```
{
 "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": null,
   "outputs": []
  },
  {
   "cell_type": "code",
   "source": [
    "s=\"Hi there sam!\"\n",
    "s=s.split()\n",
    "print(s);"
   ],
   "metadata": {
    "id": "6mGVa3SQYLkb",
    "colab": {
     "base_uri": "https://localhost:8080/"
    },
```

```
    "outputId": "24bc88b7-e988-4d96-b25c-74aae5940f2d"
   },
   "execution_count": 1,
   "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": "_ZHomI3kPqic"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "planet = \"Earth\"\n",
    "diameter = 12742\n",
    "print('The diameter of {} is {} kilometer.'.format(planet,diameter));"
  ],
  "metadata": {
    "id": "HyRyJv6CYPb4",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "0f349465-ab5c-4c1a-f539-96d36e71cdeb"
  },
  "execution_count": 2,
  "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": "fcVwbCc1QrQl"
   },
   "execution_count": null,
   "outputs": []
  },
  {
   "cell_type": "code",
   "source": [
    "d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}\n",
    "print(d['k1'][3][\"tricky\"][3]['target'][3])"
   ],
```

```
  "metadata": {
    "id": "MvbkMZpXYRaw",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "684247e7-0583-40e1-a75c-0ad87178feaf"
  },
  "execution_count": 3,
  "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": 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": "w0g8hinbgx30"
  }
},
{
  "cell_type": "code",
  "source": [
    "import numpy as np\n",
    "np.zeros(10)"
  ],
  "metadata": {
    "id": "NHrirmgCYXvU",
    "colab": {
      "base_uri": "https://localhost:8080/"
```

```
    },
    "outputId": "a4c05afd-2454-4bc9-cab9-8fd0fb73a170"
  },
  "execution_count": 4,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])"
        ]
      },
      "metadata": {},
      "execution_count": 4
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "import numpy as np\n",
    "np.ones(10)*5"
  ],
  "metadata": {
    "id": "e4005lsTYXxx",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "a99f6524-7bf9-4287-f6f8-f7bc83d34c33"
```

```
  },
  "execution_count": 5,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])"
        ]
      },
      "metadata": {},
      "execution_count": 5
    }
  ]
},
{
  "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\n",
    "print(np.arange(20,35,2))"
```

```
  ],
  "metadata": {
    "id": "oAI2tbU2Yag-",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "47035924-ae6b-4ac4-b458-88f16c2e4aa2"
  },
  "execution_count": 6,
  "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": [
    "import numpy as np\n",
    "np.arange(0,9).reshape((3,3))"
  ],
  "metadata": {
    "id": "tOlEVH7BYceE",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "568e4709-1709-4ee4-a84c-a3b07a0189cd"
  },
  "execution_count": 7,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "array([[0, 1, 2],\n",
          "       [3, 4, 5],\n",
          "       [6, 7, 8]])"
        ]
      },
      "metadata": {},
      "execution_count": 7
    }
  ]
},
{
```

```
  "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": [
    "import numpy as np\n",
    "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": "f5e5f4d1-89fc-469c-a2a9-f6a056154fe8"
  },
  "execution_count": 8,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
```

```
        "text/plain": [
          "array([1, 2, 3, 4, 5, 6])"
        ]
       },
       "metadata": {},
       "execution_count": 8
     }
   ]
  },
  {
   "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": [
    "import pandas as pd\n",
    "record={\n",
    "    \"name\":[\"Aldo\",\"Beula\",\"Chase\"],\n",
    "    \"regno\":[1,2,3]}\n",
    "df=pd.DataFrame(record)\n",
    "df\n"
  ],
  "metadata": {
    "id": "xNpI_XXoYhs0",
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 143
    },
    "outputId": "e89b3135-7e26-4927-8f08-9c1ab3ebd630"
  },
  "execution_count": 11,
  "outputs": [
```

```
{
 "output_type": "execute_result",
 "data": {
  "text/plain": [
   "    name  regno\n",
   "0   Aldo      1\n",
   "1  Beula      2\n",
   "2  Chase      3"
  ],
  "text/html": [
   "\n",
   "  <div id=\"df-79cd7c00-fd1e-4237-93d3-09d83427efe8\">\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>name</th>\n",
"      <th>regno</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>Aldo</td>\n",
"      <td>1</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>Beula</td>\n",
"      <td>2</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>Chase</td>\n",
"      <td>3</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"</div>\n",
"      <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-79cd7c00-fd1e-4237-93d3-09d83427efe8')\"\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-79cd7c00-fd1e-4237-93d3-09d83427efe8 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-79cd7c00-fd1e-4237-93d3-09d83427efe8');\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": 11

    }

  ]
```

```
},
{
  "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": [
    "import numpy as np\n",
    "import pandas as pd\n",
    "pd.date_range(start='1/1/2023',end='2/10/2023')"
  ],
  "metadata": {
    "id": "dgyC0JhVYl4F",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "bc58aca1-017d-47f3-aa20-279039d480ab"
  },
  "execution_count": 12,
  "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": 12
    }
   ]
  },
  {
   "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": "_XMC8aEt0IlB"
  },
  "execution_count": null,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "import pandas as pd\n",
    "lst = [[1, 'aaa', 22], [2, 'bbb', 25], [3,'ccc', 24]]\n",
    "df = pd.DataFrame(lst, columns =['number', 'name', 'age'],dtype=int)\n",
    "print(df)"
  ],
  "metadata": {
    "id": "knH76sDKYsVX",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "ace408fc-3336-4f60-c887-d60bc4a767ed"
```

```
    },
    "execution_count": 14,
    "outputs": [
     {
      "output_type": "stream",
      "name": "stdout",
      "text": [
       "   number name  age\n",
       "0       1  aaa   22\n",
       "1       2  bbb   25\n",
       "2       3  ccc   24\n"
      ]
     },
     {
      "output_type": "stream",
      "name": "stderr",
      "text": [
       "/usr/local/lib/python3.7/dist-packages/IPython/core/interactiveshell.py:3326: FutureWarning: Could not cast to int64, falling back to object. This behavior is deprecated. In a future version, when a dtype is passed to 'DataFrame', either all columns will be cast to that dtype, or a TypeError will be raised\n",
       "  exec(code_obj, self.user_global_ns, self.user_ns)\n"
      ]
     }
    ]
   }
  ]
}
```