{

"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": [

"t=s.split()\n",

"print(t)"

],

"metadata": {

"id": "6mGVa3SQYLkb",

"outputId": "596d6776-1873-4faf-fb94-7594aecb67ac",

"colab": {

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

}

},

"execution\_count": 2,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

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

]

}

]

},

{

"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": 3,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"print(\"The diameter of {} is {} kilometers.\".format(planet,diameter));"

],

"metadata": {

"id": "HyRyJv6CYPb4",

"outputId": "f8ca53e6-1493-4788-b3ea-ac19e35a7d3f",

"colab": {

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

}

},

"execution\_count": 4,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"The diameter of Earth is 12742 kilometers.\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": 5,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"print(d['k1'][3][\"tricky\"][3]['target'][3])"

],

"metadata": {

"id": "MvbkMZpXYRaw",

"outputId": "59b20f8d-c593-4a8d-be0e-0450d0cfcc3f",

"colab": {

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

}

},

"execution\_count": 6,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"hello\n"

]

}

]

},

{

"cell\_type": "markdown",

"source": [

"# Numpy"

],

"metadata": {

"id": "bw0vVp-9ddjv"

}

},

{

"cell\_type": "code",

"source": [

"import numpy as np"

],

"metadata": {

"id": "LLiE\_TYrhA1O"

},

"execution\_count": 7,

"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": [

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

"print(arr)"

],

"metadata": {

"id": "NHrirmgCYXvU",

"outputId": "b70310be-2c96-40ff-a996-1be53e8e3bab",

"colab": {

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

}

},

"execution\_count": 9,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"arr1=np.ones(10)\*5\n",

"print(arr1)"

],

"metadata": {

"id": "e4005lsTYXxx"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "markdown",

"source": [

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

],

"metadata": {

"id": "gZHHDUBvrMX4"

}

},

{

"cell\_type": "code",

"source": [

"array=(np.arange(20,36,2))\n",

"print(array)"

],

"metadata": {

"id": "oAI2tbU2Yag-",

"outputId": "e41fc6c8-6fdd-4316-af90-91468a15328a",

"colab": {

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

}

},

"execution\_count": 14,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"[20 22 24 26 28 30 32 34]\n"

]

}

]

},

{

"cell\_type": "markdown",

"source": [

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

],

"metadata": {

"id": "NaOM308NsRpZ"

}

},

{

"cell\_type": "code",

"source": [

"arr2=np.arange(0,9).reshape((3,3))\n",

"print(arr2)"

],

"metadata": {

"id": "tOlEVH7BYceE",

"outputId": "a61c08ff-5aa2-49aa-f231-ff63df2c73d8",

"colab": {

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

}

},

"execution\_count": 16,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"[[0 1 2]\n",

" [3 4 5]\n",

" [6 7 8]]\n"

]

}

]

},

{

"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",

"ab=np.concatenate((a,b))\n",

"print(ab)"

],

"metadata": {

"id": "rAPSw97aYfE0",

"outputId": "13939c3e-5d19-48dd-b336-15cd7b66eb63",

"colab": {

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

}

},

"execution\_count": 17,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"[1 2 3 4 5 6]\n"

]

}

]

},

{

"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": 18,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"data={\"domain\":['AI','ML','DS'],\"interest\":['yes','yes','yes']}\n",

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

"print(df)"

],

"metadata": {

"id": "xNpI\_XXoYhs0",

"outputId": "ed5c7533-3628-43a7-c3bf-52a115307b0f",

"colab": {

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

}

},

"execution\_count": 19,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

" domain interest\n",

"0 AI yes\n",

"1 ML yes\n",

"2 DS yes\n"

]

}

]

},

{

"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": [

"pd.date\_range(\"01-01-2023\",\"02-10-2023\")"

],

"metadata": {

"id": "dgyC0JhVYl4F",

"outputId": "88f1ceb9-8baa-4fea-d34d-49a7c4adaf83",

"colab": {

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

}

},

"execution\_count": 20,

"outputs": [

{

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

"data": {

"text/plain": [

"DatetimeIndex(['2023-01-01', '2023-01-02', '2023-01-03', '2023-01-04',\n",

" '2023-01-05', '2023-01-06', '2023-01-07', '2023-01-08',\n",

" '2023-01-09', '2023-01-10', '2023-01-11', '2023-01-12',\n",

" '2023-01-13', '2023-01-14', '2023-01-15', '2023-01-16',\n",

" '2023-01-17', '2023-01-18', '2023-01-19', '2023-01-20',\n",

" '2023-01-21', '2023-01-22', '2023-01-23', '2023-01-24',\n",

" '2023-01-25', '2023-01-26', '2023-01-27', '2023-01-28',\n",

" '2023-01-29', '2023-01-30', '2023-01-31', '2023-02-01',\n",

" '2023-02-02', '2023-02-03', '2023-02-04', '2023-02-05',\n",

" '2023-02-06', '2023-02-07', '2023-02-08', '2023-02-09',\n",

" '2023-02-10'],\n",

" dtype='datetime64[ns]', freq='D')"

]

},

"metadata": {},

"execution\_count": 20

}

]

},

{

"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": 22,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"df=pd.DataFrame(lists,columns=['Number','Name','ID.No'])\n",

"print(df)"

],

"metadata": {

"id": "knH76sDKYsVX",

"outputId": "4319e300-495f-41c4-9568-159e6c9431f0",

"colab": {

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

}

},

"execution\_count": 23,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

" Number Name ID.No\n",

"0 1 aaa 22\n",

"1 2 bbb 25\n",

"2 3 ccc 24\n"

]

}

]

}

]

}