{

"cells": [

{

"cell\_type": "markdown",

"metadata": {

"id": "McSxJAwcOdZ1"

},

"source": [

"# Basic Python"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "CU48hgo4Owz5"

},

"source": [

"## 1. Split this string"

]

},

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

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

]

},

"execution\_count": 1,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

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

"s.split()"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "GH1QBn8HP375"

},

"source": [

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

"\n",

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

]

},

{

"cell\_type": "code",

"execution\_count": 2,

"metadata": {

"id": "\_ZHoml3kPqic"

},

"outputs": [],

"source": [

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

"diameter = 12742"

]

},

{

"cell\_type": "code",

"execution\_count": 3,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"The diameter of Earth is 12742 kilometers.\n"

]

}

],

"source": [

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

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "KE74ZEwkRExZ"

},

"source": [

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

]

},

{

"cell\_type": "code",

"execution\_count": 5,

"metadata": {

"id": "fcVwbCc1QrQI"

},

"outputs": [

{

"data": {

"text/plain": [

"{'k1': [1,\n",

" 2,\n",

" 3,\n",

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

]

},

"execution\_count": 5,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

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

"d"

]

},

{

"cell\_type": "code",

"execution\_count": 6,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"hello\n"

]

}

],

"source": [

"g=d['k1'][3]['tricky'][3]['target'][3]\n",

"print(g)"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "bw0vVp-9ddjv"

},

"source": [

"# Numpy"

]

},

{

"cell\_type": "code",

"execution\_count": 8,

"metadata": {

"id": "LLiE\_TYrhA1O"

},

"outputs": [],

"source": [

"import numpy as np"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "wOg8hinbgx30"

},

"source": [

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

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

]

},

{

"cell\_type": "code",

"execution\_count": 11,

"metadata": {

"id": "NHrirmgCYXvU"

},

"outputs": [

{

"data": {

"text/plain": [

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

]

},

"execution\_count": 11,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#An array of 10 zeros\n",

"np.zeros(10)"

]

},

{

"cell\_type": "code",

"execution\_count": 10,

"metadata": {

"id": "e4005lsTYXxx"

},

"outputs": [

{

"data": {

"text/plain": [

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

]

},

"execution\_count": 10,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#An array of 10 fives\n",

"np.ones(10)\*5"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"# or"

]

},

{

"cell\_type": "code",

"execution\_count": 12,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"An array of 10 zeros is [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]\n",

"An array of 10 fives is [5. 5. 5. 5. 5. 5. 5. 5. 5. 5.]\n"

]

}

],

"source": [

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

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

"print(\"An array of 10 zeros is {}\".format(a))\n",

"print(\"An array of 10 fives is {}\".format(b))"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "gZHHDUBvrMX4"

},

"source": [

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

]

},

{

"cell\_type": "code",

"execution\_count": 13,

"metadata": {

"id": "oAI2tbU2Yag-"

},

"outputs": [

{

"data": {

"text/plain": [

"array([20, 22, 24, 26, 28, 30, 32, 34])"

]

},

"execution\_count": 13,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"np.arange(20,35,2,dtype=int)"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "NaOM308NsRpZ"

},

"source": [

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

]

},

{

"cell\_type": "code",

"execution\_count": 14,

"metadata": {

"id": "tOlEVH7BYceE"

},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"A 3x3 matrix with values ranging from 0 to 8 is given below\n",

"[[0 1 2]\n",

" [3 4 5]\n",

" [6 7 8]]\n"

]

}

],

"source": [

"import numpy as np\n",

"e=np.arange(9)\n",

"f=e.reshape(3,3)\n",

"print(\"A 3x3 matrix with values ranging from 0 to 8 is given below\")\n",

"print(\"{}\".format(f))"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "hQ0dnhAQuU\_p"

},

"source": [

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

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

]

},

{

"cell\_type": "code",

"execution\_count": 15,

"metadata": {

"id": "rAPSw97aYfE0"

},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Concatination of a and b is [1 2 3 4 5 6]\n"

]

}

],

"source": [

"import numpy as pd\n",

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

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

"cc=np.concatenate((a,b),axis=0)\n",

"print(\"Concatination of a and b is {}\".format(cc))"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "dlPEY9DRwZga"

},

"source": [

"# Pandas"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "ijoYW51zwr87"

},

"source": [

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

]

},

{

"cell\_type": "code",

"execution\_count": 19,

"metadata": {

"id": "T5OxJRZ8uvR7"

},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"A datafram with 3 rows and 2 columns is given below\n",

" 1 2\n",

"1 0 1\n",

"2 2 3\n",

"3 4 5\n"

]

}

],

"source": [

"import pandas as pd\n",

"d=np.arange(6).reshape(3,2)\n",

"c=['1','2']\n",

"r=['1','2','3']\n",

"dataframe=pd.DataFrame(data=d,index=r,columns=c)\n",

"print(\"A datafram with 3 rows and 2 columns is given below\")\n",

"print(\"{}\".format(dataframe))"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "UXSmdNclyJQD"

},

"source": [

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

]

},

{

"cell\_type": "code",

"execution\_count": 20,

"metadata": {

"id": "dgyC0JhVYl4F"

},

"outputs": [

{

"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')"

]

},

"execution\_count": 20,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"import pandas as pd\n",

"pd.date\_range(start='1st/jan/2023',end='10th/feb/2023',inclusive='both')"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "ZizSetD-y5az"

},

"source": [

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

"\n",

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

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

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

]

},

{

"cell\_type": "code",

"execution\_count": 21,

"metadata": {

"id": "\_XMC8aEt0llB"

},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

" S/No Name Rollno\n",

"0 1 aaa 22\n",

"1 2 bbb 25\n",

"2 3 ccc 24\n"

]

}

],

"source": [

"import pandas as pd\n",

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

"df=pd.DataFrame(lists,columns=['S/No','Name','Rollno'])\n",

"print(df)"

]

}

],

"metadata": {

"colab": {

"collapsed\_sections": [],

"provenance": []

},

"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": "3.9.12"

}

},

"nbformat": 4,

"nbformat\_minor": 1

}