{

"nbformat": 4,

"nbformat\_minor": 0,

"metadata": {

"colab": {

"provenance": [],

"collapsed\_sections": []

},

"kernelspec": {

"name": "python3",

"display\_name": "Python 3"

},

"language\_info": {

"name": "python"

}

},

"cells": [

{

"cell\_type": "markdown",

"source": [

"# Basic Python"

],

"metadata": {

"id": "McSxJAwcOdZ1"

}

},

{

"cell\_type": "markdown",

"source": [

"## 1. Split this string"

],

"metadata": {

"id": "CU48hgo4Owz5"

}

},

{

"cell\_type": "code",

"source": [

"s = \"Hi there Sam!\""

],

"metadata": {

"id": "s07c7JK7Oqt-"

},

"execution\_count": 1,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"s.split()"

],

"metadata": {

"id": "6mGVa3SQYLkb",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "6c8a8628-7f79-4f18-d9ce-94e5e813772c"

},

"execution\_count": 2,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"['Hi', 'there', 'Sam!']"

]

},

"metadata": {},

"execution\_count": 2

}

]

},

{

"cell\_type": "markdown",

"source": [

"## 2. Use .format() to print the following string. \n",

"\n",

"### Output should be: The diameter of Earth is 12742 kilometers."

],

"metadata": {

"id": "GH1QBn8HP375"

}

},

{

"cell\_type": "code",

"source": [

"planet = \"Earth\"\n",

"diameter = 12742"

],

"metadata": {

"id": "\_ZHoml3kPqic"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"print(\"The diameter of {} is {} kilometer\".format(planet,diameter\n",

" ))"

],

"metadata": {

"id": "HyRyJv6CYPb4",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "f1be3e4d-a5fa-47a2-c4c1-f01702b350e7"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"The diameter of Earth is 12742 kilometer\n"

]

}

]

},

{

"cell\_type": "markdown",

"source": [

"## 3. In this nest dictionary grab the word \"hello\""

],

"metadata": {

"id": "KE74ZEwkRExZ"

}

},

{

"cell\_type": "code",

"source": [

"d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}"

],

"metadata": {

"id": "fcVwbCc1QrQI"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"d['k1'][3]['tricky'][3]['target'][3]"

],

"metadata": {

"id": "MvbkMZpXYRaw",

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 35

},

"outputId": "562192a8-ea84-44f7-c7ad-6bd08e511512"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"'hello'"

],

"application/vnd.google.colaboratory.intrinsic+json": {

"type": "string"

}

},

"metadata": {},

"execution\_count": 9

}

]

},

{

"cell\_type": "markdown",

"source": [

"# Numpy"

],

"metadata": {

"id": "bw0vVp-9ddjv"

}

},

{

"cell\_type": "code",

"source": [

"import numpy as np"

],

"metadata": {

"id": "LLiE\_TYrhA1O"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

"## 4.1 Create an array of 10 zeros? \n",

"## 4.2 Create an array of 10 fives?"

],

"metadata": {

"id": "wOg8hinbgx30"

}

},

{

"cell\_type": "code",

"source": [

"a=np.zeros(10)\n",

"a"

],

"metadata": {

"id": "NHrirmgCYXvU",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "8376576f-0921-4780-d856-f25f33ff720a"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])"

]

},

"metadata": {},

"execution\_count": 12

}

]

},

{

"cell\_type": "code",

"source": [

"b=np.ones(10)\*5\n",

"b"

],

"metadata": {

"id": "e4005lsTYXxx",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "0131e0ec-100a-42f5-9867-ad4aab2013ec"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])"

]

},

"metadata": {},

"execution\_count": 13

}

]

},

{

"cell\_type": "markdown",

"source": [

"## 5. Create an array of all the even integers from 20 to 35"

],

"metadata": {

"id": "gZHHDUBvrMX4"

}

},

{

"cell\_type": "code",

"source": [

"s=np.arange(20,50,2)\n",

"s"

],

"metadata": {

"id": "oAI2tbU2Yag-",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "09b9e929-d046-4017-95d9-15f41f514437"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48])"

]

},

"metadata": {},

"execution\_count": 15

}

]

},

{

"cell\_type": "markdown",

"source": [

"## 6. Create a 3x3 matrix with values ranging from 0 to 8"

],

"metadata": {

"id": "NaOM308NsRpZ"

}

},

{

"cell\_type": "code",

"source": [

"b=np.arange(0,9).reshape(3,3)\n",

"b"

],

"metadata": {

"id": "tOlEVH7BYceE",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "60c7326a-9633-4425-bb39-c062e828d15d"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([[0, 1, 2],\n",

" [3, 4, 5],\n",

" [6, 7, 8]])"

]

},

"metadata": {},

"execution\_count": 17

}

]

},

{

"cell\_type": "markdown",

"source": [

"## 7. Concatenate a and b \n",

"## a = np.array([1, 2, 3]), b = np.array([4, 5, 6])"

],

"metadata": {

"id": "hQ0dnhAQuU\_p"

}

},

{

"cell\_type": "code",

"source": [

"a=np.array([1,2,3])\n",

"b=np.array([4,5,6])\n",

"np.concatenate((a,b))"

],

"metadata": {

"id": "rAPSw97aYfE0",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "7a0cce13-2d3f-4a8c-b9a4-1c9aa60b1575"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"array([1, 2, 3, 4, 5, 6])"

]

},

"metadata": {},

"execution\_count": 18

}

]

},

{

"cell\_type": "markdown",

"source": [

"# Pandas"

],

"metadata": {

"id": "dlPEY9DRwZga"

}

},

{

"cell\_type": "markdown",

"source": [

"## 8. Create a dataframe with 3 rows and 2 columns"

],

"metadata": {

"id": "ijoYW51zwr87"

}

},

{

"cell\_type": "code",

"source": [

"import pandas as pd\n"

],

"metadata": {

"id": "T5OxJRZ8uvR7"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"d = {\"names\":[\"aaa\",\"bbb\",\"ccc\",],\"age\":[21,22,20]}\n",

"df = pd.DataFrame(d)\n",

"df"

],

"metadata": {

"id": "xNpI\_XXoYhs0",

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 143

},

"outputId": "92862b6c-029d-4dff-9879-8732bed4335b"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

" names age\n",

"0 aaa 21\n",

"1 bbb 22\n",

"2 ccc 20"

],

"text/html": [

"\n",

" <div id=\"df-b5679877-3840-42e3-9a1b-a99a23e039fd\">\n",

" <div class=\"colab-df-container\">\n",

" <div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>names</th>\n",

" <th>age</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>aaa</td>\n",

" <td>21</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>bbb</td>\n",

" <td>22</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>ccc</td>\n",

" <td>20</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>\n",

" <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-b5679877-3840-42e3-9a1b-a99a23e039fd')\"\n",

" title=\"Convert this dataframe to an interactive table.\"\n",

" style=\"display:none;\">\n",

" \n",

" <svg xmlns=\"http://www.w3.org/2000/svg\" height=\"24px\"viewBox=\"0 0 24 24\"\n",

" width=\"24px\">\n",

" <path d=\"M0 0h24v24H0V0z\" fill=\"none\"/>\n",

" <path d=\"M18.56 5.44l.94 2.06.94-2.06 2.06-.94-2.06-.94-.94-2.06-.94 2.06-2.06.94zm-11 1L8.5 8.5l.94-2.06 2.06-.94-2.06-.94L8.5 2.5l-.94 2.06-2.06.94zm10 10l.94 2.06.94-2.06 2.06-.94-2.06-.94-.94-2.06-.94 2.06-2.06.94z\"/><path d=\"M17.41 7.96l-1.37-1.37c-.4-.4-.92-.59-1.43-.59-.52 0-1.04.2-1.43.59L10.3 9.45l-7.72 7.72c-.78.78-.78 2.05 0 2.83L4 21.41c.39.39.9.59 1.41.59.51 0 1.02-.2 1.41-.59l7.78-7.78 2.81-2.81c.8-.78.8-2.07 0-2.86zM5.41 20L4 18.59l7.72-7.72 1.47 1.35L5.41 20z\"/>\n",

" </svg>\n",

" </button>\n",

" \n",

" <style>\n",

" .colab-df-container {\n",

" display:flex;\n",

" flex-wrap:wrap;\n",

" gap: 12px;\n",

" }\n",

"\n",

" .colab-df-convert {\n",

" background-color: #E8F0FE;\n",

" border: none;\n",

" border-radius: 50%;\n",

" cursor: pointer;\n",

" display: none;\n",

" fill: #1967D2;\n",

" height: 32px;\n",

" padding: 0 0 0 0;\n",

" width: 32px;\n",

" }\n",

"\n",

" .colab-df-convert:hover {\n",

" background-color: #E2EBFA;\n",

" box-shadow: 0px 1px 2px rgba(60, 64, 67, 0.3), 0px 1px 3px 1px rgba(60, 64, 67, 0.15);\n",

" fill: #174EA6;\n",

" }\n",

"\n",

" [theme=dark] .colab-df-convert {\n",

" background-color: #3B4455;\n",

" fill: #D2E3FC;\n",

" }\n",

"\n",

" [theme=dark] .colab-df-convert:hover {\n",

" background-color: #434B5C;\n",

" box-shadow: 0px 1px 3px 1px rgba(0, 0, 0, 0.15);\n",

" filter: drop-shadow(0px 1px 2px rgba(0, 0, 0, 0.3));\n",

" fill: #FFFFFF;\n",

" }\n",

" </style>\n",

"\n",

" <script>\n",

" const buttonEl =\n",

" document.querySelector('#df-b5679877-3840-42e3-9a1b-a99a23e039fd button.colab-df-convert');\n",

" buttonEl.style.display =\n",

" google.colab.kernel.accessAllowed ? 'block' : 'none';\n",

"\n",

" async function convertToInteractive(key) {\n",

" const element = document.querySelector('#df-b5679877-3840-42e3-9a1b-a99a23e039fd');\n",

" const dataTable =\n",

" await google.colab.kernel.invokeFunction('convertToInteractive',\n",

" [key], {});\n",

" if (!dataTable) return;\n",

"\n",

" const docLinkHtml = 'Like what you see? Visit the ' +\n",

" '<a target=\"\_blank\" href=https://colab.research.google.com/notebooks/data\_table.ipynb>data table notebook</a>'\n",

" + ' to learn more about interactive tables.';\n",

" element.innerHTML = '';\n",

" dataTable['output\_type'] = 'display\_data';\n",

" await google.colab.output.renderOutput(dataTable, element);\n",

" const docLink = document.createElement('div');\n",

" docLink.innerHTML = docLinkHtml;\n",

" element.appendChild(docLink);\n",

" }\n",

" </script>\n",

" </div>\n",

" </div>\n",

" "

]

},

"metadata": {},

"execution\_count": 24

}

]

},

{

"cell\_type": "markdown",

"source": [

"## 9. Generate the series of dates from 1st Jan, 2023 to 10th Feb, 2023"

],

"metadata": {

"id": "UXSmdNclyJQD"

}

},

{

"cell\_type": "code",

"source": [

"m= pd.date\_range(start='1-01-2023',end='10-02-2023')\n",

"for i in m:\n",

" print(i)"

],

"metadata": {

"id": "dgyC0JhVYl4F",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "148bfe6d-e4ea-4796-fa1e-79ad4b21f7d9"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"2023-01-01 00:00:00\n",

"2023-01-02 00:00:00\n",

"2023-01-03 00:00:00\n",

"2023-01-04 00:00:00\n",

"2023-01-05 00:00:00\n",

"2023-01-06 00:00:00\n",

"2023-01-07 00:00:00\n",

"2023-01-08 00:00:00\n",

"2023-01-09 00:00:00\n",

"2023-01-10 00:00:00\n",

"2023-01-11 00:00:00\n",

"2023-01-12 00:00:00\n",

"2023-01-13 00:00:00\n",

"2023-01-14 00:00:00\n",

"2023-01-15 00:00:00\n",

"2023-01-16 00:00:00\n",

"2023-01-17 00:00:00\n",

"2023-01-18 00:00:00\n",

"2023-01-19 00:00:00\n",

"2023-01-20 00:00:00\n",

"2023-01-21 00:00:00\n",

"2023-01-22 00:00:00\n",

"2023-01-23 00:00:00\n",

"2023-01-24 00:00:00\n",

"2023-01-25 00:00:00\n",

"2023-01-26 00:00:00\n",

"2023-01-27 00:00:00\n",

"2023-01-28 00:00:00\n",

"2023-01-29 00:00:00\n",

"2023-01-30 00:00:00\n",

"2023-01-31 00:00:00\n",

"2023-02-01 00:00:00\n",

"2023-02-02 00:00:00\n",

"2023-02-03 00:00:00\n",

"2023-02-04 00:00:00\n",

"2023-02-05 00:00:00\n",

"2023-02-06 00:00:00\n",

"2023-02-07 00:00:00\n",

"2023-02-08 00:00:00\n",

"2023-02-09 00:00:00\n",

"2023-02-10 00:00:00\n",

"2023-02-11 00:00:00\n",

"2023-02-12 00:00:00\n",

"2023-02-13 00:00:00\n",

"2023-02-14 00:00:00\n",

"2023-02-15 00:00:00\n",

"2023-02-16 00:00:00\n",

"2023-02-17 00:00:00\n",

"2023-02-18 00:00:00\n",

"2023-02-19 00:00:00\n",

"2023-02-20 00:00:00\n",

"2023-02-21 00:00:00\n",

"2023-02-22 00:00:00\n",

"2023-02-23 00:00:00\n",

"2023-02-24 00:00:00\n",

"2023-02-25 00:00:00\n",

"2023-02-26 00:00:00\n",

"2023-02-27 00:00:00\n",

"2023-02-28 00:00:00\n",

"2023-03-01 00:00:00\n",

"2023-03-02 00:00:00\n",

"2023-03-03 00:00:00\n",

"2023-03-04 00:00:00\n",

"2023-03-05 00:00:00\n",

"2023-03-06 00:00:00\n",

"2023-03-07 00:00:00\n",

"2023-03-08 00:00:00\n",

"2023-03-09 00:00:00\n",

"2023-03-10 00:00:00\n",

"2023-03-11 00:00:00\n",

"2023-03-12 00:00:00\n",

"2023-03-13 00:00:00\n",

"2023-03-14 00:00:00\n",

"2023-03-15 00:00:00\n",

"2023-03-16 00:00:00\n",

"2023-03-17 00:00:00\n",

"2023-03-18 00:00:00\n",

"2023-03-19 00:00:00\n",

"2023-03-20 00:00:00\n",

"2023-03-21 00:00:00\n",

"2023-03-22 00:00:00\n",

"2023-03-23 00:00:00\n",

"2023-03-24 00:00:00\n",

"2023-03-25 00:00:00\n",

"2023-03-26 00:00:00\n",

"2023-03-27 00:00:00\n",

"2023-03-28 00:00:00\n",

"2023-03-29 00:00:00\n",

"2023-03-30 00:00:00\n",

"2023-03-31 00:00:00\n",

"2023-04-01 00:00:00\n",

"2023-04-02 00:00:00\n",

"2023-04-03 00:00:00\n",

"2023-04-04 00:00:00\n",

"2023-04-05 00:00:00\n",

"2023-04-06 00:00:00\n",

"2023-04-07 00:00:00\n",

"2023-04-08 00:00:00\n",

"2023-04-09 00:00:00\n",

"2023-04-10 00:00:00\n",

"2023-04-11 00:00:00\n",

"2023-04-12 00:00:00\n",

"2023-04-13 00:00:00\n",

"2023-04-14 00:00:00\n",

"2023-04-15 00:00:00\n",

"2023-04-16 00:00:00\n",

"2023-04-17 00:00:00\n",

"2023-04-18 00:00:00\n",

"2023-04-19 00:00:00\n",

"2023-04-20 00:00:00\n",

"2023-04-21 00:00:00\n",

"2023-04-22 00:00:00\n",

"2023-04-23 00:00:00\n",

"2023-04-24 00:00:00\n",

"2023-04-25 00:00:00\n",

"2023-04-26 00:00:00\n",

"2023-04-27 00:00:00\n",

"2023-04-28 00:00:00\n",

"2023-04-29 00:00:00\n",

"2023-04-30 00:00:00\n",

"2023-05-01 00:00:00\n",

"2023-05-02 00:00:00\n",

"2023-05-03 00:00:00\n",

"2023-05-04 00:00:00\n",

"2023-05-05 00:00:00\n",

"2023-05-06 00:00:00\n",

"2023-05-07 00:00:00\n",

"2023-05-08 00:00:00\n",

"2023-05-09 00:00:00\n",

"2023-05-10 00:00:00\n",

"2023-05-11 00:00:00\n",

"2023-05-12 00:00:00\n",

"2023-05-13 00:00:00\n",

"2023-05-14 00:00:00\n",

"2023-05-15 00:00:00\n",

"2023-05-16 00:00:00\n",

"2023-05-17 00:00:00\n",

"2023-05-18 00:00:00\n",

"2023-05-19 00:00:00\n",

"2023-05-20 00:00:00\n",

"2023-05-21 00:00:00\n",

"2023-05-22 00:00:00\n",

"2023-05-23 00:00:00\n",

"2023-05-24 00:00:00\n",

"2023-05-25 00:00:00\n",

"2023-05-26 00:00:00\n",

"2023-05-27 00:00:00\n",

"2023-05-28 00:00:00\n",

"2023-05-29 00:00:00\n",

"2023-05-30 00:00:00\n",

"2023-05-31 00:00:00\n",

"2023-06-01 00:00:00\n",

"2023-06-02 00:00:00\n",

"2023-06-03 00:00:00\n",

"2023-06-04 00:00:00\n",

"2023-06-05 00:00:00\n",

"2023-06-06 00:00:00\n",

"2023-06-07 00:00:00\n",

"2023-06-08 00:00:00\n",

"2023-06-09 00:00:00\n",

"2023-06-10 00:00:00\n",

"2023-06-11 00:00:00\n",

"2023-06-12 00:00:00\n",

"2023-06-13 00:00:00\n",

"2023-06-14 00:00:00\n",

"2023-06-15 00:00:00\n",

"2023-06-16 00:00:00\n",

"2023-06-17 00:00:00\n",

"2023-06-18 00:00:00\n",

"2023-06-19 00:00:00\n",

"2023-06-20 00:00:00\n",

"2023-06-21 00:00:00\n",

"2023-06-22 00:00:00\n",

"2023-06-23 00:00:00\n",

"2023-06-24 00:00:00\n",

"2023-06-25 00:00:00\n",

"2023-06-26 00:00:00\n",

"2023-06-27 00:00:00\n",

"2023-06-28 00:00:00\n",

"2023-06-29 00:00:00\n",

"2023-06-30 00:00:00\n",

"2023-07-01 00:00:00\n",

"2023-07-02 00:00:00\n",

"2023-07-03 00:00:00\n",

"2023-07-04 00:00:00\n",

"2023-07-05 00:00:00\n",

"2023-07-06 00:00:00\n",

"2023-07-07 00:00:00\n",

"2023-07-08 00:00:00\n",

"2023-07-09 00:00:00\n",

"2023-07-10 00:00:00\n",

"2023-07-11 00:00:00\n",

"2023-07-12 00:00:00\n",

"2023-07-13 00:00:00\n",

"2023-07-14 00:00:00\n",

"2023-07-15 00:00:00\n",

"2023-07-16 00:00:00\n",

"2023-07-17 00:00:00\n",

"2023-07-18 00:00:00\n",

"2023-07-19 00:00:00\n",

"2023-07-20 00:00:00\n",

"2023-07-21 00:00:00\n",

"2023-07-22 00:00:00\n",

"2023-07-23 00:00:00\n",

"2023-07-24 00:00:00\n",

"2023-07-25 00:00:00\n",

"2023-07-26 00:00:00\n",

"2023-07-27 00:00:00\n",

"2023-07-28 00:00:00\n",

"2023-07-29 00:00:00\n",

"2023-07-30 00:00:00\n",

"2023-07-31 00:00:00\n",

"2023-08-01 00:00:00\n",

"2023-08-02 00:00:00\n",

"2023-08-03 00:00:00\n",

"2023-08-04 00:00:00\n",

"2023-08-05 00:00:00\n",

"2023-08-06 00:00:00\n",

"2023-08-07 00:00:00\n",

"2023-08-08 00:00:00\n",

"2023-08-09 00:00:00\n",

"2023-08-10 00:00:00\n",

"2023-08-11 00:00:00\n",

"2023-08-12 00:00:00\n",

"2023-08-13 00:00:00\n",

"2023-08-14 00:00:00\n",

"2023-08-15 00:00:00\n",

"2023-08-16 00:00:00\n",

"2023-08-17 00:00:00\n",

"2023-08-18 00:00:00\n",

"2023-08-19 00:00:00\n",

"2023-08-20 00:00:00\n",

"2023-08-21 00:00:00\n",

"2023-08-22 00:00:00\n",

"2023-08-23 00:00:00\n",

"2023-08-24 00:00:00\n",

"2023-08-25 00:00:00\n",

"2023-08-26 00:00:00\n",

"2023-08-27 00:00:00\n",

"2023-08-28 00:00:00\n",

"2023-08-29 00:00:00\n",

"2023-08-30 00:00:00\n",

"2023-08-31 00:00:00\n",

"2023-09-01 00:00:00\n",

"2023-09-02 00:00:00\n",

"2023-09-03 00:00:00\n",

"2023-09-04 00:00:00\n",

"2023-09-05 00:00:00\n",

"2023-09-06 00:00:00\n",

"2023-09-07 00:00:00\n",

"2023-09-08 00:00:00\n",

"2023-09-09 00:00:00\n",

"2023-09-10 00:00:00\n",

"2023-09-11 00:00:00\n",

"2023-09-12 00:00:00\n",

"2023-09-13 00:00:00\n",

"2023-09-14 00:00:00\n",

"2023-09-15 00:00:00\n",

"2023-09-16 00:00:00\n",

"2023-09-17 00:00:00\n",

"2023-09-18 00:00:00\n",

"2023-09-19 00:00:00\n",

"2023-09-20 00:00:00\n",

"2023-09-21 00:00:00\n",

"2023-09-22 00:00:00\n",

"2023-09-23 00:00:00\n",

"2023-09-24 00:00:00\n",

"2023-09-25 00:00:00\n",

"2023-09-26 00:00:00\n",

"2023-09-27 00:00:00\n",

"2023-09-28 00:00:00\n",

"2023-09-29 00:00:00\n",

"2023-09-30 00:00:00\n",

"2023-10-01 00:00:00\n",

"2023-10-02 00:00:00\n"

]

}

]

},

{

"cell\_type": "markdown",

"source": [

"## 10. Create 2D list to DataFrame\n",

"\n",

"lists = [[1, 'aaa', 22],\n",

" [2, 'bbb', 25],\n",

" [3, 'ccc', 24]]"

],

"metadata": {

"id": "ZizSetD-y5az"

}

},

{

"cell\_type": "code",

"source": [

"lists = [[1, 'aaa', 22], [2, 'bbb', 25], [3, 'ccc', 24]]"

],

"metadata": {

"id": "\_XMC8aEt0llB"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"print(pd.DataFrame(lists))"

],

"metadata": {

"id": "knH76sDKYsVX",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "b7cf97e6-eaa3-4390-ec48-29f2d6aa92b7"

},

"execution\_count": null,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

" 0 1 2\n",

"0 1 aaa 22\n",

"1 2 bbb 25\n",

"2 3 ccc 24\n"

]

}

]

}

]

}