# **Assignments**

| 1. Accept a char input from the user and display it on the console. |
| --- |
| user\_input=input("enter a character:")  print("you entered "+user\_input) |
| 1. Accept two inputs from the user and output its sum.  | **Variable** | **Data Type** | | --- | --- | | Number 1 | Integer | | Number 2 | Float | | Sum | Float | |
| number1 = int(input("enter first number:"))  number2 = float(input("enter second number:"))  sum = float(number1 + number2)  print("sum of the numbers=",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 | |
| principal\_amount = int(input("Enter the Principal amount:"))  interest\_rate = float(input("Enter the Interest Rate:"))  number\_of\_years = float(input("Enter the number of years:"))  simple\_interest = float((principal\_amount\*interest\_rate\*number\_of\_years)/100)  print("Simple interest=",simple\_interest) |
| 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 | |
| mark = float(input("Enter your mark out of 100:"))  if mark >=50:  print("you passed the exam")  else:  print("you failed the exam") |
| 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 | |
| total\_mark = float(input("Enter your Total number:"))  if total\_mark >= 90:  print("Grade A")  elif 90>total\_mark>=80:  print("Grade B")  elif 80>total\_mark>=70:  print("Grade C")  elif 70>total\_mark>=60:  print("Grade D")  elif 60>total\_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 | |
| def main():  day\_num = int(input("Enter the number of the day:"))  match day\_num:  case 1:  print("Sunday")  case 2:  print("Monday")  case 3:  print("Tuesday")  case 4:  print("Wednesday")  case 5:  print("Thursday")  case 6:  print("Friday")  case 7:  print("Saturday")  case \_:  print("Invalid Number")    if \_\_name\_\_ == "\_\_main\_\_" :  main() |
| 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 |
| *Using for loop*      *Using while loop*  *def main():*  *number = int(input("Enter a number:"))*  *i=1*  *while i<=10:*  *print(f"{i}x{number}={i\*number}")*  *i+=1*  *if \_\_name\_\_=="\_\_main\_\_":*  *main()* |
| 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 |
| def main():  row = int(input("Enter the number of rows:"))  for i in range(1,row+1):  for j in range(1,i+1):  print(j,end=" ")  print()    if \_\_name\_\_=="\_\_main\_\_":  main() |
| 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 ]  from array import \*  arr1 = array("i",[])  arr2 =array("i",[])  size = int (input("Enter the size of the Array:"))  print("Enter the elements of the array")  for i in range(size):  value = int(input(f"enter [{i}] element of the array:"))  arr1.append(value)  arr2.extend(arr1)  new\_value = int(input("enter the new element for the copied array:"))  arr2.append(new\_value)  print("The Original Array",arr1) |
|  |
| 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 |
| *Selection sort*  *from array import \**  *arr = array("i",[])*  *size = int (input("Enter the size of the Array:"))*  *print("Enter the Values of the array")*  *for i in range(size):*  *value = int(input(f"enter [{i}] element of the array:"))*  *arr.append(value)*  *print("The array:",arr)*  *for i in range(size-1):*  *max\_index = i*  *for j in range(i, size):*  *if arr[j] > arr[max\_index]:*  *max\_index = j*  *arr[i],arr[max\_index] = arr[max\_index],arr[i]*  *print("The array after sorting:",arr)*    *Insertion sort*  *from array import \**  *arr = array("i",[])*  *size = int (input("Enter the size of the Array:"))*  *print("Enter the Values of the array")*  *for i in range(size):*  *value = int(input(f"enter [{i}] element of the array:"))*  *arr.append(value)*  *print("The array:",arr)*  *for i in range(1, size):*  *key = arr[i]*  *j = i - 1*  *while j >= 0 and arr[j] < key:*  *arr[j + 1] = arr[j]*  *j -= 1*  *arr[j + 1] = key*  *print("The array after sorting:",arr)*    *Bubble sort*  *from array import \**  *arr = array("i",[])*  *size = int (input("Enter the size of the Array:"))*  *print("Enter the Values of the array")*  *for i in range(size):*  *value = int(input(f"enter [{i}] element of the array:"))*  *arr.append(value)*  *print("The array:",arr)*  *for i in range(size - 1):*  *for j in range(size - 1 - i):*  *if arr[j] < arr[j + 1]:*  *arr[j], arr[j + 1] = arr[j + 1], arr[j]*  *print("The array after sorting:",arr)* |
| 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 |
| user\_input = input("Enter a string:")  user\_input = user\_input.lower()  length = len(user\_input)  for i in range(length//2):  if user\_input[i] != user\_input[length-i-1]:  print("The string is not a palindrome")  break  else:  print("The string is 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 |
| def user\_array():  size = int(input("Enter the size of the array:"))  array = []  print("Enter the elements of the array:")  for i in range(size):  row = []  for j in range(size):  element = int(input(f"Enter element [{i}][{j}]:"))  row.append(element)  array.append(row)  return array  def main():  print("First Array")  array1 = user\_array()  print("Second Array")  array2 = user\_array()  sum = []  if len(array1) != len(array2) or len(array1[0]) != len(array2[0]):  print("different size cannot find sum")  else:  for i in range(len(array1)):  row = []  for j in range(len(array1[0])):  row.append(array1[i][j] + array2[i][j])  sum.append(row)  print("Array sum")  for row in sum:  print(row)  if \_\_name\_\_=="\_\_main\_\_":  main() |
| 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 |
| print("Enter the marks scored by the students")  written\_test = float(input("Enter the Written Test mark:"))  lab\_test = float(input("Enter the Lab Test mark:"))  assignment = float(input("Enter the Assignment mark:"))  grade = float((written\_test \* 70)/100 + (lab\_test \* 20)/100 + (assignment \* 10)/100)  print(f"Grade of the student is {grade}") |
| 1. *Study about functions*  * *User defined* * *Types of Arguments* * *Lambda*   *Write a program using user defined functions and lambda functions* |
| *User defined functions*  *def square(num):*  *return num\*num*  *number = int(input("Enter a number:"))*  *print("The square of the number is ",square(number))*    *Lambda functions*  *nums = [3,2,6,8,5,7,1]*  *even\_nums = list(filter(lambda n : n%2==0,nums))*  *double = list(map(lambda n : n\*2,nums))*  *print(nums)*  *print(even\_nums)*  *print(double)* |
| 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 |
| def get\_array(array):  size = int(input("Enter the size of the array:"))  print("Enter the elements of the array:")  for i in range(size):  element = int(input(f"Enter element [{i+1}]:"))  array.append(element)  return array  def display\_array(array):  print(array)  def main():  array = []  get\_array(array)  display\_array(array)  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) |
| def hello\_world():  return "HELLO WORLD "  def decorator(func):  def wrapper():  return func()  return wrapper  hello\_world = decorator(hello\_world)  print(hello\_world()) |
| 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.* |
| 1. Write a program to print the following pattern using for loop   7 8 9 10  4 5 6  2 3  1 |
| row=4  count=7  for i in range(row,0,-1):  for j in range(1,i+1):  print(count,end=' ')  count+=1  count=round((count/2)-1)  print() |
| 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 |
| class Add\_Array:  def \_\_init\_\_(self):  self.result = []  def getArray(self):  size = int(input("Enter the size of the array:"))  array = []  print("Enter the elements of the array:")  for i in range(size):  row = []  for j in range(size):  element = int(input(f"Enter element [{i}][{j}]:"))  row.append(element)  array.append(row)  return array    def addArray(self,array1,array2):  if len(array1) != len(array2) or len(array1[0]) != len(array2[0]):  print("different size cannot find sum")  else:  for i in range(len(array1)):  row = []  for j in range(len(array1[0])):  row.append(array1[i][j] + array2[i][j])  self.result.append(row)  def displayArray(self):  if self.result:  print("Sum of the array1 and array2")  for row in self.result:  print(row)  matrix = Add\_Array()  array1 = matrix.getArray()  array2 = matrix.getArray()  matrix.addArray(array1,array2)  matrix.displayArray() |
| 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 |
| from abc import ABC,abstractmethod  class Calculator(ABC):  def \_\_init\_\_(self,id,name):  self.id = id  self.name = name  @abstractmethod  def add(self, x, y):  pass  @abstractmethod  def subtract(self, x, y):  pass  @abstractmethod  def multiply(self, x, y):  pass  @abstractmethod  def divide(self, x, y):  pass  class SubClass(Calculator):  def add(self, x, y):  return x + y  def subtract(self, x, y):  return x - y  def multiply(self, x, y):  return x \* y  def divide(self, x, y):  if y != 0:  return x / y  else:  return "can't divide by zero"    operator = SubClass(1,"Basic Calculator")  print("Addition:",operator.add(25,5))  print("Subtraction:",operator.subtract(25,5))  print("Multiplication:",operator.multiply(25,5))  print("Division:",operator.divide(25,5)) |
| 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 |
| class Home:  def \_\_init\_\_(self):  pass    def room1(self):  width = 100  breadth = 100  print(f"Area of the room1={width\*breadth},width={width},breadth={breadth}")  def room2(self):  width = 100  breadth = 100  print(f"Area of the room2={width\*breadth},width={width},breadth={breadth}")  def dinninghall(self):  width = 100  breadth = 100  print(f"Area of the dinninghall={width\*breadth},width={width},breadth={breadth}")  def bathroom(self):  width = 100  breadth = 100  print(f"Area of the bathroom={width\*breadth},width={width},breadth={breadth}")  def kitchen(self):  width = 200  breadth = 250  print(f"Area of the kitchen={width\*breadth},width={width},breadth={breadth}")  class FirstHome(Home):  def study\_room(self):  width = 150  breadth = 200  print(f"Area of the studyroom={width\*breadth},width={width},breadth={breadth}")    class SecondHome(Home):  def work\_Area(self):  width = 150  breadth = 150  print(f"Area of the work area={width\*breadth},width={width},breadth={breadth}")  first\_home = FirstHome()  first\_home.room1()  first\_home.room2()  first\_home.dinninghall()  first\_home.bathroom()  first\_home.kitchen()  first\_home.study\_room()  print()  second\_home = SecondHome()  second\_home.room1()  second\_home.room2()  second\_home.dinninghall()  second\_home.bathroom()  second\_home.kitchen()  second\_home.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.\_\_account\_number = account\_number*  *@property*  *def name(self):*  *return self.\_\_name*  *@name.setter*  *def name(self, new\_name):*  *self.\_\_name = new\_name*  *@property*  *def account\_number(self):*  *return self.\_\_account\_number*  *@account\_number.setter*  *def account\_number(self, new\_account\_number):*  *self.\_\_account\_number = new\_account\_number*  *account =BankAccount("Arun","1234567890")*  *print("name:",account.name)*  *print("Account Number:",account.account\_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*  *def sum\_numbers(\*numbers):*  *sum=0*  *for i in numbers:*  *sum += i*  *print(f"sum of numbers={sum}")*  *sum\_numbers(1,2,3)*  *sum\_numbers(8,20,2)*  *sum\_numbers(12.5,3.147,98.1)*  *sum\_numbers(1.1,2.2,5.5)* |
| 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.*  *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",*  *],*  *}*  *def cook\_recipe(recipe\_name):*  *global pantry*  *recipe = recipes.get(recipe\_name)*  *if recipe:*  *for ingredient in recipe:*  *if ingredient in pantry:*  *pantry[ingredient] -= 1*  *else:*  *print(f"Ingredient '{ingredient}' not found in pantry.can't make {recipe\_name}")*  *return*  *print(f"Successfully cooked {recipe\_name}!")*  *else:*  *print(f"Error: Recipe '{recipe\_name}' not found.")*  *def pantry\_items():*  *print("\nRemaining pantry items:")*  *for key, value in pantry.items():*  *print(f"{key}:{value}")*  *def menu():*  *recipe\_names = list(recipes.keys())*  *print("Menu")*  *for index, item in enumerate(recipe\_names, start=1):*  *print(f"{index}.{item}")*  *def main():*  *menu()*  *recipe\_names = list(recipes.keys())*  *choice = int(input("Enter the number of your order:"))*  *if 1 <= choice <=len(recipe\_names):*  *recipe\_name = recipe\_names[choice-1]*  *cook\_recipe(recipe\_name)*  *pantry\_items()*  *else:*  *print("Invalid Menu munber")*  *if \_\_name\_\_=="\_\_main\_\_":*  *main()* |
| 1. *create a custom exception class and raise this exception when user press one in the menu and handles this exception.*   *class OptionOne(Exception):*  *pass*  *def options():*  *print("menu\n1.none\n2.item two\n3.item three\n4.item four\nselect your option")*  *def option\_one(option):*  *if option==1:*  *raise OptionOne("option 1 is none")*  *elif(1<option<=4):*  *print(f"selected option {option}")*  *else:*  *print("invalid option")*  *def main():*  *try:*  *options()*  *option=int(input("enter your option:"))*  *option\_one(option)*  *except OptionOne as e:*  *print(f"Error: {e}")*  *print("select another option")*  *main()* |
| 1. *Write a list comprehension that returns the list ["1\*2=1", "22=4", "32=9", ...., "25\*2=625"]*   *list=[*  *f"{i}\*2={i\*i}" for i in range(1,26)*  *]*  *print(list)* |
| 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*  *}*  *dict2={*  *key: value for key,value in dict1.items() if value > 2000*  *}*  *print(dict2)* |