Day - 9

**Dictionaries, Nesting**

**and**

**the Secret Auction**

Nesting list inside list, Dictionaries inside Dictionaries or both

**9.1 Dictionaries**

A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets, and they have keys and values. Separated by commas:

|  |  |
| --- | --- |
| **{key\_1 : value\_1, key\_2 : value\_2, key\_3 : value\_3, …… }**  For assigning  **Dict\_name[Key] = value**  For defining  **Key : value** |  |

#*key - value pair*

dictionary\_test = {

    "Bug": "An error in a program that prevents the program from running as expected.",

    "Function": "A piece of code that you can easily call over and over again.",

    "Loop": "The action of doing something over and over again."

}

thisdict = {

  "brand": "Ford",

  "model": "Mustang",

  "year": 1964

}

#RETRIEVINGitems*from Dictionaries*

#*dictionary\_name[key]*

bug\_data = dictionary\_test["Bug"]

**print**(bug\_data)

**print**(thisdict["year"])

**print**(thisdict)

# ADDINGnewitems*to dictionary.*

dictionary\_test["Book"] = "Tings with papaer and ink"

**print**(dictionary\_test)

#CREATE an EMPTY dictionary*.*

new\_dict\_test = {}

#WIPE an existing dictionary

thisdict = {}

**print**(thisdict)

#EDIT an item*in a dictionary*

dictionary\_test["Book"] = "Edited Book"

#For loop*of dictionary*

**for** key **in** dictionary\_test :

**print**(key)

**print**(dictionary\_test[key])

#*python dict.py*

* Exercise 9.1: Grading program.

#*key - value pair*

studentScore = {

    "Harry": 81,

    "Ron": 72,

    "Hermione": 99,

    "Draco": 74,

    "Neville": 62

}

student\_grade = {}

**for** student **in** studentScore:

**if** 91 **<=** studentScore[student] **<** 100:

    grade = "Outstanding"

**elif** 81 **<=** studentScore[student] **<=** 90:

    grade = "Exceed Expectations"

**elif** 70 **<=** studentScore[student] **<=** 80:

    grade = "Acceptable"

**elif** studentScore[student] **<** 70:

    grade = "Failed"

  student\_grade[student] = grade

**print**(student\_grade)

#*python grade.py*

**9.2 Nesting Lists and Dictionaries**

In key-value pair value could be list or Dictionary. Key is usually String.

#*Nesting*LISTinsideDICTIONARY

carDict = {

  "brand": "Ford",

  "model": "Mustang",

  "year": [1964, 1890, 2008, 2509]

}

travel\_log = {

  "France": ["Paris", "Lille", "Dijon"],

  "Germany": ["Berlin", "Hamburg", "Stuttgrad"]

}

**print**(carDict["year"])

**print**(travel\_log["Germany"][2])

**print**(travel\_log)

#*Nesting*LISTinsideLIST

bui = ["a", "c", [1, 2, 5], "D"]

**print**(bui)

**print**(bui[2][1])

#*Nesting*DICTIONARYinsideDICTIONARY

travel\_log\_2 = {

  "France": {"cities\_visited" : ["Paris", "Lille", "Dijon"], "total\_visited": 15},

  "Germany": {"cities\_visited" : ["Berlin", "Hamburg", "Stuttgrad"], "total\_visited": 5}

}

**print**(travel\_log\_2["France"])

**print**(travel\_log\_2["France"]["cities\_visited"])

**print**(travel\_log\_2["Germany"]["total\_visited"])

#*Nesting*DICTIONARYinsideLIST

travel\_log\_3 = [

  { "country":"France",

    "cities\_visited" : ["Paris", "Lille", "Dijon"],

    "total\_visited": 15},

  { "country": "Germany",

    "cities\_visited" : ["Berlin", "Hamburg", "Stuttgrad"],

    "total\_visited": 5}

]

**print**(travel\_log\_3)

**print**(travel\_log\_3[1]["country"])

**print**(travel\_log\_3[0]["cities\_visited"][2])

#*python dict\_list\_nesting.py*

* Exercise 9.2: Nesting Exercise. Crate the function **add\_new\_country**(countryName, no\_visit, city\_list)

travel\_log\_3 = [

  { "country":"France",

    "cities\_visited" : ["Paris", "Lille", "Dijon"],

    "total\_visited": 15},

  { "country": "Germany",

    "cities\_visited" : ["Berlin", "Hamburg", "Stuttgrad"],

    "total\_visited": 5}

]

**def** **add\_new\_country**(countryName, no\_visit, city\_list):

  temp\_dict = {}

  temp\_dict["country"] = countryName

  temp\_dict["cities\_visited"] = city\_list

  temp\_dict["total\_visited"] = no\_visit

  travel\_log\_3**.append**(temp\_dict)

**add\_new\_country**("Russia", 2, ["Moscow", "Saint Petersburg"])

**add\_new\_country**("India", 20, ["Mumbai", "Gujrat", "Calcutta"])

**print**(travel\_log\_3)

#*python nesting\_excercise.py*

* Exercise 9.3: Secret Auction Project. Create a dictionary and pick max valued one.

Practice Version

**import** secret\_auctn\_art

**import** os

**print**(secret\_auctn\_art**.**logo)

**print**("WelCome to secret Auction")

bidders = []

**def** **call\_bid**(bidr\_name, bidr\_bid):

  personInfo = {}

  personInfo["Name"] = bidr\_name

  personInfo["Bid"] = bidr\_bid

  bidders**.append**(personInfo)

end = **False**

**while** end **==** **False**:

  prsn\_name = **input**("\n\tEnter your name : ")

  prsn\_bid = **int**(input("\n\tYour bid-value : "))

**call\_bid**(bidr\_name = prsn\_name, bidr\_bid = prsn\_bid)

  ask = **input**("Are there any other bidders? Type 'yes' or 'no' : ")

**if** ask **==** "yes":

    #*clear screen*

    os**.system**('cls')

**elif** ask **==** "no":

**print**(bidders)

**print**("\n\t\t\tgoodbye\n")

    end = **True**

mAx = bidders[0]["Bid"]

winner = bidders[0]["Name"]

**for** bdr **in** bidders:

**if** bdr["Bid"] **>** mAx:

    mAx = bdr["Bid"]

    winner = bdr["Name"]

**print**(f"\n\t\t The winner is : {winner}\n")

#*python secret\_auctn.py*

Instructor version

**import** secret\_auctn\_art

**import** os

**print**(secret\_auctn\_art**.**logo)

**print**("WelCome to secret Auction")

bids = {}

end = **False**

**while** **not** end:

  name = **input**("\n\tEnter your name : ")

  price = **int**(input("\n\tWhat is your bid-value? $ : "))

  bids[name] = price

  ask = **input**("Are there any other bidders? Type 'yes' or 'no' : ")

**if** ask **==** "yes":

    #*clear screen*

    os**.system**('cls')

**elif** ask **==** "no":

**print**(bids)

**print**("\n\t\t\tgoodbye\n")

    end = **True**

mAx = 0

winner = ""

**for** bidder **in** bids:

**if** bids[bidder] **>** mAx:

    mAx = bids[bidder]

    winner = bidder

**print**(f"\n\t\t The winner is : {winner} with Amount {mAx}\n")

#*python secret\_auctn.py*