{

"metadata": {

"colab": {

"provenance": [],

"collapsed\_sections": []

},

"kernelspec": {

"name": "python",

"display\_name": "Python (Pyodide)",

"language": "python"

},

"language\_info": {

"codemirror\_mode": {

"name": "python",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.8"

}

},

"nbformat\_minor": 4,

"nbformat": 4,

"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": null,

"outputs": []

},

{

"cell\_type": "code",

"source": "# split method for strings\ns = \"Hi there Sam!\"\nstuff = s.split()\nprint(stuff)",

"metadata": {

"id": "6mGVa3SQYLkb",

"trusted": true

},

"execution\_count": 1,

"outputs": [

{

"name": "stdout",

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

"output\_type": "stream"

}

]

},

{

"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\"\ndiameter = 12742",

"metadata": {

"id": "\_ZHoml3kPqic"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": "planet = \"Earth\"\ndiameter = 12742\nprint( 'The diameter of {} is {} kilometers.' .format(planet,diameter));",

"metadata": {

"id": "HyRyJv6CYPb4",

"trusted": true

},

"execution\_count": 3,

"outputs": [

{

"name": "stdout",

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

"output\_type": "stream"

}

]

},

{

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

"metadata": {

"id": "MvbkMZpXYRaw",

"trusted": true

},

"execution\_count": 4,

"outputs": [

{

"name": "stdout",

"text": "hello\n",

"output\_type": "stream"

}

]

},

{

"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": "import numpy as np\narray=np.zeros(10)\nprint(\"An array of 10 zeros:\")\nprint(array)",

"metadata": {

"id": "NHrirmgCYXvU",

"trusted": true

},

"execution\_count": 10,

"outputs": [

{

"name": "stdout",

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

"output\_type": "stream"

}

]

},

{

"cell\_type": "code",

"source": "import numpy as np\narray=np.ones(10)\*5\nprint(\"An array of 10 fives:\")\nprint(array)",

"metadata": {

"id": "e4005lsTYXxx",

"trusted": true

},

"execution\_count": 13,

"outputs": [

{

"name": "stdout",

"text": "An array of 10 fives:\n[5. 5. 5. 5. 5. 5. 5. 5. 5. 5.]\n",

"output\_type": "stream"

}

]

},

{

"cell\_type": "markdown",

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

"metadata": {

"id": "gZHHDUBvrMX4"

}

},

{

"cell\_type": "code",

"source": "import numpy as np\narray=np.arange(30,71,2)\nprint(\"Array of all the even integers from 20 to 35\")\nprint(array) ",

"metadata": {

"id": "oAI2tbU2Yag-",

"trusted": true

},

"execution\_count": 14,

"outputs": [

{

"name": "stdout",

"text": "Array of all the even integers from 20 to 35\n[30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70]\n",

"output\_type": "stream"

}

]

},

{

"cell\_type": "markdown",

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

"metadata": {

"id": "NaOM308NsRpZ"

}

},

{

"cell\_type": "code",

"source": "np.arange(0,9).reshape((3,3))",

"metadata": {

"id": "tOlEVH7BYceE",

"trusted": true

},

"execution\_count": 15,

"outputs": [

{

"execution\_count": 15,

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

"data": {

"text/plain": "array([[0, 1, 2],\n [3, 4, 5],\n [6, 7, 8]])"

},

"metadata": {}

}

]

},

{

"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])\nb = np.array([4,5,6])\nnp.concatenate((a,b),axis=0)",

"metadata": {

"id": "rAPSw97aYfE0",

"trusted": true

},

"execution\_count": 16,

"outputs": [

{

"execution\_count": 16,

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

"data": {

"text/plain": "array([1, 2, 3, 4, 5, 6])"

},

"metadata": {}

}

]

},

{

"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": "",

"metadata": {},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": "import pandas as pd\nd = {\"names\":[\"Harini\",\"alin\",\"keerthana\"],\"age\":[20,19,20]}\ndf = pd.DataFrame(d)\ndf",

"metadata": {

"id": "xNpI\_XXoYhs0",

"trusted": true

},

"execution\_count": 19,

"outputs": [

{

"execution\_count": 19,

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

"data": {

"text/plain": " names age\n0 Harini 20\n1 alin 19\n2 keerthana 20",

"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>names</th>\n <th>age</th>\n </tr>\n </thead>\n <tbody>\n <tr>\n <th>0</th>\n <td>Harini</td>\n <td>20</td>\n </tr>\n <tr>\n <th>1</th>\n <td>alin</td>\n <td>19</td>\n </tr>\n <tr>\n <th>2</th>\n <td>keerthana</td>\n <td>20</td>\n </tr>\n </tbody>\n</table>\n</div>"

},

"metadata": {}

}

]

},

{

"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": "P = pd.date\_range(start='1-1-2023',end='10-2-2023')\nfor val in P:\n print(val);",

"metadata": {

"id": "dgyC0JhVYl4F",

"trusted": true

},

"execution\_count": 20,

"outputs": [

{

"name": "stdout",

"text": "2023-01-01 00:00:00\n2023-01-02 00:00:00\n2023-01-03 00:00:00\n2023-01-04 00:00:00\n2023-01-05 00:00:00\n2023-01-06 00:00:00\n2023-01-07 00:00:00\n2023-01-08 00:00:00\n2023-01-09 00:00:00\n2023-01-10 00:00:00\n2023-01-11 00:00:00\n2023-01-12 00:00:00\n2023-01-13 00:00:00\n2023-01-14 00:00:00\n2023-01-15 00:00:00\n2023-01-16 00:00:00\n2023-01-17 00:00:00\n2023-01-18 00:00:00\n2023-01-19 00:00:00\n2023-01-20 00:00:00\n2023-01-21 00:00:00\n2023-01-22 00:00:00\n2023-01-23 00:00:00\n2023-01-24 00:00:00\n2023-01-25 00:00:00\n2023-01-26 00:00:00\n2023-01-27 00:00:00\n2023-01-28 00:00:00\n2023-01-29 00:00:00\n2023-01-30 00:00:00\n2023-01-31 00:00:00\n2023-02-01 00:00:00\n2023-02-02 00:00:00\n2023-02-03 00:00:00\n2023-02-04 00:00:00\n2023-02-05 00:00:00\n2023-02-06 00:00:00\n2023-02-07 00:00:00\n2023-02-08 00:00:00\n2023-02-09 00:00:00\n2023-02-10 00:00:00\n2023-02-11 00:00:00\n2023-02-12 00:00:00\n2023-02-13 00:00:00\n2023-02-14 00:00:00\n2023-02-15 00:00:00\n2023-02-16 00:00:00\n2023-02-17 00:00:00\n2023-02-18 00:00:00\n2023-02-19 00:00:00\n2023-02-20 00:00:00\n2023-02-21 00:00:00\n2023-02-22 00:00:00\n2023-02-23 00:00:00\n2023-02-24 00:00:00\n2023-02-25 00:00:00\n2023-02-26 00:00:00\n2023-02-27 00:00:00\n2023-02-28 00:00:00\n2023-03-01 00:00:00\n2023-03-02 00:00:00\n2023-03-03 00:00:00\n2023-03-04 00:00:00\n2023-03-05 00:00:00\n2023-03-06 00:00:00\n2023-03-07 00:00:00\n2023-03-08 00:00:00\n2023-03-09 00:00:00\n2023-03-10 00:00:00\n2023-03-11 00:00:00\n2023-03-12 00:00:00\n2023-03-13 00:00:00\n2023-03-14 00:00:00\n2023-03-15 00:00:00\n2023-03-16 00:00:00\n2023-03-17 00:00:00\n2023-03-18 00:00:00\n2023-03-19 00:00:00\n2023-03-20 00:00:00\n2023-03-21 00:00:00\n2023-03-22 00:00:00\n2023-03-23 00:00:00\n2023-03-24 00:00:00\n2023-03-25 00:00:00\n2023-03-26 00:00:00\n2023-03-27 00:00:00\n2023-03-28 00:00:00\n2023-03-29 00:00:00\n2023-03-30 00:00:00\n2023-03-31 00:00:00\n2023-04-01 00:00:00\n2023-04-02 00:00:00\n2023-04-03 00:00:00\n2023-04-04 00:00:00\n2023-04-05 00:00:00\n2023-04-06 00:00:00\n2023-04-07 00:00:00\n2023-04-08 00:00:00\n2023-04-09 00:00:00\n2023-04-10 00:00:00\n2023-04-11 00:00:00\n2023-04-12 00:00:00\n2023-04-13 00:00:00\n2023-04-14 00:00:00\n2023-04-15 00:00:00\n2023-04-16 00:00:00\n2023-04-17 00:00:00\n2023-04-18 00:00:00\n2023-04-19 00:00:00\n2023-04-20 00:00:00\n2023-04-21 00:00:00\n2023-04-22 00:00:00\n2023-04-23 00:00:00\n2023-04-24 00:00:00\n2023-04-25 00:00:00\n2023-04-26 00:00:00\n2023-04-27 00:00:00\n2023-04-28 00:00:00\n2023-04-29 00:00:00\n2023-04-30 00:00:00\n2023-05-01 00:00:00\n2023-05-02 00:00:00\n2023-05-03 00:00:00\n2023-05-04 00:00:00\n2023-05-05 00:00:00\n2023-05-06 00:00:00\n2023-05-07 00:00:00\n2023-05-08 00:00:00\n2023-05-09 00:00:00\n2023-05-10 00:00:00\n2023-05-11 00:00:00\n2023-05-12 00:00:00\n2023-05-13 00:00:00\n2023-05-14 00:00:00\n2023-05-15 00:00:00\n2023-05-16 00:00:00\n2023-05-17 00:00:00\n2023-05-18 00:00:00\n2023-05-19 00:00:00\n2023-05-20 00:00:00\n2023-05-21 00:00:00\n2023-05-22 00:00:00\n2023-05-23 00:00:00\n2023-05-24 00:00:00\n2023-05-25 00:00:00\n2023-05-26 00:00:00\n2023-05-27 00:00:00\n2023-05-28 00:00:00\n2023-05-29 00:00:00\n2023-05-30 00:00:00\n2023-05-31 00:00:00\n2023-06-01 00:00:00\n2023-06-02 00:00:00\n2023-06-03 00:00:00\n2023-06-04 00:00:00\n2023-06-05 00:00:00\n2023-06-06 00:00:00\n2023-06-07 00:00:00\n2023-06-08 00:00:00\n2023-06-09 00:00:00\n2023-06-10 00:00:00\n2023-06-11 00:00:00\n2023-06-12 00:00:00\n2023-06-13 00:00:00\n2023-06-14 00:00:00\n2023-06-15 00:00:00\n2023-06-16 00:00:00\n2023-06-17 00:00:00\n2023-06-18 00:00:00\n2023-06-19 00:00:00\n2023-06-20 00:00:00\n2023-06-21 00:00:00\n2023-06-22 00:00:00\n2023-06-23 00:00:00\n2023-06-24 00:00:00\n2023-06-25 00:00:00\n2023-06-26 00:00:00\n2023-06-27 00:00:00\n2023-06-28 00:00:00\n2023-06-29 00:00:00\n2023-06-30 00:00:00\n2023-07-01 00:00:00\n2023-07-02 00:00:00\n2023-07-03 00:00:00\n2023-07-04 00:00:00\n2023-07-05 00:00:00\n2023-07-06 00:00:00\n2023-07-07 00:00:00\n2023-07-08 00:00:00\n2023-07-09 00:00:00\n2023-07-10 00:00:00\n2023-07-11 00:00:00\n2023-07-12 00:00:00\n2023-07-13 00:00:00\n2023-07-14 00:00:00\n2023-07-15 00:00:00\n2023-07-16 00:00:00\n2023-07-17 00:00:00\n2023-07-18 00:00:00\n2023-07-19 00:00:00\n2023-07-20 00:00:00\n2023-07-21 00:00:00\n2023-07-22 00:00:00\n2023-07-23 00:00:00\n2023-07-24 00:00:00\n2023-07-25 00:00:00\n2023-07-26 00:00:00\n2023-07-27 00:00:00\n2023-07-28 00:00:00\n2023-07-29 00:00:00\n2023-07-30 00:00:00\n2023-07-31 00:00:00\n2023-08-01 00:00:00\n2023-08-02 00:00:00\n2023-08-03 00:00:00\n2023-08-04 00:00:00\n2023-08-05 00:00:00\n2023-08-06 00:00:00\n2023-08-07 00:00:00\n2023-08-08 00:00:00\n2023-08-09 00:00:00\n2023-08-10 00:00:00\n2023-08-11 00:00:00\n2023-08-12 00:00:00\n2023-08-13 00:00:00\n2023-08-14 00:00:00\n2023-08-15 00:00:00\n2023-08-16 00:00:00\n2023-08-17 00:00:00\n2023-08-18 00:00:00\n2023-08-19 00:00:00\n2023-08-20 00:00:00\n2023-08-21 00:00:00\n2023-08-22 00:00:00\n2023-08-23 00:00:00\n2023-08-24 00:00:00\n2023-08-25 00:00:00\n2023-08-26 00:00:00\n2023-08-27 00:00:00\n2023-08-28 00:00:00\n2023-08-29 00:00:00\n2023-08-30 00:00:00\n2023-08-31 00:00:00\n2023-09-01 00:00:00\n2023-09-02 00:00:00\n2023-09-03 00:00:00\n2023-09-04 00:00:00\n2023-09-05 00:00:00\n2023-09-06 00:00:00\n2023-09-07 00:00:00\n2023-09-08 00:00:00\n2023-09-09 00:00:00\n2023-09-10 00:00:00\n2023-09-11 00:00:00\n2023-09-12 00:00:00\n2023-09-13 00:00:00\n2023-09-14 00:00:00\n2023-09-15 00:00:00\n2023-09-16 00:00:00\n2023-09-17 00:00:00\n2023-09-18 00:00:00\n2023-09-19 00:00:00\n2023-09-20 00:00:00\n2023-09-21 00:00:00\n2023-09-22 00:00:00\n2023-09-23 00:00:00\n2023-09-24 00:00:00\n2023-09-25 00:00:00\n2023-09-26 00:00:00\n2023-09-27 00:00:00\n2023-09-28 00:00:00\n2023-09-29 00:00:00\n2023-09-30 00:00:00\n2023-10-01 00:00:00\n2023-10-02 00:00:00\n",

"output\_type": "stream"

}

]

},

{

"cell\_type": "markdown",

"source": "## 10. Create 2D list to DataFrame\n\nlists = [[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": "lists = [[1, 'aaa', 22], [2, 'bbb', 25], [3, 'ccc', 24]]\ndf = pd.DataFrame(lists)\ndf",

"metadata": {

"trusted": true

},

"execution\_count": 21,

"outputs": [

{

"execution\_count": 21,

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

"data": {

"text/plain": " 0 1 2\n0 1 aaa 22\n1 2 bbb 25\n2 3 ccc 24",

"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>0</th>\n <th>1</th>\n <th>2</th>\n </tr>\n </thead>\n <tbody>\n <tr>\n <th>0</th>\n <td>1</td>\n <td>aaa</td>\n <td>22</td>\n </tr>\n <tr>\n <th>1</th>\n <td>2</td>\n <td>bbb</td>\n <td>25</td>\n </tr>\n <tr>\n <th>2</th>\n <td>3</td>\n <td>ccc</td>\n <td>24</td>\n </tr>\n </tbody>\n</table>\n</div>"

},

"metadata": {}

}

]

},

{

"cell\_type": "code",

"source": "",

"metadata": {},

"execution\_count": null,

"outputs": []

}

]

}