```
{
 "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": 1,
   "metadata": {
    "id": "UhvE4PBC85j3",
    "outputId": "a05565aa-db43-4716-e87d-41c5c8a6f95e"
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "2401"
      ]
     },
     "execution_count": 1,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "import math\n",
    "math.pow(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": {
  "id": "GD_Tls3H85j7"
 },
 "outputs": [],
 "source": [
  "str=\"Hi there dad!\""
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "RRGOKoai85j8",
  "outputId": "cc52f0d8-2ed1-4b4d-e956-5bbeb332cdc2"
 },
 "outputs": [
  {
   "data": {
    "text/plain": [
     "['Hi', 'there', 'dad!']"
    ]
   },
   "execution_count": 3,
   "metadata": {
    "tags": []
   },
   "output_type": "execute_result"
  }
 ],
 "source": [
  "list(str.split())"
 ]
},
{
 "cell_type": "raw",
 "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-"
   },
   "outputs": [],
   "source": [
    "planet = \"Earth\"\n",
    "diameter = 12742"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "id": "s_dQ7_xc85j_",
    "outputId": "4235fdfb-5591-4dd9-f9d2-77f311977633",
    "scrolled": true
   },
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "The diameter of Earth is 12742 kilometers.\n"
     ]
    }
   ],
   "source": [
    "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": "c3417d1c-3081-4e24-8489-154cdce1b06b"
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "'hello'"
      ]
     },
     "execution_count": 14,
     "metadata": {
      "tags": []
     },
     "output_type": "execute_result"
    }
   ],
   "source": [
    "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": 2,
   "metadata": {
    "id": "vrYAxSYN85kG"
   },
   "outputs": [],
   "source": [
    "d =
{'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}
]}]}"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {
    "id": "FlILSdm485kH",
    "outputId": "4232540d-95c2-461d-c78d-24ea62398e08"
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "'hello'"
      ]
     },
     "execution_count": 3,
```

```
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "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": [
    "list is mutable whereas Tuple is immutable."
   ]
  },
  {
   "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": 5,
   "metadata": {
    "id": "unvEAwjk85kL"
   },
   "outputs": [],
   "source": [
    "def domainGet(email):\n",
    "    return email.split('@')[-1]"
   ]
  },
  {
   "cell_type": "code",
```

```
   "execution_count": 6,
   "metadata": {
    "id": "Gb9dspLC85kL",
    "outputId": "4216116b-da08-45a2-9545-d6b13bcefaeb"
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "'domain.com'"
      ]
     },
     "execution_count": 6,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "domainGet('user@domain.com')"
   ]
  },
  {
   "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": 9,
   "metadata": {
    "id": "Q4ldLGV785kM"
   },
   "outputs": [],
   "source": [
    "def Dog(st):\n",
    "    return 'dog' in st.lower().split()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 10,
   "metadata": {
    "id": "EqH6b7yv85kN",
    "outputId": "e7909af1-8df1-4534-fc8c-27b03d7369e5"
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "True"
      ]
```

```
     },
     "execution_count": 10,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "Dog('The dog kept barking all night')"
   ]
  },
  {
   "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": 11,
   "metadata": {
    "id": "6hdc169585kO"
   },
   "outputs": [],
   "source": [
    "def count_Dog(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": 12,
   "metadata": {
    "id": "igzsvHb385kO",
    "outputId": "0602a2b5-0b18-48d8-e2d4-fe644cbccf8a"
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "2"
      ]
     },
     "execution_count": 12,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "count_Dog('This dog runs faster than the other dog dude!')"
   ]
```

```
  },
  {
   "cell_type": "markdown",
   "metadata": {
    "id": "3n7jJt4k85kP"
   },
   "source": [
    "### Final 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": 13,
   "metadata": {
    "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": 16,
   "metadata": {
    "id": "p1AGJ7DM85kR",
    "outputId": "ca80629f-5949-4926-8d27-1b61576669ac"
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "'Small Ticket'"
      ]
     },
```

```
    "execution_count": 16,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "caught_speeding(83,True)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 17,
  "metadata": {
   "id": "BU_UZcyk85kS",
   "outputId": "699de8ef-a18c-436b-fdd9-60dc44979906"
  },
  "outputs": [
   {
    "data": {
     "text/plain": [
      "'Big Ticket'"
     ]
    },
    "execution_count": 17,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "caught_speeding(83,False)"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "id": "QL7sY6NR85kT"
  },
  "source": [
   "# Great job!"
  ]
 }
],
"metadata": {
 "colab": {
  "name": "python exercise.ipynb",
  "provenance": []
 },
 "kernelspec": {
  "display_name": "Python 3 (ipykernel)",
  "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": "3.9.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 1
}
```