```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "metadata": {
    "collapsed": false
   },
   "outputs": [
    {
     "ename": "NameError",
     "evalue": "name 'null' is not defined",
     "output_type": "error",
     "traceback": [
      "\u001b[1;31m----------------------------------------------------------------------
-----\u001b[0m",
      "\u001b[1;31mNameError\u001b[0m                                 Traceback (most
recent call last)",
      "\u001b[1;32m<ipython-input-1-80b20dff0a42>\u001b[0m                                 in
\u001b[0;36m<module>\u001b[1;34m()\u001b[0m\n\u001b[0;32m                     21\u001b[0m
{\n\u001b[0;32m                                                           22\u001b[0m
\u001b[1;34m\"cell_type\"\u001b[0m\u001b[1;33m:\u001b[0m
\u001b[1;34m\"code\"\u001b[0m\u001b[1;33m,\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u
001b[1;32m--->                                                        23\u001b[1;33m
\u001b[1;34m\"execution_count\"\u001b[0m\u001b[1;33m:\u001b[0m
\u001b[0mnull\u001b[0m\u001b[1;33m,\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0
m\u001b[0;32m      24\u001b[0m      \"metadata\": {\n\u001b[0;32m      25\u001b[0m
\u001b[1;34m\"id\"\u001b[0m\u001b[1;33m:\u001b[0m            \u001b[1;34m\"s07c7JK7Oqt-
\"\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",
      "\u001b[1;31mNameError\u001b[0m: name 'null' is not defined"
     ]
    }
   ],
   "source": [
    "{\n",
    " \"cells\": [\n",
    "  {\n",
    "   \"cell_type\": \"markdown\",\n",
    "   \"metadata\": {\n",
    "    \"id\": \"McSxJAwcOdZ1\"\n",
    "   },\n",
    "   \"source\": [\n",
    "    \"# Basic Python\"\n",
    "   ]\n",
    "  },\n",
    "  {\n",
    "   \"cell_type\": \"markdown\",\n",
    "   \"metadata\": {\n",
    "    \"id\": \"CU48hgo4Owz5\"\n",
    "   },\n",
    "   \"source\": [\n",
    "    \"## 1. Split this string\"\n",
    "   ]\n",
    "  },\n",
    "  {\n",
    "   \"cell_type\": \"code\",\n",
"   \"execution_count\": null,\n",
"   \"metadata\": {\n",
"    \"id\": \"s07c7JK7Oqt-\"\n",
"   },\n",
"   \"outputs\": [],\n",
"   \"source\": [\n",
"    \"s = \\\"Hi there Sam!\\\"\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": 1,\n",
"   \"metadata\": {\n",
"    \"id\": \"6mGVa3SQYLkb\"\n",
"   },\n",
"   \"outputs\": [\n",
"    {\n",
"     \"data\": {\n",
"      \"text/plain\": [\n",
"       \"['Hi', 'there', 'Sam!']\"\n",
"      ]\n",
"     },\n",
"     \"execution_count\": 1,\n",
"     \"metadata\": {},\n",
"     \"output_type\": \"execute_result\"\n",
"    }\n",
"   ],\n",
"   \"source\": [\n",
"    \"s = \\\"Hi there Sam!\\\"\\n\",\n",
"    \"s.split()\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"markdown\",\n",
"   \"metadata\": {\n",
"    \"id\": \"GH1QBn8HP375\"\n",
"   },\n",
"   \"source\": [\n",
"    \"## 2. Use .format() to print the following string. \\n\",\n",
"    \"\\n\",\n",
"    \"### Output should be: The diameter of Earth is 12742 kilometers.\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": null,\n",
"   \"metadata\": {\n",
"    \"id\": \"_ZHoml3kPqic\"\n",
"   },\n",
"   \"outputs\": [],\n",
"   \"source\": [\n",
"    \"planet = \\\"Earth\\\"\\n\",\n",
"    \"diameter = 12742\"\n",
"   ]\n",
"  },\n",
"  {\n",
"      \"cell_type\": \"code\",\n",
"      \"execution_count\": 2,\n",
"      \"metadata\": {\n",
"        \"id\": \"HyRyJv6CYPb4\"\n",
"      },\n",
"      \"outputs\": [\n",
"        {\n",
"          \"name\": \"stdout\",\n",
"          \"output_type\": \"stream\",\n",
"          \"text\": [\n",
"            \"The diameter of Earth is 12742 kilometers.\\n\"\n",
"          ]\n",
"        }\n",
"      ],\n",
"      \"source\": [\n",
"        \"planet = \\\"Earth\\\"\\n\",\n",
"        \"diameter = 12742\\n\",\n",
"        \"print(\\\"The diameter of {} is {} kilometers.\\\".format(planet,diameter))\"\n",
"      ]\n",
"    },\n",
"    {\n",
"      \"cell_type\": \"markdown\",\n",
"      \"metadata\": {\n",
"        \"id\": \"KE74ZEwkRExZ\"\n",
"      },\n",
"      \"source\": [\n",
"        \"## 3. In this nest dictionary grab the word \\\"hello\\\"\"\n",
"      ]\n",
"    },\n",
"    {\n",
"      \"cell_type\": \"code\",\n",
"      \"execution_count\": null,\n",
"      \"metadata\": {\n",
"        \"id\": \"fcVwbCc1QrQI\"\n",
"      },\n",
"      \"outputs\": [],\n",
"      \"source\": [\n",
"        \"d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}\"\n",
"      ]\n",
"    },\n",
"    {\n",
"      \"cell_type\": \"code\",\n",
"      \"execution_count\": 3,\n",
"      \"metadata\": {\n",
"        \"id\": \"MvbkMZpXYRaw\"\n",
"      },\n",
"      \"outputs\": [\n",
"        {\n",
"          \"data\": {\n",
"            \"text/plain\": [\n",
"              \"'hello'\"\n",
"            ]\n",
"          },\n",
"          \"execution_count\": 3,\n",
"          \"metadata\": {},\n",
"          \"output_type\": \"execute_result\"\n",
"    }\n",
"   ],\n",
"   \"source\": [\n",
"    \"d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}\\n\",\n",
"    \"d['k1'][3]['tricky'][3]['target'][3]\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"markdown\",\n",
"   \"metadata\": {\n",
"    \"id\": \"bw0vVp-9ddjv\"\n",
"   },\n",
"   \"source\": [\n",
"    \"# Numpy\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": null,\n",
"   \"metadata\": {\n",
"    \"id\": \"LLiE_TYrhA1O\"\n",
"   },\n",
"   \"outputs\": [],\n",
"   \"source\": [\n",
"    \"import numpy as np\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"markdown\",\n",
"   \"metadata\": {\n",
"    \"id\": \"wOg8hinbgx30\"\n",
"   },\n",
"   \"source\": [\n",
"    \"## 4.1 Create an array of 10 zeros? \\n\",\n",
"    \"## 4.2 Create an array of 10 fives?\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": 4,\n",
"   \"metadata\": {\n",
"    \"id\": \"NHrirmgCYXvU\"\n",
"   },\n",
"   \"outputs\": [\n",
"    {\n",
"     \"data\": {\n",
"      \"text/plain\": [\n",
"       \"array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])\"\n",
"      ]\n",
"     },\n",
"     \"execution_count\": 4,\n",
"     \"metadata\": {},\n",
"     \"output_type\": \"execute_result\"\n",
"    }\n",
"   ],\n",
"   \"source\": [\n",
"    \"import numpy as np\\n\",\n",
"    \"np.zeros(10)\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": 5,\n",
"   \"metadata\": {\n",
"    \"id\": \"e4005lsTYXxx\"\n",
"   },\n",
"   \"outputs\": [\n",
"    {\n",
"     \"data\": {\n",
"      \"text/plain\": [\n",
"       \"array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])\"\n",
"      ]\n",
"     },\n",
"     \"execution_count\": 5,\n",
"     \"metadata\": {},\n",
"     \"output_type\": \"execute_result\"\n",
"    }\n",
"   ],\n",
"   \"source\": [\n",
"    \"import numpy as np\\n\",\n",
"    \"np.ones(10)*5\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"markdown\",\n",
"   \"metadata\": {\n",
"    \"id\": \"gZHHDUBvrMX4\"\n",
"   },\n",
"   \"source\": [\n",
"    \"## 5. Create an array of all the even integers from 20 to 35\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": 6,\n",
"   \"metadata\": {\n",
"    \"id\": \"oAI2tbU2Yag-\"\n",
"   },\n",
"   \"outputs\": [\n",
"    {\n",
"     \"data\": {\n",
"      \"text/plain\": [\n",
"       \"array([20, 22, 24, 26, 28, 30, 32, 34])\"\n",
"      ]\n",
"     },\n",
"     \"execution_count\": 6,\n",
"     \"metadata\": {},\n",
"     \"output_type\": \"execute_result\"\n",
"    }\n",
"   ],\n",
"   \"source\": [\n",
"    \"import numpy as np\\n\",\n",
"    \"np.arange(20,35,2)\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"markdown\",\n",
"   \"metadata\": {\n",
"    \"id\": \"NaOM308NsRpZ\"\n",
"   },\n",
"   \"source\": [\n",
"    \"## 6. Create a 3x3 matrix with values ranging from 0 to 8\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": 7,\n",
"   \"metadata\": {\n",
"    \"id\": \"tOlEVH7BYceE\"\n",
"   },\n",
"   \"outputs\": [\n",
"    {\n",
"     \"data\": {\n",
"      \"text/plain\": [\n",
"       \"array([[0, 1, 2],\\n\",\n",
"       \"       [3, 4, 5],\\n\",\n",
"       \"       [6, 7, 8]])\"\n",
"      ]\n",
"     },\n",
"     \"execution_count\": 7,\n",
"     \"metadata\": {},\n",
"     \"output_type\": \"execute_result\"\n",
"    }\n",
"   ],\n",
"   \"source\": [\n",
"    \"import numpy as np\\n\",\n",
"    \"np.arange(0,9).reshape(3,3)\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"markdown\",\n",
"   \"metadata\": {\n",
"    \"id\": \"hQ0dnhAQuU_p\"\n",
"   },\n",
"   \"source\": [\n",
"    \"## 7. Concatenate a and b \\n\",\n",
"    \"## a = np.array([1, 2, 3]), b = np.array([4, 5, 6])\"\n",
"   ]\n",
"  },\n",
"  {\n",
"   \"cell_type\": \"code\",\n",
"   \"execution_count\": 8,\n",
"   \"metadata\": {\n",
"    \"id\": \"rAPSw97aYfE0\"\n",
"   },\n",
"   \"outputs\": [\n",
"    {\n",
"     \"data\": {\n",
"          \"text/plain\": [\n",
"            \"array([1, 2, 3, 4, 5, 6])\"\n",
"          ]\n",
"        },\n",
"        \"execution_count\": 8,\n",
"        \"metadata\": {},\n",
"        \"output_type\": \"execute_result\"\n",
"      }\n",
"    ],\n",
"    \"source\": [\n",
"      \"import numpy as np\\n\",\n",
"      \"a=np.array([1,2,3])\\n\",\n",
"      \"b=np.array([4,5,6])\\n\",\n",
"      \"np.concatenate((a,b))\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"markdown\",\n",
"    \"metadata\": {\n",
"      \"id\": \"dlPEY9DRwZga\"\n",
"    },\n",
"    \"source\": [\n",
"      \"# Pandas\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"markdown\",\n",
"    \"metadata\": {\n",
"      \"id\": \"ijoYW51zwr87\"\n",
"    },\n",
"    \"source\": [\n",
"      \"## 8. Create a dataframe with 3 rows and 2 columns\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"code\",\n",
"    \"execution_count\": null,\n",
"    \"metadata\": {\n",
"      \"id\": \"T5OxJRZ8uvR7\"\n",
"    },\n",
"    \"outputs\": [],\n",
"    \"source\": [\n",
"      \"import pandas as pd\\n\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"code\",\n",
"    \"execution_count\": 10,\n",
"    \"metadata\": {\n",
"      \"id\": \"xNpI_XXoYhs0\"\n",
"    },\n",
"    \"outputs\": [\n",
"      {\n",
"        \"data\": {\n",
"          \"text/html\": [\n",
"            \"<div>\\n\",\n",
"        \"<style scoped>\\n\",\n",
"        \"    .dataframe tbody tr th:only-of-type {\\n\",\n",
"        \"        vertical-align: middle;\\n\",\n",
"        \"    }\\n\",\n",
"        \"\\n\",\n",
"        \"    .dataframe tbody tr th {\\n\",\n",
"        \"        vertical-align: top;\\n\",\n",
"        \"    }\\n\",\n",
"        \"\\n\",\n",
"        \"    .dataframe thead th {\\n\",\n",
"        \"        text-align: right;\\n\",\n",
"        \"    }\\n\",\n",
"        \"</style>\\n\",\n",
"        \"<table border=\\\"1\\\" class=\\\"dataframe\\\">\\n\",\n",
"        \"  <thead>\\n\",\n",
"        \"    <tr style=\\\"text-align: right;\\\">\\n\",\n",
"        \"      <th></th>\\n\",\n",
"        \"      <th>0</th>\\n\",\n",
"        \"      <th>1</th>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"  </thead>\\n\",\n",
"        \"  <tbody>\\n\",\n",
"        \"    <tr>\\n\",\n",
"        \"      <th>0</th>\\n\",\n",
"        \"      <td>1</td>\\n\",\n",
"        \"      <td>2</td>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"    <tr>\\n\",\n",
"        \"      <th>1</th>\\n\",\n",
"        \"      <td>3</td>\\n\",\n",
"        \"      <td>4</td>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"    <tr>\\n\",\n",
"        \"      <th>2</th>\\n\",\n",
"        \"      <td>5</td>\\n\",\n",
"        \"      <td>6</td>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"  </tbody>\\n\",\n",
"        \"</table>\\n\",\n",
"        \"</div>\"\n",
"       ],\n",
"       \"text/plain\": [\n",
"        \"   0  1\\n\",\n",
"        \"0  1  2\\n\",\n",
"        \"1  3  4\\n\",\n",
"        \"2  5  6\"\n",
"       ]\n",
"      },\n",
"      \"execution_count\": 10,\n",
"      \"metadata\": {},\n",
"      \"output_type\": \"execute_result\"\n",
"     }\n",
"    ],\n",
"    \"source\": [\n",
"     \"import pandas as pd\\n\",\n",
"     \"pd.DataFrame([[1,2],[3,4],[5,6]])\\n\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"markdown\",\n",
"    \"metadata\": {\n",
"     \"id\": \"UXSmdNclyJQD\"\n",
"    },\n",
"    \"source\": [\n",
"     \"## 9. Generate the series of dates from 1st Jan, 2023 to 10th Feb, 2023\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"code\",\n",
"    \"execution_count\": 11,\n",
"    \"metadata\": {\n",
"     \"id\": \"dgyC0JhVYl4F\"\n",
"    },\n",
"    \"outputs\": [\n",
"     {\n",
"      \"data\": {\n",
"       \"text/plain\": [\n",
"        \"DatetimeIndex(['2023-01-01', '2023-01-02', '2023-01-03', '2023-01-04',\\n\",\n",
"           \"                         '2023-01-05', '2023-01-06', '2023-01-07', '2023-01-
08',\\n\",\n",
"        \"               '2023-01-09', '2023-01-10', '2023-01-11', '2023-01-12',\\n\",\n",
"        \"               '2023-01-13', '2023-01-14', '2023-01-15', '2023-01-16',\\n\",\n",
"        \"               '2023-01-17', '2023-01-18', '2023-01-19', '2023-01-20',\\n\",\n",
"        \"               '2023-01-21', '2023-01-22', '2023-01-23', '2023-01-24',\\n\",\n",
"        \"               '2023-01-25', '2023-01-26', '2023-01-27', '2023-01-28',\\n\",\n",
"        \"               '2023-01-29', '2023-01-30', '2023-01-31', '2023-02-01',\\n\",\n",
"           \"                         '2023-02-02', '2023-02-03', '2023-02-04', '2023-02-
05',\\n\",\n",
"           \"                         '2023-02-06', '2023-02-07', '2023-02-08', '2023-02-
09',\\n\",\n",
"        \"               '2023-02-10'],\\n\",\n",
"        \"              dtype='datetime64[ns]', freq='D')\"\n",
"       ]\n",
"      },\n",
"      \"execution_count\": 11,\n",
"      \"metadata\": {},\n",
"      \"output_type\": \"execute_result\"\n",
"     }\n",
"    ],\n",
"    \"source\": [\n",
"     \"import pandas as pd\\n\",\n",
"     \"pd.date_range(start='1/1/2023', end='02/10/2023')\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"markdown\",\n",
"    \"metadata\": {\n",
"     \"id\": \"ZizSetD-y5az\"\n",
"    },\n",
"    \"source\": [\n",
"     \"## 10. Create 2D list to DataFrame\\n\",\n",
"     \"\\n\",\n",
"     \"lists = [[1, 'aaa', 22],\\n\",\n",
"     \"         [2, 'bbb', 25],\\n\",\n",
"     \"         [3, 'ccc', 24]]\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"code\",\n",
"    \"execution_count\": null,\n",
"    \"metadata\": {\n",
"     \"id\": \"_XMC8aEt0llB\"\n",
"    },\n",
"    \"outputs\": [],\n",
"    \"source\": [\n",
"     \"lists = [[1, 'aaa', 22], [2, 'bbb', 25], [3, 'ccc', 24]]\"\n",
"    ]\n",
"   },\n",
"   {\n",
"    \"cell_type\": \"code\",\n",
"    \"execution_count\": 12,\n",
"    \"metadata\": {\n",
"     \"id\": \"knH76sDKYsVX\"\n",
"    },\n",
"    \"outputs\": [\n",
"     {\n",
"      \"data\": {\n",
"       \"text/html\": [\n",
"        \"<div>\\n\",\n",
"        \"<style scoped>\\n\",\n",
"        \"    .dataframe tbody tr th:only-of-type {\\n\",\n",
"        \"        vertical-align: middle;\\n\",\n",
"        \"    }\\n\",\n",
"        \"\\n\",\n",
"        \"    .dataframe tbody tr th {\\n\",\n",
"        \"        vertical-align: top;\\n\",\n",
"        \"    }\\n\",\n",
"        \"\\n\",\n",
"        \"    .dataframe thead th {\\n\",\n",
"        \"        text-align: right;\\n\",\n",
"        \"    }\\n\",\n",
"        \"</style>\\n\",\n",
"        \"<table border=\\\"1\\\" class=\\\"dataframe\\\">\\n\",\n",
"        \"  <thead>\\n\",\n",
"        \"    <tr style=\\\"text-align: right;\\\">\\n\",\n",
"        \"      <th></th>\\n\",\n",
"        \"      <th>0</th>\\n\",\n",
"        \"      <th>1</th>\\n\",\n",
"        \"      <th>2</th>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"  </thead>\\n\",\n",
"        \"  <tbody>\\n\",\n",
"        \"    <tr>\\n\",\n",
"        \"      <th>0</th>\\n\",\n",
"        \"      <td>1</td>\\n\",\n",
"        \"      <td>aaa</td>\\n\",\n",
"        \"      <td>22</td>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"    <tr>\\n\",\n",
"        \"      <th>1</th>\\n\",\n",
"        \"      <td>2</td>\\n\",\n",
"        \"      <td>bbb</td>\\n\",\n",
"        \"      <td>25</td>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"    <tr>\\n\",\n",
"        \"      <th>2</th>\\n\",\n",
"        \"      <td>3</td>\\n\",\n",
"        \"      <td>ccc</td>\\n\",\n",
"        \"      <td>24</td>\\n\",\n",
"        \"    </tr>\\n\",\n",
"        \"  </tbody>\\n\",\n",
"        \"</table>\\n\",\n",
"        \"</div>\"\n",
"       ],\n",
"       \"text/plain\": [\n",
"        \"   0    1   2\\n\",\n",
"        \"0  1  aaa  22\\n\",\n",
"        \"1  2  bbb  25\\n\",\n",
"        \"2  3  ccc  24\"\n",
"       ]\n",
"      },\n",
"      \"execution_count\": 12,\n",
"      \"metadata\": {},\n",
"      \"output_type\": \"execute_result\"\n",
"     }\n",
"    ],\n",
"    \"source\": [\n",
"     \"import pandas as pd\\n\",\n",
"     \"lists = [[1, 'aaa', 22], [2, 'bbb', 25], [3, 'ccc', 24]]\\n\",\n",
"     \"pd.DataFrame(lists)\"\n",
"    ]\n",
"   }\n",
" ],\n",
" \"metadata\": {\n",
"   \"colab\": {\n",
"    \"collapsed_sections\": [],\n",
"    \"provenance\": []\n",
"   },\n",
"   \"kernelspec\": {\n",
"    \"display_name\": \"Python 3\",\n",
"    \"language\": \"python\",\n",
"    \"name\": \"python3\"\n",
"   },\n",
"   \"language_info\": {\n",
"    \"codemirror_mode\": {\n",
"     \"name\": \"ipython\",\n",
"     \"version\": 3\n",
"    },\n",
"    \"file_extension\": \".py\",\n",
"    \"mimetype\": \"text/x-python\",\n",
"    \"name\": \"python\",\n",
"    \"nbconvert_exporter\": \"python\",\n",
"    \"pygments_lexer\": \"ipython3\",\n",
"    \"version\": \"3.8.8\"\n",
    "   }\n",
    " },\n",
    " \"nbformat\": 4,\n",
    " \"nbformat_minor\": 1\n",
    "}\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "collapsed": true
   },
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
   "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": "3.5.1"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 0
}
```