{

"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": 2,

"metadata": {

"id": "UhvE4PBC85j3",

"outputId": "a05565aa-db43-4716-e87d-41c5c8a6f95e"

},

"outputs": [

{

"data": {

"text/plain": [

"2401"

]

},

"execution\_count": 2,

"metadata": {},

"output\_type": "execute\_result"

}

],

"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": 13,

"metadata": {

"id": "GD\_Tls3H85j7"

},

"outputs": [],

"source": [

"s=\"Hi there dad!\""

]

},

{

"cell\_type": "code",

"execution\_count": 33,

"metadata": {

"id": "RRGOKoai85j8",

"outputId": "cc52f0d8-2ed1-4b4d-e956-5bbeb332cdc2"

},

"outputs": [

{

"data": {

"text/plain": [

"['Hi', 'there', 'dad!']"

]

},

"execution\_count": 33,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"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": 14,

"metadata": {

"id": "2TrzmDcS85j-"

},

"outputs": [],

"source": [

"planet=\"Earth\"\n",

"diameter=12742"

]

},

{

"cell\_type": "code",

"execution\_count": 15,

"metadata": {

"id": "s\_dQ7\_xc85j\_",

"outputId": "4235fdfb-5591-4dd9-f9d2-77f311977633"

},

"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": 8,

"metadata": {

"id": "-7dzQDyK85kD"

},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"hello\n"

]

}

],

"source": [

"lst = [1,2,[3,4],[5,[100,200,['hello']],23,11],1,7]\n",

"print(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": 16,

"metadata": {

"id": "vrYAxSYN85kG"

},

"outputs": [],

"source": [

"d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}"

]

},

{

"cell\_type": "code",

"execution\_count": 17,

"metadata": {

"id": "FlILSdm485kH",

"outputId": "4232540d-95c2-461d-c78d-24ea62398e08"

},

"outputs": [

{

"data": {

"text/plain": [

"'hello'"

]

},

"execution\_count": 17,

"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": [

"tuple object is immutable whereas list object 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": 19,

"metadata": {

"id": "unvEAwjk85kL"

},

"outputs": [],

"source": [

"def grabDomain(email):\n",

" return email.split('@')[-1]"

]

},

{

"cell\_type": "code",

"execution\_count": 20,

"metadata": {

"id": "Gb9dspLC85kL",

"outputId": "4216116b-da08-45a2-9545-d6b13bcefaeb"

},

"outputs": [

{

"data": {

"text/plain": [

"'domain.com'"

]

},

"execution\_count": 20,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"grabDomain('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": 21,

"metadata": {

"id": "Q4ldLGV785kM"

},

"outputs": [],

"source": [

"def Dog(t):\n",

" return 'dog'in t.lower().split()"

]

},

{

"cell\_type": "code",

"execution\_count": 26,

"metadata": {

"id": "EqH6b7yv85kN",

"outputId": "e7909af1-8df1-4534-fc8c-27b03d7369e5"

},

"outputs": [

{

"data": {

"text/plain": [

"True"

]

},

"execution\_count": 26,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"Dog('I love this dog')"

]

},

{

"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": 27,

"metadata": {

"id": "6hdc169585kO"

},

"outputs": [],

"source": [

"def countDog(s):\n",

" c=0\n",

" for word in s.lower().split():\n",

" if word=='dog':\n",

" c+=1\n",

" return c"

]

},

{

"cell\_type": "code",

"execution\_count": 28,

"metadata": {

"id": "igzsvHb385kO",

"outputId": "0602a2b5-0b18-48d8-e2d4-fe644cbccf8a"

},

"outputs": [

{

"data": {

"text/plain": [

"2"

]

},

"execution\_count": 28,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"countDog('My dog name is sweety and her dog name is puppy')"

]

},

{

"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": 30,

"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": 31,

"metadata": {

"id": "p1AGJ7DM85kR",

"outputId": "ca80629f-5949-4926-8d27-1b61576669ac"

},

"outputs": [

{

"data": {

"text/plain": [

"'Small Ticket'"

]

},

"execution\_count": 31,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"caught\_speeding(81,True)"

]

},

{

"cell\_type": "code",

"execution\_count": 32,

"metadata": {

"id": "BU\_UZcyk85kS",

"outputId": "699de8ef-a18c-436b-fdd9-60dc44979906"

},

"outputs": [

{

"data": {

"text/plain": [

"'Big Ticket'"

]

},

"execution\_count": 32,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"caught\_speeding(81,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": "3.9.12"

}

},

"nbformat": 4,

"nbformat\_minor": 1

}