```
{
 "cells": [
  {
   "cell_type": "markdown",
   "metadata": {
    "id": "fwU2iooz85jt"
   },
   "source": [
    "## Exercises\n",
    "\n",
    "Answer the questions or complete the tasks outlined in bold below, use the specific method described if applicable."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {
    "id": "SzBQQ_ml85j1"
   },
   "source": [
    "** What is 7 to the power of 4?**"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
```

```
  "metadata": {
    "id": "UhvE4PBC85j3",
    "outputId": "799c6f1c-9790-43cf-8d6e-e3b23d964159",
    "colab": {
      "base_uri": "https://localhost:8080/"
    }
  },
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "2401"
        ]
      },
      "metadata": {},
      "execution_count": 1
    }
  ],
  "source": [
    "7**4"
  ]
 },
 {
  "cell_type": "markdown",
```

```
  "metadata": {
   "id": "ds8G9S8j85j6"
  },
  "source": [
   "** Split this string:**\n",
   "\n",
   "    s = \"Hi there Sam!\"\n",
   "    \n",
   "**into a list. **"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
   "collapsed": true,
   "id": "GD_Tls3H85j7"
  },
  "outputs": [],
  "source": [
   "s = \"Hi there Sam!\"\n"
  ]
 },
 {
  "cell_type": "code",
```

```
  "execution_count": null,
  "metadata": {
    "id": "RRGOKoai85j8",
    "outputId": "45f80735-2a12-4335-f84b-f8185bde1638",
    "colab": {
      "base_uri": "https://localhost:8080/"
    }
  },
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "['Hi', 'there', 'Sam!']"
        ]
      },
      "metadata": {},
      "execution_count": 3
    }
  ],
  "source": [
    "\n",
    "s.split()"
  ]
},
```

```
{
 "cell_type": "markdown",
 "metadata": {
  "id": "_bBNOu-785j9"
 },
 "source": [
  "** Given the variables:**\n",
  "\n",
  "    planet = \"Earth\"\n",
  "    diameter = 12742\n",
  "\n",
  "** Use .format() to print the following string: **\n",
  "\n",
  "    The diameter of Earth is 12742 kilometers."
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "collapsed": true,
  "id": "2TrzmDcS85j-",
  "outputId": "3e2ec4b7-1f1f-4401-a045-b0789cf77b7a",
  "colab": {
   "base_uri": "https://localhost:8080/"
```

```
   }
  },
  "outputs": [
   {
    "output_type": "stream",
    "name": "stdout",
    "text": [
     "The diameter of Earth is 12742 kilometers.\n"
    ]
   }
  ],
  "source": [
   "planet = \"Earth\"\n",
   "diameter = 12742\n",
   "print(\"The diameter of {} is {} kilometers.\".format(planet,diameter))"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "id": "QAKtN7Hh85kB"
  },
  "source": [
   "** Given this nested list, use indexing to grab the word \"hello\" **"
  ]
```

```
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "collapsed": true,
    "id": "-7dzQDyK85kD"
  },
  "outputs": [],
  "source": [
    "lst = [1,2,[3,4],[5,[100,200,['hello']],23,11],1,7]"
  ]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "6m5C0sTW85kE",
    "outputId": "568de926-cd6e-4ada-b5e5-b480e0c264b3",
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 36
    }
  },
  "outputs": [
```

```
    {
     "output_type": "execute_result",
     "data": {
      "text/plain": [
       "'hello'"
      ],
      "application/vnd.google.colaboratory.intrinsic+json": {
       "type": "string"
      }
     },
     "metadata": {},
     "execution_count": 7
    }
   ],
   "source": [
    "\n",
    "lst[3][1][2][0]"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {
    "id": "9Ma7M4a185kF"
   },
   "source": [
```

```
    "** Given this nest dictionary grab the word \"hello\". Be prepared, this will be annoying/tricky **"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "vrYAxSYN85kG"
   },
   "outputs": [],
   "source": [
    "d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "FlILSdm485kH",
    "outputId": "c3d2b415-c271-4bbc-e2a6-4849ad2e1f49",
    "colab": {
     "base_uri": "https://localhost:8080/",
     "height": 36
    }
   },
```

```
  "outputs": [
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
      "'hello'"
     ],
     "application/vnd.google.colaboratory.intrinsic+json": {
      "type": "string"
     }
    },
    "metadata": {},
    "execution_count": 9
   }
  ],
  "source": [
   "\n",
   "d['k1'][3]['tricky'][3]['target'][3]"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "id": "FInV_FKB85kI"
  },
```

```
  "source": [
    "** What is the main difference between a tuple and a list? **"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "collapsed": true,
    "id": "_VBWf00q85kJ"
  },
  "outputs": [],
  "source": [
    "#Tuple is immutable where list is mutable"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
    "id": "zP-j0HZj85kK"
  },
  "source": [
    "** Create a function that grabs the email website domain from a string in the form: **\n",
    "\n",
    "    user@domain.com\n",
```

```
    "    \n",
    "**So for example, passing \"user@domain.com\" would return: domain.com**"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "collapsed": true,
    "id": "unvEAwjk85kL"
   },
   "outputs": [],
   "source": [
    "def domainGet(email):\n",
    "    return email.split('@')[-1]"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "Gb9dspLC85kL",
    "outputId": "4216116b-da08-45a2-9545-d6b13bcefaeb"
   },
   "outputs": [
```

```
    {
     "data": {
      "text/plain": [
       "'domain.com'"
      ]
     },
     "execution_count": 26,
     "metadata": {
      "tags": []
     },
     "output_type": "execute_result"
    }
   ],
   "source": []
  },
  {
   "cell_type": "markdown",
   "metadata": {
    "id": "gYydb-y085kM"
   },
   "source": [
    "** Create a basic function that returns True if the word 'dog' is contained in the input string. Don't worry about edge cases like a punctuation being attached to the word dog, but do account for capitalization. **"
   ]
  },
```

```
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "collapsed": true,
  "id": "Q4ldLGV785kM"
 },
 "outputs": [],
 "source": [
  "def findDog(st):\n",
  "    return 'dog' in st.lower().split()"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "EqH6b7yv85kN",
  "outputId": "349f2262-4f46-46ca-d8c0-2d98716fff18",
  "colab": {
   "base_uri": "https://localhost:8080/"
  }
 },
 "outputs": [
  {
```

```
      "output_type": "execute_result",
      "data": {
       "text/plain": [
        "True"
       ]
      },
      "metadata": {},
      "execution_count": 13
     }
    ],
    "source": [
     "findDog('Is there a dog here?')"
    ]
   },
   {
    "cell_type": "markdown",
    "metadata": {
     "id": "AyHQFALC85kO"
    },
    "source": [
     "** Create a function that counts the number of times the word \"dog\" occurs in a string. Again ignore edge cases. **"
    ]
   },
   {
    "cell_type": "code",
```

```
  "execution_count": null,
  "metadata": {
    "id": "6hdc169585kO"
  },
  "outputs": [],
  "source": [
    "def countDog(st):\n",
    "    count = 0\n",
    "    for word in st.lower().split():\n",
    "        if word == 'dog':\n",
    "            count += 1\n",
    "    return count"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "igzsvHb385kO",
    "outputId": "a5a34aa4-723f-4574-fa1a-c3a2b5634668",
    "colab": {
      "base_uri": "https://localhost:8080/"
    }
  },
  "outputs": [
```

```
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
      "2"
     ]
    },
    "metadata": {},
    "execution_count": 17
   }
  ],
  "source": [
   "countDog('This dog runs faster than the other dog dude!')\n"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "id": "3n7jJt4k85kP"
  },
  "source": [
   "### Problem\n",
   "**You are driving a little too fast, and a police officer stops you. Write a function\n",
   "  to return one of 3 possible results: \"No ticket\", \"Small ticket\", or \"Big Ticket\". \n",
   "  If your speed is 60 or less, the result is \"No Ticket\". If speed is between 61 \n",
```

```
      "  and 80 inclusive, the result is \"Small Ticket\". If speed is 81 or more, the result is \"Big    Ticket\". Unless it is your birthday (encoded as a boolean value in the parameters of the function) -- on your birthday, your speed can be 5 higher in all \n",

      "  cases. **"

    ]

   },

   {

    "cell_type": "code",

    "execution_count": null,

    "metadata": {

     "collapsed": true,

     "id": "nvXMkvWk85kQ"

    },

    "outputs": [],

    "source": [

     "def caught_speeding(speed, is_birthday):\n",

     "    \n",

     "    if is_birthday:\n",

     "        speeding = speed - 5\n",

     "    else:\n",

     "        speeding = speed\n",

     "    \n",

     "    if speeding > 80:\n",

     "        return 'Big Ticket'\n",

     "    elif speeding > 60:\n",

     "        return 'Small Ticket'\n",
```

```
    "    else:\n",
    "        return 'No Ticket'"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "BU_UZcyk85kS",
    "outputId": "e3b735f8-018f-4550-9a3f-8be11a61cebf",
    "colab": {
     "base_uri": "https://localhost:8080/",
     "height": 36
    }
   },
   "outputs": [
    {
     "output_type": "execute_result",
     "data": {
      "text/plain": [
       "'Big Ticket'"
      ],
      "application/vnd.google.colaboratory.intrinsic+json": {
       "type": "string"
      }
```

```
    },
    "metadata": {},
    "execution_count": 24
   }
  ],
  "source": [
   "caught_speeding(81,False)\n"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
   "id": "p1AGJ7DM85kR",
   "outputId": "fa871e9c-8e4a-4d6b-f06d-228b7d3cbdf2",
   "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 36
   }
  },
  "outputs": [
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
```

```
          "'Small Ticket'"
         ],
         "application/vnd.google.colaboratory.intrinsic+json": {
          "type": "string"
         }
        },
        "metadata": {},
        "execution_count": 25
       }
     ],
     "source": [
      "caught_speeding(70,True)"
     ]
    },
    {
     "cell_type": "markdown",
     "source": [
      "Create an employee list with basic salary values(at least 5 values for 5 employees)  and using a for loop retreive each employee salary and calculate total salary expenditure. "
     ],
     "metadata": {
      "id": "Tie4rC7_kAOC"
     }
    },
    {
     "cell_type": "code",
```

```
  "source": [
    "employee = [30000,23000,25000,40000,33000]\n",
    "tot = 0\n",
    "for i in employee:\n",
    "    tot = tot + i\n",
    "print(\"Total Salary :\",tot)"
  ],
  "metadata": {
    "id": "R5-CdXSKjacN",
    "outputId": "80033baf-6c99-4bbc-e7e0-604feb0e4179",
    "colab": {
      "base_uri": "https://localhost:8080/"
    }
  },
  "execution_count": null,
  "outputs": [
    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "Total Salary : 151000\n"
      ]
    }
  ]
},
```

```
{
 "cell_type": "markdown",
 "source": [
  "Create two dictionaries in Python:\n",
  "\n",
  "First one to contain fields as Empid,  Empname,  Basicpay\n",
  "\n",
  "Second dictionary to contain fields as DeptName,  DeptId.\n",
  "\n",
  "Combine both dictionaries. "
 ],
 "metadata": {
  "id": "-L1aiFqRkF5s"
 }
},
{
 "cell_type": "code",
 "source": [
  "dict1 = {\n",
  "    \"EmpId\":[1001,1002,1003,1004],\n",
  "    \"EmpName\":[\"Dharani.M\",\"Shree vidhyaa.S\",\"Kaviya.S.D\",\"Shanmugapriya.M\"],\n",
  "    \"Basicpay\":[45000,42000,43000,44000]\n",
  "}\n",
  "\n",
  "dict2 = {\n",
```

```
  "    \"DeptName\":[\"CSE\",\"ECE\",\"EEE\",\"IT\"],\n",
  "    \"DeptId\":[\"19CS\",\"19ECE\",\"19EEE\",\"19IT\"]\n",
  "}\n",
  "\n",
  "dict1.update(dict2)\n",
  "dict3 = dict1.copy()\n",
  "dict3"
 ],
 "metadata": {
  "id": "8ugVoEe0kOsk",
  "outputId": "d9422a1c-de5b-488a-b804-e5420e2ea012",
  "colab": {
   "base_uri": "https://localhost:8080/"
  }
 },
 "execution_count": null,
 "outputs": [
  {
   "output_type": "execute_result",
   "data": {
    "text/plain": [
     "{'EmpId': [1001, 1002, 1003, 1004],\n",
     " 'EmpName': ['Dharani.M', 'Shree vidhyaa.S', 'Kaviya.S.D', 'Shanmugapriya.M'],\n",
     " 'Basicpay': [45000, 42000, 43000, 44000],\n",
     " 'DeptName': ['CSE', 'ECE', 'EEE', 'IT'],\n",
```

```
          " 'DeptId': ['19CS', '19ECE', '19EEE', '19IT']}"
         ]
        },
        "metadata": {},
        "execution_count": 27
      }
    ]
  }
],
"metadata": {
  "colab": {
    "provenance": [],
    "collapsed_sections": []
  },
  "kernelspec": {
    "display_name": "Python 3",
    "language": "python",
    "name": "python3"
  },
  "language_info": {
    "codemirror_mode": {
      "name": "ipython",
      "version": 3
    },
    "file_extension": ".py",
```

```
    "mimetype": "text/x-python",
    "name": "python",
    "nbconvert_exporter": "python",
    "pygments_lexer": "ipython3",
    "version": "3.8.5"
   }
  },
  "nbformat": 4,
  "nbformat_minor": 0
}
```