**Week 4- Programs on Iterative constructs, Lists and Tuples**

**Name: NAREN CHANDRASHEKHAR**

**SRN: PES2UG20CS216**

**Section: G**

|  |  |
| --- | --- |
| Program 1 | 1. Write a program to generate fibonacci series till n terms 2. Find factorial of a number 3. prints all prime numbers from 2 - n |
|  | **Algorithm:**  **For Fibonacci Series**  **Step1: Start**  **Step2: Input the value of number of terms, n**  **Step3: Set value of fib1 and fib2 as 0 and 1 respectively**  **Step4: Set counter value as 3**  **Step5: fib3 = fib1+fib2**  **Step6: Print fib3**  **Step7: fib1 = fib2**  **Step8: fib2 = fib3**  **Step9: counter=counter+1**  **Step10: Repeat step5 to 9 till value of counter is <=n**  **For Factorial**  **Step1: Start**  **Step2: Input number to find factorial, m**  **Step3: Set counter, j and factorial as 1**  **Step4: factorial=factorial\*j**  **Step5: j=j+1**  **Step6: while j<=m repeat steps 4 and 5**  **Step7: Print factorial**  **Step8: End**  **For Prime numbers between 2 and p**  **Step1: Start**  **Step2: Input value of p**  **Step3: num = 2;**  **Step4: k=2; isPrime=True**  **Step5: if num%k == 0:**  **isPrime = False; go to step7**  **Step6: k=k+1**  **Step7: Repeat steps 5 and 6 till k<num**  **Step8: Print the prime number, num = num + 1**  **Step9: Repeat steps 4 to 8 till num < p**  **Step9: End** |
|  | **Program with comments:**  **#Fibonacci Series**  **n = int(input("Enter the number of terms in Fibonacci series "))**  **fib1 = 0 #assigning first term in Fibonacci as 0**  **fib2 = 1 #assigning first term in Fibonacci as 1**  **print(fib1)**  **print(fib2)**  **i = 3 #Set counter as 3 as two terms are already printed**  **while i<=n:**  **fib3 = fib1+fib2 #assign next fibonacci term as the sum of previous two terms**  **print(fib3)**  **#changing values of fib1 and fib2 to the next values**  **fib1 = fib2**  **fib2 = fib3**  **i+=1**  **#Factorial of a number**  **m = int(input("Enter a positive number to find its factorial "))**  **j = 1 #counter**  **factorial = 1 #assigning factorial as 1, as 0 and 1 factorial is 1**  **while j<=m:**  **factorial = factorial\*j #calculating factorial**  **j+=1**  **print("The factorial of the number is",factorial)**  **#Prime numbers**  **p = int(input("Enter a positive number to print all prime number from 2 "))**  **print("Prime numbers between 2 to",p,"are")**  **for num in range(2,p):**  **k = 2**  **isPrime = True #assuming all numbers to be prime**  **while k<num:**  **if num%k == 0:**  **isPrime = False #segregating prime numbers from non prime numbers**  **break**  **k+=1**  **if(isPrime):**  **print(num)** |
|  | **Output:** |
| Program 2 | Write a python program to perform the following operations using given list as input:   1. **Given a heterogenous list, create separate lists for different types of data. Write a python program to achieve the same**. 2. **Sort in ascending and descending order**   **i)list of strings ii) list of number** |
|  | **Algorithm:**  **Step1: Start**  **Step2: Set, l1=[1,2,"hello",(5,8,9),10.5,[6,7],3.5,(1,2,3),[11,12],{23,34},"aaa"]**  **Step3: Create empty lists l\_int, l\_float, l\_str, l\_tuple, l\_list, l\_set**  **Step4: set i = 0**  **Step5: if type of l1[i] = int, add l1[i] to l\_int[]**  **Step6: elif type of l1[i] = float, add l1[i] to l\_float[]**  **Step7: elif type of l1[i] = str, add l1[i] to l\_str[]**  **Step8: elif type of l1[i] = tuple, add l1[i] to l\_tuple[]**  **Step9: elif type of l1[i] = list, add l1[i] to l\_list[]**  **Step10: elif type of l1[i] = set, add l1[i] to l\_set[]**  **Step11: i=i+1**  **Step12: Repeat steps 5 to 11 till end of the list**  **Step13: End**  **Sort in ascending and descending order**  **Step1: Start**  **Step2: Initialise strlist=['aaa','xxx','bbb','abs','xyz','bcd']**  **numlist=[23,12,34,56,26,33,22]**  **Step3: strlist.sort()**  **Step4: Print strlist**  **Step5: strlist.sort(reverse=True)**  **Step6: Print strlist**  **Step7: numlist.sort()**  **Step8: Print numlist**  **Step9: numlist.sort(reverse=True)**  **Step10:print numlist**  **Step11: End** |
|  | **Program with Comments:**  **#A**  **l1=[1,2,"hello",(5,8,9),10.5,[6,7],3.5,(1,2,3),[11,12],{23,34},"aaa"]**  **#creating empty lists for every data type**  **l\_int = []**  **l\_float = []**  **l\_str = []**  **l\_tuple = []**  **l\_list = []**  **l\_set = []**  **#checking data type of elements**  **for i in l1:**  **c = type(i)**  **if(c == int):**  **l\_int.append(i)**  **elif(c == float):**  **l\_float.append(i)**  **elif(c == str):**  **l\_str.append(i)**  **elif(c == tuple):**  **l\_tuple.append(i)**  **elif(c == list):**  **l\_list.append(i)**  **elif(c == set):**  **l\_set.append(i)**  **print("List of integers",l\_int)**  **print("List of floating point numbers",l\_float)**  **print("List of strings",l\_str)**  **print("List of tuples",l\_tuple)**  **print("List of lists",l\_list)**  **print("List of sets",l\_set)**  **#B**  **strlist=['aaa','xxx','bbb','abs','xyz','bcd']**  **numlist=[23,12,34,56,26,33,22]**  **#organising list in ascending and descending order respectively**  **strlist.sort()**  **print("The string in ascending order is",strlist)**  **strlist.sort(reverse=True)**  **print("The string in descending order is",strlist)**    **numlist.sort()**  **print("The numbers in ascending order is",numlist)**  **numlist.sort(reverse=True)**  **print("The numbers in descending order is",numlist)** |
|  | **Output:** |
| Program 3 | Generate heart rate randomly between 50 to 120 at time interval of 3 hours for 24 hours.   1. **If heart rate is between 50-65 print as bradycardia(lower heart rate) if greater than 100 print as tachycardia(higher heart rate). Else print as normal.** 2. **Count number of Bradycardia and tachycardia if any of this is greater than 3 display as risk.** 3. **Print the maximum heart rate and minimum heart rate** |
|  | **Algorithm:**  **Step1: Start**  **Step2: Set brady\_cardia = 0; tachy\_cardia = 0; c = 0; heart\_rate=[]**  **Step3: Print 8 random values between 50 and 120**  **Step4: if i>50 and i<65:**  **brady\_cardia= brady\_cardia+1**  **print Bradycardia**  **elif i>100:**  **tachy\_cardia=tachy\_cardia+1**  **print Tachycardia**  **else:**  **print Normal**  **Step5: if (brady\_cardia or tachy\_cardia)>3:**  **Print Risk**  **Step6: Print maximum and minimum heart rate**  **Step7: End** |
|  | **Program with comments:**  **import random**  **#setting limit of Bradycardia, Trachycardia and counter as 0**  **brady\_cardia = 0**  **tachy\_cardia = 0**  **c = 0**  **heart\_rate=[] #creating empty list**  **#getting 8 random values of heart rate between 50 and 120**  **while(c<8):**  **i = random.randint(50,120)**  **heart\_rate.append(i)**  **c=c+1**  **print("The heart rate at intervals of 3 hours for 24 hours is",heart\_rate)**  **for i in heart\_rate:**  **if i>50 and i<65:**  **brady\_cardia+=1 #incrementing count of Bradycardia**  **print("Heart rate",i,"is Bradycardia")**  **elif i>100:**  **tachy\_cardia+=1 #incrementing count of Trachycardia**  **print("Heart rate",i,"is Tachycardia")**  **else:**  **print("Heart rate",i,"is Normal")**  **if (brady\_cardia or tachy\_cardia)>3:**  **print("Risk because count of Bradycardia or Tachycardia is greater than 3")**  **#printing maximum and minimum heart rate**  **print("The maximum heart rate is",max(heart\_rate))**  **print("The minimum heart rate is",min(heart\_rate))** |
|  | **Output:** |
| Program 4 | Enter marks of students till you need to stop.   1. **Find maximum marks** 2. **Find number of students who have scored highest** 3. **Find second highest marks** 4. **Enter fail marks and remove if fail marks present in list** |
|  | **Algorithm:**  **Step1: Start**  **Step2: Initialize student\_marks = []; n = 0;count = 0**  **Step 3: Read the marks in student\_marks[] until the input is negative value**  **Step 4: Print max(student\_marks)**  **Step 5: max\_marks = max(student\_marks)**  **Step 6: if student\_marks[n] == max\_marks: count+=1**  **Step 7: Repeat step 6 for each element in student\_marks**  **Step 8: print - "The number of students who have scored highest marks",count**  **Step 9: new\_list =[]**  **Step 10: if student\_marks[n] != max\_marks:**  **Add marks to new\_list**  **Step 11: Repeat step 10 for all elements in studeny\_marks**  **Step 12: max\_marks2 = max(new\_list)**  **Step 13: print("The second highest marks is",max\_marks2 )**  **Step 14: Read the value of fail\_marks**  **Step 15: pass\_list = [] #creating new list for pass students**  **Step 16: for each element n in student\_marks:**  **if n>fail\_marks:**  **pass\_list.append(n)**  **Step 17: print("The marks of students who have passed are",pass\_list)**  **Step 18: End** |
|  | **Program with comment:**  **student\_marks = []**  **n = 0**  **count = 0**  **print("Enter the marks of each student for a given subject(to stop enter a negative number) ")**  **#input value of marks**  **while (n>=0):**  **n = int(input())**  **if n>=0:**  **student\_marks.append(n)**  **print(student\_marks)**  **print("The highest marks scored in the subject is",max(student\_marks))**  **#counting number of maximum marks by comparing and incrementing value of count**  **max\_marks = max(student\_marks)**  **for n in student\_marks:**  **if n == max\_marks:**  **count+=1**  **print("The number of students who have scored highest marks",count)**  **#counting second highest marks**  **new\_list =[] #creating a new list**  **for n in student\_marks:**  **if n != max\_marks:**  **new\_list.append(n) #new list without highest marks**  **max\_marks2 = max(new\_list)**  **print("The second highest marks is",max\_marks2 )**  **fail\_marks = int(input("The fail marks is ")) #input failed marks**  **pass\_list = [] #creating new list for pass students**  **for n in student\_marks:**  **if n>fail\_marks:**  **pass\_list.append(n)**  **print("The marks of students who have passed are",pass\_list)** |
|  | **Output:** |
| Program 5 | Write a python program which accepts a sequence of comma-separated values from console and generate as a list and as a tuple. |
|  | **Algorithm:**  **Step1: Start**  **Step2: Input comma separated values in x**  **Step3: y=list(x.split(','))**  **Step4: Print y**  **Step5: z=tuple(x.split(','))**  **Step6: Print z**  **Step7: End** |
|  | **Program with comment**  **x=input("Enter the comma separated values ")**  **print(x)**  **#using split function to separate the input values**  **y=list(x.split(','))**  **print("The comma separated list is",y)**  **z=tuple(x.split(','))**  **print("The comma separated tuple is",z)** |
|  | **Output:** |