```
{
  "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": [
        "x = s.split()\n",
        "print(x)"
      ],
      "metadata": {
        "id": "6mGVa3SQYLkb",
        "outputId": "32137f82-fcbb-4806-8709-da0a074be2bb",
        "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": "5bc9acfe-f7c9-47de-dddb-d2bfe1007da1",
        "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": "5854619a-1da9-45a8-84d6-b369cbef0695",
        "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": [
        "np.zeros(10)"
      ],
      "metadata": {
        "id": "NHrirmgCYXvU",
        "outputId": "3b2a6bd6-c120-493a-ca96-be9f13f9ac8b",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 8,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])"
            ]
          },
          "metadata": {},
          "execution_count": 8
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "np.ones(10) * 5"
      ],
      "metadata": {
        "id": "e4005lsTYXxx",
        "outputId": "77992313-f971-4344-88b8-9a41722065ba",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 10,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])"
            ]
          },
          "metadata": {},
          "execution_count": 10
        }
      ]
    },
    {
      "cell_type": "markdown",
      "source": [
        "## 5. Create an array of all the even integers from 20 to 35"
      ],
      "metadata": {
        "id": "gZHHDUBvrMX4"
      }
    },
    {
      "cell_type": "code",
      "source": [
        "print(np.arange(20,35,2))"
      ],
      "metadata": {
        "id": "oAI2tbU2Yag-",
        "outputId": "778a6ecb-dc60-4925-d300-4135622c5b3a",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 11,
      "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": [
        "np.arange(0,9).reshape((3,3))"
      ],
      "metadata": {
        "id": "tOlEVH7BYceE",
        "outputId": "5dd46d92-11fc-4ee4-ac46-bb1fe8740ad2",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 12,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([[0, 1, 2],\n",
              "       [3, 4, 5],\n",
              "       [6, 7, 8]])"
            ]
          },
          "metadata": {},
          "execution_count": 12
        }
      ]
    },
    {
      "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), axis=None)"
      ],
      "metadata": {
        "id": "rAPSw97aYfE0",
        "outputId": "8999534d-fe75-4022-8bf2-cad50afa7ba5",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 17,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "array([1, 2, 3, 4, 5, 6])"
            ]
          },
          "metadata": {},
          "execution_count": 17
        }
      ]
    },
    {
      "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": [
        "d = {'col1': [1,2,3], 'col2': [4,5,6]}\n",
        "df = pd.DataFrame(data=d)\n",
        "df"
      ],
      "metadata": {
        "id": "xNpI_XXoYhs0",
        "outputId": "acf1db3b-0515-4015-d4e3-c83bcb1fef31",
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 143
        }
      },
      "execution_count": 23,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "   col1  col2\n",
              "0     1     4\n",
              "1     2     5\n",
              "2     3     6"
            ],
            "text/html": [
              "\n",
              "  <div id=\"df-ffdf65f2-ecfd-448c-8b54-795bd43777b3\">\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>col1</th>\n",
              "      <th>col2</th>\n",
              "    </tr>\n",
              "  </thead>\n",
              "  <tbody>\n",
              "    <tr>\n",
              "      <th>0</th>\n",
              "      <td>1</td>\n",
              "      <td>4</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>1</th>\n",
              "      <td>2</td>\n",
              "      <td>5</td>\n",
              "    </tr>\n",
              "    <tr>\n",
              "      <th>2</th>\n",
              "      <td>3</td>\n",
              "      <td>6</td>\n",
              "    </tr>\n",
              "  </tbody>\n",
              "</table>\n",
              "</div>\n",
              "      <button class=\"colab-df-convert\"
onclick=\"convertToInteractive('df-ffdf65f2-ecfd-448c-8b54-
795bd43777b3')\"\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-ffdf65f2-ecfd-448c-
8b54-795bd43777b3 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-
ffdf65f2-ecfd-448c-8b54-795bd43777b3');\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": 23
        }
      ]
    },
    {
      "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 pandas as pd\n",
        "cal = pd.date_range(start ='1-1-2023',end ='02-10-2023', freq
='12H')\n",
        "print(cal)"
      ],
      "metadata": {
        "id": "dgyC0JhVYl4F",
        "outputId": "38bad67b-a3be-4fea-b1d4-ff69110a9c88",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 30,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "DatetimeIndex(['2023-01-01 00:00:00', '2023-01-01
12:00:00',\n",
            "               '2023-01-02 00:00:00', '2023-01-02
12:00:00',\n",
            "               '2023-01-03 00:00:00', '2023-01-03
12:00:00',\n",
            "               '2023-01-04 00:00:00', '2023-01-04
12:00:00',\n",
            "               '2023-01-05 00:00:00', '2023-01-05
12:00:00',\n",
            "               '2023-01-06 00:00:00', '2023-01-06
12:00:00',\n",
            "               '2023-01-07 00:00:00', '2023-01-07
12:00:00',\n",
            "               '2023-01-08 00:00:00', '2023-01-08
12:00:00',\n",
            "               '2023-01-09 00:00:00', '2023-01-09
12:00:00',\n",
            "               '2023-01-10 00:00:00', '2023-01-10
12:00:00',\n",
            "               '2023-01-11 00:00:00', '2023-01-11
12:00:00',\n",
            "               '2023-01-12 00:00:00', '2023-01-12
12:00:00',\n",
            "               '2023-01-13 00:00:00', '2023-01-13
12:00:00',\n",
            "               '2023-01-14 00:00:00', '2023-01-14
12:00:00',\n",
            "               '2023-01-15 00:00:00', '2023-01-15
12:00:00',\n",
            "               '2023-01-16 00:00:00', '2023-01-16
12:00:00',\n",
            "               '2023-01-17 00:00:00', '2023-01-17
12:00:00',\n",
            "               '2023-01-18 00:00:00', '2023-01-18
12:00:00',\n",
            "               '2023-01-19 00:00:00', '2023-01-19
12:00:00',\n",
            "               '2023-01-20 00:00:00', '2023-01-20
12:00:00',\n",
            "               '2023-01-21 00:00:00', '2023-01-21
12:00:00',\n",
            "               '2023-01-22 00:00:00', '2023-01-22
12:00:00',\n",
            "               '2023-01-23 00:00:00', '2023-01-23
12:00:00',\n",
            "               '2023-01-24 00:00:00', '2023-01-24
12:00:00',\n",
            "               '2023-01-25 00:00:00', '2023-01-25
12:00:00',\n",
            "               '2023-01-26 00:00:00', '2023-01-26
12:00:00',\n",
            "               '2023-01-27 00:00:00', '2023-01-27
12:00:00',\n",
            "               '2023-01-28 00:00:00', '2023-01-28
12:00:00',\n",
            "               '2023-01-29 00:00:00', '2023-01-29
12:00:00',\n",
            "               '2023-01-30 00:00:00', '2023-01-30
12:00:00',\n",
            "               '2023-01-31 00:00:00', '2023-01-31
12:00:00',\n",
            "               '2023-02-01 00:00:00', '2023-02-01
12:00:00',\n",
            "               '2023-02-02 00:00:00', '2023-02-02
12:00:00',\n",
            "               '2023-02-03 00:00:00', '2023-02-03
12:00:00',\n",
            "               '2023-02-04 00:00:00', '2023-02-04
12:00:00',\n",
            "               '2023-02-05 00:00:00', '2023-02-05
12:00:00',\n",
            "               '2023-02-06 00:00:00', '2023-02-06
12:00:00',\n",
            "               '2023-02-07 00:00:00', '2023-02-07
12:00:00',\n",
            "               '2023-02-08 00:00:00', '2023-02-08
12:00:00',\n",
            "               '2023-02-09 00:00:00', '2023-02-09
12:00:00',\n",
            "               '2023-02-10 00:00:00'],\n",
            "              dtype='datetime64[ns]', freq='12H')\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": 33,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "df = pd.DataFrame([lists])\n",
        "df.columns =['col1','col2','col3']\n",
        "print(df)"
      ],
      "metadata": {
        "id": "knH76sDKYsVX",
        "outputId": "d0661608-d235-4eb4-b323-4916c7b0af2b",
        "colab": {
          "base_uri": "https://localhost:8080/"
        }
      },
      "execution_count": 34,
      "outputs": [
        {
          "output_type": "stream",
          "name": "stdout",
          "text": [
            "           col1          col2          col3\n",
            "0  [1, aaa, 22]  [2, bbb, 25]  [3, ccc, 24]\n"
          ]
        }
      ]
    }
  ]
}
```