**Python Lab Assignment 2**

**Akshay.G**

**16231179**

1. String sorting

Solution:

*#Taking the input string from terminal and storing it.*str = input(**"enter the string"**)  
*#converting string to lower case  
#splitting string into words and storing the sorted elements (duplication will be removed)*str1 = sorted(set((str.lower().split())))  
*#joining the unique sorted words*str2 = **" "**.join(str1)  
print(str2)

2. Dictionary of squares

*#This code generates dictionary of squares of integers(>=1)*a = int(input(**"Enter an integer upto which you want to generate the dictinary of squares"**))  
*#Function dict() to generate dictionary of squares from 1 to given number  
#If any wrong input is given (i.e <1) then it asks again until valid input is given***def** dict(a):  
 **if** (a>=1):  
 dictionry = {x: x \* x **for** x **in** range(1, a + 1)}  
 print(dictionry)  
 **else**:  
 print(**"Given input is not valid"**)  
 a = int(input(**"Enter valid number"**))  
 dict(a) *#Recursion - calling the function again until valid input is given*dict(a)

4. Random vector using numpy.

**import** numpy **as** np  
*#creating vector of size 15 with some random values between 1-15*myvector = np.random.randint(1,15, size=15)  
print(myvector)  
x=myvector.max()  
*#printing the max value in the vector*print(**"Max value in the vector is:"**,x)  
print(**"Replacing max value with 100"**)  
*# Replacing the max value in the vector with 100***for** i **in** range(0,14):  
 **if**(myvector[i]==x):  
 myvector[i]=100  
print(myvector)

5. Expense Tracker

print(**" Welcome to Budget Tracker application \n Please provide your details:"**)  
name = input(**"Your Name:"**)  
age = int(input(**"Age:"**))  
email = input(**"Your email"**)  
num = int(input(**"Your mobile number"**))  
SSN = int(input(**"Enter your SSN"**))  
lim = int(input(**"Enter tracking limit amount"**))  
*#\_\_init\_\_ constructor and self are implemented in every class.  
#class 1 created***class** Details():  
 **def** \_\_init\_\_(self, n, a, e, nu):  
 self.name = n  
 self.age = a  
 self.email = e  
 self.num = nu  
  
 **def** details(self,s):  
 self.\_\_ssn = s *#private variable used here* print(**"\n Name: "**, self.name + **"\n Email: "**, self.email)  
 print(**" Age:"**, self.age)  
 print(**" Mobile no.:"**, self.num)  
 print(**" SSN:"**, self.\_\_ssn)  
*#class 2, Inheritance and super implemented***class** Income(Details):  
 income = 0  
  
 **def** \_\_init\_\_(self, n, a, e, nu):  
 super(Income, self).\_\_init\_\_(n, a, e, nu) *#super implemented here* **def** upd\_salary(self):  
 self.a = int(input(**"Enter your savings"**))  
 self.\_\_class\_\_.income = self.\_\_class\_\_.income + self.a  
 print(**"Total savings: "**, self.\_\_class\_\_.income)  
*#class 3, Ineritance implemented***class** Addexp(Details):  
 a = 0  
  
 **def** \_\_init\_\_(self, n, a, e, nu):  
 Details.\_\_init\_\_(self, n, a, e, nu)  
  
 **def** upd\_exp(self):  
 b = []  
 c = int(input(**"Enter the number of expenses you want to add:"**))  
 **for** x **in** range(0, c):  
 t = int(input(**"Add your expense here "**))  
 b.append(t)  
 print(**"List of expenses added"**, b)  
 self.exp = sum(b)  
 self.\_\_class\_\_.a = self.\_\_class\_\_.a + self.exp  
 print(**"Sum of expenses added:"**, self.\_\_class\_\_.a)  
  
*#class4, multiple Inheritance implemented below***class** Balance(Addexp, Income):  
 bal = 0  
 **def** \_\_init\_\_(self):  
 print(**"Check your balance here:"**)  
 **def** bala(self):  
 self.\_\_class\_\_.bal = Income.income - Addexp.a  
 print(**"Total Savings:"**,Income.income)  
 print(**"Total Expenses:"**, Addexp.a)  
 print(**"Amount Left:"**, self.\_\_class\_\_.bal)  
  
*#class 5, Inheritance implemented***class** Tracking(Balance):  
 **def** \_\_init\_\_(self, l):  
 self.tr = l  
 **if** (Balance.bal < self.tr):  
 print(**"Your remaining balance:"**,Balance.bal)  
 print(**"Your balance is less than your set limit"**)  
 **elif** (Balance.bal > self.tr):  
 print(**"Your remaining balance:"**, Balance.bal)  
  
  
  
a = Details(name, age, email, num)  
a.details(SSN)  
b = Income(name, age, email, num) *#Instance of class Income 'b' is created and it extends instance of class Details 'a'*b.upd\_salary() *#updating income of instance 'b'*c = Addexp(name, age, email, num) *#Instance of class Addexp 'c' is created and it extends instance of class Details 'a'*c.upd\_exp() *#updating expenses of instance 'c'*d = Balance() *#Instance of class Balance 'd' is created and it extends instance of classes Income 'b' and Addexp 'c'*d.bala() *#checking balancing with the instance 'd'*e = Tracking(lim) *#Instance of class Tracking 'e' is created and it extends instance of class Balance 'd'*b.upd\_salary()  
c.upd\_exp()  
d.bala()  
e.\_\_init\_\_(lim)