{

"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": "06412194-a190-485f-e232-91421aaa5fd6",

"colab": {

"base\_uri": "https://localhost:8080/"

}

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"2401\n"

]

}

],

"source": [

"print(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": 2,

"metadata": {

"collapsed": true,

"id": "GD\_Tls3H85j7"

},

"outputs": [],

"source": [

"s= \"Hi there Sam\"\n",

"s.split(\" \");"

]

},

{

"cell\_type": "code",

"execution\_count": 4,

"metadata": {

"id": "RRGOKoai85j8"

},

"outputs": [],

"source": [

"s= \"Hi there dad!\"\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": 5,

"metadata": {

"collapsed": true,

"id": "2TrzmDcS85j-"

},

"outputs": [],

"source": [

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

"diameter = 12742"

]

},

{

"cell\_type": "code",

"execution\_count": 6,

"metadata": {

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

"outputId": "589d989f-2b3c-40ea-edb9-166f11ec067f",

"colab": {

"base\_uri": "https://localhost:8080/"

}

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"The diameter of Earth is 12742 kilometers.\n"

]

}

],

"source": [

"print(\"The diameter of \", planet ,\" is \", diameter , \" kilometers.\")"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "QAKtN7Hh85kB"

},

"source": [

"\*\* Given this nested list, use indexing to grab the word \"hello\" \*\*"

]

},

{

"cell\_type": "code",

"execution\_count": 7,

"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": 8,

"metadata": {

"id": "6m5C0sTW85kE",

"outputId": "722fc51f-e4e4-4ffd-dc73-181be417a9c6",

"colab": {

"base\_uri": "https://localhost:8080/"

}

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"['hello']\n"

]

}

],

"source": [

"print(lst[3][1][2])"

]

},

{

"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": 9,

"metadata": {

"id": "vrYAxSYN85kG"

},

"outputs": [],

"source": [

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

]

},

{

"cell\_type": "code",

"execution\_count": 10,

"metadata": {

"collapsed": true,

"id": "\_VBWf00q85kJ",

"outputId": "c95c18a5-216e-48c4-d8d0-f8a3d601dd79",

"colab": {

"base\_uri": "https://localhost:8080/"

}

},

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"tuple==> (1, 2, 3, 4, 5)\n",

"List==> [1, 2, 3, 4]\n"

]

}

],

"source": [

"tup =(1,2,3,4,5)\n",

"print(\"tuple==>\",tup)\n",

"lis=[1,2,3,4]\n",

"print(\"List==>\",lis)"

]

},

{

"cell\_type": "markdown",

"metadata": {

"id": "FInV\_FKB85kI"

},

"source": [

"\*\* What is the main difference between a tuple and a list? \*\*"

]

},

{

"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": 11,

"metadata": {

"collapsed": true,

"id": "unvEAwjk85kL"

},

"outputs": [],

"source": [

"def domain(text):\n",

" x = text.split(\"@\")\n",

" print(x[-1])"

]

},

{

"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": 12,

"metadata": {

"id": "6hdc169585kO"

},

"outputs": [],

"source": [

"def dogcount(value):\n",

" count = 0\n",

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

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

" count = count + 1\n",

" print(count)"

]

},

{

"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": "igzsvHb385kO",

"outputId": "abdfbdba-cb7b-4a5c-97b1-ed7fb879fbf1",

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 166

}

},

"outputs": [

{

"output\_type": "error",

"ename": "NameError",

"evalue": "ignored",

"traceback": [

"\u001b[0;31m---------------------------------------------------------------------------\u001b[0m",

"\u001b[0;31mNameError\u001b[0m Traceback (most recent call last)",

"\u001b[0;32m<ipython-input-21-0075f38eebbe>\u001b[0m in \u001b[0;36m<module>\u001b[0;34m\u001b[0m\n\u001b[0;32m----> 1\u001b[0;31m \u001b[0mfindDog\u001b[0m\u001b[0;34m(\u001b[0m\u001b[0;34m\"my dog name is Tom\"\u001b[0m\u001b[0;34m)\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n\u001b[0m",

"\u001b[0;31mNameError\u001b[0m: name 'findDog' is not defined"

]

}

],

"source": [

"findDog(\"my dog name is Tom\")"

]

},

{

"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": 13,

"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": 14,

"metadata": {

"id": "BU\_UZcyk85kS",

"outputId": "76630933-e59b-4daf-8b99-b9b0d0130434",

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 35

}

},

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"'Big Ticket'"

],

"application/vnd.google.colaboratory.intrinsic+json": {

"type": "string"

}

},

"metadata": {},

"execution\_count": 14

}

],

"source": [

"caught\_speeding(85,False)"

]

},

{

"cell\_type": "code",

"execution\_count": 15,

"metadata": {

"id": "p1AGJ7DM85kR",

"outputId": "baf5fdd1-dea4-4aaf-cddb-e4def72a09bb",

"colab": {

"base\_uri": "https://localhost:8080/",

"height": 35

}

},

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"'Small Ticket'"

],

"application/vnd.google.colaboratory.intrinsic+json": {

"type": "string"

}

},

"metadata": {},

"execution\_count": 15

}

],

"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": [

"emp = [10000,25000,12000,20000,18000]\n",

"for i in emp:\n",

" if(i>=10000 & i<15000):\n",

" print(\"total salary ==>\",i )\n",

" print(\"total expenditure ==>\" , int(i\*0.6)) \n",

"\n",

" if(i>15000 & i<25000):\n",

" print(\"total salary ==>\",i )\n",

" print(\"total expenditure ==>\" , int(i\*0.7)) \n",

" else:\n",

" print(\"total salary ==>\",i )\n",

" print(\"total expenditure ==>\" , int(i\*0.8)) "

],

"metadata": {

"id": "R5-CdXSKjacN",

"outputId": "c92be8b3-1252-4703-af2b-c519c6091050",

"colab": {

"base\_uri": "https://localhost:8080/"

}

},

"execution\_count": 16,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"total salary ==> 10000\n",

"total expenditure ==> 6000\n",

"total salary ==> 10000\n",

"total expenditure ==> 7000\n",

"total salary ==> 25000\n",

"total expenditure ==> 15000\n",

"total salary ==> 25000\n",

"total expenditure ==> 17500\n",

"total salary ==> 12000\n",

"total expenditure ==> 7200\n",

"total salary ==> 12000\n",

"total expenditure ==> 8400\n",

"total salary ==> 20000\n",

"total expenditure ==> 12000\n",

"total salary ==> 20000\n",

"total expenditure ==> 14000\n",

"total salary ==> 18000\n",

"total expenditure ==> 10800\n",

"total salary ==> 18000\n",

"total expenditure ==> 12600\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": [

"d1 = { \"Empid\":1,\"Empname\":\"Lokesh\",\"Basicpay\": 20000}\n",

"d2 = {\"deptname\":\"CSE\" , \"DEPTID\": 'CSE1024'}\n",

"d3 = {\*\*d1 , \*\*d2}\n",

"print(d3)"

],

"metadata": {

"id": "8ugVoEe0kOsk",

"outputId": "9d25fac4-003b-42f2-b96d-1f3336281ecd",

"colab": {

"base\_uri": "https://localhost:8080/"

}

},

"execution\_count": 17,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"{'Empid': 1, 'Empname': 'Lokesh', 'Basicpay': 20000, 'deptname': 'CSE', 'DEPTID': 'CSE1024'}\n"

]

}

]

}

],

"metadata": {

"colab": {

"provenance": []

},

"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

}