```
{
  "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!\"\n",
        "\n"
      ],
      "metadata": {
        "id": "s07c7JK7Oqt-",
        "colab": {
          "base_uri": "https://localhost:8080/"
        },
        "outputId": "a806b6ae-aafd-4c1c-e18f-862f0fa10283"
      },
      "execution_count": null,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "['Hi', 'there', 'Sam!']"
            ]
          },
          "metadata": {},
          "execution_count": 3
        }
        ]
    },
    {
        "cell_type": "code",
        "source": [
            "s.split()"
        ],
        "metadata": {
            "id": "6mGVa3SQYLkb",
            "colab": {
                "base_uri": "https://localhost:8080/"
            },
            "outputId": "22ae3ae6-1732-43a4-e3ca-afb98496378a"
        },
        "execution_count": null,
        "outputs": [
            {
                "output_type": "execute_result",
                "data": {
                    "text/plain": [
                        "['Hi', 'there', 'Sam!']"
                    ]
                },
                "metadata": {},
                "execution_count": 8
            }
        ]
    },
    {
        "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\n"
        ],
        "metadata": {
            "id": "_ZHoml3kPqic"
        },
        "execution_count": null,
        "outputs": []
    },
    {
        "cell_type": "code",
        "source": [
            "print('The diameter of {} is {} kilometers.'.format(planet,diameter));"
        ],
        "metadata": {
            "id": "HyRyJv6CYPb4",
            "colab": {
                "base_uri": "https://localhost:8080/"
    },
    "outputId": "9a20c0d8-5b8f-46ec-f818-ed9d5e94e245"
  },
  "execution_count": null,
  "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']}]}]}\n",
    "d['k1'][3]['tricky'][3]['target'][3]"
  ],
  "metadata": {
    "id": "fcVwbCc1QrQI",
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 36
    },
    "outputId": "3b48168a-fd64-4ff4-8f72-844064e6e4d9"
  },
  "execution_count": 13,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "'hello'"
        ],
        "application/vnd.google.colaboratory.intrinsic+json": {
          "type": "string"
        }
      },
      "metadata": {},
      "execution_count": 13
    }
  ]
},
{
  "cell_type": "code",
  "source": [
    "d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}\n",
    "d['k1'][3]['tricky'][3]['target'][3]"
  ],
      "metadata": {
        "id": "MvbkMZpXYRaw",
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 36
        },
        "outputId": "5b5d5f70-dc7f-4300-e930-4e30fef250d5"
      },
      "execution_count": 14,
      "outputs": [
        {
          "output_type": "execute_result",
          "data": {
            "text/plain": [
              "'hello'"
            ],
            "application/vnd.google.colaboratory.intrinsic+json": {
              "type": "string"
            }
          },
          "metadata": {},
          "execution_count": 14
        }
      ]
    },
    {
      "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\n",
        "a = np.zeros(10)"
    ],
    "metadata": {
      "id": "NHrirmgCYXvU"
    },
    "execution_count": 15,
    "outputs": []
  },
  {
    "cell_type": "code",
    "source": [
      "import numpy as np\n",
      "a"
    ],
    "metadata": {
      "id": "e4005lsTYXxx",
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "outputId": "23faeb97-f83e-4091-8a42-86fab9792947"
    },
    "execution_count": 16,
    "outputs": [
      {
        "output_type": "execute_result",
        "data": {
          "text/plain": [
            "array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])"
          ]
        },
        "metadata": {},
        "execution_count": 16
      }
    ]
  },
  {
    "cell_type": "markdown",
    "source": [
      "## 5. Create an array of all the even integers from 20 to 35"
    ],
    "metadata": {
      "id": "gZHHDUBvrMX4"
    }
  },
  {
    "cell_type": "code",
    "source": [
      "b = np.ones(10)*5\n",
      "b"
    ],
    "metadata": {
      "id": "oAl2tbU2Yag-",
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "outputId": "74ae7374-e44e-4896-f32a-99862f8548ba"
    },
    "execution_count": 17,
    "outputs": [
      {
        "output_type": "execute_result",
      "data": {
        "text/plain": [
          "array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])"
        ]
      },
      "metadata": {},
      "execution_count": 17
    }
  ]
},
{
  "cell_type": "markdown",
  "source": [
    "## 6. Create a 3x3 matrix with values ranging from 0 to 8"
  ],
  "metadata": {
    "id": "NaOM308NsRpZ"
  }
},
{
  "cell_type": "code",
  "source": [
    "c = np.arange(0,9).reshape(3,3)\n",
    "c"
  ],
  "metadata": {
    "id": "tOlEVH7BYceE",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "d77d9706-214a-48ee-d012-28fb0401ddc5"
  },
  "execution_count": 18,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "array([[0, 1, 2],\n",
          "       [3, 4, 5],\n",
          "       [6, 7, 8]])"
        ]
      },
      "metadata": {},
      "execution_count": 18
    }
  ]
},
{
  "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",
      "np.concatenate((a,b),axis=0)"
    ],
    "metadata": {
      "id": "rAPSw97aYfE0",
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "outputId": "476b1599-603a-4e89-90b0-e9944a0c01fc"
    },
    "execution_count": 19,
    "outputs": [
      {
        "output_type": "execute_result",
        "data": {
          "text/plain": [
            "array([1, 2, 3, 4, 5, 6])"
          ]
        },
        "metadata": {},
        "execution_count": 19
      }
    ]
  },
  {
    "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": [
      "a = np.array([1,2,3])\n",
      "b = np.array([4,5,6])\n",
      "np.concatenate((a,b),axis=0)"
    ],
    "metadata": {
      "id": "xNpI_XXoYhs0",
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "outputId": "f7fb4440-736c-4db5-feef-94001b95c983"
    },
    "execution_count": 20,
    "outputs": [
      {
        "output_type": "execute_result",
        "data": {
          "text/plain": [
            "array([1, 2, 3, 4, 5, 6])"
          ]
        },
        "metadata": {},
        "execution_count": 20
      }
    ]
  },
  {
    "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",
      "P = pd.date_range(start='1-1-2023',end='10-2-2023')\n",
      "for val in P:\n",
      "    print(val);"
    ],
    "metadata": {
      "id": "dgyC0JhVYl4F",
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "outputId": "7d96286b-09c1-4d9b-dc96-28d52701e347"
    },
    "execution_count": 22,
    "outputs": [
      {
        "output_type": "stream",
        "name": "stdout",
        "text": [
          "2023-01-01 00:00:00\n",
          "2023-01-02 00:00:00\n",
          "2023-01-03 00:00:00\n",
          "2023-01-04 00:00:00\n",
          "2023-01-05 00:00:00\n",
          "2023-01-06 00:00:00\n",
          "2023-01-07 00:00:00\n",
"2023-01-08 00:00:00\n",
"2023-01-09 00:00:00\n",
"2023-01-10 00:00:00\n",
"2023-01-11 00:00:00\n",
"2023-01-12 00:00:00\n",
"2023-01-13 00:00:00\n",
"2023-01-14 00:00:00\n",
"2023-01-15 00:00:00\n",
"2023-01-16 00:00:00\n",
"2023-01-17 00:00:00\n",
"2023-01-18 00:00:00\n",
"2023-01-19 00:00:00\n",
"2023-01-20 00:00:00\n",
"2023-01-21 00:00:00\n",
"2023-01-22 00:00:00\n",
"2023-01-23 00:00:00\n",
"2023-01-24 00:00:00\n",
"2023-01-25 00:00:00\n",
"2023-01-26 00:00:00\n",
"2023-01-27 00:00:00\n",
"2023-01-28 00:00:00\n",
"2023-01-29 00:00:00\n",
"2023-01-30 00:00:00\n",
"2023-01-31 00:00:00\n",
"2023-02-01 00:00:00\n",
"2023-02-02 00:00:00\n",
"2023-02-03 00:00:00\n",
"2023-02-04 00:00:00\n",
"2023-02-05 00:00:00\n",
"2023-02-06 00:00:00\n",
"2023-02-07 00:00:00\n",
"2023-02-08 00:00:00\n",
"2023-02-09 00:00:00\n",
"2023-02-10 00:00:00\n",
"2023-02-11 00:00:00\n",
"2023-02-12 00:00:00\n",
"2023-02-13 00:00:00\n",
"2023-02-14 00:00:00\n",
"2023-02-15 00:00:00\n",
"2023-02-16 00:00:00\n",
"2023-02-17 00:00:00\n",
"2023-02-18 00:00:00\n",
"2023-02-19 00:00:00\n",
"2023-02-20 00:00:00\n",
"2023-02-21 00:00:00\n",
"2023-02-22 00:00:00\n",
"2023-02-23 00:00:00\n",
"2023-02-24 00:00:00\n",
"2023-02-25 00:00:00\n",
"2023-02-26 00:00:00\n",
"2023-02-27 00:00:00\n",
"2023-02-28 00:00:00\n",
"2023-03-01 00:00:00\n",
"2023-03-02 00:00:00\n",
"2023-03-03 00:00:00\n",
"2023-03-04 00:00:00\n",
"2023-03-05 00:00:00\n",
"2023-03-06 00:00:00\n",
"2023-03-07 00:00:00\n",
"2023-03-08 00:00:00\n",
"2023-03-09 00:00:00\n",
"2023-03-10 00:00:00\n",
"2023-03-11 00:00:00\n",
"2023-03-12 00:00:00\n",
"2023-03-13 00:00:00\n",
"2023-03-14 00:00:00\n",
"2023-03-15 00:00:00\n",
"2023-03-16 00:00:00\n",
"2023-03-17 00:00:00\n",
"2023-03-18 00:00:00\n",
"2023-03-19 00:00:00\n",
"2023-03-20 00:00:00\n",
"2023-03-21 00:00:00\n",
"2023-03-22 00:00:00\n",
"2023-03-23 00:00:00\n",
"2023-03-24 00:00:00\n",
"2023-03-25 00:00:00\n",
"2023-03-26 00:00:00\n",
"2023-03-27 00:00:00\n",
"2023-03-28 00:00:00\n",
"2023-03-29 00:00:00\n",
"2023-03-30 00:00:00\n",
"2023-03-31 00:00:00\n",
"2023-04-01 00:00:00\n",
"2023-04-02 00:00:00\n",
"2023-04-03 00:00:00\n",
"2023-04-04 00:00:00\n",
"2023-04-05 00:00:00\n",
"2023-04-06 00:00:00\n",
"2023-04-07 00:00:00\n",
"2023-04-08 00:00:00\n",
"2023-04-09 00:00:00\n",
"2023-04-10 00:00:00\n",
"2023-04-11 00:00:00\n",
"2023-04-12 00:00:00\n",
"2023-04-13 00:00:00\n",
"2023-04-14 00:00:00\n",
"2023-04-15 00:00:00\n",
"2023-04-16 00:00:00\n",
"2023-04-17 00:00:00\n",
"2023-04-18 00:00:00\n",
"2023-04-19 00:00:00\n",
"2023-04-20 00:00:00\n",
"2023-04-21 00:00:00\n",
"2023-04-22 00:00:00\n",
"2023-04-23 00:00:00\n",
"2023-04-24 00:00:00\n",
"2023-04-25 00:00:00\n",
"2023-04-26 00:00:00\n",
"2023-04-27 00:00:00\n",
"2023-04-28 00:00:00\n",
"2023-04-29 00:00:00\n",
"2023-04-30 00:00:00\n",
"2023-05-01 00:00:00\n",
"2023-05-02 00:00:00\n",
"2023-05-03 00:00:00\n",
"2023-05-04 00:00:00\n",
"2023-05-05 00:00:00\n",
"2023-05-06 00:00:00\n",
"2023-05-07 00:00:00\n",
"2023-05-08 00:00:00\n",
"2023-05-09 00:00:00\n",
"2023-05-10 00:00:00\n",
"2023-05-11 00:00:00\n",
"2023-05-12 00:00:00\n",
"2023-05-13 00:00:00\n",
"2023-05-14 00:00:00\n",
"2023-05-15 00:00:00\n",
"2023-05-16 00:00:00\n",
"2023-05-17 00:00:00\n",
"2023-05-18 00:00:00\n",
"2023-05-19 00:00:00\n",
"2023-05-20 00:00:00\n",
"2023-05-21 00:00:00\n",
"2023-05-22 00:00:00\n",
"2023-05-23 00:00:00\n",
"2023-05-24 00:00:00\n",
"2023-05-25 00:00:00\n",
"2023-05-26 00:00:00\n",
"2023-05-27 00:00:00\n",
"2023-05-28 00:00:00\n",
"2023-05-29 00:00:00\n",
"2023-05-30 00:00:00\n",
"2023-05-31 00:00:00\n",
"2023-06-01 00:00:00\n",
"2023-06-02 00:00:00\n",
"2023-06-03 00:00:00\n",
"2023-06-04 00:00:00\n",
"2023-06-05 00:00:00\n",
"2023-06-06 00:00:00\n",
"2023-06-07 00:00:00\n",
"2023-06-08 00:00:00\n",
"2023-06-09 00:00:00\n",
"2023-06-10 00:00:00\n",
"2023-06-11 00:00:00\n",
"2023-06-12 00:00:00\n",
"2023-06-13 00:00:00\n",
"2023-06-14 00:00:00\n",
"2023-06-15 00:00:00\n",
"2023-06-16 00:00:00\n",
"2023-06-17 00:00:00\n",
"2023-06-18 00:00:00\n",
"2023-06-19 00:00:00\n",
"2023-06-20 00:00:00\n",
"2023-06-21 00:00:00\n",
"2023-06-22 00:00:00\n",
"2023-06-23 00:00:00\n",
"2023-06-24 00:00:00\n",
"2023-06-25 00:00:00\n",
"2023-06-26 00:00:00\n",
"2023-06-27 00:00:00\n",
"2023-06-28 00:00:00\n",
"2023-06-29 00:00:00\n",
"2023-06-30 00:00:00\n",
"2023-07-01 00:00:00\n",
"2023-07-02 00:00:00\n",
"2023-07-03 00:00:00\n",
"2023-07-04 00:00:00\n",
"2023-07-05 00:00:00\n",
"2023-07-06 00:00:00\n",
"2023-07-07 00:00:00\n",
"2023-07-08 00:00:00\n",
"2023-07-09 00:00:00\n",
"2023-07-10 00:00:00\n",
"2023-07-11 00:00:00\n",
"2023-07-12 00:00:00\n",
"2023-07-13 00:00:00\n",
"2023-07-14 00:00:00\n",
"2023-07-15 00:00:00\n",
"2023-07-16 00:00:00\n",
"2023-07-17 00:00:00\n",
"2023-07-18 00:00:00\n",
"2023-07-19 00:00:00\n",
"2023-07-20 00:00:00\n",
"2023-07-21 00:00:00\n",
"2023-07-22 00:00:00\n",
"2023-07-23 00:00:00\n",
"2023-07-24 00:00:00\n",
"2023-07-25 00:00:00\n",
"2023-07-26 00:00:00\n",
"2023-07-27 00:00:00\n",
"2023-07-28 00:00:00\n",
"2023-07-29 00:00:00\n",
"2023-07-30 00:00:00\n",
"2023-07-31 00:00:00\n",
"2023-08-01 00:00:00\n",
"2023-08-02 00:00:00\n",
"2023-08-03 00:00:00\n",
"2023-08-04 00:00:00\n",
"2023-08-05 00:00:00\n",
"2023-08-06 00:00:00\n",
"2023-08-07 00:00:00\n",
"2023-08-08 00:00:00\n",
"2023-08-09 00:00:00\n",
"2023-08-10 00:00:00\n",
"2023-08-11 00:00:00\n",
"2023-08-12 00:00:00\n",
"2023-08-13 00:00:00\n",
"2023-08-14 00:00:00\n",
"2023-08-15 00:00:00\n",
"2023-08-16 00:00:00\n",
"2023-08-17 00:00:00\n",
"2023-08-18 00:00:00\n",
"2023-08-19 00:00:00\n",
"2023-08-20 00:00:00\n",
"2023-08-21 00:00:00\n",
"2023-08-22 00:00:00\n",
"2023-08-23 00:00:00\n",
"2023-08-24 00:00:00\n",
"2023-08-25 00:00:00\n",
"2023-08-26 00:00:00\n",
"2023-08-27 00:00:00\n",
"2023-08-28 00:00:00\n",
"2023-08-29 00:00:00\n",
"2023-08-30 00:00:00\n",
"2023-08-31 00:00:00\n",
"2023-09-01 00:00:00\n",
"2023-09-02 00:00:00\n",
"2023-09-03 00:00:00\n",
"2023-09-04 00:00:00\n",
        "2023-09-05 00:00:00\n",
        "2023-09-06 00:00:00\n",
        "2023-09-07 00:00:00\n",
        "2023-09-08 00:00:00\n",
        "2023-09-09 00:00:00\n",
        "2023-09-10 00:00:00\n",
        "2023-09-11 00:00:00\n",
        "2023-09-12 00:00:00\n",
        "2023-09-13 00:00:00\n",
        "2023-09-14 00:00:00\n",
        "2023-09-15 00:00:00\n",
        "2023-09-16 00:00:00\n",
        "2023-09-17 00:00:00\n",
        "2023-09-18 00:00:00\n",
        "2023-09-19 00:00:00\n",
        "2023-09-20 00:00:00\n",
        "2023-09-21 00:00:00\n",
        "2023-09-22 00:00:00\n",
        "2023-09-23 00:00:00\n",
        "2023-09-24 00:00:00\n",
        "2023-09-25 00:00:00\n",
        "2023-09-26 00:00:00\n",
        "2023-09-27 00:00:00\n",
        "2023-09-28 00:00:00\n",
        "2023-09-29 00:00:00\n",
        "2023-09-30 00:00:00\n",
        "2023-10-01 00:00:00\n",
        "2023-10-02 00:00:00\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]]\n",
    "df = pd.DataFrame(lists)\n",
    "df"
  ],
  "metadata": {
    "id": "_XMC8aEt0llB",
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 172
    },
    "outputId": "b2a54d62-58f6-49bb-eaa0-5b848672572e"
  },
"execution_count": 25,
"outputs": [
  {
    "output_type": "execute_result",
    "data": {
      "text/plain": [
        "     0     1     2\n",
        "0    1   aaa   22\n",
        "1    2   bbb   25\n",
        "2    3   ccc   24"
      ],
      "text/html": [
        "\n",
        "  <div id=\"df-87a6eded-382d-4f7c-a9d3-c4956a6c7313\">\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>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>\n",
"        <button class=\"colab-df-convert\" onclick=\"convertToInteractive('df-87a6eded-382d-4f7c-a9d3-c4956a6c7313')\"\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-87a6eded-382d-4f7c-a9d3-
c4956a6c7313 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-87a6eded-382d-4f7c-
a9d3-c4956a6c7313');\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": 25
        }
      ]
    }
  ]
}
```