

ASSIGNMENT-3

PYTHON CODING WITH GOOGLE COLABORATORY

```
{
  "cells": [
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "view-in-github",
        "colab_type": "text"
      },
      "source": [
        "<a href=\"https://colab.research.google.com/github/IBM-EPBL/IBM-Project-18081-1659678980/blob/main/C_Magimai_Assignment_3_Python.ipynb\" target=\"_parent\"><img src=\"https://colab.research.google.com/assets/colab-badge.svg\" alt=\"Open In Colab\"/></a>"
      ]
    },
    {
      "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": "a05565aa-db43-4716-e87d-41c5c8a6f95e"
  },
  "outputs": [
    {
      "data": {
        "text/plain": [
          "2401"
        ]
      },
      "execution_count": 1,
      "metadata": {
        "tags": []
      },
      "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": null,  
  "metadata": {  
    "collapsed": true,  
    "id": "GD_Tls3H85j7"  
  },  
  "outputs": [],  
  "source": [  
    "s = \"Hi there Sam!\"\n"  
  ]
```

```
},  
{  
  "cell_type": "code",  
  "execution_count": null,  
  "metadata": {  
    "id": "RRGOKoai85j8",  
    "outputId": "cc52f0d8-2ed1-4b4d-e956-5bb332cdc2"  
  },  
  "outputs": [  
    {  
      "data": {  
        "text/plain": [  
          "['Hi', 'there', 'dad!']"  
        ]  
      },  
      "execution_count": 3,  
      "metadata": {  
        "tags": []  
      },  
      "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": null,
  "metadata": {
    "collapsed": true,
    "id": "2TrzmDcS85j-"
  },
  "outputs": [],
```

```
"source": [  
  "planet = \"Earth\\n\",  
  "diameter = 12742\\n"  
],  
{  
  "cell_type": "code",  
  "execution_count": null,  
  "metadata": {  
    "id": "s_dQ7_xc85j_",  
    "outputId": "4235fdbf-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 {} kilometres.\".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": [
    "\n",
    "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": 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": "FIILSdm485kH",
    "outputId": "4232540d-95c2-461d-c78d-24ea62398e08"
  },
  "outputs": [
```

```
{
  "data": {
    "text/plain": [
      "hello"
    ]
  },
  "execution_count": 16,
  "metadata": {
    "tags": []
  },
  "output_type": "execute_result"
},
{
  "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": null,
  "metadata": {
    "collapsed": true,
    "id": "_VBWf00q85kJ"
  },
  "outputs": [],
  "source": [
    "#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": 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": [
    "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": 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": "e7909af1-8df1-4534-fc8c-27b03d7369e5"
  },
  "outputs": [
    {
      "data": {
        "text/plain": [
          "True"
        ]
      }
    ]
  }
}
```

```
    },  
    "execution_count": 28,  
    "metadata": {  
      "tags": []  
    },  
    "output_type": "execute_result"  
  }  
],  
  "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": "0602a2b5-0b18-48d8-e2d4-fe644cbccf8a"
  },
  "outputs": [
    {
      "data": {
        "text/plain": [
          "2"
        ]
      }
    ]
  }
}
```



```

    ]
  },
  "execution_count": 31,
  "metadata": {
    "tags": []
  },
  "output_type": "execute_result"
}
],
"source": [
  "countDog('This dog runs faster than the other dog dude!')"
]
},
{
  "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": "699de8ef-a18c-436b-fdd9-60dc44979906"
  },
  "outputs": [
    {
      "data": {
        "text/plain": [
          "'Big Ticket'"
        ]
      },
      "execution_count": 6,
      "metadata": {
        "tags": []
      },
      "output_type": "execute_result"
    }
  ],
  "source": [
    "caught_speeding(81,False)"
  ]
}
```

```
]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {
    "id": "p1AGJ7DM85kR",
    "outputId": "ca80629f-5949-4926-8d27-1b61576669ac"
  },
  "outputs": [
    {
      "data": {
        "text/plain": [
          "'Small Ticket'"
        ]
      },
      "execution_count": 5,
      "metadata": {
        "tags": []
      },
      "output_type": "execute_result"
    }
  ],
  "source": [
    "caught_speedin(81,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 retrieve each employee salary and calculate total salary expenditure. "

```
],
```

```
"metadata": {
```

```
"id": "Tie4rC7_kAOC"
```

```
}
```

```
},
```

```
{
```

```
"cell_type": "code",
```

```
"source": [
```

```
"for(i=0;i<=5;i++)\n",
```

```
"sample_dict = {\n",
```

```
"  \"name\": \"Kelly\", \n",
```

```
"  \"age\": 28, \n",
```

```
"  \"salary\": 20000, \n",
```

```
"  \"city\": \"New york\" \n",
```

```
"  \"emp id\": '62345' } \n",
```

```
"key=[\"name\", \"salary\"]"
```

```
],
```

```
"metadata": {
```

```
"id": "R5-CdXSKjacN"
},
"execution_count": null,
"outputs": []
},
{
"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": [
"def Merge(dict_1,dict_2):\n",
"    result = dict_1| dict_2\n",
```

```
" return result\n",  
"dict_1={ 'Empid': 76543, 'Empname': muthu, 'Basicpay': 9000}\n",  
"dict_1={ 'DeptName': computer science engineering, 'DeptName': 98761,}\n",  
"dict_3 = Merge(dict_1,dict_2)\n",  
"print(dict_3)\n",  
"\n"  
],  
"metadata": {  
  "id": "8ugVoEe0kOsk"  
},  
"execution_count": null,  
"outputs": []  
}  
],  
"metadata": {  
  "colab": {  
    "provenance": [],  
    "include_colab_link": true  
  },  
  "kernelpec": {  
    "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  
}
```


Browser tabs: Welcome to Project! Delight..., IBM, Created using Colaboratory, My Drive - Google Drive, M.Suba_Assignment_3_Pyth...

Address bar: colab.research.google.com/drive/1r7kz79df1HfMG_TuyEsAjyCz1Ktdpnjg

Navigation bar: Gmail, YouTube, Maps, TCS iON| Digital Le..., TCS iON| Digital Le..., I have seek this cou..., Sent Mail - aswiniy..., Anna University - C..., online-gaming.web...

M.Suba_Assignment_3_Python.ipynb

File Edit View Insert Runtime Tools Help Last saved at 9:16 AM

Comment Share Settings

+ Code + Text

Connect Editing

7 **4

2481

** What is 7 to the power of 4?**

Indented block

** Split this string:**

```
s = "Hi there Sam!"
```

into a list.

```
[ ] s = "Hi there Sam!"
```

```
[ ] s.split()
```

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

Windows taskbar: Type here to search, 28°C Partly cloudy, ENG, 09:19, 07-10-2022

The screenshot shows a Google Colaboratory notebook interface. The browser tabs at the top include 'Welcome to Project! Delight...', 'IBM', 'Created using Colaboratory', 'My Drive - Google Drive', and the active tab 'M.Suba_Assignment_3_Python'. The notebook title is 'M.Suba_Assignment_3_Python.ipynb' with a star icon and a 'Last saved at 9:16 AM' timestamp. The interface includes a left sidebar with icons for file explorer, search, and code execution. The main area displays the following Python code:

```
[ ] s.split()

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

** Given the variables:**

planet = "Earth"
diameter = 12742

** Use .format() to print the following string: **

The diameter of Earth is 12742 kilometers.

[ ] planet = "Earth"
diameter = 12742

[ ] print("The diameter of {} is {} kilometres.".format(planet,diameter))

The diameter of Earth is 12742 kilometres
```

The bottom of the image shows a Windows taskbar with a search bar, several application icons, and system status indicators including 'Good air', 'ENG', and the date '07-10-2022'.

The screenshot shows a Google Colab notebook with the following content:

```
[ ] print("The diameter of {} is {} kilometres.".format(planet,diameter))

The diameter of Earth is 12742 kilometers.

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

[ ]
1st = [1,2,[3,4],[5,[100,200,['hello']],23,11],1,7]

[ ] 1st[3][1][2][0]

'hello'

** Given this nest dictionary grab the word "hello". Be prepared, this will be annoying/tricky **

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

[ ] d['k1'][3]['tricky'][3]['target'][3]

'hello'
```

Browser tabs: Welcome to Project! Delight..., IBM, Created using Colaboratory, My Drive - Google Drive, M.Suba_Assignment_3_Python.ipynb

Address bar: colab.research.google.com/drive/1r7kz79df1HfMG_TuyEsAgyCz1Ktdpnjg

Navigation: Gmail, YouTube, Maps, TCS iONJ Digital Le..., TCS iONJ Digital Le..., I have seek this cou..., Sent Mail - aswiniy..., Anna University - C..., online-gaming.web...

File M.Suba_Assignment_3_Python.ipynb ☆

Menu: File Edit View Insert Runtime Tools Help Last saved at 9:16 AM

Buttons: Comment, Share, Settings, Profile

Code Editor: + Code + Text Connect Editing

```
'hello'
```

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

```
[ ] #Tuple is immutable
```

**** Create a function that grabs the email website domain from a string in the form: ****

```
user@domain.com
```

So for example, passing "user@domain.com" would return: domain.com

```
[ ] def domainGet(email):  
    return email.split('@')[-1]
```

```
[ ] domainGet('user@domain.com')
```

```
'domain.com'
```

**** 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**

Windows Taskbar: Type here to search, 28°C Partly cloudy, 09:24 07-10-2022

Browser tabs: Welcome to Project! Delight..., IBM, Created using Colaboratory, My Drive - Google Drive, M.Suba_Assignment_3_Python.ipynb

Address bar: colab.research.google.com/drive/1r7kz79df1HfMG_TuyEsAjyCz1Ktdpnjg

Navigation bar: M.Suba_Assignment_3_Python.ipynb, File, Edit, View, Insert, Runtime, Tools, Help, Last saved at 9:16 AM, Comment, Share, Settings, Profile

Code editor tabs: + Code, + Text, Connect, Editing

Code:

```
'domain.com'
```

Comments:

** 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. **

```
[ ] def findDog(st):  
    return 'dog' in st.lower().split()  
  
[ ] findDog('Is there a dog here?')
```

Output:

```
True
```

Comments:

** Create a function that counts the number of times the word "dog" occurs in a string. Again ignore edge cases. **

```
[ ] def countDog(st):  
    count=0  
    for word in st.lower().split():  
        if word == 'dog':  
            count += 1  
    return count  
  
[ ] countDog('This dog runs faster than the other dog dude!')
```

Windows taskbar: Type here to search, 28°C Partly cloudy, 09:26 07-10-2022

Welcome to Project! Delighti x IBM Created using Colaboratory x My Drive - Google Drive x M.Suba_Assignment_3_Pythc x

colab.research.google.com/drive/1r7kz79df1HfMG_TuyEsAjyCz1Ktdpnjg

Gmail YouTube Maps TCS iONJ Digital Le... TCS iONJ Digital Le... I have seek this cou... Sent Mail - aswiniy... Anna University - C... online-gaming.web...

M.Suba_Assignment_3_Python.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 9:16 AM

Comment Share Settings

+ Code + Text

Connect Editing

[] countDog('This dog runs faster than the other dog dude!')

2

Problem

**You are driving a little too fast, and a police officer stops you. Write a function to return one of 3 possible results: "No ticket", "Small ticket", or "Big Ticket". If your speed is 60 or less, the result is "No Ticket". If speed is between 61 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 cases. **

[] def caught_speeding(speed, is_birthday):

if is_birthday:

speeding = speed - 5

else:

speeding = speed

if speeding > 80:

return 'Big Ticket'

elif speeding > 60:

return 'Small Ticket'

else:

return 'No Ticket'

return speeding

Type here to search

28°C Partly cloudy ENG 09:27 07-10-2022

colab.research.google.com/drive/1r7kz79df1HfMG_TuyEsAjyCz1Ktdpnjg

M.Suba_Assignment_3_Python.ipynb

File Edit View Insert Runtime Tools Help Last saved at 9:16 AM

Comment Share

+ Code + Text

speed can be 5 higher in all cases. *

```
[ ] def caught_speeding(speed, is_birthday):  
  
    if is_birthday:  
        speeding = speed - 5  
    else:  
        speeding = speed  
  
    if speeding > 80:  
        return 'Big Ticket'  
    elif speeding > 60:  
        return 'Small Ticket'  
    else:  
        return 'No Ticket'  
  
[ ] caught_speeding(81,False)  
  
'Big Ticket'  
  
[ ] caught_speeding(81,True)  
  
'Small Ticket'
```

Type here to search

28°C Partly cloudy 09:28 07-10-2022

Welcome to Project! Delighti x IBM x Created using Colaboratory x My Drive - Google Drive x M.Suba_Assignment_3_Pythi x +

colab.research.google.com/drive/1r7kz79df1HfMG_TuyEsAjyCz1Ktdpnjg

Gmail YouTube Maps TCS iONJ Digital Le... TCS iONJ Digital Le... I have seek this cou... Sent Mail - aswiniy... Anna University - C... online-gaming.web...

M.Suba_Assignment_3_Python.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 9:16 AM

Comment Share Settings

+ Code + Text

Connect Editing

Create an employee list with basic salary values(at least 5 values for 5 employees) and using a for loop retrieve each employee salary and calculate total salary expenditure.

```
[ ] for(i=0;i<=5;i++)
    sample_dict = {
        "name": "Kelly",
        "age": 28,
        "salary": 20000,
        "city": "New york"
        "emp id": '62345' }
    key=["name", "salary"]
```

Create two dictionaries in Python:
First one to contain fields as Empid, Empname, Basicpay
Second dictionary to contain fields as DeptName, DeptId.
Combine both dictionaries.

```
[ ] def Merge(dict_1,dict_2):
    result = dict_1| dict_2
    return result
dict_1 = {'Empid': 76543, 'Empname': 'mithu', 'Basicpay': 9999}
```

Type here to search

28°C Partly cloudy 09:28 07-10-2022

Browser tabs: Welcome to Project! Delight..., IBM, Created using Colaboratory, My Drive - Google Drive, M.Suba_Assignment_3_Pyth...

Address bar: colab.research.google.com/drive/1r7kz79df1HfMG_TuyEsAiyCz1Ktdpnjg

Navigation bar: M.Suba_Assignment_3_Python.ipynb, File Edit View Insert Runtime Tools Help, Last saved at 9:16 AM, Comment, Share, Settings, Profile

Code editor: + Code + Text, Connect, Editing

Code:

```
[ ] key-[name], salary ]
```

Create two dictionaries in Python:

First one to contain fields as Empid, Empname, Basicpay

Second dictionary to contain fields as DeptName, DeptId.

Combine both dictionaries.

```
[ ] def Merge(dict_1,dict_2):  
    result = dict_1| dict_2  
    return result  
dict_1 ={'Empid': 76543, 'Empname': muthu, 'Basicpay': 9000}  
dict_1 ={'DeptName': computer science engineering, 'DeptName': 98761,}  
dict_3 = Merge(dict_1,dict_2)  
print(dict_3)
```

Windows taskbar: Type here to search, 28°C Partly cloudy, 09:29, 07-10-2022

