OQIhwhHNrPQ72fgVyPpmlq+dlI9vMP2mxJ/6i6ZtdVoQ+pb1TZzRRI6pKGJ3k3uLLe0+fl5yS6x6Xh6xm1+y+tM9tHQ6hwzzA/qpDN0RhiRHu2pjT1hnQjKHw3h5aVMTeTGxbNXeo3Pyw2TltYBBrvCb7pLhFpNOqvIi2pc3oOfUJ5I3COK26JDHI5RkXrtqFE/3vR30AUnBjhLCwXpd8Nn3SJ9qtXTnFshmvI2KZjSHlXw+nCVexj8kUGrpXgm/6Hqpw0Fn51KZxAeJRVJAOZEA13zYzdmyCNKYK2dZPTvqFPhv6sdklVXm5tY8TnATpzOa6nnTwsqXOR6NLrVXH8cLdgp1hcQeYO0UAw+QGVdY7W5t4aGBVMkS21y5P5lMhsAe4pi9CKRW7AvRRRTiiKr2v8Xbw7YjUh/GcG7S3nUGuVQq8eZpPwY5lbWvwl1zlr0Sz1jDQrQQL4PhyCyD+JDe5OrIuZ/X44PWOuSkfc5jkEtBHA3J6wL4IwnlkB68YF6wGq0pk21Uae1SYUmrtnEW7GqdTrLNbyJrVC0pSmNyF5sOBUiN1YrgvJPc1RiNrF/vPbFa3UWg6aSbYkKYeuH47sAC/+xvEmAZtxZgePBUIc3tjrYrV0VlXbyNTxeCr7tHrZ5+o2e8LVkx6N0w2+F46qhP7JnMq+sBPXUNhTpQO2QFqzG8PDzOtB73XCbDZd94Cx+/kH8nnbw6jC4mNdeMhCewKcJ48dlJgyQPgWfF6jbZH0R1d/IYbA+Od+z4Fm21UGqwMw6y3yCrPX4HJoPBhPXiJIShK9e85HLClSXfBwMRkRA+0ZhM8Fr7f+B6z7pLDNTLYsGFd6BmiXPzunvgzyqiGOD0NN6oqX4WlSf1Z/c4afo6V0RGQb8XoCeDrvHRkxtpEHaJjrlKl18Af4DuS7MSgVLmAqs+I5H6jBflWbcQi8B4WPlGPzpRZjvEW3zqtVLp1MGo1qzmLqXr2evfsuI/kdlL1tTLAeDBETvZEBi+X6y1kahM9Dt+r2ezpH0E0BTdfG+2Dgq/wzwjfjcvX8qKjjOVjq+jh4Xh3CuTftrIuSBS54VCPfaZJcsu/3lcTtfYqXC+/RH7uhvGxYRHOEzjqDpW7kJVyWUUDL9crJsl58B4+HWaZfcq3CWJ34GoRDRovzc6ZzUU3Y/4JnkH05iEK9ZIsBm6vv6j/+RZDqAgLVzebxlx/VVI+2CGXm6sRmAo3KPI8mKe2Gqe9lrDBJuBO4bhNlaSbHJUyqfMu4JrtiL8hxuG6FYcC5xIHzhqaBLUgCKm7PqDZ7L+Ym6+a5O/3RJakMrQ/RVJoRjHT8QomuPNsvwbzA5N9TyCMdwyAQK8r4eL6pTPiSBBsvIrx6QxvcgWHz7rnXL0WEfe6vQhbmuyBRTN6VXe1sP+ezTIUvQqOxQAU9EaAcFvDDsgPnDkqhdoJJVdDlO8BK3bVGgP7r+H4EbUj7UT4e4ZrUk9GR9L0PcI3NsQwqaKyhr0VFWUlwkM+FD9JiGr6I8tL9h57W/0KVoKnT4j17tJ4Ss0OFQZTFAYuzuUnJjbXJpn2ij/w49ccFnJPBrJZoORLrnixQl+/U5hU6cdVlv6P+vgKfgP2O8Uqd9++oXVfz06GaFkF8a9fB/9hs44cMMpzVsDzyn7V6h0CUoM6Kmt+Cr6N2qaYyCEJ5dSMtd4IHz+oRzC3FAEUQXD0cP0f/w3mYqXvXIu7nE5Ad+ZlBi4UvP8Mkh9LoVlLP/B4iLv6/5lXXbvTOrJRcrizKuaArQcN/HgobT3RnPs2J1O4l3wCJ1MaHWRxy1apWZoJ1E8a/UDdVmBuOnp5bNQ8uOfsj3LuDtpPOXJ4crrqmfJSI2wcZojT1+HdDRQMY15i0b8IzAzIhTrwE4Gq6HkbHCaUCcNpE7+ejPgyzvfcFqg8QhByTaN4TUPcZpL8sMIGxD/3nFXLXUdeiU8L3KuXXc66KrfLPjrvzAa8PCZxEAbr1Po3xKBJCbQ7vhX4ogi1vs8SNnsQ3Jdl6lv00PWu1WBC+hHd9NhqhmkmnONoAxW+Dnh5otq9FD4zIvA0eP79QFnY8HnZzijo4ZsoMEejC3vJ5MRMW+7/0GdnuXMLvUEPJvt3kgPgRpzAHgaeukt0PwsBEKLNQW3nzkaVVyZ5mmnTW4yH7yX4yxH1r+aFgH9UtMNlDN4xJXXrWmFfPnQH3OHMzjzXiX94lhQKMMvm7RnD5cuMovN34/lfFKqdEjkkbTf/JAtWZWBu7ORADmbHrtpk6ZdcnkjVPMFjwxP3YcNBWw7aiCOkc1I2HrDSt3FbqxWMIWgVuGkpJB8+6/ZrxVIsxNmbXKnrFl8sLZkn8cN3jGZKvc8eyE2NKP3JMB0SOyPGjfl0k+aTzvumhWmKzdxUkd0dyq+A81ykeS1DRdX3eWhqOtBNqNhcNlbXfCUU4CoZDRFgSQUSjg3zlCFaQrzKcY3S6fD0axc9f+OfyIzh7UqszrHdmjPWVFfzl6ppZAfm3vLY1dyQfLGRdLUvqZb62IRIIWz/+hSH8dgOyCcQ3bBb+0N/m5BDhnUOfr5LfNeoFOc/1Nfy/Xy9OAhnmXqqg/TwlcrJsJc+9t/S0AU+mBrYkjBtf8CTwhVm2BWCGNu3mnrlNkUGlMu1UGX7I4F8tx91qRBeyDR/PNLT/yx5wkcuwk/mIhsSAsP1qcx6VRrcsGNxqFq0hK8FFTGSyatyZY8lwvCUw35fZjQBuPFc3dZhUtNFcjmOTC5pgMRk+v/2kA7G5v4NNn2gyT9wj3zeW0uxOvYxWr4BiOH48lYZnv9I1rg1uwVJg8XSmdUkDr2pd4KfRJgmqKbJ3QZVZ0YyPWO/zO1NaUazZ+kPFHkw5Tmps2blsHRQAdrxLfbuTFlxOuawfd0TnVUAalBypbzWAkgGy3BvjIwRKqkaZbOQVwk/eaEqTbSsPaBvB9RJYwnIijXzH4zjHlnMbgA1XKtLYuf6+CNyYbeZ3kCm8r+Zu8SEmyQvkEzm4XIRK4VO1Fx8yRdNQxjNczOtDdvnmtBCGe0IMYUVMK8BWCwT4+5PCHXzbewbKJ0mgAOGQf5cvjDY+FNvwF9b7cPQIaFd5WMGXO4YEL5l6DVZhk86D3vwLMHRjG2hsc2NJK4Gv4eADH0/aPB9x6Zaou5t5y9xm6qE2DASm05hlMFcl1rhKADsiZZLppSFyY88pZD/oUUtcmsqexmUS76YF/qp5rgLVHr5hSGHyVpYwMgwhBbNnyJySbqrqtCmTUsgMH5uKDnTnizzDWlfzm0BEKNONZNUa19hN7VaPyhD7apY09qqdvjSb+nlSlC1iTW3EwYR5PDb4RiDJdXfHB5E+eT1T1ybvrSg8qY9zZEhVvLEY5A9utL9qhhgWyTQ0oXEF2rb0L9RqSYFcZr4+pIPBBMwzcmeZJHvFzZPLe1nD/lZwgQydSI+jBKGDBNyyRtVXnjHWORjd6ouBo/HG6XSWqYAUdaDNSJGlqp2q/h3nd17Hdf2UNA4zBl4QQx6g1Nb6PuPTkKNyDKxlb5n9kXUuRMlUADHyVORMLYmQbKAZtfa/8iEMcrjO9o+FUMfW1Kd2VK7aWMUhw0YEm85vqEHHMm43w8MDrCfKEfVAadB6pyRXjI0ypbMsYQ5obKwrwhf/AqFcKnvntYSAL2KEzapQ+mY5dH6SKjgfeWe2dHJHcy+4ubMwZhJ7RFYGnBKW3zOFC2lXqNGRwafBZv64lUo9ClTRdgjyLxm3uU+kGBTNHGOchOa/vr5BrWp8lZLlyAdTxAHsNdnPIT0Fsz7R6QVX7LkUoJOn+xqomXyLvpAB6MOLOjfE1hLDvbOgUP7H2fNTLTn9hQKh8ifcUgRpQSn8NL4tGr2y6WvVp7mpLqN9IGrYCq+Rj8PVejxrNyGkYa3kvQl0+fJVNkyYDoYCMJnXoIlPTVnQxWXdLC8jLzrk3VT4PyS8IdBivQl8DrSTgdoMsEVZR34ymLbcE1x+gSyPwdi6n3R9c4Jv3Y66k5aWvbuqmG4mqWD2mGmf+WSWm7EqNu3sHw1cs7Ljd9i5c8yZYfB97bE7/H+a+pz3VyzaHA6pqGHdHjn9R6BeQv6rEHPY/TimNRJqFlDEZn3rmj1G/gGtrfIZnIqxh2rWJjYPmbbu7RIX7XbDgfY8tH+EtkoNp2O3kwL4m/GvPUQpSjrV6HBSlX7/LdfS9yVYMSPb7csAUQ1RnSBqAFaUlDWjNj5kuYQvd5Ds6w9qEV3G6Bv4VnjjjBTyPBa1QcsjP/UJm/ffvJiQqt5dQHJE2RhthcBotNAX6trIHxhqDwMqBFjnLiVDmcSgpOYvOcCTBYhDiEQg2d/WFhz5J2NQSsY/Ab8g14fkeg14ltp1J9PTljXgzC2+5xQyfSDR4wC3hXfh5o9jBW79CUgnAO8B4bAzQ/ljqUyqohNEnIAw61a8OMtHxffLnGfeJ0O/Y1wm59tm4+KefBIoRZm6CFsjB9nhRv8L8h5dqZ0LCjwSDfvukUtjwMH+yEDsgg4/uj1XGpwBNnJWr+VjzNldagyUmqX5UFpv8gb23D84P6zjmswvHFW4iKIOfYyXyjn6s4T/O1zlaCu17BqXeR2UQMx9qM6gCUZ6jg2y1AskHHCVOK48m3dwSZ4Ud/rP0ohdLEXV+4tHCBCG5tt7p0lwm6xt8QXhEKjdNUsrYAAOh6DIfnbYXsRaygA57EaNlrkbBXQmj87nl7XWw4m8AqxhQuQOMwfLEN40mfFEwlbdb0+6gp3LBt6ndEPL85KfWJOLvRKRTaY2GcJnE0jW8l0RXDK38GRbANNPz6jj8JM3uDFCmSOrlqn3aZHTcKWJhB0zWqU6fwRqg4vW30mgQa4iIQzXyqMvyoLqYhMjZCLjnT/i+3VpFd7b3R4fTlmGZwa4nEUXVy+UrKoUq+eoNQE32Li6nw9UmCRuBoZFp+Ccc5SJhZQZ8OImZc5Mqtf3snMz8hAPEe8mgKLFnYmU9VA/5uHamjF7sQWBIzjM3m28nn7jgmKE2pfXoJPhXGV4gj2s2J7y4xV0PL6Uqh9wxlQwKakPWiXrWZHlEVs5BSzpRHTFUCx9w1Dhhm13RL9yH6g/ELeuZ7Oc79BXc9T6eCFMYeTgmgAzRZxWmpbxQBaRrO0J7TuFAz72jrqVPQkTUbdX0L/VZvk3DI8mBvXb7j64Ip/8Yq1YsYYBB4aUbcr5NypvJnaGQJrLDvFtmJDOZQeBnB+F6+uS0BwUIn7qCLEyiWx8fsEKd7CzjgVT4vl5Pad6DH6ybLmxS4GGlhJpl8PawD3CGm27ksxE7LFl8sqfbcpW1wPRVXE9sjYHLWmzt8tdyBotTTzTOWvwbK0+lN4QNA/7PEd8TFCqejBsgL38PyomxMqzyuBqPFTUu95bd8pH7bExzPkZG7VDA9PprD+iAfyd3vmRmaj7UoOuSxa1A7XMbThxXcRVwOj9QsPQojvGlAK/K1AbnmwDFxI9ZBmAa/CHo35yXbxm59cqPl6w4V2PTDs1jXOv0qkPdlKBMdVqgjKOfWStu6NDgn/8qP9Pefm85qjbsEMyIuRCJI1Sxg57QXDOc3O5O3mt4JfJC+HPPh2xiWT00Ml8K3crFLeuSjGMn7Cw3kzo3KQyMvJ3pKP3pXWqZvRrfmTQUeDhqO8lzUaFjWxtNzwczqm5dR0VBE5YF0HKhqpUA3QeQqhnoJ7WCX4wLifHBQ5Q/A4j4rDj+1Fnf+bsVwH/VyWW7MAaaM0kVuoaaqnqpD2lM6r1cCnd6tQWsG5CfDqOtcxMOLdCxrFHJdhWhSQMypTm+q6cC9vnxANWLxTRXFXSSoALfUvHVUU9BvLTQMPgD2nmkvZdmNxdm3qc8lAqYk7V/mwGuyHXfh5Zr2hBxFQqOuXoNrEnCgClLMR5hnJ9uy1+Ffnp8GzcbRnYMlnlpbcQP4c7jyWXjflr7KfKVZg/ILEQ+4FgKv6wSNJWEpT57i5QQR4cz+yfiPexMTvMqFpZyzDIswXTuHJoO5t0wUSgQrsMAuNUg9Ibe8tB0R2b1pH1XWUAnvP3atfUV0mMfoL0YZFFNrWlGMAbmBsN6Jxp+JC7NLmRSxwD95WKRcphVDe7w+vZ/ZyGrPnwhBCMUzHIh5oYHGtfZ9WbqrQSuvuUjUnAt+qL7lyfbrnxRa5G9QL/YHH0F3Oa2nnWwMuzbzTC5t4AN7lnLWlw16x/f7DJcWCd/OBjhQ8tDB7OGYcUcjyaaHw2EPMwPLh59GXDCTLEV5kfx1S7ZZ62jm0xkmdvxSKwm2VSzxEf7fbvxFNPKEbpCkJ4mKeCS1wbsIZ7WtBtx8CodfKK5Hmjo0T8OsGJzAqSzvEwZYnwWWM7Tk6/YDKzSpA/hr5cWpgiOwm3MQnHvKZlmRApMqVqgbFBSbU0YYZ0K9qbEMTUjS+f+oBIiTEBzZQ/eK6OvXVyvOlY0CSBnTuK6hsIFuQpHG7ohCx1KlPm2Xzr8R8hxJtdtBNQhrscDKHCOKOXVyfhZ3+y04mSgqX4IZy9wJHXwnhXaXEY4XkYsGyxjGvTukWJBaEv5iQeAq2N5TOzGJbFh+vpJpW8MeGg/Tp1aRyuqOOlSRimt/MYgUvLAyjewSkd9qWefW4Bu/EvoChaXSp6ApJCNnkkDjTjhgKi2eJ94M8Yrj19s1H7ChLL5RYkqbRdrJr+hzzK0O7fJT4fRfgYHTNRV/j9Ea3XbbzJ0N+4KDeJh+qp5w/sBKO7VO3XFPrdNBmgT2p6PEDAI+CkMtm+u6J34evn5eKgTXHZbJxXhF4EhdE5R6fkFmapMkZC7b6K+4xFy3N4RCmpzLpw4vCm5SNSXZ+0xzvSahx4bJx7fgcJmOevt4spAaq8fm0kKRGw0Xe6jMnGfRHCyNPtz6apTxBqyHj5nubU0Nc1iJoyyIyj/PmybAY8S6gppnGEW9GanDEsNSIjCAD0C9kt/96d3vuWCZxE3c8a8gvEU52s3F+CR5FHe60s6I8CUeLIZtEtrkCbTyOeiy/EvOCK+MR3kt1F+TcNPbAlTFdBwgZpJQfl9GRZEZPdgPB7/APyEB+OixQqbaqP3TX58rmEMNoUY5UlzqELsHqdGxNtG6EAJb8ZmBrw4X71jYeUtTg3lSBwlnrqd36bNdHNKzSY2w3Ntgn37eVf6/pQhYMezffZG3iBF5yyWCVKb9ptmrBSiq1gxkbgE6Ae0TOnqFpWPq4A/SfLvsT/bx0N9ptQO2RKkvDOSfsW3F1oJqK8SlmAuvdQDrq3gO8yvI49pE4x2H4AfQW0zkM/2c4u4kHswIQpkdLnwKl31kIxyrhCY8nSSIcjW+Ti7eDiKd84PJb0aBN8SPHl6vAr8YBugQYCvp3FxRv6umlYGZowUNeHDY5CfzKyqsQyIYyOXAgMsIUJn4fNUzLN/6gwKhwUhU3sxl1J8XGYOAlN3Oz0k/ixL5RHjWKzK4iaZfkehXc2yzg7NH3V7qShWtDzJ4CrcXUPU0jzcm0u7lRlYWPF8WB8jodVfW9A6EbJ1utLRPj8SkDOW+Y7V/TCLDDpWTFAJqB8dXLdPNeBj/TzeIL8ZLUYRAOFlIUrjt5I5Y04aoyg5/L3TfQ56v6Eq5+sP5E1VNmeYqD+jTfY9iXYVKQ0r0Li05jZFwEXKFAfG4hzR9sC/hAAFzCe9AujLDmvv5uWVlxDRrQ3A8ye8vj0qNaUXT46ABQWhE0mAt6bWlgWO/Bw2uw9bG5BbLg4VB6m2dL9jKS2HHSDu5Vjv8yl0hv0xohwaDjSmPgBmDaOFXxszOvYFZtO5YAmWYtwp9n7O1GvvyTSavuaKq0d6uS/qtn+KpP1uIbpgEvCehJyg7nBzzE4/rx+BJ2BIGz4VHCfmJV9xmrVKDspfovCGzX3rsl7DrDTyLnRgDffGNd2RO8xbpWHcH4XtGUVKc9cVLk9L2/Dsw6cLeLrBOJK7I6aYHoIa59pXYAVxBjf9LvLwxlZ6Dyaez1OuyvhmDJdxrPzb4qQRszxKrw5u91HNjWQ5lxYKx7enN/0zCLQTj9rU8g/BlZr2a8fMRlwY9zaBYYvuVxm6MgfNA9XUhpmzAeGHx7Oj5cb7D5HOVk6ukCf1l+DUIxwQQmOuXn6LwHBZbK+QMAwrh11mhSHZkgYTDH0AUrn3lMTLxBYiDeAC2EktOB7ZEtPs1lYldtiORB7ypQvHr8o5BzlQXhnhdzFvEY5YQBnBgQi2ikOvvqMZfW9Q3CvqaEQ/1IUgaSAn1g2ZcsNvI+MuGWuh1oJ/3Ifes6xN3VeaQtkdZ/cZ/jDijo6idzBx/dm2RqxJIJAmDHPIY7zAefHLm0w9qQcwxOi12VBqu8CEgVB9y8tEMMd5bf5ZxrPRddLkNdFZEgQ4U4Z/gkg2T4rpSiQcRI6YjGeJDvGQGBF0wUmJuGnAnl6LJhg/QPmiOpRq5BR+0i5VbDdzMPUD4p6CwZZsXGmQ1JNxXH6ZKVldAVgm7tiu4QFebmWB/N+K5bnDG2s5DhC5HrX5GgmwDIyuNlcsXYJgXgNfmmmNSy71xW8wxAbffKBDy9QHkWGI87p/X4aYDhpbAVGmdJYgIOpjqMgRM3f64HU4OIe1hkhM3uQ9oKA3MZy7ZwDL0VXoAfwhWgfaa0lkuwU8h4Xb5bGJ0D0mw76N0N4c8mN4kfc6p1fNvnZLbo63dfYCFrd/tnyKoL/PKN8v+PUt/RnqpP0mqUwTJCQ9rZEKNM6HQ5fCGq0ul0w4xHe8vo+oK+UQ4yxPuegHfKT9XzazNGQK4b9uFtvIDFh8e4dU3U9Fqa2jTxdpOhBZhsYcO5anFy9nGHNuHpwjQPUdSNq/uSX9zf1LWAF5tALpQQlrFPdJ7TTSQc+Fms3w6LlGeB/G6kJ2gR0Ry0d/DCu2ISpm/T6Lb1M39F0vlBbDBEBDRBRLaYeHaDI0pdI7cuwqiYEKqIMQ5+JF9uKD85PImG8P/K94wD7u7sv+ntjKb8ILu9Nbw3tIhk+UX7dJXdTr5HUfB0wmKBI6hOITfHWHWHsGBpdz1NYYZIRlEe4LzXZZTlz2Go+rbEA9j4sq/EMHnj5NSE5I9vrbGBnWaDO6i/C+Zmr7tK0mcsUZ8+N3tVkKuPzhXVjGgxoSpF5JlHAy2UaDLcp1kpYhsJXqDx8C54OGhSf3ub51smHPIdpUxbHTKFoBRP3+0a4ZKU2j7s+cgEKgmT6kQ2iCFmSJqzJOvCSSyW+1FvYHsaw1BC1HMXgUUexhPrHA9AJhB6lBCkZ0JvqXmZc9rYrgCxNbL6vHqxpbUpWP7CNxKf2fpCE/UJPU+iLOahRVCBPPUCeKQzVU4v3KdjDFzYzyPqdzsjYCBZVIvLAWO8EiytgwgfCHn3ijh4TD2mEfYhMbmvDQpfFu8hMZn+yyEFt2XRC6mP6AIc7YmNmS/jSMDt2ogSf/TELwjjkAgE9aK1hPDR0WA6l72BWKfAQYSFC4rtxRylnrCP8MEvYJRzyF4pbEsdpIHlUnRW0iBYlBexpoZHrMXzi5FTKvPpKl1ulmJXhLGpjPjQ3FemtU/nM6d5Scmw23WjqGBn3LfdcsswQ64DsdZr7kRaXKBXHqRvgVEeYr9dlK8pDoGrAC+4WeZCrY04hIJ9d6rCRr8yVq8lwIkFHMfwbLj9nVHLwNhqMXVEE6GXPr1CtKbfiB1dNOQx1/3ji7wwCzTvEwqzL4w9uxH1hYbb0YEc31g3P1PaP4Fc5nn5qlJAF1xwvMG2wC4kQF2bJ1+e9pmBuXzuA2VJnfgB5nLDNomOrghuU7nKBYOA1nKjcbFP7dpWj/1+YzjLfKtJefuPPGWGJL2kN86pjHZ/yaYgUm5P6A81jsqlxb2jBDI48XeWKQIPkrHvfTGhNLOvMWQ7iVpQDEaYgzWyuDtOmpNTLiZTM/XhE3mzRJSRhZNS5R2Fjh9BLNRMn4A+iFc+vCJZNXVsOMsc8fkVJymbRXBDxXOvvYRJPp3O/QqI89fRV4wRbzBBXDMRkp3mY1fTaBf7d5DvhkPbqgfdIxJ1M7DfvSgWdAUXNRUoejjCoshZ7Kc1n5W6xbaPJR7NaECD8cVe88pimK3wZ0kr4RV2/LhP2T8rCq3gyBvvCWZTvPxmHp6wE5ZGW1eXq9oJNNmGOk0CmGCxKlUB1OJpwb6vpBZOTHgDnS+RrDTg9Ib3wkY5EUDD2CqT+x/Tbp0NBE3A0S0ohZ+H7CuKKOXLYbvAeojFgYvs0y37upAlnmTN/6BiZ81SVsJAWRdyxN4yZaJ8fOzjyKQwsx6w5ZQPEpP3QE7Sjyb5lsXbIxWjCOyVKvU+FM9LQOTj8iqkuB0gCPp8Md4SQqiGpBGfOl+2l9BuEO8lZ2BhS2aNXRSgFe4w7eTjZq/5/lrZ2ozbMS2B0Mtl6AHK0eb5+1A9VL8iS7xgcAYmqV4BcFDHbsXIEcUXWsN8ZqHdkwBcIS+M5KnPPlUJiMrDAMyFB4EYzwpqEi0dRmo5KmlLGFFCqyC9KDhpJhAfOCq8g6lV/xeF28MZjHHFWjMeAj0/1QrJVJSbA4hpY5RMNpNhsfaxwBhN1q+R1eSzWoIZHfd0YrIDg3jPx0h71RSxa4mcaLgTgnRt/tKY6WBnnwNWXaPYMkHjaSJ7CiLqbUmm1MxVpo7Cd7PFgGMI8RLfpN8lPfzs7AAgjsCR0BjEIYv4ofL3Yh80T3ezo9UaTmJyt4LByxZSR8xlOxTWgaFEBBIkX8w29tc17vGTNtyMbZuRvWyrCuP6FJ8xqjRo3OOFPUfsi1bLmb6GCYu+ZcJdmU6Df9KGgpLu0uvoHvkQ+8QbcWDQBumyj49VDKq4VuMeN36K5ATzTVWHCEUKQl9iiaXtBLl6F/MpA7Y6gCmcWWLtTPLBfo4KcjfkJG5DqkoaMXPmAvdE9sGoUlOUqCQ7vHyfR71sDIhUbxHgCOGGSsUZYuWK/lF8ZwM5hJFbdr7oHuU3Ev7IwrGaXv8xSTa5bKG8UuiDcdHJrn94TrDqxKg5lImKaX0nsnNY6JsZrsBO18tlM+a2ZOs1MpdyMoUjVSRqmhU5RpcqgMF3TIuZBVZUzWBqvrY7QK7HMXU+SJmqDq/o5YtJnA8OZhCBNchh6VYoNu2TEWMuf8b01u2IxO3F+mlcBEjqrLr+f1Rt2w4+CHkOw1aIWWoa9+AHHKt8QyCZoBvNyxHQOKd1aYu/t1qbdvQrkLbjEhKwPMHTh0l5R7jRiJ3Q8Ecrh0JAp8yrhk8Z4p2AMbhyPGaBx9cp8CY3RQ2dSwK1tUk6DXJh2IMnBPySm35mZkY5rSRdtQHHtAm29ZYvYkRXZP8CsNbD7KZ0kGxHoFc1yXQv2RNdYY9hRTyK1mSvDYIFUyb/+6zymbQkQXr2UeCSw2OhWL6bAgRlv4RKCh8JH7IQ9UBgZ5kkoJXjXuD38mcVF7HbuipqZF04sGFfDUyRV92+2ckD1ZLFrmaTmfsr2t8qozBkw38kIZUn0cKAYNFJMzLp6wH4J4a5fUZdIpXzFaG9OvVvQ5swbPVXgtRk6ZUxw0m3pZ8FXKMHD3X3m/tQkngqJoUbYGGKyF3UwwgbMbSSM82PtgYfVh2mwvmDnEhpKT+fIIdDsYS1cejRGS2mcQRIaLHX3avdPd1cqUNOyd7pR4PgpJOcsCzD+0aqWRb+D1eGdRFOx8IhR7rhWrWiTOq2yM4wHsA7jEgEKB5BwXTR6Y8o3RW6xtx2FfXhdPgeCq3WSPn9xvSHfpzoMd4bn8DJQ2I2fHOw0LWqsDWmI9ps+B4EXXhoDsBGnXwHKHrQu3NCIZDbqGdTKw0AaIN70s/7MGJTIvBRkLd0lrG+rBT1NIeTnJaXSy8rCPEbDqGj3m5tFLfIbDM/qNcnbqgezgFXhDHCmQ5kfeEatvmLp4IO7ZCHHhzj6aUuv2pNM+o9Nvj2xJK6HZjU4HqI0cD+xF6kv9QVtnNfnTzvzhhz1eG35lOvnXy3dQKNKYSBY7VvrhiIwW2xPmdiOe0pVrl3qEJ3fH+Jd2xzbxeTbm4yJqjNeOvYjmjlPjFgeQeDKNqxS7y4iFPNjXFMN5IjS6A4z5PjPjmyLfH7I3s5Zb0ktvHRPsU4z5zl6JYZ7cNOl/hjXZzxcfjfbm6P3YPDJ3UQMbbrJyugkcFsVZPYz5sqBnDjE9nQE4oeeIsKDB8aWHBcXmpEcmt+CzRJP3Ne9d+wvVlSywKvhcNjpA1HwCK7nbvlIpvZ4wXDH8k5ijObFUHXs8aIvvmSzCSoWaAXyx6Rr0GbYIYtEyD2ZcQ6W0PKoNLdcaWbtXcg+OvrPbVGQk7bMiKTHeMYZcI6m1oZ4wmJb1nVJ5E/ZHtSOS5lLojzpNZDsNEZq8TD1oAdqbUXnIh0KDpNtJI7DpmdLkvPIm7k4OQhkbFhAZmBoHpBfq2srK3BZ8NsI8T5lHmpOH5xWNyIxRF+xqM2XXKW0AjyUJDxMl/KlXYhkHE5YKsjRjOlVFDmHadyS0gOytk/2RuvQG9/c0Pz44/Q4AJtTXIfC3R+UtjROcPwFkr7QXtxTi0mIiUngoTeRdSKsoizxRpRxAgBwuEEEZjfXm/vSNZKsEHJ3hkpSySoKsO6s31zsTnOkPzOUWp1cJHpZ4nTsUzibeB23tjjA+8wGJeC6ni4Klk9Xd0iNd5smQNuYtJCL4dibpZoC9sCBhz+gvTTWF/P7UfaRqGVpGXjhb2+ZcHoPfZmsuPE3xBgqyxNg3y5klrmnIAmhwHZZA3n5z7QaD3wrCyUW4ScdamMdfG8qKhDBVFXtGnJDRjoK9s8N8LYRAIaT8NbJTrL37WTmNMUwEzKkYLsW0P9eXIUvU6BUQybSN7BTxsZRIMQKWoW3a5dW6FPJvbrDfhQpk1bBp9VO1eFGDHFY/PmWW1LHSHkUuyCV9+6pjLphqEdpkz0V/63z6OZ1gXOe35CGXUdGnZa0JPD06ZpHDtaqbY39zgedRdbFkKCRZwYJbcjOYsLhrh32dvcsHd2CQjIhWQpPZSlyDmYSHhNt3RP1cbp5ryIt8bMeXbwf5lh3XRLcsmr1yDVtHAvj3ROXnz9HGHWqgbTMqbIAgGMKe1cVe0kqiwnDsl6yThzyVCzmDmTKi4D8UOhzqBKnAe0flHE4anzajwqjpS5eIUoK/lQdNAQwgUnHWLXz1BwyJOnvVLdKEGuBBadbjCjeb+9PE3LDg1kuhnaJlcQmjE8PQBa4cUzY6ETHmgZBLYWb3kG2FDG6rIELuoKrjAqZl3GtPb3eXd0ITKqT7Pyx1uUjjYlsrLd4ZDB0atNStgdz4QEWbeCO6J0Q/Xuw87G2o1EewiOcpjEqoTTcuPCpFzG5UG5s6mw6NxtG9UZKaMKbOtJH8O9NU8Zqja8j2ZqzHrchBkR5Va60PyDXyy9BzkEtS+WjTEV57tWmBBSldWAUS6f4eymTavIjnR/tDd9T6kUlNyVwB6GGp/Y69LfqpG2pBGbE1VSl0HSBAIm7BifOQy03/VYpj8b4EfuqBCmfhhCtZNAYhb1T4PXtCwJ6B7SZFZHsiB0UbeXGDBehZNlZWGQeZzhkbl3X2vfQPMfbmkuO6xBRnL8CO9fgbSqwlvawA9AbB2wqqWlB76JS+BQrwjVGFLGH9bCAUA1Ssb8h2lOUBtU4fJbzySKivBfVO1B1+d2bOLi+RQ8wn0iBHKYvt7aBmfOjQBTAU9tJcvXeqAVMFKlm6d81tRrAD0l4VWW1rp9wkuoo2Pu6Dmc2vKnapDBJWNIlfCx6+eLHmhFQeBBx+5e66ywD4COb66vVrH3DNjjzFjj4F/fgi2F1wjMgtbA3hOu1XNrjpHd8ZvHR+GPJeY4b8ovxqQgF7aKU4Eiq1O0IFfy4W8oTgc4aTAXyqAD1uPYa6BipUTnTgAFe8lYJs/OIUfXa1Yrf1pLnh1g57PWUyUoSmrnWYSumd/vjgDfkmnKa6gpkgL53AqBv8PAvKrT9GRWhWDZ1tkPSbIo+Rb2tSBZTvkA0gRCHrZi8V0ApZ1r4b8GN5jZaxJr5t03pYnFgppp+Dv+HahkoHXqNSo24FwfIPbfvITM97/4jeIoizOTtVBzUmjm/MT+CKMjvvMn76Ug3ruSBnYIiHtNmDS3Aj3rcjyEArdxMTKxfCVGpMbwwfqW+UtyOY8kr/cPdzCwJhTBfyjqlM5C6AeYzDKwJwwcN/TLjlK5OKppx9Ghwq5lbBklXW+SzfNcP89FQvqHxlGtfqRJS8wb4rpNOvCnJxyBYlTBjbGPs0m3t3NyLEdVm4bPrV25e5gXoo82R4c4xqShAMLfcB0XqB/y1q3wcAEnHRpk5phWE4o52cEF7cwewXqCqe1zh37O0aeBgC+8rK6NbL4uNvr7DkeGDQuwl29lQjyf12T8mi8z9tLDY7mWXhsHLSIDBNJPnZgRmQOcWBbzgtU+KzlU0tIcCwxHdUm/dL2Glf/9qUnahm8XEm0XzwIRtHR4UiH3PnXjWrGUQEj1FQtt4PK5XYCPAU+zCNQLQUlVhzzc5SPeevBn+YmVqlkQ4aUWHSi1OC6RLnhX+G/nmZJnc6hFJ8IaqJcGpPbH6Eyy1Ibr6/vXhxLYn0uu9u77cZ+ojmZ6avvsJ3GLu/XgaJ/oY8+gCGX2sDOwUNfWWCBsEMM5WONNEbmDWt7yNCAumZpM0Unsse3dWo3DxphDNzsIOvi6XHfoF8Ybu7BypUR7yOAmxq9RgRzYHd/YKCzy5T5WGbbbbG80anU8owkzZ2bdpDcKtDzGBMENm8vejx2YLr7SgHh3mvoCrT5xtdwNKH1fslowW0EUvTfcqxaa7mkmZPfiA6K6eDeoH7vbZzAfeGpv/YuTGlQrNYVpDKvDDnJ3sMC0xYwmR+u3Ypsp8WmbnfMn+FRP/x1u2TfAR5kHFOz1sqmyEvx58qRUdfbbv63JuJ2XjxWffhsTXKWAd3H3AH3pJXRR2HZtklddNtZexhS6y9eSCDpFXKFMQ1wpONSNWbIiiWjSkrEUwY4TlAPwOD9uJVCr9acsgWfgUWnLQw6sd7HyVxL+fVyHDq8WL9JfJf0b5jJsdOVrZPjqkYkI1TGfBiGf7R4k/9bXZ9vgsMcJrbvf1JyJL8R5/6lI3eRT6IfDfbGfutvL7bwL/e9VzMOSYfd1U2Fa3K6A2p25nuPvuh6UGHFhkfMXl3Ip1O0qk5qY/+TI+3BrR/58z9Ub60Oj3LHdpzthxDu4t/rzY1O6fCzQvOOWgsRnqwT9PgeeWLDF0yl77oJg8WhtOKYVoFK1JOzUBXeuyWmmSXmvMbgdLTfubeG1x3ttX2lJhntcUOZGhGxf7Tdx0QkhB1eNZx/2lBSr2+35LJ6R/IGoJyuHp9BfgaBz3hvoS09SaicQU6yfnKmg8v6r5OtrI1rBbJupaBUDs8tT2OFOZ+B1vw6Nt/Yj9OJhZsg7ipEgwHzZ3L6vTwqnAfuIfuRlPSVVzPmD3JOXY3lHWHXbQaMYfIh2fwuDPfhbZWXezRUT7WKK9z7R3ZUFwHb3bFz1EdzbnYz86CeWwt6HMshG7gDTSc7qHqk0EucTIEKUFDZRHdvkLdi5IrNV17QuxxUDWM6WkMBcfR6vWnqmPdTfzJXcyqzxSbvgtGHlTxPMbgvNUtAwnMsYn3DJtKqZjX1qUUM3sD7t8x61Im1Bl6Vc4K60GrDBnZY48MHFKGRrJfTkNw8JYuu4Ydl8RFc4xJhA1jeoJOwlmJ4NDCXJtU3LnR19ulNWhf59JPdxEf8LTM96gi/lvMKcYaGhUV7b5Y3XzHKjWyEwSXi7Uc2Fi/KAwN7fd5g8dZv+EOT6NcAVJrFeozYLQSw3H2Z2vKxP+b34oPALvA40c40V7VfGzwNV1EBDb9HlvupL/SpuySQZFhZhJ2QGiF0vpfyzIBFiXYtCyO/f2QMMxTJFDFhqMn09ZwqBXE5r9ZOe3vPNdAKkiujNUqbbX8Frt0lYFJ0UX9ro+EMMH4qMjIcgMCazaWb4b9EFoDDOVUb4bHdRP9xCXNrxSqqNePK5/BGIHB2xJ0UqwBahDugP6OIkj8d4lEa9JKq4Iktbu0aWVDwlUXN2fjuWzkKkO1OiyBlNHWwEX3HVHGFvC4xNl4xKgTIo6g6Gcet3FdHMS/B6LJqflBuGsR27zLqfmgNuWyAGxDrTGQgptSr+tKbf2h/6GEC+1zfvnfUi9ebYgczHpF8O02EX6rZaxQBFUQZEf3urfwBTjfTmXiJbVkBs8OoQQ+xaDWZ1iqVEr8Hj0o/aDn9tRq8u3Ys/FWKmMnqAJPmn4dGJw4aPZ+GTHWOa8WdOjGltnPvW5ePo4zb6whp6TaRzzJZx5xI2HW4i1Tgh5e3gSd/l3BaIbEB3A1Hl/HZEKqOmwBfEoq0Tq8l9QCgw8a33X9WRPgd1tGL2RXnjQS7/GnAZ02ukwqQ5YAufsx8WO4rMHlOhOvlvcZ6sTKkMyGoVmbXEqQZ5h0u8+2ZnYeVxiuU1uKeM3J28vmGlVVdTNqOwhW5Kx3o1kTOR73lumU1Y9aF89g/ZdH+XbvYLNNO2HQ7HzA8YMpOk+R3Bw+qfbDL4Z3eRCiA0WZmCS4AuIZ3iFaKSiUNT2Gal5yqzY40l/L2iVD1H9DZK9jSxGNbnEJfaFBCpWQyt14GvxpRgkDdq9khYIAYS1AlXEDVXbVNRZfjWmtSTSjiqHlJJZgB/tmWxWrP+hEtxg6gIbKVmiN8ct6TzIM5pEwvybLyj5W89xyUSCUuemPHkN2e6gbgbx9XU5iSS5BRcTwSWmDYbUl9Qf26uCX6Ez3Ly09S7/GLc/Qod9XtFv2p8kBj8lmmFUuh64KUKpc0jcX2UZDuzjT9XCMxSmI2FbKPUWjdr1g3hFA2lbHMl+zs07nkkQDIY0gYb18LFeebRyGdWukNo8CbtxbB6MJ6FaQ9kE1CmmbxqrYwPhQ1DrFlSPlnGxeYWDA3s5CgUVwkRIyIgN7vvO48nrKu6hF+Mm+OSDz4BQz14CUNnZNeI5jsrDnbi/bsmW3HjKD0fb9oxLj3jaB+cAEFAdyFLnIZR7UTdX4qF89hi8SbBunV4S62AVrZMk+MXAfqrbf+N1URPHv1gVz/g4fbE9vLtrUxrVFQ1l8VBf4XjCmcTAEvDjj4ohPCRRxpv4MFJjDxNSXqpKHvvW1jNWJoMh7iqlVoerwmn1cl96AK7sPYEHsaeOyATDHzWNrdW5+a4nmQRLPjEaXdKB5oh9NHcFXfwAyL2lkIIifSN/b3D7MtbNjrvptqRK9Oa8p5ssqnvgdp8TK6S3lxOQSdHlrtCNQeRwcx3NAVbIBWBDvgsDMgsYQvC1BgfXXPNdi99u33+cTkdp5Z6vS/aVC1aUpOXH01KSLX00exVL2OsIXU28KNrJLp27kxtqpgJjZsqpRr5h5txhv7NyNmrMCSChZPTzgTrp7UptVK2VJqHWHPSwcSPFTZdrCx2g8NV5GY0/xW9+ct8JsAF2zxS7+yjJHGhFIyIvgqPoJi/GgrI9hbl8JjZAz3RMfZlC54D9sLUXuBgSNLHYa5K1prtQ/j/Btd6ML/hi+f2YdmQh7APlvkBUFG7bwDP5OTmxzQXTyR86uZ3FRTqX0/YEZIlQ0h1mQYQ44n9uL+K+GdHHE3H/i/BDbBifmRuj9crl5DwSMA0KDNmXcp/oL98SJAhNxsxw0crn4gQ5+qa/6n/4VjjEwrYaohlNZXwSQq3oRkM3FbqJzQXwTAytl9O5NF+jr9xOa6o9fa7GxeIPCrfgLCX/vnm3kxmTuazStPxwfkRp2VAMz55HdH+O6oxSvUUK6MGHOE8b2j6SoX9FHUQhRbe+RbAM11bubqXx3mVqkLN2YgO0umvUtrnWqMOwdnEa5fjN6cvL42ew4HQ2sgdNpmcIlKytsy/yhSVpvpeTlI4AsNUSIuSPkpMnVIZm1HNqJDLfDXSLi2vXm+qWoNPrzDMrQH+x5MGWHhEMHV3TwAWOTuhSxdD4yjoVyvIpMoVvI9VvT0WHAU8qBNpOeHijiC+Hhl21IpOgZt3nTtiXbz+lp26Ua+kjPVxpG4IJNLfp8TSFiKCtB1uzfqdZiDYR0ybTwt8N1znAQfjTXXzmpZPhqL/7lGmJ7HHfohQ/KfowK9XhK0CJiOUxdvdPSWBe8rL0jKzXvIu2O76AzSm3V/rt5fKdeU2OnHX2mG2dUMhmNTYJUDhP5lxRwrcdWG7C+nJTXKAWSf5afz5TXqs4D8dP4Gbc9ZhQa7/fhGqchb6TGMcRndGtRTGIdy3PY6JJU3BxFfAy56r7LhlBBx8Ze21vhQ7o4sZ6lnmm6HBYdJvkPYx1z5NAXgK1YR4dIK28QZtQRaXSmTXi2aF5ESBJOqjQdtnVD70iaNdKaRljtDCHHnK6g2HqAsowG1Hge/I8hNhCj4i3+lwMlFc/WQv5etl1dJdTMaTp1yWQmc3mM/LGoAI4sz3t/GNQxajx8YLwPwsKiEP7l98Eixo15FWmn4GEKhXiS2xxTJOzZ4/IoAnRsM11CxOg4r3hdwBRu+dgJVAmrAGmL5TBgq4Igk7KzKbqbvKr7bF/uEudlO7jafhCRtULhs4YLxLGt1kqjHtXIqSzM2WiVDSP2avkvbQyk2jgvq5/sHHDUeb8b6VaJShnJR1lCLa6rjCpDZr/mHytAdcNFaRijTqoxnTfrVeE06OKrCXPkaBwfUGnROdrwZj4H3C8S/WmCdt5rmZ2cKD6L9JAsH9lTP5/7PxOeqP0kDZhvd0i6ZLPS/pTOHmZvH4I/6k2WxzyalRk6HbgJmV9k1Ok3Owi9L3hZnbWVNvLE2aLUGfCD2R9CTsWxKNYGavYJfYN9dxN6kskRHwEEDpzhA0XmQmpAdFHrgkiiomddFrPTLFdC7NPHPAR9rYGJpvzaKB4y8UKqVzfrmpWG+OUqQ5KRbDNVk7yjb9BDbP0gx0J+dLyVa1/WkyxJWCr8lr2kr1hNQVPwRB5Bqix3t86BXeGNZnGxtx5Jb39rM26eeNLBnKgxFbTqAHjhWpdaOYZeZ1Yz2VRumWk2tg1PJOFf/NODnCkRAIrG/FSAUrp/xJ7UI0EDTqGN0K5IHPxZJFfo4LB7HNkWm/J9a+/9W6LH5PN92qgYKbKx1xAm/33U2F5TX3xNzBNm5G5SJJDK38hmo8tNhUSKR3DL0FmEXEWPV6IHZUc5EW4jqDO+j7nDzuqlpsDqCuh7vEq3jqAxJ4w8GXPd0GfDcqRNm/v6R2+1uR/RX1gbfF5a+nJj0+NCqOGm+b6XXkAvYxvR1/wcQ8Hq+Uh3XsWhxtbhDUmd4XMK92fhZtRNfrSeEHA4c8G06PVT/SWbA9uoQi+JsrVrZGaUuFMhR9NwD2SNXAPAS7XFZ/OkYe6ncyfgbU9hwETfGdJaxadV1L0Hm9zkBlsYm9lCrc2cFVPujPPUgXSK5CPDcrLBStvt3GnHxDgwyBPhO2Cy9BVc7IKTXzQpazYjiyZ+zeH4Xf35rXFx+/fqVWEqvTXgpAIaqCfr8eVMI6nyU1TybY7VXoNy6nWw6mFiJ4AWjpsijd4ZSzdf8u6gZFO8QMjjuZxKBMhV0Tg7glt0IiOx1ALWaKXmxymfEqj9oc7Hmq/BjlbyXK5+ZAiO/vDzTRtk2YNsb6ly9LGg/Y1sk3bc0pNXmLDzSBjlLq9bOQgnmyT9bV5K/SuxA5nhY3s7F3rDFW+XtYwqEFgqVpQf9kpGMMnLPFHb9mWV1WsLPF9MaljK0uZHG97s3JoMh0tfh2lOOxSwB5HK2Bi3ARCrW6mlLeBP8RoObYzwfsrfyxpetiTdEC/pNpPJAVtWBiqymyOvUbW5SOUQ2wFoBo07UnaJfgDQo9zUdN1B2ZXwLj6WjUwfsg2ObJ1M7UdJWn6iGkBU1cz6g1ZFY2D/9cN3bQ7k88Wa8xxZ2Cs2+HtBJx2swW08e3JQOm5RY9qIJ7cJZIHcQeCi9ygz4VQbuJ+7gkB8DD3ZtJLLm6MnZGl4zLnsWuPzBKeXKCKrJNCiGK/2mpFWp2QuXKVFiUNKALDz/tSzKD6gfXY1QyOeB2+FMA8j6CALpe2Cl0k4DJmLqYmexgPzRCQbdG0N/2G8C3m3mYtH9n/qECnixkjv/gUHi9iO+NpXxqMYVzIthGvr8yvIj3NVZlg+VZUF6cBcukVnhdJFx3BKTeqoW0FKHozQTmLMUhBKGwqC4nL48QqrzrAr94tNRXG0/yHlWtKa3crrW+RAMZ/GYVc91VoE6AFIg0D9RRA1lKjjQEPn4mjzs5m6smMyLabZWHedIsR9WWDTtS3pNZpX5RrGtM2r7mymj5tPRpXkPx2d6RR4PK3MqwBDOsdO7kpOt30RFm2kn94tBTt61gMRPgDqaCJP6nQALXIpNvm+/e/0ywhZ5Cc3ax5nMy1OT5cO/g4nMLf8ej2OBbHWzdzofrV5WcNM8X3OeJ/b9oJ6G/mCFMI/RH3Rjr+Ek+fLWhGfVnBdfWPC/hjrzC2w54hAXLxE0OHU99NJyNnvRtXe56ZFkSVCFvmgIgntLPONyzU9ouH5Us1vkGfwfNvs2d0ykcqQWIOcgQS+e0x/t8pMkx8MWIVHrdtfn6odGJ4wzmFBAXhaRV3/XMAQYrUFiwsCFL8I0d+ssBK30L+bmrDTMjKqwYswnOaZTsw1hHBbyQbD1eyTUVS9vK8F5BgiJ2yp8pKqp1XSzjkJ7oXwsQeQOVEyRqR7aZMvVIdEDQVt4pG/YpO4oNptTD7ZLdC+13Ug3C5T3N8yQM5UcYu+rIsNobvlgHdlbvD/SeNAjhnCZOeyKXrxRvQMbOSsbUVWcRkXK4aSt12ldkNtvUf2ocUpnRn9WbqQyjLtAgpy+vA8FVMEbixrl0AqIYl+jqyI7YzwbKvsHDQaVnPBFRKDb/IcICmy3lJCvQdZXJOd21dvV/8e0ERSuBpa9+3q+vnrrmf0e/Gg7Ma5LyxZQgOD2mT6lp3bJz8OPgI3yzBzq9eYA8tW/c+tJr4iBaKYk6dsHZbIqf2d5GbYEWYRmR8M2HqaPejaAJ6i+jOK9vpRGTIv1ltOEr9FRIkv8S/Jc/0gs1SX7HTEhedaYNY8q0uKwt5VCbb5XdV3r/cdybVgytdsHHUsMn3G64uLkpYNPE1hBcAuj6ZKKTdvMcXSl7TUbvV71a7jAIGbA7LHFF/WDE8jb7cozBx9PZ3IIQiKrIOfPNtuxavWKEiX3KJR3tcVYwZ/1u5LmWNhl/ahlq4bNuK8dF8jJwLqi/QZuywuUdNO0u3LGB/AOyHSsX2Pm0yiRaT/+9IUTNuxNmzaSZO4yE5vv7ccfrZeZ5u4BsANz8aWRTmYKHjmqaDjd2+Y0o48tjtY4kMgY79CQuYlt2nJF59tIbcSyYYEhr4JdcZPCMhVbcpOuokgDyv4lvS0oBQOdtVMYchbtmpQfEkC4g3TuYaRkHSK3azy2zXi5lIeUzvYqd0pjRRjNuWtXXuWMqNTZB2Jy84iRcg0tZ09eAgkbZz6BHJSHPgLyNrxyqw1L1hQCihhi7tjfd8rYfcNkGbKLV0hpwwRvqavgpgE1lUqQnLO/ZUj4SaJmY/eOUjFhdOVOxcTvwXS5CVkShLLmpsdWFEb8ZOLoQ4hwnJyI58HxkVZLIbv82Xdwo1PeR3CEfR7ltXFSl0AyvRH1wyWiRjGWbDCgIHlxuvmsNejLwV4vWv0+JfDpfqFYaoNmmqPOjsp+yK6J2sEOxjfMfP9b5IUQCeIdyBPFEeofqldCAmtHTA+0ZSrkhskyZmNN7v1jGhLr1n9sPkZmgnbEjT3U5T/YsvdsH8oPZSaPAHMnYtC1f8a+JMULbXEgPTvLOWuqSmIfLXc+GPcOYXZeyRb5evM3iNpNVXOAmw/uQe2pHfuHl+b9PLqXgcVJyUtX9qOyTXcVPEcvYSIhHYbFx8YqjMbPePZQpJ8xZvXJqA7uj5pFoaDmMZu34ok9yKGzXmEaT7w1RFL1L4PjDMHR89ihpYCbfD4kL9WsdEbSn6DJdUbRVPlbcYf5R0oI2pemHEb7N4g+RvOHvtFnJKhZjhq2lIvr8vx+L3bLSaAKWPEIg2s9oGXnyfYFO4dsbNsBF/SLm2ezGkhcWGwUl5grTUOHr2bbYfjojaRj7y8M8QO2J4Fmw01v45HvtcbKKd4xrJpHDAeRunN7BD/GYbxzzzEcgnlKO5MjtcSg3pPtMHtwPJu5qgzLz6DCO7q7lVxx13QkNrbEeDhsG/Kt9SjEaPDwxLsuXP9mwf7EokwKaOFd/T2DTmE9BFfJBc2zM0dC8VdtqqdC6/4xmwgMR2BJMD7FdzlRaPo5IBpkQezPSlBtItTFtm4OEsQ2BhJVctX73M9Ht+tpPA1sWfgxXDrsHeqwOLvNV/zyjUytobIONIOsFJsGmEL+/ApqAs0bAs5wMV2Yn04ERjBZPPoSEWhNrPNrIzz/pM+1L4UY92fTRWWLmyeDP6TLWz18O8VAOkBFAEBsSsKdPHHkJdL8R+RWa8y7IBZiiHrzB3BZ/d8aOgF/A7rw8UGEsXZfgupumH11hOnA1ncz9MHx/Az8WMzDoBJUfiLkoJQmW1mNZu7uaN8+RZpjG0ocyLEtvDHFGSSwswJNjy3uc3tQJCj0tEqQLjJEQH/AqCfd/DRyTaWSYzfoxPWWyVlA6Ghjar5kfgvJaLfaWsN47FvlvqXsUwkYVHdC+OrX9b3U8DzzjT1hoPiJWtPo3WE6U+o9kSz7qdd2TtnNMfIMwVGqjU8xPQQHkeVY7MBrRZqkAjRcmmBmYdPTJMlt+FP3Sx2OTv0RytfjsUGaR0cL1RWz5QeURq97RLUQRu2eTd5slwG5DH8Tb/NLN9u5H9lSu2p5X8tGE7o2F5nCKmN/QUEEvqSRtRd1VkRKQWaeopLExzmyrAv8kFpg8bHl8BgJ4UU/oB6n/I/nG4mvRS6ZxVN3LcCm8CFm5B9n+pCNVk2rZP+H0fqY+jvJXyQvqb5Fu4TnXStMyzOZfR4NVvzOsttwn6kJxCsMRcIgK78zx8s29EdR9T2fGSyA+mVNq7/bkY3tFaBv/0tTpsmtxuBdF9rDKg2Db0MpOyQGJB0SSLKzahYDAm6p2T3ms5ZcncdnChzMY1Y0rVKqUvyeRxeM2vbsGdP6Y/CF4ts8B8DDYa/GpELgCdjtsKBIntbO0d7fwTNOq42t9eHAxnLEUqNmXr48YyWDzOTZkQKTaysCjuU3SHiGnmKvUdpF/wmXwnptxiwIgJMpG9UzUXgoIrFfAQQ3nJPNQmnKHyA/ORfFLZvR7E/44WWNYI/HzQGbLDzvgSxCa9IfaWwWuQsOHwfrNQaEHAOnP6I2jWzwIW/sFGvUjRvXZy3PiM/sdnmlvDNSptOmnh46tChBgZgtfGCVHQSfLOFHH9bI9u76uvdkL7uW58eAzJqVRBiapYHMYuWcuFR+jPmIzNNqY4fIJPM34dRcnB8UoNMVDMIHiuX+MQ2k2M/R/GanvRk51H9NNWbNVB/uD6DZA4MaulFtTg11ldr1DmbGqCMVf6NNDWz7tdhDGoXZYWbnHqXq42t/7rbi4iH3YXwk/HPuy92fhYJuaiATmsXDPhM/KOUjaD2C1+HyDc7yqk9mfzaZnDn0Ez8Cqi0AT2QMnzh9O2sBHqe/g+X7Z5ELbeLJBV+yY7/Ws7kXw3G2zjbRi7KGy5KQNKvDNGO6hVUR+rx0jwVzJtfHzlnb+uowV3jKx8yOD6aVBPvH+yQ1Y2UOhpnV8qudaJvuXlg7ofm84rwqtYrsXs7fyULoPKpAOwnbt20FnwhqcLeLmdIk99yj2tLr7RIaoiOTwEGYgtALxN5vsqnqlIdH0Mhp9yL7Z/Xz5IsfrRd8ehYJ4BUfWG4KDTQTBv8T/AJF+ePm+I3dr4Y0LDzojOn3q/c+nUc4uRFOWIsiR1lxPhnQhQsWwF+u7QtXU3U48QiMHLtMe4XXyjFAJRTHOaSYLdMmuO8oGljAS+vNUEg8Ch1HHsH1SbmR1zOyJqaKMF6KxsAeAVmyewT7tFPBT1qDVe/WGNYi+Hfua7JpNMn9hIr+AGRWGrHwPoGBbYP2aDshHp43k1QeKaMl8PHlObjiscwlVrB/qS2kq5vvrPmDlXeU2gAtbANGkYbs/07IBk/DTK5mtm3J/RcoTt4VvbvNzw0O5aypGTz3j06wsLw9CwcdSjiv6F6v3hvnAaIF7pc0Gc5hu1bU8qGiIk/e7k5+uJzuCcSP2bDNUBfsaIE16PeWC5ShKQ/r8pJjtbP/gk8vO8C5d6xyNfqM64lqR7jpuE4HAlHF11+vKClxCcBK5WiJIGeYDr4VDO4o5eDhwdMfQp6sM1MrGvgi08DYjFf6b/23ASYEnCZjFDH4J0LMhn9ruyRn0L+sjBnTnyanTq/7i7c2NJuiNI8/JUKqDDUtNwmT9m5EOjAUPZWv5Z5iHVqtEgAJsvLHYWZv6PwjSgtGZt8IGW0IEYJsfNpdmQA4rg5BYD/m1VDWq0ZwxQTIdxGUhS37JQ1u3O4IVEKHj7zcobk35Q2mOfvvs+OJ3TGb7aS6g4Gp7UBzAo8t1x6Qx5HuZi8hPknXS81/b3LzcI0EmhPM5P5R4BTfCqF/XyE9CYwZ30o5TXpG+IaQXUpcLDcfgCN9kXc+6NmlbIb7oDnCrt1Oi9G8CnWZNmpCtULLjQIfmfUKNvW+5eEK6xGRcyIpxngsRkAiLm5wBl9uD6WRfECL1WvZyfRe36r35p4ssgcmnSivkY4J2zXDYzDCC+xOU0AfDoXq+Nqa60TzMq0/mqAcl3Vtw2Ej9cHeIs+j8YF9wAXmujdsf8XHK+DaOu09NxFaTwNwlIOGn3IdX91XKGDwNRpQhfMNq9D0vQs8f2q13sCUf1PhlbFYPHOOMl5OmNWvbAkXzcUtZfpSf8L8BcS1NBizEv10YbgoRv+7ec0rIY6I6XgvTsbCSsaUmxi27YpxtNA/PBP7XdzOzoSNbMZcULaG5Yvt/pKMPJ8Vs2R3DfYTQ3aEPd0+8cdqB+Ie3a8hh9YS00UE+V8+1ibvTuf7LOEN2xBjMxKzvHavmIV8LKPdnq6FNpyN+kuEL9a+vEsE8ybsfLLpKFqjnmLamxB/b4pO4I9nkNaywupM7EH5VP4LsuMEb3fs5XMdV7V6QmJFtBwaqkJ4m9CkLfgTn+1zfDjIGwBuuhDwwXzYzdLwPXcbOjCw8RF/dzvNkWYIIv4RbvwzRAhVzIDcdPMHKPKkF7phIAbhLiCIIsiS3sbPEixaflopMsnOhpvDhXOLxXm7DbJCfdB/X8bgPMCV7V6C/ANX3Cq26G1yWgWsy80E39HCQ+x+dhAc0ctYN1zbyqCCNHtcgvH3Zj1YvW5dv6mL8UclHPRxOS0dTC0Yft0gd5EtajIo5SEPN8/hr9MYp2+YBdKPVL/U8GbFqmdxKubIybKBxqO+jjPCrzPH0ahPdQCtVX/3y6rp2KmS4I8aQaPqZMMaCuTsHgv1Kg/X2rJmBqZD+/G2w9LpP8r4lOcdiQHKSc1TIS/4U/6y1hEo0zR4sUt7Ge8Jbf6HT16QC6OvLcT7KPYrhTP+dRwb1CE/VKtbnnkNr1/tz5sWi9oPEjR6K0ggdkMpNeG93Ot8WbaLCxrcgtHCMcyKBe4l+mbgyb6QKRvM4c2Y0m/Cl+DSlPmdAD/oO7gz+UtHnkVDugaNT9gJq30Ohl3Ohav5BnTeE9bXpa9Y/HHhyxR3Asuwi8MG8D7SU/zgeeXVI27u8sbJbDCTskmrUTvnZr8ilwKvnZ8ZdXwoxDX3vGt9ra2e5mnCSM/pSUMl5NRrz6W/sbkuR8iDI6q1XpcQ5ShUVrpV7pldMSdHaQPHyfszyOcVVuJ4KgfvgIO7XdfqqLv/qMNv75FhRm3YUnvJlKw1a19jOEo/CLvq7r7yZU28ooPmTufUyfB2PzIiTJNZumyLlmJ/GB78oRX7/8oihRlpqDvO6B6WKO2OHVkh23fcVBM/LMtvWiCkaS6zjDrzXNehZpbV73LIFlAPcKvDOjbzldm+ASBWYK/SSt7Yub4qZNoB8q5Tfh77bA8fLA7HjyJ4+9h0FpZZWoDyKaSS8yCn6KlJQCBQ5kNZz1qVm3OjHL3BXvNdjYYnBimhMOBOlgc01MkUcZHOyJabb9F9CeiV1NM20ETw7BUeyKR/C6eXDKLqUiYgg3DeXWkiJG9Cd5xBsvhMNS7yodvoUdqOk/9IxKKzaXCjekyxcGtk1NB1mR8GJWNPFZhdYHo26iUYsI6TJd4uSMzzFcwL5D7N3ehHHSmeS9Ko00X4Cka0uqSQbtYNwyNgyhMAGcZjVvDzhzJXSBg54cLiHRxTdj2fxkALIcNhdpf5UiAPYiz1M2qeH8YaHY8GBMXYsqqcAxmYdoR7EpdN4O4GMnq1AVzIOSsi7Qyo1S3vfa8xbxdp1HWEuwNRU3Pq0s8pIc9gA5gGu62fuXpUkHvyXi13YpCe3u5ft8ScSPIxGlcMGpDVmxxObeyb8l4L8YVSkoAN+/zSTLOPBC9NIo4tLqPsa8+CGIMzaKsk4yBz7zPeqeJGnFS86dhqH0b8g0zJvStCo7s4w8bxO9FUTqJPgc3CxTihRaZbqSFYpuKVe1bXA9ajw0NfbR/VS9qaYb8AFjK/nLTYmmQ9V4qyXEwlxxq95dIgQA0TOnH7OnE0qtZIszxSMj9nAN7G7CUgB35H+9hR4i0lS/4qiMQbSTb1NCZUgUsaf8UNtB5pPe7ZdWX5ziKt0zGD6GcGO2OMZiklDXzQwf7TC/MmeBbqrVsPkCn5ryoQ1AozJvjm12SgCSDSowWAQALN35Do8GQmdvcsc8fkX2M3d9kBWgrDLGnKqGHR9T61eUxY6ANbz/lsCpX3zzeu/2eP8+3i38DQRjv/U8VIQYxlB4HU6gXUkd6WBN8tGtrnl+Vo5JLtwh/7T96xaqsspdQ4yZq8MUEm0OpDrbfUMrQ2WYzLn4RqGmCsNj+H1mLuUgC8mvYA2aMvuJ6p/3/DexRYbRuQVh9WzCIsuZ1rUfyPe4EQ6zK8ezzgQJNJ1ofFG4EuGqRCHxIG3eXq7slrMBIZHs/4zC3+SE9RWFiU5iCrUImTTEo7zKJBNsRTxt2x/TVwHiuljUHnfQBGqp32Z8NJ8WFKhW2a+LzWaBF3GX8rUw7hbhSwxrpmqWGn/UIKKZjfwgnTXGLYLXJNL7r1gbZXvTaPnUoKr3iRsDx9QjRLjhmYWDBhC8wYyy+fqLcXjrWsmY1lyGKuDz7r+eRvb9zgZyBxtbUxd0dI5pF8fH5JUwoJ5OpAfewx6wfGB8ZxRmDYcX64JkkpP59kY6zGWCA9fo6D9SIUsLi+ku5jwwHZsuwrimzLmUY6blA3BwNV1SzyUw3xnYqxVnJ5ltR30oNGqM6FNGm8vSf9hwuSEzA2ZuTsq8uZUZ2rSLKSYdpN9k4w7WL/dHxo7fu6KMGNm5ZS60GRk9WiDBP0DKVmQ5rZ1bSgbuxB/+gbqJNeZEoyomvyqgeLQTv1WYOLDSVA8GLSwuGWw5o9aQAAuqAyXbZD6Pki57EBdTFKP2Sj66xp9gXPK1+e3TvQEf5sQO6kH0XAawj6nSq2F/VP+0dMYiRfS3sAbzVOhUuQakE7tTr/KU3/CfajYXC8iP3WzzT8EOlJxumZBBXOCiRjh6M5RGy4J8AvCdPcOB3KzADWzV7H7A0Ff+IFd3vjoLLYj9BL36CHPVhAiJTFNAuIIeoqpmSIx3b0wTwKf+GHEHRVbkf/Du4tG70rKtljp+D9XfxXBR4N6u1JqLWj9gD/2/h/e1G0UnZc6zVqtZBi7yDpmrwsGXXMV4m2aGJPhoJzwcsLgYH55eymLRhDIWAzD7xK+hb/YzzSUUlLG86PzUUP+QHS8sFMjeTBpqAi2TC/OfM4RM6aSg6/bTqWb4ozR6mEqSDj1uP0Q9v89fA0lVPDhsWB7ACJkjKvYpM2ofwaKdMOzkjsyYqFl0SVdAyLmXrcXVKoSHSCuzkrrYF6kntWprxEOMSQ0xdF/TjkFtk7wl0YgJjIAAtuFmuaaDyAWHH8bz8ytp9G+e1Yl0b/ZnUzODo+RY6GczCf+d2dcxm5OlRTsn9Hd7Gc3FX48fDDGc8O0j6XkTyTXE0OPmNOfd3swOk8BxYOpMwEC4FajK4yv4z6Glt0bTB1rUiHB8rmw68xy7Yo0iwCFT2VXs+1J7VrEb5gjfT8AdTjC5I6xDTnvXdNDrLIBBD9y5ONVTNH/LTK8K3tvJXmjUCXnyNHSy40MfsmL6tbAafFTMUT9IHqnmlsxjdRIBrbrH2uVsCgqRl+GcsyIuJqrbkhImemF7bFkP6MqK8DgTp9AjcoTzEDHcoXbp0dmWU70MC7rydVvqmBzS5b3HCFxDPUSSDnDCtvO6YxJjq9SN88ItsDGHBabc7IBWazGFQfgvJ7LR7hUIJzs/1GrHEgsE4uvr3UJXgwnG5OgTFtHtOLbMQahZUfTR5DEgyWD92BI4O6iW1yATxAEX7PdYGFQtfmdcsLzRw5PRrZejP3g7RxZdHtv6PWhmgIP2fV6KTUsvFNUwg8Vh9vxWKeRtj4jRlHqnqG34VSI2K29AH1kI5x80oaxRCgvsKJfZ3ZnMFY1UJehGVqEycyY7mNb8kzCtp9JJ7xb0E1ZGOZ8OaPOSJo+e8GtGk3rsy+p2JaOuGRt9DnIeAONWEhg/WCoh1lsbYIpFQzkDO93BHtHO/mJa7/kR5YR9ZqNuz4Fg+IdJVxuWmhMt+uGD2rDN9SqZgupqLxgkwDcXFQB0gjGxMv/yUDNGt0H5wd5wKUOG3ltTIqiXq23ed8+HtT6LVwyGoAeWLKD2oZN55PiTFS6KL+a8PvxMl8XCgL0jCSbBxqOMR7kvpbOXj0AUKAI7OBoRxz02ZGJWpYkfeRgxC42akqWsP12Hpv/kBfW/PQ0KsjPw6gpF95X1iLRE3qBbq7tb6hPWQ4MyWq+RZRovx7Vfmvm4TsmBExrQ8ysH4dXn3kJZnaTJAFVsZ1TZpYrlh6uAGzu5kvEw2rmeCKuiWt89RIIWRhjq7QrWH0RC0HP5fmOgQZ7wLLiwGnfWMN3V0dCOh2jjGNYWTAgj0L+DTkA3B4LWbAaVqCbvm4NY+cQa+SKAUrEwNQnvRmwWtFDzeOtpjQsVpAwHdSUxk8B6JHQ5LF1W6Uh0VOd6ONf5sWFSE/4aTyn2RkVgK/xjg7F8EF9XtZvhqQeUwgeY3gzlbPW+ljZ+CLgR8Z3lB1VSLxaUQ5+VFfU/tWaRE7lQTLOQhi9Jzr4cuZvbU9B5aGkZUzKLxhgzQiQc5A2nlq1/t3/LyUK76yImlmAOOWYpF0TtpqO8EbrRCtSifBqPZL7rbxnvtT+tTNYlb4v6mcdQlOWkTbPQqQ6MOt+hkzTbT7Vq3XLDi6ycKu6r/UlPqkKoZ2lVQk4hUUU/n0+sw71NwWY0dTB8f0mVlRGzzrGyAnUB2gH8Bcbt9lSJ0vsDP9KQ9ju58+0XHSh2HvWoT4DUn6odV9Hfa5gAAJkFcpUU1QIUzXuN1mFZoIAAAAAElFTkSuQmCC
```

```
8DwBunnXpO/Y9AqKbg9Ysn/Egqfs3lm/
XbugDdLvlFWbd3YhTeZn2alKNRLTsS6cbXkYZjCSsmcfOXz+oENPphleIbN0fU3Eha2ePJ/
PoDgCaDxhH4x43o7UmO2k3a2c/grm0CP+olnS9dRJGV3fMJhPr5z1NxuFtdm5ZzYU260Wba/6LUgYOr/
+GfcusvzrAO6QhcTdNxi+qJbt+mcr+FZ+Wr+tw4yTCWvSlKD+DZZOH2Dv1Wrd+6yHe+2w5ZrNmaP/
W3HpNF6v1qzckMPOrNRgMCqPBwIOj+9KmeZSDsrW9mU9PYFi3XqRlZdL69pGOTqfWrd92mPdnr7K83/1acs
vVHa3W/7F6D/N+3AyARx1XHr2zL7ENggH4dvE2fl62Ew0M69eSkUPb2zv9i7Z1XxazFh/
GbNb0jwvlP70jrdav2naC71dbvrndw83AuOENaRDuRXp2Ee8tTCA7vwSlYGDHUIZ1C3dEE2zCmXo8anQfD6
XU67ZK5NyvbaDX6Cn8Mnkk85/
oSuNu1+If2dQqpl67gfiGNeLLR+NY8el4eo+ZWr6ux6iJHN2+jPmPd+Hrp3uSlbS3fJ13QCRRrfuQd+KY3d
pTU0oZ6HLz6yx77xZ+nNCbBp2uxje8iVVMZKt+1A1tyKLnu7HuiyfocstEADz8wmjW705+eW0wP73UF2UwE
tNpBABx173A9p/f4udXBrL9x8nEXfuC3dtWY8pA22tfYu2M0fwxaRBR7a/CJzTWKqSkIId/vnuZ/
X9+WmHzIxu+5a8Zo+2VrU2ZTGbe/nQpU567ljlvj2bZmj0cPpZuFRPXuh6fTb2dWW/
eztP3DWbyR787KFv7mL3kJwY/
fr+j07AJk9nMO7NWMPmZEXz+1q0sW7uPw4kZVjHhIb68N+FaPptyC6P+05k3P1kOwMGjGfy8bCfTX7+BmZN
vZt2WwyQmZzugFRfOZNZ88tMhnhvVnHceaseaHekcSyuwigkJqMMrd7Xk7Qfbcl3fKKb/
cBAAo1Fxx5D6vPdwOybe3Zpf16dU2NaZabOpxg9HqbLwUEq9d9bjfeC+/
y3bMUcAQmLjyEk9SG7aEcymEhLWfUdM3JVWMTFxV7J39VcApCZsws3TF0+/
UFw9fIho1o3dK74AwGwqobggt3y77re9xrp5L6LR9mtQDQXGtCcv7TD56Ucxm0o4vPEHotsOsoqJbjeYA+s
WAJB+aAtuHnXx8A0BwGAwYnStgzIYcXHzoCAn1bKR1rjV8QbA1aMuhdkp9mvUBQqo15aT6UcoyDiGNpWQuP
VnwlsNtIopys8g69g/
aFNphe0zDm6k+GS2nbK1rd0JKUSG+RMR6oerq5H+3ZuxZuMBqxhPDzeUUgAUFpWAckSm9rN6+xYyc3McnYZ
N7E5IJTLUj4hQX1xdjPTr1pg1Gw9axbRqGo6Pdx0AWjYO40RGPgBHkjJp0TiMOu6uuBgNtG0RyaoNByq8xq
UsITGfsMA6hAXUwdXFQI/
WQWzcnWUV06yeD94els78JtE+ZOQUAeDv40bDCMu5zsPdSFSwB5m5xfZtgA1pbarxw1GqG2r5D7AC+J3Tp6
obgc02zqlSXv7h5GcklS/nZx4nNDauYkzm6ZiTmcfx8g/HbC6lMC+dfnd/
SGD9Vpw4tI01c56htKiABh2GcDIrmYyjO+3Wlgvh6RfGyTPaVpCdTFBM+woxBVnHT8dkJePpF07Gke3E/
z6dayduwlRyiuO7VpK8ayUAG7/+LwPGzyfuOsuwxZJJw+3ToItQxy+Mwuzk8uXCnGQC6rVzXEIOlJ6ZR0iQ
T/lycKA3u/YnV4hbtX4/M75cTVZuAZOe+Y89UxS1KD0zn5BA7/Ll4EBvdiekVhn/
y5+76NKuPgAx0YF8+vU6cvIKcXdz4e+th2naMMTmOdemzNxignzdy5cD6rqxPzGvyvhlm9No38S/
wvNpWac4lHySxlHelWzlnJzpPh7VDbU0B9KBwcBSrfXnQJ7W+vOyn+3qf5/
YzqS1PmcMaAwGF4IbtGXn0lkseLY3pUUFdBg+Hhc3D+KufpQNC96wUda1p9K2ndV+qvgduXn6Et1uEN8924
UFT7bDxd2TmC7XAtCk9yg2fjOBhU93ZOM3E+h2+9QK+7jUVP6B/
dLtrbKlsw8BqPxY6dWlMXPfG8NrT45g5ldr7JCZsIXK3u+qbNl5jF+Wx3P3Ld0BaBAVwM3D43js1UU88foP
xNYPwsXoXN+aUVnzKz/vw46DOSzbnMZtg+pZPV9YZGLK/
H2MvrIBnnWcappjtZypx6PKo05rnae1Hg9MBeYqpR6vLv5MSqlxSqlNSqlNaxKKaiXR/
MzjeAeenkTkHRBBQVZKxZiA0zFeARGczEohP/
M4+ZnHSTtg6aw5sP5Hghu0pW5oDD7B9bl+4mpufXc73gERjHxtZfnwxKXkZFYyXme0zdMvnIJs6086BVnJe
PpHnI7xD6cwJ4Xw5j3JTz9KUX4G2lTK0S2LCWlkmZDWqNv1HN3yCwBHNv9EYINLf7JZYXYKHn6nJ4V5+IZT
mJPmwIwcJzjQh7T005/4TmTkE+Rf9ae4di2iSUrNJjv38hnb/
jcJDvQmrWzoBP73fntViDtwJJ0pM5bx+hPD8PXxKH9+aL+WfDrpJt5/6Tp8vOsQGeZnj7RrTWBdN9JzTv9N
ycwtJsDHrULc4ZSTfPT9AZ6+pSk+nq7lz5eazEyZv5eebYPo2jLQLjnby+UyxyMaQGu9GegHFAJrytb1rG6
nWusZWuuOWuuOPWLdqws9b2kHtuAb1gif4HoYjK7EXvEfDm1eYhVzePMSmva8EYDQ2I4UF+ZSkJ1KYU4a+R
lJ+IVbJiBGtepFZtJeMo/tYva9TZj7cFvmPtyW/
MzjLHiu9yX5Ryzj8DZ8QmLwDozGYHSlQacRHNv+m1XMse2/0egKyyz+oJgOlBTmUZiTxsnMJIIbxmF0s5yA
wpv1ICd5PwAF2amENrkCgLBmPchLO2THVl2YrGP/4B3cAM+AKJTRlaj2w0iOX+rotByiWWwYiclZHE/
NpqTExLK1e+jeqZFVTGJyVnnv4N6DqZSWmq3+GAnn0axRKIkp2SSn5VBSamL5X/
vp3rGhVUxqeh4vTP2F5+4fRHSE9TBDVk5BeczqDQcY0N16gvqlLjbSm+SMU6RmnqKk1MyaHel0bGbdxhPZR
UyZt5eHRjYmIuj0ca61Ztr3B4gK9mB494izd+30nKnHo7p+ppVKqenAW1rrUuBDpdRCpdRcoCnQyS4ZltFm
E6tnP8lVTy9EGYzsWfElWUl7aNnfcnVC/LLPOLLtd+q1G8gtb2+htKiQ5R+fntm++vMnGXD/
DIwubuSkHebPj51r1rs2m9gw/1kGjJ9vuZx27VfkJO+jSa9RAOxbNYekHcuIbNWfa15bR2lxIX/
NfgSA9ENbObL5Z4Y9/ztmUymZx3ayb/
VcAP7+4nE63fAKymDEVFrEui+ecFgbz5c2m9i28EW63/05ymDgyPoF5KXsJ6bbzQAc+mse7j5B9Hv0B1zqe
KO1Jrb3aP6YOIjSonw63fYuwbFdcPPyZ8iEtez69V2OrP/Gwa26MC5GA+Pv6s/jry7EbDZzZb/
WxEQH8cNv2wAYMagdK//ex28rd+HiYsDdzYUXHxlWZff05WDehDfo0z6OIF8/
ji38lQmzpjPrl0WOTqtWuBgNjB/Th8df/8HyfvdpSUx0ID/8YblsfsTA1nz+7Xpy8k/x9sw/ATAaDcx4w/
KB7IW3FpObV4iL0cj4MX3KJ6E6C6NRcdewGF75fDdms6ZfXAj1Qj35bYOl93tQ5zAW/JlIXkEpn/
xYdjWLQTH5vjbsOZLHym3p1Av15LEPtgNw88B6xDWtOAfEGTnTHA919jyJ8hVK+QMTgW7Aw0Br4BFgCvCR1
vq8WjntZv9/5+B7GW8n+49d27y9/t2frLuPecbRKThM2H3THJ2CQyW/f5ejU3Co9DPuk/
Rv1GrkbLtW96k7Pqnx39rQ1mMd8gmkyh4PrXUWcLdS6mFgKXAc6KrUTrXS3VzfHwU0p9D
```

IzGcmXLt8ASpVQ/
eyUnhBBCiHO7XOZ4bAGmAfeXzfH4XSnVDpimlDqitb7JHgkKIYQQ4vJRXeHR6+xhFa31NqCbUmqsTbMSQgg
hxHlzpsml1c3xqHIuh9b6E9ukI4QQQoiacqYvibt8btsmhBBC/
Es58oZgNSWFhxBCCOHkpMdDCCGEEHZzWczxEEIIIYRzkB4PIYQQQtiNzPEQQgghhN1Ij4cQQggh7EZ6PIQQ
QghhN9okhYcQQggh7ER6PIQQQghhN9LjcYa+14+09Utc0ur3m+DoFBzqg1FNHJ2CQw0O/ve2P/
n9uxydgkOFP/ipo1NwqIyZTzs6hX8Vs/
R4CCGEEMJepMdDCCGEEHYjhYcQQggh7EabSx2dwnkzODoBIYQQQvx7SI+HEEII4eTMMtQihBBCCHtxpvt4y
FCLEEII4eS0yVTjx/lQSg1WSu1VSiUopSpcI62UukUp9U/
Z4y+lVNtz7VN6PIQQQggnZ4seD6WUEfgQGAgkAhuVUj9qrXedEXYI6K21zlJKDFmAF2q268UHkIIIYSTs9
Ecj85Agtb6IIBS6itgBFBeeGit/zoj/
m8g6lw7lcJDCCGEcHI2muMRCRw7YzmR6nsz7gSWnGunUngIIYQQTu5CbiCmlBoHjDvjqRla6xlnhlT2UlXs
qy+WwqPHuV5XCg8hhBDCyWlTzW8gVlZkzKgmJBGIPmM5Cjh+dpBSqg3wKTBEa51xrteVwkMIIYRwcjb6kri
NQGOlVAyQBNwI3HxmgFKqHvAdcJvWet/57FQKDyGEEMLJ2eK7WrTWpUqpB4DDfACMwS2sdr5S6p2z9dOC/
QCAwTSkFUKq17ljdfs+78CireNoDu7TWey6sGbVry95MPv0xAbPWDOwUzrV961mtX7k1le9WWObF1HEzcs8
1jYmJ8OZE9ine/XoP2XklKAX/1yWcq3qccyLuJWXtXxuYMvUDzGYzV4+4kjF33FxpXHz8HkaNeYCJr7/
AwP69KSoq5s5xD1NcUoKp1MSA/r259+477Jt8LYhpP5D+d01GGYz888fnrP9uaoWY/
ndNoWHcIEqKClny3t2kHtxWvk4ZDIx6cw35GcdZ+Np1dsy8dqxdv5Mp739tef+H9mDMLUOs1v+5ZhsfzfwB
ZVAYjUaeeOB62rdpDEBeXgEvTZnDgUNJKBQTnrqdtq0aOaIZF2z9tsO8P3sVZrNmaL+W3HK19Xnuj9V7mPf
jZgA86rjy6J19iW0QDMC3i7fx87KdaGBYv5aMHNre3unb1MynJzCsWy/
SsjJpfftIR6dT69Zt3s87nyzGZNYMH9iBUSN7Wa1f9fduZny5HINSGI0Gxt81hLYt65N6IoeX315IRlY+Bq
UYMbgjNwy/
wkGtqH22uoGY1noxsPis56af8fNdwF012WeVhYdSapHW+uqyn0cA7wArgDeUUm9orWfX5IVqm8ms+XjRfl6
6qw2Bvu488cEWOrcIJDrUqzwm1L8Or93dFm9PVzbvyWDad/
uY8kAHjAbF6GGNaBTpQ2FRKY+9t4V2jf2ttr2UmUwmJk5+l48+mEJoaDC33H4vvXt1o1HDBhXi3v1gBld0P
X1SdnNzZcZHb+Hp6UFJaSlj7nqI7t0606Z1Czu34sIpg4EBd7/FNxOuIi8jiVFTVpOw4RcyEk/
Xww3jBuEfHssn97YhvEknBt7zDnOf7FO+Pm7Y/WQk7sXdw8cBLbg4JpOZie/
M46OpjxAa7M8td79O7+5tadQgojymS4dm9OneFqUU+w4k8tSLH/
P9F68AMPn9r+nWuSVvvnwPJSWlnDpV7KimXBCT2cw7s1Yw9blrCA705u5nvqZ7xxgaRAWWx4SH+PLehGvx8
a7D31sP8+Yny5n+2g0cPJrBz8t2Mv31G3BxMfLk6z9wRYcYosL9HNegWjZ7yU988N3XzHnuFUenUutMJjNT
p//Mu6/
cTkhgXcY8+jE9uzQjpl5IeUzHtg3p2aUZSikSDqXw3KRv+Hr6QxiNBh4aM5imsRGcLChi9CPT6dyukdW2zs
yZvp22ujuX1j/j56eAflrr0UB34BGbZnUe9h/
LJTzQg7BAD1xdDPRoG8L6XdZzWpo18MXb0xWApvXqkpFTBEBAXXcaRVr+4Hi4uxAV4lm+zhnsjN9DdHQkUV
ERuLq6MmhgP1as/KtC3Fdff0//
vr0I8Pcvf04phaenBwClpaWUlpZS1j3mNMIbdyQ7+SA5qYcxl5awe823xHYZZhUT23ko8SvmAZC8byN1vHz
x8g8DwDswgkYdB/
PPH7PtnXqt2Ln7ENGRIURFBOPq6sKgfp1YsWa7VYynZ53y97WwsAhVNjk9/2QhW7bv45qhlonnrq4u+Ph42
rcBF2l3QiqRoX5EhPri6mKkX7fGrNl40CqmVdNwfLzrANCycRgnMvIBOJKUSYvGYdRxd8XFaKBti0hWbThg
9zbY0urtW8jMzXF0Gjaxa38iUeEBRIYF4OrqwoBerVm13roD3tPD/fSxX1TM/
05vQQE+NI21FOdenu40iA7mREauXfO3JbPZVOOHo1Q31HLmJTMuWutDAFrrdKWU2bZpnVtmTjFBfu7ly4G+
7uw/
WvVBtHRjCh2aBlR4PjXzFAeT8mlSr65N8rSFtBPphIaertJDQ4PYuXO3dUzaCZavWMOMj6YSv2uv1TqTycT
Nt93DscQkbhh5Na1bNbdL3rXFOyCCvPTE8uW8jCQiGlt3tfsERJBrFXMcn4BwTmal0P/Oyaz4/
DncnLC3AyAtPZvQkNPHcmiwHzt3H6oQt3zVVt7/5Dsys/
J4b+KDACQdT8ffz4cJE2ezLyGR5k3r8+SDN+Dh4V5h+0tVemY+IYHe5cvBgd7sTkitMv6XP3fRpZ3lc1RMd
CCffr2OnLxC3N1c+HvrYZo2vDw+8f4bnMjIIyTIt3w5JLAu8fsSK8StWLeLjz5fSlbOSaZOuKXC+uTULPYd
SKZlU+caYq/
O5dLj0VYplauUygPaKaXCAJRSblgmmThU5RcSVx6740AWSzemMGpIQ6vnC4tMTJobz53DG+FZx4nm2epKWn
9Wr8WUtz7k4QfHYTRWfKuMRiNfz/
uE3375hp3xe0hIqPhH61JWWQ+NPvuIqCKmUcfBFOScIPXANhtlZweVvf+VHPz9erXn+y9e4a3X7mParB8AK
DWZ2LP/KCNH9OarmS/gUceNWfN+tXHCtavS5ldhy85j/
LI8nrtv6Q5Ag6gAbh4ex2OvLuKJ138gtn4QLkb5yipnoSt58ys7H/S5ogVfT3+ISc/
dxIy5y63WFRQW8cwbXzF+7BC8POvYLFd70+bSGj8cpbq/tjFa66OVPO8B3F/dTs+8KcmL9/bg+v+r/U/
Ugb5upGefHh7JyCkioG7FT22Hk/P54Nt9/HdMa+p6uZY/
X2oyM+mLeHq3C+GKVsG1np8thYQEk5qaVr6cmppOcyu3ft4umyMNzs7hzV/
rcfFaKRvn9P3dvHx8aZjXFv+WreB2NgY+yRfC/IykvAJOv1JxScwkvzMlAoxdYOiSCqPiSA/
M4WmV1xDbKehNIwbhNG1Du6ePgwdP5Nf3rnTji24OCHB/qSmZZYvp57IJjjIr8r4uLZNSEw6QVZ2HqHB/
oQE+9O6haUIH9A7js/mnfNGg5eU4EBv0sqGTgBOZOQT5F9xftaBI+lMmbGMyU+PwNfHo/
z5of1aMrRfSwBmzP+L4ADvCtuKS1NIUF3S0k8PI6Vl5BIUUHXPZftWDUhKziQ75yR+vl6Ulpp49o2vGNSnD

X26Oc+8tvNxufR4rFBKPamUKi9OlFKhWL4w5q3qdqq1nqG17qi17miLogOgcVRdkjMKSc0spKTUzJrtaXRu
HmgVcyLrFBO/
iOeRG5oRGXx6HFtrzQff7iMqxJMRvaLP3vUlr2WLZhw9mkRSUjIlJSX89sdy+vSynp39yw/zWPzjfBb/
OJ8B/XrzzFMP07dPDzKzssnLs5y0T50qYv2GLTRoUK+yl7lkJe/
fjH94I3xD6mNwcaV5j+tI2PCLVUzChl9o2cdypU94k04UnczlZFYKq+ZO4KO7mvDxuBb8NPV2jv6z0qmKDo
CWzRpwNDGNpOR0SkpK+W35Rvp0t/5CyKOJaeWfDnfvO0JJqQk/X2+CAn0JC/
bn8FFLobZhy24anjEp1Rk0axRKYko2yWk5lJSaWP7Xfrp3tO7NTE3P44Wpv/Dc/
YOIjvC3WpeVU1Aes3rDAQZ0b2K33MXFad44kmPHMzmekkVJSSlLV+2gZ+dmVjHHjmeUH/
t7E45TUmrCt64nWmtee28R9aODuenq7o5IX5SprscjDpgIbFVKPQy0Bh4FJgOj7JBbtYxGxdgRsbw0cwcms
2ZApzDqhXnx69+Wm6oN7hrB18uOkFdQyvRF+y3bGBRTH4pj9+FcVmxJpX6YF+Pf2QTArYNj6NgssMrXu5S4
uBh56skHue+hpzCbTIwYPoRGjWJYsPBHAEZeO7zKbdPTM/
jvi5Mwm82YzWYGDuhDr57OdUmZNptY+sljjJzwA8poZMfSOWQc2027QZYCYttvMzm4+Tcaxg1i7PQdlJZdT
nu5cHEx8tT4m7jv8Xcwm82MuLI7jWIiWPDDSgBGjujNslVb+Pm3dbi4GHF3c2PShLHlXdJPPXwTz746k9KS
UiIjgnjp6Tsc2JqaczEaGD+mD4+//gNms5kr+7QkJjqQH/7YAcCIga35/
Nv15OSf4u2ZfwJgNBqY8caNALzw1mJy8wpxMRoZP6ZP+STUy8W8CW/
Qp30cQb5+HFv4KxNmTWfWL4scnVatcDEaeeyeoYyfMAez2cywAR1oWD+E75ZsBOA/
Qzqx4q9dLFm+rezYd+HVJ69HKcX2+CP8+ud2GjUIZdRD0wC4Z9QAunW8PApPR04WrSlV2ZiZVYCl6Hgby21
Su2qtK87kqcbuReNqMCJ7+anfb4KjU3CoD0ZdHv+pL9QD0xefO+gylZuy09EpOFT4g586OgWHypj5tKNTcK
iAJjfY9XLBFS+3qfHf2j7//cchlzRWOdSilPJTSn0MjAYGA98CS5RS/
eyVnBBCCCHOzWzWNX44SnVDLVuAacD9WutS4HelVDsst0U9rW+yR4JCiGEEKJ6ZrPD73Jx3qorPHqdPayi
td4GdFNKjbVpVkIIIYQ4b47swaipKguP6uZyaK0/
sU06QgghhKipy6LwEEIIIYRzMOvLY6hFCCGEEE5AejyEEEIIYTeXy+RSIYQQQjgB6fEQQgghhN1I4SGEEEI
Iu5GhFiGEEELYjfR4CCGEEMJupPAQQgghhN3IfTyEEEIIYTfO1ONR5bfTCiGEEELUNpv3eGQmxNv6JS5pDX
oXOjoFh9p4pMTRKTjUvj+mOToFh3Gp4+HoFBwqY+bTjk7BoQLvnOjoFBxKr77Brq8nV7UIIYQQwm6caahFC
g8hhBDCyUnhIYQQQgi7kaEWIYQQQtiN9HgIIYQQwm6k8BBCCCGE3chQixBCCCHsxqylx0MIIYQQdiI9HkII
IYSwG5njIYQQQgi7kcJDCCGEEHYjhYcQQggh7MaJ6g4pPIQQQghnJ4WHEEIIIezG5ESVR5WFh1KqHpCmtT6
llFLAHUAHYBfwida61D4pVs2vQRca9hsPykDqjp9I2jDXar1HQD1iBz+Hd0gTjqyZwfFN8y3P+9ejyVUvl8
fV8Y3g6NpPSd7yjT3Tvyhr121i8tszMJvNX
DP8/xgz6vpK43bu2seoux5j0qtPMbBfDwAmvPoOq9ZuIMDfj4XznOtr20c/
+RYdug+m6FQhH064k0N7tlWICYlowPiJc/H29efQ7m28//
wdlJaWANAirhejn5iK0cWVvOx0Jtw1gMDQKB54ZRZ+gWFobWbpwk9ZPP8DO7fswvlEtCWy4x0oZSAjYTlp8
T9YrfeP6UFIy+EAmEtOcWzDTE5lHXFEqrVm674sZi0+jNms6R8Xyn96R1qtX7XtBN+vPg6Ah5uBccMb0iDc
i/
TsIt5bmEB2fglKwcCOoQzrFu6IJlywdZv3884nizGZNcMHdmDUyF5W61f9vZsZXy7HoBRGo4Hxdw2hbcv6p
J7I4eW3F5KRlY9BKUYM7sgNw69wUCtsZ+bTExjWrRdpWZm0vn2ko9OxGyeqO6rt8VgMdC77eSLQCFgE9AM6
AWNsmtm5KAMNBzxG/ILxFOel0ezWjykdgPmV5Wzw8fOoA
brxvG8y+/Zc+0L1r7HoMJrxfLgyNa0Lh1Z8Y++wHPjupRIe6Wh1/n5y/f46/fvmHscx/Q75rR/
L5gBp7evox99n1eu38Y6SnHqOsfDIDJVMqct57k0J5t1PH0ZtK89fyzfhmJB3fbu4k1pxRRncdwYOlrlBRk
0GTIG+QkbqIoJ6k8pCg/jYTfX8JUfBKfiHZEdx3L/iXPOzDpi2Myaz756RD/
Hd2CwLpuPDV9B52a+xMd4lkeExJQh1fuaom3hwtb9mUx/
YeDTLynNUaj4o4h9WkY4U1hkYknpv1D21hfq20vZSaTmanTf+bdV24nJLAuYx79jJ5dmhFTL6Q8pmPbhvTs
0gylFAmHUnhu0jd8Pf0hjEYDD40ZTNPYCE4WFDH6kel0btfIatvLwewlP/
HBd18z57lXHJ2KqIKhunVa64KynwcA12ut52qtxwBxtk+tej5hzTmVlUhRznG0gIzZQTe5YR0KinVUxJQTb5K
XvQ5qo7Z/zqdeRUdhJFuam2TrnW7Ny1j+ioCKIiw3F1dWXQwF6sWPV3hbj5C36if9/
uBPj7Wj0f174Vdev62CvdWtOp91Ws/PlLAPbv2ICXjx9+QWEV4lp16sPfSxcCsPKnL+jUx/Jpv8eQG1m/
bBHpKccAyM06AUB2ekp5z8mpgnySDu0hIDjC1s2pFZ6BsRTlpVKcn4Y2m8g68he+0Z2sYgpO7MNUfNLyc/
p+XD0DHZFqrUlIzCcssA5hAXVwdTHQo3UQG3dnWcU0q+eDt4flc1WTaB8ycooA8Pdxo2GENwAe7kaigj3Iz
C22bwMuwq79iUSFBxAZFoCrqwsDerVm1Vfo9VjGeHu5YOqmhsKiYsh8JCvChaazluPbydKdBdDAnMnLtmr89
rN6+hczcHEenYXdmXfOHo1RXeBxTSvUr+/kwEA2glLokzlpuPsEU56WVLxfnp+HuE1zj/
QQ168+JPUtrMzWbSzuRQVhIUPlyaEgQaScyrGJS09L5c+U6Rl4zxN7p2UxASAQZZUUDQEZqIgEh1gWCj18g
BXnZmE2mspgkAkIs3fAR9RvjVdePFz/5g0lf/k2vYbdWeI3g8PrENG3L/
p0bbNiS2uPqGUDJydPvfcnJDFw9/
KuMD4jtS17SNjtkZjuZucUE+bqXLwfUdSU9t2oT6jK+GWb02jfpOLvJC3rFIeST9I4ytsmedrCiYw8QoJOf5AI
CaxbafGwYt0ubrjnPR576Uuee/
jqCuuTU7PYdyCZlk2jbJmusKPLpfC4C3hBKbUKcAO2KaWWA0uBR6vbqVW9/
p9RettavUuEZXcN71SuDCwGNepCxd3ltJWUXlbXz7N/GlHdm8PD9ozEajfZJyg7+9ynOylm/
i+pijEYXGjbvwBsPjuDV+4dy3dhnCK/XuDysjocXj7/5NZ+9+TiFJ/NqNXfbqaS9VfAObUlgbD+Ob/
nShvnYXmX/yyt934EdB3NYtjmN2wZZD0MWFpmYMn8fo69sgGcd55ljX+n//Ura3ueKFnw9/

```
SEmPXcTM+Zan98KCot45o2vGD92CF6edWyWq7Avs7nmj/OhlBqslNqrlEpQSj1dyXqllHqvbP0/
SqkOle3nTNX+j9Na91VKNQeaALOBRGAj0P0c280AZgCsfbO7Teqq4rw03HxOj026eYdQnJ9eo334x3QlP20
fJQVZ5w6+hISGBJGSdrqtqWnpBAdbd0Tt2p3AU89PAiA7J5c16zZhNBrp19u5JpMNuv4eBvznTgAS4jcRGB
Zdvi4wNIrME8lW8blZ6Xj6+GEwGjGbTASGRpJ5wjLJMCMtidzsDIpOFVB0qoDdW9ZQv0kbko/
ux+jiwmNvfs3qJfPZsHyR3dp3sUoKMnD1Ov3eu3oFUlJY8Xiu41eP6CvGcXDZREzF+fZMsdYF1nUjPed0D0
dmbjEBPm4V4g6nnOSj7w/w/O3N8fF0LX+
+1GRmyvy99GwbRNeWl0QH7nkLCapLWvrpYYS0jFyCAqoeNm3fqgFJyZlk55zEz9eL0lITz77xFYP6tKFPtx
b2SFnYickGXxKnlDICHwIDKfv7r5T6UWu964ywIUDjskcX4KOyf6tUXY/
HSqXUk8B+rfUPWuuFWIZc5gAOn5WYl7IHD/8o3H3DUQYXgpv1J/PAmhrtI6j5QNL3/
GGjDG2nZfMmHD2WRNLxFEpKSvjtj1X07mn9Pi/+fhZLFn3GkkWfMaBvd5594j6nKzoAfvtmOk/
c2IknbuzExj9/
pPewWwBo3LozBfk5ZKdX7FGL37SSrgOuBaD3VbexccVPAGxc8RPN23fHYDTiVseD2FadSTpkGR+/
d8IMkg7t4ee579qpZbWjIOMA7j5huHkHowxG/Ot3I/fYJqsYV89AYno/
xpG1H1KUl1zFnpxHbKQ3yRmnSM08RUmpmTU70unYzHoo5UR2EVPm7eWhkY2JCPIof15rzbTvDxAV7MHw7s4
xj+dMzRtHcux4JsdTsigpKWXpqh307NzMKubY8YzynpG9CccpKTXhW9cTrTWvvbeI+tHB3HR1tZ8dhROy0V
BLZyBBa31Qa10MfAWMOCtmBDBHW/
wN+Cmlqr1UrLoejzgsV7NsVUo9DLTGMsQyGRh1XinbkjZxcNnbtLz2LTAYSdvxM4UZhwhrezUAKdsX4eoZQ
NvbZmJ08wJtJiLuerZ+dgum4gIMLu741e/Egd8nO7YdF8DFxcjTj9/LvQ+/
gNlsZsSwgcQ2rM+C7xYDMPI/V1a7/
dMvTGLTlh1kZ+fyf1eN4t6xt3DN8EH2SP2ibFmzhPY9BvP+j7spPlXIhy/eVb7umfd/
YPrL95B1Ipm57z7LIxPnctN9L3Jo73aWL/oMgKRDe9j21+9M/WYLZrOZZd/
P4tiBeJq160bvYbdyZN8Opny1EYB5H7zA1jW/
OqSdNaLNJG6YRcP+z6KUgcyEFZzKSSSw8QAAMvYvJazNdRjdvYnubOk50trEvsXPOjLri2I0Ku4aFsMrn+/
GbNb0iwuhXqgnv22wFKGDOoex4M9E8gpK+eTHg5ZtDIrJ97Vhz5E8Vm5Lp16oJ499sB2AmwfWI65p1fNiLi
UuRiOP3TOU8RPmYDabGTagAw3rh/DdEstx+58hnVjx1y6WLN+Gi4sRdzcXXn3yepRSbI8/wq9/
bqdRg1BGPWS5jP6eUQPo1rGJI5tU6+ZNeIM+7eMI8vXj2MJfmTBrOrN+WeTotGzORl9OGwkcO2M5kYq9GZX
FRAJVfspR55oXUVZ0vA0cB7pqrRPPP2fbDbU4iw53fu7oFBxqVL9/
d3fuc49e4+gUHMaljse5gy5jEW0vn4ndFyLwzomOTsGh9Oqt5z8BqxY83t+jxn9rpy4/
dTcw7oynZpRNlQBAKTUSGKS1vqts+Tags9b6wTNifgHe0FqvKVteBjyptd5c1etWdwMxP2ASlupmMHAlsEQ
p9bDW2rlmYwohhBCXsQu5SuXM+ZhVSKTsitYyUVg6IWoaY6W6OR5bgP1AR63171rr8cBtwKtKqfnV7VQIIY
QQ9mPSusaP87ARaKyUilFKuQE3Aj+eFfMjMKrs6pauQI7WutrJZNXN8eh19rCK1nob0E0pNfZ8MhZCCCGE7
dlijofWulQp9QDwG2AEZmmt45VS95Stn47lLudXAglAATD6XPutsvCobi6H1vqTmqUvhBBCCFux1Q3BtNaL
sRQXZz43/YyfNXB/
TfbpPHfOEUIIIUSlLpcviRNCCCGEE7DR5bQ2IYWHEEII4eRscedSW5HCQwghhHByMtQihBBCCLuRoRYhhBB
C2I0z9XhUdwMxIYQQQohaJT0eQgghhJNzph4PKTyEEEIIJydXtQghhBDCbmRyqRBCCCHsxpmGWpR2ou6ZC6
GUGlf21b//StL+f2/7/81tB2m/tP/f3f5L2b/hqpZxjk7AwaT9/17/5raDtF/aLy5J/
4bCQwghhBCXCCk8hBBCCGE3/4bC498+xift//
f6N7cdpP3SfnFJuuwnlwohhBDi0vFv6PEQQgghxCXisiw8lMUapdSQM567Xin1qyPzsiWlVLRS6pBSKqBs2
b9sub5S6lelVLZS6mdH52kL1bS9t1JqnVIqXin1j1LqBkfnagvneO83K6W2lf0O7nF0rrZQXfvLlusqpZKU
Uh84NlPbOMf7byp7/7cppX50dK617Rxtr6eU+l0ptVsptUsp1cDB6Yoyl+1Qi1KqFbAAaA8YgW3AYK31AUf
mZUtKqSeBWK31OKXUx8BhrfUbSqn+gCdwt9Z6mGOztI3K2g4sBLTWer9SKgLYDDTXWmc7LlPbqKL9U7H8Hy
9SSnkDO4FuWuvjDkzVJqo69svWvQsEA5la6wccmaetVPN/P19r7e3o/
GypmravAF7TWv9RdvybtdYFDk1WAJdx4QGglJoMnAS8gDyt9SsOTsmmlFKuWP64zgLGAu211sVl6/oAj1/
GhUeVbT8jZjtwndZ6vwNStKlztV8pFQhsBbpepoVHpe1XSsUBTwC/Ah0v48Kj0v48Kjqvb/
GwqPCm0HYoEZWusejsxNVO5yv2X6S8AWoBjo6OBcbE5rXaKU+t9J9v/O/sN7OTtX25VSnQE34LLs8aqq/
UqpaOAXLCfiJy7HogOq3Pe3Ao7/
OkqpTUApMFFrvchROdpKFe99EsBZ7WWpscmauwuCznePyP1vok8DXwhda6yNH52MkQIBlo5eh
EHKDStiulwoEvgNFaayf6KqUaq9B+rfUxrXUbLIXH7UqpUEclZwdnt/8+YLHW+pjjUrKryo7/
elrrjsDNwDtKqQUYOycz2zm67C9ATeBzoBDQE7nBIZqKCy7rwKGMue1z2lFLtgIFAV+CRsj+4/
wpVtV0pVRfLJ/7ntdZ/Oy5D2zrXe1/W0xGP5WR82ami/VcADyilDgNvotEUUhMdlqQNVfX+/
6+HS2t9EFiBZRjislJF2xOBrVrrg1rrUmAR0N0MFhSQor/
4bC419BKaWAj4DxWuujwBQsJ9vLXlVtV0q5Ad8Dc7TWCxyZoy1V0/4opZRHWu/
0B3Y67hMbaOq9mutb9Fa19NaN8DyyXeO1vppB6ZqE9W8//5KKfeymCAs7/8ux2Va+6o5720E/JVSwWWh/
bjM2u7MpPC4fIwFjmqt/
yhbngY0U5ZLSldjucKnv1IqUk1yGFZ2kalbQeeAXoBd5xxSWE7B+VoS1W1/05gfdmk2pVY/
hjvcFCOtlTlse/AnOypqve/DbCp7P3/E8scj8vtj29Vbe+BpdhcppTaASAScjgE8ekKM52WV/VIoQQQohLi/
R4CCGEEMJupPAQQgghhN1I4SGEEEIIu5HCQwghhBB2I4WHEEIIIexGCg8hhBBC2I0UHkIIIYSwGyk8hBBCC
GE3/w+0P5lDVu8O/QAAAABJRU5ErkJggg==\n",
      "text/plain": [
       "<Figure size 720x360 with 2 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "#Heatmap, relations between variables\n",
    "\n",
    "#x1 to x2 = .06 correlation       #x2 to x3 =  .18     #x3 to x4 =   .3 
#x4 to x5 = .29        #x5 to x6 = .32     \n",
    "#x1 to x3=  .28 correlation       #x2 to x4 =  .11     #x3 to x5 =   .36 
#x4 to x6 = .22                                           \n",
    "#x1 to x4=  .088 correlation      #x2 to x5 =  .04     #x3 to x6 =   .2 
\n",
    "#x1 to x5=  .43 correlation       #x2 to x6 = -.062    \n",
    "#x1 to x6=  .41 correlation        \n",
    "\n",
    "\n",
    "plt.figure(figsize=(10,5))\n",
    "c= df.corr()\n",
    "sns.heatmap(c,cmap=\"BrBG\",annot=True)\n",
    "c"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 19,
   "id": "bfbe0bf8",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "5    65\n",
       "4    40\n",
       "3    20\n",
       "1     1\n",
       "Name: X1, dtype: int64"
      ]
     },
     "execution_count": 19,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df[\"X1\"].value_counts()\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 20,
   "id": "dae42eb5",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "3    42\n",
       "2    34\n",
       "1    27\n",
       "4    17\n",
       "5     6\n",
       "Name: X2, dtype: int64"
      ]
     },
     "execution_count": 20,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df[\"X2\"].value_counts()\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 21,
   "id": "844560c9",
   "metadata": {
    "scrolled": true
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "3    55\n",
       "4    33\n",
       "5    17\n",
       "2    14\n",
       "1     7\n",
       "Name: X3, dtype: int64"
      ]
     },
     "execution_count": 21,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "\n",
    "df[\"X3\"].value_counts()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 22,
   "id": "e1793dde",
   "metadata": {
    "scrolled": true
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "4    53\n",
       "3    41\n",
       "5    25\n",
       "2     5\n",
       "1     2\n",
       "Name: X4, dtype: int64"
      ]
     },
     "execution_count": 22,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df[\"X4\"].value_counts()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 23,
   "id": "d62c475b",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "4    50\n",
       "5    31\n",
       "3    22\n",
       "2    16\n",
       "1     7\n",
       "Name: X5, dtype: int64"
      ]
     },
     "execution_count": 23,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df[\"X5\"].value_counts()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 24,
   "id": "35ef8caf",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "5    57\n",
       "4    47\n",
       "3    20\n",
       "2     1\n",
       "1     1\n",
       "Name: X6, dtype: int64"
      ]
     },
     "execution_count": 24,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df[\"X6\"].value_counts()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 25,
   "id": "66fc5fa5",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Requirement already satisfied: xgboost in d:\\downloads\\anucleus\\lib\\
site-packages (1.6.1)\n",
      "Requirement already satisfied: numpy in d:\\downloads\\anucleus\\lib\\site-
packages (from xgboost) (1.20.3)\n",
      "Requirement already satisfied: scipy in d:\\downloads\\anucleus\\lib\\site-
packages (from xgboost) (1.7.1)\n"
     ]
    }
   ],
   "source": [
    "import sys\n",
    "!{sys.executable} -m pip install xgboost"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 26,
   "id": "ecd5b2a6",
   "metadata": {},
   "outputs": [],
   "source": [
    "#import libraries for training and testing several ML models\n",
    "from sklearn.model_selection import train_test_split\n",
    "from sklearn.ensemble import RandomForestClassifier\n",
    "from sklearn.linear_model import LogisticRegression\n",
    "from sklearn.svm import SVC\n",
    "from xgboost import XGBClassifier\n",
    "from sklearn.metrics import confusion_matrix, classification_report,
accuracy_score\n",
    "from sklearn.metrics import f1_score\n",
    "from sklearn.metrics import recall_score\n",
    "from sklearn.metrics import precision_score"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 27,
   "id": "341ab3bc",
   "metadata": {},
   "outputs": [],
   "source": [
    "#Define a function to sequentially test Logistic Regression, Support Vector, 
Random Forest Classifier and XG Boost Classifier\n",
    "# models\n",
    "\n",
    "def model_sequence(df, rand_state):\n",
    "    \"\"\" \n",
    "    Input: Dataframe, random state for train-test split\n",
    "    Output : Dataframe containining training and testing accuracy scores, 
precision, recall and f1-score\n",
    "    Output : Feature importances for the Random Forest and XG Boost models\n",
    "    \"\"\"\n",
    "    X = df.drop('Y', axis=1)\n",
    "    y = df.Y\n",
    "    X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.3, 
random_state = rand_state)\n",
    "\n",
    "    score_dict = {'Logistic Regression':{},\n",
    "                  'Support Vector':{},\n",
    "                  'Random Forest':{}, \n",
    "                  'XG Boost':{}\n",
    "                 }\n",
    "    \n",
    "    def score_dict_edit (key, ML_model):\n",
    "        ML_model.fit(X_train, y_train)\n",
    "        train_pred = ML_model.predict(X_train)\n",
    "        test_pred = ML_model.predict(X_test)\n",
    "        score_dict[key]['Train Accuracy'] = accuracy_score(y_train, 
train_pred)\n",
    "        score_dict[key]['Test Accuracy'] = accuracy_score(y_test, test_pred)\
n",
    "        score_dict[key]['Precision GP 0'] = precision_score(y_test, test_pred, 
average=None)[0]\n",
    "        score_dict[key]['Precision GP 1'] = precision_score(y_test, test_pred, 
average=None)[1]\n",
    "        score_dict[key]['Recall GP 0'] = recall_score(y_test, test_pred, 
average=None)[0]\n",
    "        score_dict[key]['Recall GP 1'] = recall_score(y_test, test_pred, 
average=None)[1]\n",
    "        score_dict[key]['F1 Score GP 0'] = f1_score(y_test, test_pred, 
average=None)[0]\n",
    "        score_dict[key]['F1 Score GP 1'] = f1_score(y_test, test_pred, 
average=None)[1]\n",
    "\n",
    "\n",
    "    # create objects for the ML models to be tested \n",
    "    model_logreg = LogisticRegression(max_iter=1000, C=1)\n",
    "    model_SVC=SVC(kernel='rbf', C=5, gamma='auto')\n",
    "    model_rfc = RandomForestClassifier(n_estimators = 200)\n",
    "    model_xgb = XGBClassifier(use_label_encoder=False, eval_metric='mlogloss', 
n_estimators=100, learning_rate=0.5, max_depth=5)\n",
    "\n",
    "    models=[model_logreg, model_SVC, model_rfc, model_xgb]\n",
    "\n",
    "    for model in models:\n",
    "\n",
    "        if model==model_logreg:\n",
    "            score_dict_edit ('Logistic Regression', model_logreg)\n",
    "        elif model==model_SVC:\n",
    "            score_dict_edit ('Support Vector', model_SVC)\n",
    "        elif model==model_rfc:\n",
    "            score_dict_edit ('Random Forest', model_rfc)\n",
    "        else:\n",
    "            score_dict_edit ('XG Boost', model_xgb)\n",
    "    df_metrics=pd.DataFrame(score_dict).transpose()\n",
    "    rfc_imp = pd.DataFrame({'Feature':X.columns,
'Importance':model_rfc.feature_importances_}).sort_values(by='Importance',
ascending=False)\n",
    "    xgb_imp = pd.DataFrame({'Feature':X.columns,
'Importance':model_xgb.feature_importances_}).sort_values(by='Importance',
ascending=False)\n",
    "    return df_metrics, rfc_imp, xgb_imp"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 28,
   "id": "f104b553",
   "metadata": {},
   "outputs": [],
   "source": [
    "df_metrics, rfc_imp, xgb_imp = model_sequence(df, 5)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 29,
   "id": "698fb398",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<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>Train Accuracy</th>\n",
       "      <th>Test Accuracy</th>\n",
       "      <th>Precision GP 0</th>\n",
       "      <th>Precision GP 1</th>\n",
       "      <th>Recall GP 0</th>\n",
 "      <th>Recall GP 1</th>\n",
 "      <th>F1 Score GP 0</th>\n",
 "      <th>F1 Score GP 1</th>\n",
 "    </tr>\n",
 "  </thead>\n",
 "  <tbody>\n",
 "    <tr>\n",
 "      <th>Logistic Regression</th>\n",
 "      <td>0.659091</td>\n",
 "      <td>0.526316</td>\n",
 "      <td>0.428571</td>\n",
 "      <td>0.583333</td>\n",
 "      <td>0.3750</td>\n",
 "      <td>0.636364</td>\n",
 "      <td>0.400000</td>\n",
 "      <td>0.608696</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>Support Vector</th>\n",
 "      <td>0.829545</td>\n",
 "      <td>0.631579</td>\n",
 "      <td>0.583333</td>\n",
 "      <td>0.653846</td>\n",
 "      <td>0.4375</td>\n",
 "      <td>0.772727</td>\n",
 "      <td>0.500000</td>\n",
 "      <td>0.708333</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>Random Forest</th>\n",
 "      <td>0.943182</td>\n",
 "      <td>0.657895</td>\n",
 "      <td>0.615385</td>\n",
 "      <td>0.680000</td>\n",
 "      <td>0.5000</td>\n",
 "      <td>0.772727</td>\n",
 "      <td>0.551724</td>\n",
 "      <td>0.723404</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>XG Boost</th>\n",
 "      <td>0.931818</td>\n",
 "      <td>0.578947</td>\n",
 "      <td>0.500000</td>\n",
 "      <td>0.625000</td>\n",
 "      <td>0.4375</td>\n",
 "      <td>0.681818</td>\n",
 "      <td>0.466667</td>\n",
 "      <td>0.652174</td>\n",
 "    </tr>\n",
 "  </tbody>\n",
 "</table>\n",
 "</div>"
],
"text/plain": [
 "                     Train Accuracy  Test Accuracy  Precision GP 0  \\\n",
 "Logistic Regression        0.659091       0.526316        0.428571   \n",
 "Support Vector             0.829545       0.631579        0.583333   \n",
 "Random Forest              0.943182       0.657895        0.615385   \n",
       "XG Boost                   0.931818       0.578947        0.500000   \n",
       "\n",
       "                     Precision GP 1  Recall GP 0  Recall GP 1  F1 Score GP
0  \\\n",
       "Logistic Regression        0.583333       0.3750     0.636364
0.400000   \n",
       "Support Vector             0.653846       0.4375     0.772727
0.500000   \n",
       "Random Forest              0.680000       0.5000     0.772727
0.551724   \n",
       "XG Boost                   0.625000       0.4375     0.681818
0.466667   \n",
       "\n",
       "                     F1 Score GP 1  \n",
       "Logistic Regression       0.608696  \n",
       "Support Vector            0.708333  \n",
       "Random Forest             0.723404  \n",
       "XG Boost                  0.652174  "
      ]
     },
     "execution_count": 29,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df_metrics"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 30,
   "id": "23905c48",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<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>Feature</th>\n",
       "      <th>Importance</th>\n",
       "    </tr>\n",
     "  </thead>\n",
     "  <tbody>\n",
     "    <tr>\n",
     "      <th>1</th>\n",
     "      <td>X2</td>\n",
     "      <td>0.213869</td>\n",
     "    </tr>\n",
     "    <tr>\n",
     "      <th>4</th>\n",
     "      <td>X5</td>\n",
     "      <td>0.210036</td>\n",
     "    </tr>\n",
     "    <tr>\n",
     "      <th>2</th>\n",
     "      <td>X3</td>\n",
     "      <td>0.182676</td>\n",
     "    </tr>\n",
     "    <tr>\n",
     "      <th>0</th>\n",
     "      <td>X1</td>\n",
     "      <td>0.140330</td>\n",
     "    </tr>\n",
     "    <tr>\n",
     "      <th>5</th>\n",
     "      <td>X6</td>\n",
     "      <td>0.127447</td>\n",
     "    </tr>\n",
     "    <tr>\n",
     "      <th>3</th>\n",
     "      <td>X4</td>\n",
     "      <td>0.125641</td>\n",
     "    </tr>\n",
     "  </tbody>\n",
     "</table>\n",
     "</div>"
    ],
    "text/plain": [
     "  Feature  Importance\n",
     "1      X2    0.213869\n",
     "4      X5    0.210036\n",
     "2      X3    0.182676\n",
     "0      X1    0.140330\n",
     "5      X6    0.127447\n",
     "3      X4    0.125641"
    ]
   },
   "execution_count": 30,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "rfc_imp"
 ]
},
{
 "cell_type": "code",
 "execution_count": 31,
 "id": "5e3edf95",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/html": [
    "<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>Feature</th>\n",
    "      <th>Importance</th>\n",
    "    </tr>\n",
    "  </thead>\n",
    "  <tbody>\n",
    "    <tr>\n",
    "      <th>2</th>\n",
    "      <td>X3</td>\n",
    "      <td>0.200417</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>4</th>\n",
    "      <td>X5</td>\n",
    "      <td>0.200328</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>0</th>\n",
    "      <td>X1</td>\n",
    "      <td>0.176122</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>1</th>\n",
    "      <td>X2</td>\n",
    "      <td>0.173256</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>3</th>\n",
    "      <td>X4</td>\n",
    "      <td>0.136417</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>5</th>\n",
    "      <td>X6</td>\n",
    "      <td>0.113461</td>\n",
    "    </tr>\n",
    "  </tbody>\n",
       "</table>\n",
       "</div>"
      ],
      "text/plain": [
       "  Feature  Importance\n",
       "2      X3    0.200417\n",
       "4      X5    0.200328\n",
       "0      X1    0.176122\n",
       "1      X2    0.173256\n",
       "3      X4    0.136417\n",
       "5      X6    0.113461"
      ]
     },
     "execution_count": 31,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "#Checking feature importance in the XG Boost Model\n",
    "xgb_imp"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 32,
   "id": "416d55fc",
   "metadata": {},
   "outputs": [],
   "source": [
    "df2 = df.drop(['X6'], axis=1)\n",
    "# Drop X6 (usefulness of the app), appears to be the least important feature.\
n",
    "# Create a new dataframe df2 with just the single feature X6 being droppped"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 33,
   "id": "ec17e4c6",
   "metadata": {},
   "outputs": [],
   "source": [
    "df_metrics2, rfc_imp2, xgb_imp2 = model_sequence(df2, 3)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 34,
   "id": "23da28a7",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<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>Train Accuracy</th>\n",
"      <th>Test Accuracy</th>\n",
"      <th>Precision GP 0</th>\n",
"      <th>Precision GP 1</th>\n",
"      <th>Recall GP 0</th>\n",
"      <th>Recall GP 1</th>\n",
"      <th>F1 Score GP 0</th>\n",
"      <th>F1 Score GP 1</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>Logistic Regression</th>\n",
"      <td>0.534091</td>\n",
"      <td>0.657895</td>\n",
"      <td>0.700000</td>\n",
"      <td>0.642857</td>\n",
"      <td>0.411765</td>\n",
"      <td>0.857143</td>\n",
"      <td>0.518519</td>\n",
"      <td>0.734694</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>Support Vector</th>\n",
"      <td>0.795455</td>\n",
"      <td>0.578947</td>\n",
"      <td>0.533333</td>\n",
"      <td>0.608696</td>\n",
"      <td>0.470588</td>\n",
"      <td>0.666667</td>\n",
"      <td>0.500000</td>\n",
"      <td>0.636364</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>Random Forest</th>\n",
"      <td>0.886364</td>\n",
"      <td>0.736842</td>\n",
"      <td>0.818182</td>\n",
"      <td>0.703704</td>\n",
"      <td>0.529412</td>\n",
"      <td>0.904762</td>\n",
"      <td>0.642857</td>\n",
"      <td>0.791667</td>\n",
"    </tr>\n",
"    <tr>\n",
       "      <th>XG Boost</th>\n",
       "      <td>0.886364</td>\n",
       "      <td>0.763158</td>\n",
       "      <td>0.750000</td>\n",
       "      <td>0.772727</td>\n",
       "      <td>0.705882</td>\n",
       "      <td>0.809524</td>\n",
       "      <td>0.727273</td>\n",
       "      <td>0.790698</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "</div>"
      ],
      "text/plain": [
       "                     Train Accuracy  Test Accuracy  Precision GP 0  \\\n",
       "Logistic Regression        0.534091       0.657895        0.700000   \n",
       "Support Vector             0.795455       0.578947        0.533333   \n",
       "Random Forest              0.886364       0.736842        0.818182   \n",
       "XG Boost                   0.886364       0.763158        0.750000   \n",
       "\n",
       "                     Precision GP 1  Recall GP 0  Recall GP 1  F1 Score GP
0  \\\n",
       "Logistic Regression        0.642857     0.411765     0.857143
0.518519   \n",
       "Support Vector             0.608696     0.470588     0.666667
0.500000   \n",
       "Random Forest              0.703704     0.529412     0.904762
0.642857   \n",
       "XG Boost                   0.772727     0.705882     0.809524
0.727273   \n",
       "\n",
       "                     F1 Score GP 1  \n",
       "Logistic Regression       0.734694  \n",
       "Support Vector            0.636364  \n",
       "Random Forest             0.791667  \n",
       "XG Boost                  0.790698  "
      ]
     },
     "execution_count": 34,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df_metrics2"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 35,
   "id": "8a64e129",
   "metadata": {},
   "outputs": [],
   "source": [
    "from sklearn.model_selection import GridSearchCV"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 36,
   "id": "89c8aa73",
   "metadata": {},
   "outputs": [],
   "source": [
    "X = df2.drop('Y', axis=1)\n",
    "y = df2.Y\n",
    "X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.3, random_state = 5)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 37,
   "id": "72bbcde1",
   "metadata": {},
   "outputs": [],
   "source": [
    "xgb = XGBClassifier(use_label_encoder=False, eval_metric='mlogloss')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 38,
   "id": "58e1d6d3",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Fitting 5 folds for each of 108 candidates, totalling 540 fits\n",
      "Best parameters: {'colsample_bytree': 0.8, 'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 100}\n"
     ]
    }
   ],
   "source": [
    "params = { 'max_depth': [3,5, 6],\n",
    "           'learning_rate': [0.01, 0.5, 0.6, 0.7],\n",
    "           'n_estimators': [100, 500, 1000],\n",
    "           'colsample_bytree': [0.7, 0.8, 0.9]}\n",
    "\n",
    "clf = GridSearchCV(estimator=xgb, \n",
    "                   param_grid=params,\n",
    "                   scoring='neg_mean_squared_error', \n",
    "                   verbose=1)\n",
    "\n",
    "clf.fit(X,y)\n",
    "\n",
    "print(\"Best parameters:\", clf.best_params_)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "479b7a17",
   "metadata": {},
   "outputs": [],
   "source": [
    "xgb2.fit(X_train, y_train)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "53e4341b",
   "metadata": {},
   "outputs": [],
   "source": [
    "xgb2.predict(X_train, y_train)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "9cdb87d9",
   "metadata": {},
   "outputs": [],
   "source": [
    "#train a new xgb model with the best parameters\n",
    "xgb2 = XGBClassifier(use_label_encoder=False, eval_metric='mlogloss',
colsample_bytree = 0.8, learning_rate = 0.01, max_depth = 3, n_estimators = 100)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "8c256d7c",
   "metadata": {},
   "outputs": [],
   "source": [
    "#the 'best parameters' did not yield metrics better than what worked with this
model previously. Why is that?\n",
    "pred = xgb2.predict(X_test) #accuracy was not able to go over 76%\n",
    "print(classification_report(y_test,pred))\n",
    "print(confusion_matrix(y_test,pred)) #The user found the learning rate .01,
max-depth 3, n_estimators:100 as the best parameters."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "da35369b",
   "metadata": {},
   "outputs": [],
   "source": [
    "#Concluding Remarks:\n",
    "#The user found the learning rate .01, max-depth 3, n_estimators:100 as the
best parameters.\n",
    "#accuracy was not able to go over 76%. The accuracy is not able to be
improved.\n",
    "#XG Boost fares better than a Random Forest model for this specific dataset in
predicting customer happiness. X6 (the app makes ordering easy for me) can be
excluded in the future from the questionnaire. "
   ]
  }
```

```
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.9.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```