

MAHENDRA ENGINEERING COLLEGE FOR WOMEN

ASSIGNMENT-1 SOLUTION

NAME OF THE STUDENT: M. Abitha

REGISTER NUMBER:611419106004

YEAR/DEPARTMENT:IV-ECE

```
{
  "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": [
  "txt = \"Hi there Sam!\\n\",
  "\\n",
  "x = txt.split()\\n",
  "\\n",
  "print(x)"
],
"metadata": {
  "id": "6mGVa3SQYLkb",
  "colab": {
    "base_uri": "https://localhost:8080/"
  },
  "outputId": "826edc4f-3e69-41e8-bffc-c94dbbf01d67"
},
"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": [
    "txt = \"The diameter of Earth {diameter:} is   kilometers\\n\",
    "print(txt.format(diameter = 12742))\\n"
  ]
}

```

```

],
"metadata": {
  "id": "HyRyJv6CYPb4",
  "colab": {
    "base_uri": "https://localhost:8080/"
  },
  "outputId": "f6753ae9-465e-4c1a-b2aa-584c5b085109"
},
"execution_count": 7,
"outputs": [
  {
    "output_type": "stream",
    "name": "stdout",
    "text": [
      "The diameter of Earth 12742 is  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": "fcVwbCc1QrQl"
  },
  "execution_count": 8,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "print(d)"
  ],
  "metadata": {
    "id": "MvbkMZpXYRaw",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "e6d7ee94-2ffb-4bd8-a5a7-005f5b117e7e"
  },
  "execution_count": 15,
  "outputs": [

```

```

    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "{ 'k1': [1, 2, 3, {'tricky': ['oh', 'man', 'inception', {'target': [1, 2, 3, '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": 18,
    "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": [
      "array=np.zeros(10)\n",
      "print(\"An array of 10 zeros:\")"
    ],
    "metadata": {
      "id": "NHrirmgCYXvU",
      "colab": {
        "base_uri": "https://localhost:8080/"
      }
    },
    "outputId": "82730e66-fb70-48b6-90d8-85a831736b5a"
  },
  "execution_count": 19,

```

```

"outputs": [
  {
    "output_type": "stream",
    "name": "stdout",
    "text": [
      "An array of 10 zeros:\n"
    ]
  }
],
{
  "cell_type": "code",
  "source": [
    "array=np.zeros(10)\n",
    "print(\"An array of 5 fives:\")"
  ],
  "metadata": {
    "id": "e4005lsTYXxx",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "3bf02af0-7bd0-4299-8d16-68347a566a1e"
  },
  "execution_count": 20,
  "outputs": [
    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "An array of 5 fives:\n"
      ]
    }
  ]
},
{
  "cell_type": "markdown",
  "source": [
    "## 5. Create an array of all the even integers from 20 to 35"
  ],
  "metadata": {
    "id": "gZHHdUBvrMX4"
  }
},
{
  "cell_type": "code",
  "source": [
    "array=np.arange(20,35,2)\n",
    "print(\"Array of all the even integers from 20 to 35\")\n",
    "print(array)"
  ],
  "metadata": {
    "id": "oAI2tbU2Yag-",
    "colab": {

```

```

        "base_uri": "https://localhost:8080/"
    },
    "outputId": "28ef5cb3-93cb-4ff8-a886-fbffc66193c3"
},
"execution_count": 21,
"outputs": [
    {
        "output_type": "stream",
        "name": "stdout",
        "text": [
            "Array of all the even integers from 20 to 35\n",
            "[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": [
        "x = np.arange(0, 9).reshape(3,3)\n",
        "print(x)"
    ],
    "metadata": {
        "id": "tOIEVH7BYceE",
        "colab": {
            "base_uri": "https://localhost:8080/"
        }
    },
    "outputId": "80cd8b42-95ea-4b83-ad7a-9453f0613c69"
},
"execution_count": 22,
"outputs": [
    {
        "output_type": "stream",
        "name": "stdout",
        "text": [
            "[[0 1 2]\n",
            " [3 4 5]\n",
            " [6 7 8]]\n"
        ]
    }
]
},
{
    "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 = [1, 2,3]\n",
    "b = [4,5,6]\n",
    "  \n",
    "\n",
    "for i in b :\n",
    "    a.append(i)\n",
    "  \n",
    "\n",
    "print (\nConcatenated list a and b is : \n \n",
    "          + str(a))"
  ],
  "metadata": {
    "id": "rAPSw97aYfE0",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "445a4c3e-58ac-4a80-852e-67e724926cad"
  },
  "execution_count": 24,
  "outputs": [
    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "Concatenated list a and b is : [1, 2, 3, 4, 5, 6]\n"
      ]
    }
  ]
},
{
  "cell_type": "markdown",
  "source": [
    "# Pandas"
  ],
  "metadata": {
    "id": "dIPEY9DRwZga"
  }
},
{
  "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": 25,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "\n",
    "  \n",
    "\n",
    "data = [['tom', 10], ['nick', 15], ['juli', 14]]\n",
    "  \n",
    "\n",
    "df = pd.DataFrame(data, columns=['Name', 'Age'])\n",
    "\n",
    "df"
  ],
  "metadata": {
    "id": "xNpl_XXoYhs0",
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 143
    },
    "outputId": "2402a0ee-40d1-4e6a-dcd5-5cdea1985c78"
  },
  "execution_count": 26,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "   Name  Age\n0   tom   10\n1  nick   15\n2   juli   14"
        ],
        "text/html": [
          "\n",
          "  <div id=\"df-a344f79d-1761-4ba3-b335-c8666e11be17\">\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 tthead th {\n",  
    text-align: right;\n",  
}\n",  
</style>\n",  
<table border=\\\"1\\\" class=\\\"dataframe\\\">\n",  
  <tthead>\n",  
    <tr style=\\\"text-align: right;\\\">\n",  
      <th></th>\n",  
      <th>Name</th>\n",  
      <th>Age</th>\n",  
    </tr>\n",  
  </tthead>\n",  
  <tbody>\n",  
    <tr>\n",  
      <th>0</th>\n",  
      <td>tom</td>\n",  
      <td>10</td>\n",  
    </tr>\n",  
    <tr>\n",  
      <th>1</th>\n",  
      <td>nick</td>\n",  
      <td>15</td>\n",  
    </tr>\n",  
    <tr>\n",  
      <th>2</th>\n",  
      <td>juli</td>\n",  
      <td>14</td>\n",  
    </tr>\n",  
  </tbody>\n",  
</table>\n",  
</div>\n",  
  <button class=\\\"colab-df-convert\\\" onclick=\\\"convertToInteractive('df-a344f79d-1761-4ba3-b335-c8666e11be17')\\\">\n",  
    title=\\\"Convert this dataframe to an interactive table.\\\"\n",  
    style=\\\"display:none;\\\">  
  
    \n",  
    <svg xmlns=\\\"http://www.w3.org/2000/svg\\\" height=\\\"24px\\\" viewBox=\\\"0 0 24 24\\\">  
      width=\\\"24px\\\">  
        <path d=\\\"M0 0h24v24H0V0z\\\" fill=\\\"none\\\"/>  
        <path d=\\\"M18.56 5.44l-.94 2.06-.94-2.06-.94-2.06-.94-2.06-.94 2.06-2.06.94zm-11  
11.8.5 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-.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.39.39.95.59 1.41.59 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\\\"/>
```

```

"   </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-a344f79d-1761-4ba3-b335-c8666e11be17\n",
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-a344f79d-1761-4ba3-b335-\n",
c8666e11be17');\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": 26
}
]
},
{
  "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",
    "\n",
    "\n",
    "dates = pd.date_range('2023-01-01', periods=41, freq='D')\n",
    "\n",
    "s = pd.Series(dates)\n",
    "print (s)"
  ],
  "metadata": {
    "id": "dgyC0JhVYI4F",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "f9c818dd-bcf2-480d-ab74-9fc46403210b"
  },
  "execution_count": 29,

```

```

"outputs": [
  {
    "output_type": "stream",
    "name": "stdout",
    "text": [
      "0    2023-01-01\n",
      "1    2023-01-02\n",
      "2    2023-01-03\n",
      "3    2023-01-04\n",
      "4    2023-01-05\n",
      "5    2023-01-06\n",
      "6    2023-01-07\n",
      "7    2023-01-08\n",
      "8    2023-01-09\n",
      "9    2023-01-10\n",
      "10   2023-01-11\n",
      "11   2023-01-12\n",
      "12   2023-01-13\n",
      "13   2023-01-14\n",
      "14   2023-01-15\n",
      "15   2023-01-16\n",
      "16   2023-01-17\n",
      "17   2023-01-18\n",
      "18   2023-01-19\n",
      "19   2023-01-20\n",
      "20   2023-01-21\n",
      "21   2023-01-22\n",
      "22   2023-01-23\n",
      "23   2023-01-24\n",
      "24   2023-01-25\n",
      "25   2023-01-26\n",
      "26   2023-01-27\n",
      "27   2023-01-28\n",
      "28   2023-01-29\n",
      "29   2023-01-30\n",
      "30   2023-01-31\n",
      "31   2023-02-01\n",
      "32   2023-02-02\n",
      "33   2023-02-03\n",
      "34   2023-02-04\n",
      "35   2023-02-05\n",
      "36   2023-02-06\n",
      "37   2023-02-07\n",
      "38   2023-02-08\n",
      "39   2023-02-09\n",
      "40   2023-02-10\n",
      "dtype: datetime64[ns]\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": "_XMC8aEt0IIB"
  },
  "execution_count": 33,
  "outputs": []
},
{
  "cell_type": "code",
  "source": [
    "import pandas as pd\n",
    "\n",
    "\n",
    "lst = [[1, 'aaa', 22], [2, 'bbb', 25], [3, 'ccc', 24]]\n",
    "\n",
    "\n",
    "\n",
    "df = pd.DataFrame(lst, columns=['NO', 'name', 'age'])\n",
    "print(df)"
  ],
  "metadata": {
    "id": "knH76sDKYsVX",
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "outputId": "19affc1b-734e-4740-cb8a-40d4f6d423a5"
  },
  "execution_count": 37,
  "outputs": [
    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "   NO name  age\n",
        "0    1  aaa   22\n",
        "1    2  bbb   25\n",
        "2    3  ccc   24\n"
      ]
    }
  ]
}

```

```
}  
  ]  
}  
  ]  
}  
  ]  
}
```