# **Assignments**

| 1. Accept a char input from the user and display it on the console. |
| --- |
| *Code of the program & screenshot of the output.*  *a = input('Enter Some Words: ')*  *print(a)* |
| 1. Accept two inputs from the user and output its sum.  | **Variable** | **Data Type** | | --- | --- | | Number 1 | Integer | | Number 2 | Float | | Sum | Float | |
| *Code of the program & screenshot of the output.*  *num1 = int(input("Enter number 1 : "))*  *num2 = float(input("Enter number 2 : "))*  *sum = num1 + num2*  *print(float(sum))* |
| 1. Write a program to find the simple interest.    1. Program should accept 3 inputs from the user and calculate simple interest for the given inputs. Formula: SI=(P\*R\*n)/100)  | **Variable** | **Data Type** | | --- | --- | | Principal amount (P) | Integer | | Interest rate (R) | Float | | Number of years (n) | Float | | Simple Interest (SI) | Float | |
| *Code of the program & screenshot of the output.*  *p = int(input("Enter princple amount: "))*  *r = float(input("Enter Intrest rate: "))*  *n = float(input("Enter Number of years: "))*  *si = (p\* r\* n)/100*  *print(float(si))* |
| 1. Write a program to check whether a student has passed or failed in a subject after he or she enters their mark (pass mark for a subject is 50 out of 100). 2. Program should accept an input from the user and output a message as “Passed” or “Failed”  | **Variable** | **Data type** | | --- | --- | | mark | float | |
| *Code of the program & screenshot of the output.*  *mark = float(input("Enter the Mark: "))*  *if(50 < mark):*  *print("Passed")*  *else:*  *print("Failed")* |
| 1. Write a program to show the grade obtained by a student after he/she enters their total mark percentage. 2. Program should accept an input from the user and display their grade as follows  | **Mark** | **Grade** | | --- | --- | | > 90 | A | | 80-89 | B | | 70-79 | C | | 60-69 | D | | 50-59 | E | | < 50 | Failed |  | **Variable** | **Data type** | | --- | --- | | Total mark | float | |
| *Code of the program & screenshot of the output.*  *mark = float(input("Enter you Mark: "))*  *if(mark > 90):*  *print("Grade A")*  *elif(mark > 80):*  *print("Grade B")*  *elif(mark > 70):*  *print("Grade c")*  *elif(mark > 60):*  *print("Grade D")*  *elif(mark > 50):*  *print("Grade E")*  *else:*  *print("Failed")* |
| 1. Using the ‘switch case’ write a program to accept an input number from the user and output the day as follows.  | **Input** | **Output** | | --- | --- | | 1 | Sunday | | 2 | Monday | | 3 | Tuesday | | 4 | Wednesday | | 5 | Thursday | | 6 | Friday | | 7 | Saturday | | Any other input | Invalid Entry | |
| *Code of the program & screenshot of the output.*  *day = int(input("Enter a number: "))*  *if (day == 1):*  *print("Sunday")*  *elif (day == 2):*  *print("Monday")*  *elif(day == 3):*  *print("Tuesday")*  *elif(day == 4):*  *print("Wensday")*  *elif(day == 5):*  *print("Thursday")*  *elif(day == 6):*  *print("Friday")*  *elif(day == 7):*  *print("Saturday")*  *else:*  *print("invalid entry")* |
| 1. Write a program to print the multiplication table of given numbers. Using for and while 2. Accept an input from the user and display its multiplication table   Eg:  **Output**: Enter a number  **Input**: 5  **Output**:  1 x 5 = 5  2 x 5 = 10  3 x 5 = 15  4 x 5 = 20  5 x 5 = 25  6 x 5 = 30  7 x 5 = 35  8 x 5 = 40  9 x 5 = 45  10 x 5 = 50 |
| *Code of the program & screenshot of the output.*  *num = int(input("Enter a number: "))*  *for i in range(1, 11):*  *print(i, 'x', num, '=', num\*i)* |
| 1. Write a program to print the following pattern (**hint**: use nested loop)   1  1 2  1 2 3  1 2 3 4  1 2 3 4 5 |
| *Code of the program & screenshot of the output.*  *num = int(input("Enter the rows: "))*  *for i in range(1, num + 1):*  *for j in range(1, i + 1):*  *print(j, end=' ')*  *print('')* |
| 1. Write a program to create a copy of array and add an element to copied array . show both arrays. 2. Program should accept an array from the user, swap the values of two arrays and display it on the console   Eg: **Output**: Enter the size of Array 1  **Input**: 5  **Output**: Enter the values of Array 1  Array 1: 10, 20, 30, 40, 50  Array 2 : Copy of Array 1  Array 2 : 10, 20, 30, 40, 50 + add a element  **Output :** Array 1 **[**10, 20, 30, 40, 50 ]  **Output :** Array 2 **[**10, 20, 30, 40, 50, 60, 70 ] |
| *Code of the program & screenshot of the output.*  *from array import \**  *arr1 = array('i',[])*  *n = int(input("Enter the length: "))*  *for i in range(n):*  *x = int(input("Enter the values {}: ".format(i+1)))*  *arr1.append(x)*  *print(arr1)*  *arr2 = arr1*  *arr2.append(70)*  *arr2.append(80)*  *print(arr2)* |
| 1. Write a program to sort an array in descending order without sort() and sorted() Program should accept and array, sort the array values in descending order and display it  * Selection * Insertion * bubble   Eg: **Output**: Enter the size of an array  **Input**: 5  **Output**: Enter the values of array  **Input**: 20, 10, 50, 30, 40  **Output**: Sorted array:  50, 40, 30, 20, 10 |
| *Code of the program & screenshot of the output.*  *def sort\_des(arr):*  *n = len(arr)*  *for i in range (n):*  *for j in range (0, n-i-1):*  *if arr[j] < arr[j+1]:*  *arr[j], arr[j+1] = arr[j+1], arr[j]*  *return arr*  *n = int(input("Enter the Size of Array: "))*  *arr = []*  *for i in range(n):*  *num = int(input("Enter the element {}: ".format(i+1)))*  *arr.append(num)*  *sorted = sort\_des(arr)*  *print("Sorted values: ",sorted)* |
| 1. Write a program to identify whether a string is a palindrome or not without using reverse(), slicing 2. A string is a palindrome if it reads the same backward or forward eg: MALAYALAM   Program should accept a string and display whether the string is a palindrome or not  Eg: **Output**: Enter a string  **Input**: MALAYALAM  **Output**: Entered string is a palindrome  Eg 2: **Output**: Enter a string  **Input**: HELLO  **Output**: Entered string is not a palindrome |
| *Code of the program & screenshot of the output.*  *def palin(s):*  *left=0*  *right = len(s) - 1*  *while left < right:*  *if s[left] != s[right]:*  *return False*  *left +=1*  *right -=1*  *return True*  *s = input("Enter a string: ")*  *if palin(s):*  *print(format(s)," is a palindrome")*  *else:*  *print(format(s),"is not a palindrome",)* |
| 1. Write a program to add to two dimensional arrays, understand the memory management of list 2. Program should accept two 2D arrays and display its sum   Eg: **Output**: Enter the size of arrays  **Input**: 3  **Output**: Enter the values of array 1  **Input**:  1 2 3  4 5 6  7 8 9  **Output**: Enter the values of array 2  **Input**:  10 20 30  40 50 60  70 80 90  **Output**: Sum of 2 arrays is:  11 22 33  44 55 66  77 88 99 |
| *Code of the program & screenshot of the output.* |
| 1. Grades are computed using a weighted average. Suppose that the written test counts 70%, lab exams 20% and assignments 10%.   If Arun has a score of  Written test = 81  Lab exams = 68  Assignments = 92  Arun’s overall grade = (81x70)/100 + (68x20)/100 + (92x10)/100 = 79.5  Write a program to find the grade of a student during his academic year.   * 1. Program should accept the scores for written test, lab exams and assignments   2. Output the grade of a student (using weighted average)   Eg:  Enter the marks scored by the students  Written test = 55  Lab exams = 73  Assignments = 87  Grade of the student is 61.8 |
| *Code of the program & screenshot of the output*  *wt = int(input("Written Test = "))*  *le = int(input("Lab exams = "))*  *ass = int(input("Assigments = "))*  *total = float(wt\*70)/100 + (le\*20)/100 + (ass \*10)/100*  *print(total)* |
| 1. *Study about functions*  * *User defined* * *Types of Arguments* * *Lambda*   *Write a program using user defined functions and lambda functions* |
| *Code of the program & screenshot of the output*  *nums = [1,2,3,4,5,6,7]*  *even = list(filter(lambda n : n%2 == 0,nums))*  *print(even)* |
| 1. Write a program to accept an array and display it on the console using functions 2. Program should contain 3 functions including main() function   **main()**   1. Declare an array 2. Call function getArray() 3. Call function displayArray()   **getArray()**   1. Get values to the array   **displayArray()**   1. Display the array values 2. Study about global, local, non-local |
| *Code of the program & screenshot of the output.*  *def getArray():*  *global arr*  *arr = []*  *n = int(input("Enter the size of array: "))*  *for i in range(n):*  *num = int(input("Enter the number: "))*  *arr.append(num)*  *def displayArray():*  *print("Array elements ar: ")*  *for i in arr:*  *print(i)*  *def main():*  *getArray()*  *displayArray()*  *if \_\_name\_\_ == "\_\_main\_\_":*  *main()* |
| 1. Write a program to print “HELLO WORLD “using function without using print inside of function. (“HELLO WORLD “must be inside Decorator function) |
| *Code of the program & screenshot of the output.*  *def my\_decorator(func):*  *def wrapper():*  *return "HELLO WORLD"*  *return wrapper*  *@my\_decorator*  *def my\_function():*  *pass*  *result = my\_function()*  *print(result)* |
| 1. Write a menu driven program to do the basic mathematical operations such as addition, subtraction, multiplication and division (**hint**: use if else ladder or switch) 2. Program should have 4 functions named addition(), subtraction(), multiplication() and division() 3. Should create a class object and call the appropriate function as user prefers in the main function |
| *Code of the program & screenshot of the output.*  *class Calculator:*  *def addition(self, num1, num2):*  *return num1 + num2*  *def subtraction(self, num1, num2):*  *return num1 - num2*  *def multplication(self, num1, num2):*  *return num1 \* num2*  *def divsion(self, num1, num2):*  *return num1 / num2*  *calculator = Calculator()*  *print("Select an operator")*  *print("1. Addition\n2. Subtraction\n3. Multiplication\n4. Division")*  *choice = int(input("Enter the choce: "))*  *num1 = float(input("Enter the first number: "))*  *num2 = float(input("Enter the second number: "))*  *if choice == 1:*  *result = calculator.addition(num1, num2)*  *print("Result = ",result)*  *elif choice == 2:*  *result = calculator.subtraction(num1, num2)*  *print("Result = ",result)*  *elif choice == 3:*  *result = calculator.multplication(num1, num2)*  *print("Result = ",result)*  *elif choice == 4:*  *result = calculator.divsion(num1, num2)*  *print("Result = ",result)* |
| 1. Write a program to print the following pattern using for loop   7 8 9 10  4 5 6  2 3  1 |
| *Code of the program & screenshot of the output.*  *n,k=7,11*  *a,l=3,4*  *for i in range(1,5):*  *for j in range(n,k):*  *print(j,end=" ")*  *print("")*  *n=n-a*  *k=k-l*  *a-=1*  *l-=1* |
| 1. Write a program to add the values of two 2D arrays 2. Program should contain a class, Functions should be inside the class 3. Call function getArray() using an object 4. Call function addArray() using an object 5. Call function displayArray() using an object   **getArray()**   1. Get values to the array   **getArray()**   1. Add array 1 and array 2   **displayArray()**   1. Display the array values   Eg:  Enter the size of array  2  Enter the values of array 1  1 2  3 4  Enter the values of array 2  5 6  7 8  Output:  Sum of array 1 and array 2:  6 8  10 12 |
| *Code of the program & screenshot of the output* |
| 1. Write a program to include all the functionalities of a calculator using ABSTRACT class and abstract method. All the methods (add, sub, mul, div) should be inside of abstract class. Abstract method definition should be in another class.   Examples :  from abc import ABC, abstractmethod  class Calculator(ABC):  def \_\_init\_\_(self, id, name):  self.id = id self.name = name  @abstractmethod  def add (self):  pass |
| *Code of the program & screenshot of the output*  *from abc import ABC, abstractmethod*  *class Calculator(ABC):*  *@abstractmethod*  *def add(self, x, y):*  *pass*  *@abstractmethod*  *def sub(self, x, y):*  *pass*  *@abstractmethod*  *def mul(self, x, y):*  *pass*  *@abstractmethod*  *def div(self, x, y):*  *pass*  *class BasicClaculator(Calculator):*  *def add(self, x, y):*  *return x + y*  *def sub(self, x, y):*  *return x - y*  *def mul(self, x, y):*  *return x \* y*  *def div(self, x, y):*  *if y == 0:*  *return "Cannot Divide by zero"*  *return x / y*    *Calculator = BasicClaculator()*  *print("1 + 2 =", Calculator.add(1, 2))*  *print("4 - 2 =", Calculator.sub(4, 2))*  *print("3 \* 5 =", Calculator.mul(3, 5))*  *print("6 / 3 =", Calculator.div(6, 3))*  *print("10 / 1 =", Calculator.div(10, 1))* |
| 1. Write a program to build a home. The Home class should define all the attributes of each room in a home. From the Home class create two homes. FirstHome and SecondHome. First home should have an extra study room as a method. SecondHome should have the work\_area as an extra method. should use the concept of inheritance.   class Home:  def \_\_init\_\_(self):  pass  def room1:  width=100  breadth = 100  print('are of room1',width\*breath)  def kitchen:  width = 1222  breadth = 4888  print('are of kitchen',width\*breath)  you should have mentioned all the plans of home here as methods.  Class FirstHome(Home):  # define the extra method's  pass  class SecondHome(Home):  # define the extra method's  pass |
| *Code of the program & screenshot of the output*  *class Home:*  *def room1(self):*  *width = 100*  *breadth = 100*  *print('area of room1', width \* breadth)*  *def kitchen(self):*  *width = 1222*  *breadth = 4888*  *print('area of kitchen', width \* breadth)*  *class FirstHome(Home):*  *def \_\_init\_\_(self):*  *self.home = "First Home"*  *def study\_room(self):*  *width = 120*  *breadth = 120*  *print('area of study room in first home ', width \* breadth)*  *class SecondHome(Home):*  *def \_\_init\_\_(self):*  *self.home = "Second Home"*  *def work\_area(self):*  *width = 100*  *breadth = 100*  *print('area of work area in second home ', width \* breadth)*  *fh = FirstHome()*  *sh = SecondHome()*  *print(fh.home)*  *fh.room1()*  *fh.kitchen()*  *fh.study\_room()*  *print(sh.home)*  *sh.room1()*  *sh.kitchen()*  *sh.work\_area()* |
| 1. *Write a program to create Class with name and account number and implement get and set, with property decorator and making account number and name private.*   *class BankAccount:*  *def \_\_init\_\_(self, name, account\_number):*  *self.name = name*  *self.accont\_number = account\_number*  *@property*  *def name(self):*  *return self.name*    *@name.setter*  *def name(self, name):*  *self.name = name*  *@property*  *def account\_number(self):*  *return self.account\_number*    *@account\_number.setter*  *def account\_number(self, accont\_number):*  *self.accont\_number = account\_number*  *account = BankAccount("jhon","123978655")*  *print(account.name)*  *print(account.accont\_number)*  *account.name = "dasan"*  *account.accont\_number = "6572130347"*  *print(account.name)*  *print(account.accont\_number)* |
| 1. *Write a function to calculate the sum of all numbers passed as its arguments. Your function should be called sum\_numbers and should define a single variable argument (i.e. a star argument) that will get the values to sum.*   *Test the function with the following values:*  *Values Result*  *1, 2, 3 6*  *8, 20, 2 30*  *12.5, 3.147, 98.1 113.747*  *1.1, 2.2, 5.5 8.8* |
| 1. *pantry = {*   *"chicken": 500,*  *"lemon": 2,*  *"cumin": 24,*  *"paprika": 18,*  *"chilli powder": 7,*  *"yogurt": 300,*  *"oil": 450,*  *"onion": 5,*  *"garlic": 9,*  *"ginger": 2,*  *"tomato puree": 125,*  *"almonds": 75,*  *"rice": 500,*  *"coriander": 20,*  *"lime": 3,*  *"pepper": 8,*  *"egg": 6,*  *"pizza": 2,*  *"spam": 1,*  *}*  *recipes = {*  *"Butter chicken": [*  *"chicken",*  *"lemon",*  *"cumin",*  *"paprika",*  *"chilli powder",*  *"yogurt",*  *"oil",*  *"onion",*  *"garlic",*  *"ginger",*  *"tomato puree",*  *"almonds",*  *"rice",*  *"coriander",*  *"lime",*  *],*  *"Chicken and chips": [*  *"chicken",*  *"potatoes",*  *"salt",*  *"malt vinegar",*  *],*  *"Pizza": [*  *"pizza",*  *],*  *"Egg sandwich": [*  *"egg",*  *"bread",*  *"butter",*  *],*  *"Beans on toast": [*  *"beans",*  *"bread",*  *],*  *"Spam a la tin": [*  *"spam",*  *"tin opener",*  *"spoon",*  *],*  *}*  *observe the dictionary above and write a menu driven python program to create recipes. Once one recipe is done then the quantity of the items in pantry should also be reduced*  *Eg : If you cook beans on toast the beans quantity and bread quantity need to decrease i.e., one from the total quantity each.* |
| 1. *create a custom exception class and raise this exception when user press one in the menu and handles this exception.* |
| 1. *Write a list comprehension that returns the list ["1\*2=1", "22=4", "32=9", ...., "25\*2=625"]* |
| 1. *Using dict comprehension and a conditional argument create a dictionary from the current dictionary where only the key:value pairs with value above 2000 will be taken to the new dictionary.*   *dict1={"NFLX":4950,"TREX":2400,"FIZZ":1800, "XPO":1700}*  *dict2={}*  *dict1={"NFLX":4950,*  *"TREX":2400,*  *"FIZZ":1800,*  *"XPO":1700}*  *new\_dict = {k: v for k, v in dict1.items() if v > 2000}*  *print(new\_dict)* |