Assignment -3

Python Programming

Assignment Date	29 September 2022
Student Name	Kameshwaran P
Student Roll Number	19P114
Maximum Marks	2 Marks


```
{
 "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": 44,
   "metadata": {
    "id": "UhvE4PBC85j3",
    "outputId": "06b31237-e1b3-449a-cf10-e456ad3a04d4",
    "colab": {
     "base_uri": "https://localhost:8080/"
    }
   },
   "outputs": [
```

```
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
      "2401"
     ]
    },
    "metadata": {},
    "execution_count": 44
   }
  ],
  "source": [
   "#pow(7, 4)\n",
   "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",
   "l1 = s.split()"
  ]
```

```
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "RRGOKoai85j8",
  "outputId": "70eb8a75-0c1c-4355-f574-0f544a7d1a50",
  "colab": {
   "base_uri": "https://localhost:8080/"
  }
 },
 "outputs": [
  {
   "output_type": "stream",
   "name": "stdout",
   "text": [
    "['Hi', 'there', 'Sam!']\n"
   ]
  }
 ],
 "source": [
  "print(l1)"
 ]
},
{
 "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-",
  "colab": {
   "base_uri": "https://localhost:8080/",
   "height": 36
  },
  "outputId": "0bd62df0-12c8-4ede-b639-695c4fc5a794"
 },
 "outputs": [
  {
   "output_type": "execute_result",
   "data": {
    "text/plain": [
     "'The diameter of Earth is 12742 kilometers'"
    ],
    "application/vnd.google.colaboratory.intrinsic+json": {
     "type": "string"
    }
   },
   "metadata": {},
   "execution_count": 5
  }
 ],
 "source": [
  "planet = \"Earth\"\n",
  "diameter = 12742\n",
  "input_s = \"The diameter of {} is {} kilometers\"\n",
  "input_s.format(planet, diameter)"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "s_dQ7_xc85j_",
  "outputId": "4582877c-db0b-41ae-dc50-a8a49212d623",
  "colab": {
   "base_uri": "https://localhost:8080/",
```

```
    "height": 36
   }
  },
  "outputs": [
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
      "'The diameter of Earth is 12742 kilometers'"
     ],
     "application/vnd.google.colaboratory.intrinsic+json": {
      "type": "string"
     }
    },
    "metadata": {},
    "execution_count": 7
   }
  ],
  "source": [
   "input_s.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": 42,
  "metadata": {
  "collapsed": true,
   "id": "-7dzQDyK85kD"
  },
  "outputs": [],
  "source": [
   "lst = [1,2,[3,4],[5,[100,200,['hello']],23,11],1,7]\n"
```

```
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 43,
    "metadata": {
     "id": "6m5C0sTW85kE",
     "outputId": "43eca2eb-b911-4c64-f39f-10e1929a7a47",
     "colab": {
      "base_uri": "https://localhost:8080/"
     }
    },
    "outputs": [
     {
      "output_type": "stream",
      "name": "stdout",
      "text": [
       "hello\n"
      ]
     }
    ],
    "source": [
     "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": 4,
    "metadata": {
     "id": "vrYAxSYN85kG"
    },
    "outputs": [],
```

```
    "source": [
     "d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 23,
    "metadata": {
     "id": "FlILSdm485kH",
     "outputId": "63e1d09d-32e1-4845-9aca-1de23b80ad4f",
     "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 36
     }
    },
    "outputs": [
     {
      "output_type": "execute_result",
      "data": {
       "text/plain": [
        "'hello'"
       ],
       "application/vnd.google.colaboratory.intrinsic+json":            {
"type": "string"
       }
      },
      "metadata": {},
      "execution_count": 23
     }
    ],
    "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": [
 "thisdict = {\n",
  "  \"brand\": \"Ford\",\n",
  " \"model\": \"Mustang\",\n",
  "  \"year\": 1964\n",
  "}"
 ]
},
{
 "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": 39,
 "metadata": {
  "collapsed": true,
  "id": "unvEAwjk85kL"
 },
 "outputs": [],
 "source": [
  "def getDomain(email):\n",
  "  return email[email.index('@') + 1:]"
```

```
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 40,
   "metadata": {
    "id": "Gb9dspLC85kL",
    "outputId": "9dac19d5-d292-4b99-b730-da5ed57dfa39",
    "colab": {
     "base_uri": "https://localhost:8080/",
     "height": 36
    }
   },
   "outputs": [
    {
     "output_type": "execute_result",
     "data": {
      "text/plain": [
       "'domain.com'"
      ],
      "application/vnd.google.colaboratory.intrinsic+json": {
       "type": "string"
      }
     },
     "metadata": {},
     "execution_count": 40
    }
   ],
   "source": [
    "getDomain(\"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": 37,
 "metadata": {
 "collapsed": true,
  "id": "Q4ldLGV785kM"
 },
 "outputs": [],
 "source": [
  "#def check(input):\n",
  " # if(input.__contains__('dog')):\n",
  "  #  return True\n",
  "  #else:\n",
  "  # return False \n",
  "#input = \"The dog is the most loveable animal in the world\"\n",
  "#check(input)\n",
  "def findDog(st):\n",
  "   return 'dog' in st.lower().split()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 38,
 "metadata": {
  "id": "EqH6b7yv85kN",
  "outputId": "5d06ad86-d448-4ce3-d144-8e519e0f58ee",
  "colab": {
   "base_uri": "https://localhost:8080/"
  }
 },
 "outputs": [
  {
   "output_type": "execute_result",
   "data": {
    "text/plain": [
     "True"
    ]
   },
   "metadata": {},
   "execution_count": 38
```

```
      }
     ],
     "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": 35,
     "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": 36,
     "metadata": {
      "id": "igzsvHb385kO",
      "outputId": "06b698f3-4e2d-4597-ccfa-ddf32cdfb796",
      "colab": {
       "base_uri": "https://localhost:8080/"
      }
```

```
  },
  "outputs": [
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
      "2"
     ]
    },
    "metadata": {},
    "execution_count": 36
   }
  ],
  "source": [
   "countDog('This dog runs faster than the other dog')"
  ]
 },
 {
  "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": 32,
  "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": 33,
  "metadata": {
   "id": "BU_UZcyk85kS",
   "outputId": "9ac79e12-74e2-49b7-8215-ecf6d341ac13",
   "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 36
   }
  },
  "outputs": [
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
      "'Big Ticket'"
     ],
     "application/vnd.google.colaboratory.intrinsic+json": {
      "type": "string"
     }
    },
    "metadata": {},
    "execution_count": 33
   }
  ],
```

```
    "source": [
     "caught_speeding(81, False)"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 34,
    "metadata": {
     "id": "p1AGJ7DM85kR",
     "outputId": "10ad5ad9-c431-48d8-a2a1-807dd4055b24",
     "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 36
     }
    },
    "outputs": [
     {
      "output_type": "execute_result",
      "data": {
       "text/plain": [
        "'Small Ticket'"
       ],
       "application/vnd.google.colaboratory.intrinsic+json":            {
"type": "string"
       }
      },
      "metadata": {},
      "execution_count": 34
     }
    ],
    "source": [
     "caught_speeding(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 retreive each employee salary and calculate total salary expenditure. "
    ],
    "metadata": {
     "id": "Tie4rC7_kAOC"
```

```
  }
 },
 {
  "cell_type": "code",
  "source": [
   "emp_list = [120000, 150000, 90000, 45000, 28000]\n",
   "salary_expenditure = 0\n",
   "for i in emp_list:\n",
   "  salary_expenditure += i\n",
   "print(salary_expenditure)"
  ],
  "metadata": {
   "id": "R5-CdXSKjacN",
   "colab": {
    "base_uri": "https://localhost:8080/"
   },
   "outputId": "3a8063ca-cdff-467d-8cc0-642e3e923111"
  },
  "execution_count": null,
  "outputs": [
   {
    "output_type": "stream",
    "name": "stdout",
    "text": [
     "433000\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": [
    "dict1 = { \"Empid\" : 1,\n",
    "         \"Empname\" : \"abc\",\n",
    "         \"Basicpay\" : 1200\n",
    "        }\n",
    "dict2 = { \"DeptName\" : \"CSE\",\n",
    "         \"DeptId\" : 1,\n",
    "        }    \n",
    "print({**dict1, **dict2})"
   ],
   "metadata": {
    "id": "8ugVoEe0kOsk",
    "colab": {
     "base_uri": "https://localhost:8080/"
    },
    "outputId": "f88963ed-526f-4202-9d01-1caa14927123"
   },
   "execution_count": null,
   "outputs": [
    {
     "output_type": "stream",
     "name": "stdout",
     "text": [
      "{'Empid': 1, 'Empname': 'abc', 'Basicpay': 1200, 'DeptName': 'CSE', 'DeptId': 1}\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
}
```