ASSIGNMENT 3


```
{
 "cells": [
  {
   "cell_type": "markdown",
   "source": [
    "Assignment Date\t   : 2 October 2022\n",
    "Student Name\t    : KRUTHIKA S\n",
    "Student Roll Number: 113219041054\n",
    "Maximum Marks\t    : 2 Marks\n",
    "\n",
    "\n"
   ],
   "metadata": {
    "id": "Gt0k_DFEjdFZ"
   }
  },
  {
   "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": 7,
   "metadata": {
    "id": "UhvE4PBC85j3",
    "outputId": "ba7603be-d9d7-4d45-83e0-a7f4e6170597",
    "colab": {
     "base_uri": "https://localhost:8080/"
    }
   },
   "outputs": [
    {
     "output_type": "execute_result",
     "data": {
      "text/plain": [
```

```
          "2401"
         ]
        },
        "metadata": {},
        "execution_count": 7
      }
    ],
    "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": 8,
```

```
  "metadata": {
    "collapsed": true,
    "id": "GD_Tls3H85j7"
  },
  "outputs": [],
  "source": [
    "s=\"Hi there Sum!\"\n",
    "x=s.split()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 9,
  "metadata": {
    "id": "RRGOKoai85j8",
    "outputId": "e5151c78-b748-4a41-d3ff-a3f0d286b9d3",
    "colab": {
      "base_uri": "https://localhost:8080/"
    }
  },
  "outputs": [
    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "['Hi', 'there', 'Sum!']\n"
      ]
```

```
    }
   ],
   "source": [
    "print(x)"
   ]
  },
  {
   "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": 10,
   "metadata": {
    "collapsed": true,
```

```
    "id": "2TrzmDcS85j-"
   },
   "outputs": [],
   "source": [
    "planet=\"Earth\"\n",
    "diameter=12742"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 11,
   "metadata": {
    "id": "s_dQ7_xc85j_",
    "outputId": "ac9f9a44-1867-4a3f-d1ac-a0b69704c7ea",
    "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 \"+str(planet)+\" is \"+str(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": 12,
   "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": 13,
"metadata": {
  "id": "6m5C0sTW85kE",
  "outputId": "1f768d09-5f2c-4742-90f2-63762b99e001",
  "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 35
  }
},
"outputs": [
  {
    "output_type": "execute_result",
    "data": {
      "text/plain": [
        "'hello'"
      ],
      "application/vnd.google.colaboratory.intrinsic+json": {
        "type": "string"
      }
    },
    "metadata": {},
    "execution_count": 13
  }
],
"source": [
  "lst[3][1][2][0]\n"
]
```

```
   },
   {
    "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": 14,
    "metadata": {
     "id": "vrYAxSYN85kG"
    },
    "outputs": [],
    "source": [
     "d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 15,
    "metadata": {
     "id": "FlILSdm485kH",
     "outputId": "790505fc-7fe2-46ec-f5c5-08a0e1be8614",
```

```
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 35
    }
   },
   "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "'hello'"
        ],
        "application/vnd.google.colaboratory.intrinsic+json": {
          "type": "string"
        }
       },
      "metadata": {},
      "execution_count": 15
    }
   ],
   "source": [
    "d['k1'][3]['tricky'][3]['target'][3]\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {
```

```
      "id": "FInV_FKB85kI"
     },
     "source": [
      "* What is the main difference between a tuple and a list? *"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 16,
     "metadata": {
      "collapsed": true,
      "id": "_VBWf00q85kJ"
     },
     "outputs": [],
     "source": [
      "# Tuple is immutable and 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": 17,
   "metadata": {
    "collapsed": true,
    "id": "unvEAwjk85kL"
   },
   "outputs": [],
   "source": [
    "def domainGet(input):\n",
    "    return input.split('@')[1]\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 18,
   "metadata": {
    "id": "Gb9dspLC85kL",
    "outputId": "54676fa4-d13c-4501-fe81-18ee4ac72a6f",
    "colab": {
     "base_uri": "https://localhost:8080/",
     "height": 35
    }
```

```
  },
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "'domain.com'"
        ],
        "application/vnd.google.colaboratory.intrinsic+json": {
          "type": "string"
        }
      },
      "metadata": {},
      "execution_count": 18
    }
  ],
  "source": [
    "domainGet('user@domain.com')\n"
  ]
},
{
  "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": 19,
"metadata": {
"collapsed": true,
"id": "Q4ldLGV785kM"
},
"outputs": [],
"source": [
"def findDog(input):\n",
"   return 'dog' in input.lower().split()\n"
]
},
{
"cell_type": "code",
"execution_count": 20,
"metadata": {
"id": "EqH6b7yv85kN",
"outputId": "8bbd443e-1e50-478b-fcd6-2ffafd91a6f7",
"colab": {
"base_uri": "https://localhost:8080/"
}
},

```
   "outputs": [
    {
     "output_type": "execute_result",
     "data": {
      "text/plain": [
       "True"
      ]
     },
     "metadata": {},
     "execution_count": 20
    }
   ],
   "source": [
    "findDog('Is there a dog here?')\n"
   ]
  },
  {
   "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": 21,
  "metadata": {
    "id": "6hdc169585kO"
  },
  "outputs": [],
  "source": [
    "def countDog(inp):\n",
    "    dog = 0\n",
    "    for x in inp.lower().split():\n",
    "        if x == 'dog':\n",
    "            dog += 1\n",
    "    return dog\n"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 22,
  "metadata": {
    "id": "igzsvHb385kO",
    "outputId": "51db3854-8581-4fed-cd2d-2efb2d82917b",
    "colab": {
      "base_uri": "https://localhost:8080/"
    }
  },
  "outputs": [
    {
      "output_type": "execute_result",
```

```
      "data": {
        "text/plain": [
          "2"
        ]
      },
      "metadata": {},
      "execution_count": 22
    }
  ],
  "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": 23,
"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": 24,
 "metadata": {
  "id": "BU_UZcyk85kS",
  "outputId": "01ff5049-68bc-47fb-d468-77e34dc4d9cd",
  "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": 24
  }
 ],
```

```
  "source": [
    "caught_speeding(81,True)\n"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 25,
  "metadata": {
    "id": "p1AGJ7DM85kR",
    "outputId": "f9476ac1-ae44-41e3-ec47-88e7cd146195",
    "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": 25
     }
    ],
    "source": [
     "caught_speeding(81,False)\n"
    ]
   },
   {
    "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": [
     "list1=[{'Empname': 'Krishna', 'Basic_salary': '60000'}, {'Empname': 'Abi', 'Basic_salary': '50000'},\n",
     "{'Empname': 'Raju', 'Basic_salary': '40000'},{'Empname': 'Lakshmi', 'Basic_salary': '80000'}]\n",
     "key, value='Empname', 'Basic_salary'\n",
     "res=dict()\n",
     "list2=[]\n",
     "for i in list1:\n",
```

```
    "   res[i[key]]=i[value]\n",
    "print(\"Basic salary :\"+str(res)) \n",
    "arr=[60000, 50000, 40000, 80000]\n",
    "sum=0\n",
    "val=0\n",
    "for v in range(0, len(arr)):\n",
    "  sum=arr[v]/6\n",
    "  val+=arr[v]\n",
    "print(\"Total salary:\",val) \n",
    "print(\"Total Expenditure:\", sum) "
   ],
   "metadata": {
    "id": "R5-CdXSKjacN",
    "colab": {
     "base_uri": "https://localhost:8080/"
    },
    "outputId": "4447ca29-e801-49d0-d63b-0c9424478c97"
   },
   "execution_count": 39,
   "outputs": [
    {
     "output_type": "stream",
     "name": "stdout",
     "text": [
      "Basic salary :{'Krishna': '60000', 'Abi': '50000', 'Raju': '40000', 'Lakshmi': '80000'}\n",
      "Total salary: 230000\n",
      "Total Expenditure: 13333.333333333334\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": [
     "employee={'Empid':101,'EmpName':'Krishna','Basicpay':60000}\n",
     "department={'DeptName':'CSE','DeptId':100}\n",
     "result={*employee,*department}\n",
     "print(result)"
    ],
```

```
    "metadata": {
      "id": "8ugVoEe0kOsk",
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "outputId": "5353e69b-728f-4df1-9ede-62c84c3a856e"
    },
    "execution_count": 6,
    "outputs": [
      {
        "output_type": "stream",
        "name": "stdout",
        "text": [
          "{'Empid': 101, 'EmpName': 'Krishna', 'Basicpay': 60000, 'DeptName': 'CSE', 'DeptId': 100}\n"
        ]
      }
    ]
  }
 ],
 "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
}
```