**PRACTICAL – 1 (1.1)**

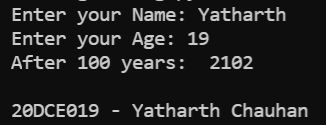
**AIM:**

|  |
| --- |
| **Create a program that asks the user to enter their name and their age. Printout a message addressed to them that tells them the year that they will turn 100 years old.** |

**PROGRAM:**

|  |
| --- |
| import datetime  name = input("Enter your Name: ")  age = int(input("Enter your Age: "))  date = datetime.datetime.now()  temp = 100 - age  update = temp + date.year  print(update)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – 1 (1.2)**

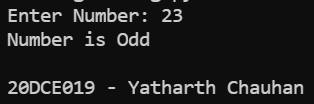
**AIM:**

|  |
| --- |
| **Write a program to ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user. Hint: how does an even / odd number react differently when divided by 2?** |

**PROGRAM:**

|  |
| --- |
| n = int(input("Enter Number: "))  if(n % 2 == 0):  print("Number is Even")  else:  print("Number is Odd")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – 2 (2.1)**

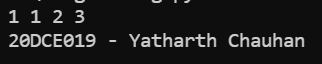
**AIM:**

|  |  |
| --- | --- |
| |  | | --- | | **Write a program as mentioned below:**  **Take a list, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], and write a program that print out all the elements of the list that are less than 5.** | |

**PROGRAM:**

|  |
| --- |
| list = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]  for i in range(len(list)):  if(list[i] < 5):    print(list[i], end=" ")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – 2 (2.2)**

**AIM:**

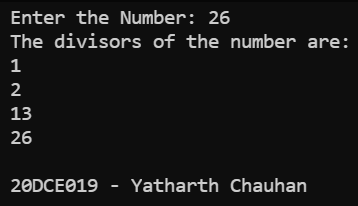
|  |
| --- |
| **Create a program that asks the user for a number and then prints out a list of all the divisors of that number. (If you don’t know what a divisor is, it is a number that divides evenly into another number. For example, 13 is a divisor of 26 because 26 / 13 has no remainder.)** |

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| n = int(input("Enter the Number: "))  print("The divisors of the number are:")  for i in range(1, n + 1):  if(n % i == 0):  print(i)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – 3 (3.1)**

**AIM:**

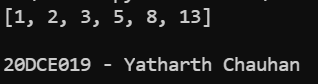
**Take two lists, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes.**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]  b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]  common = []  a = set(a)  b = set(b)  for i in a:  if i in b:  common.append(i)  print(common)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – (3.2)**

**AIM:**

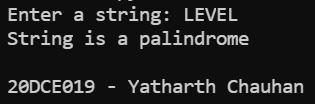
**Write a program by asking the user for a string and print out whether this string is a palindrome or not. (A palindrome is a string that reads the same forwards and backwards.)**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| string = input(("Enter a string: "))  if(string == string[::-1]):  print("String is a palindrome")  else:  print("String is not a palindrome")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – (4.1)**

**AIM:**

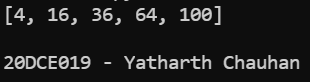
**Write one line of Python that takes list a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100] and makes a new list that has only the even elements of this list in it.**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| list = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]  ans = [i for i in list if i % 2 == 0]  print(ans)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – (4.2)**

**AIM:**

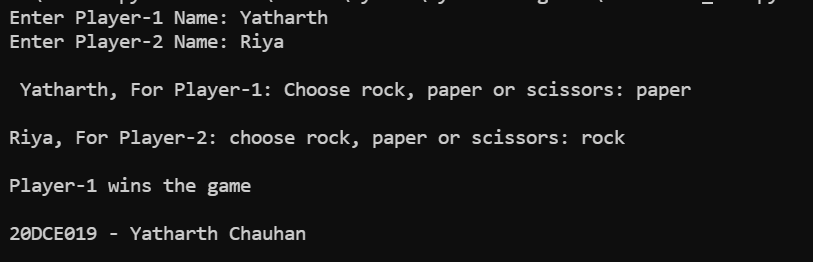
**Write a program to make a two-player Rock-Paper-Scissors game. (Hint: Ask for player plays (using input), compare them, print out a message of congratulations to the winner, and ask if the players want to start a new game) Rules: Rock beats scissors, Scissors beats paper, Paper beats rock**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| player1 = input("Enter Player-1 Name: ")  player2 = input("Enter Player-2 Name: ")  p1 = input(  "\n %s, For Player-1: Choose rock, paper or scissors: " % player1)  p2 = input(  "\n %s, For Player-2: choose rock, paper or scissors: s" % player2)  def compare(u1, u2):  if u1 == u2:  return("It's a tie")  elif u1 == 'rock':  if u2 == 'scissors':  return("\nPlayer-1 wins the game")  else:  return("\nPlayer-2 wins the game")  elif u1 == 'scissors':  if u2 == 'paper':  return("\nPlayer-1 win the game")  else:  return("\nPlayer-2 wins the game")  elif u1 == 'paper':  if u2 == 'rock':  return("\nPlayer-1 wins the game")  else:  return("\nPlayer-2 win the game")  else:  return("\nInvalid input")  print(compare(p1, p2))  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:**

**PRACTICAL – (5.1)**

**AIM:**

**Write a program to generate a random number between 1 and 9 (including 1 and 9). Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right. (Hint: remember to use the user input lessons from the very first practical)**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**

**PRACTICAL – (5.2)**

**AIM:**

**Take two lists, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes. Write this in one line of Python using at least one list comprehension**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**

**PRACTICAL - 2**

**AIM:**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**

**PRACTICAL - 2**

**AIM:**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**

**PRACTICAL - 2**

**AIM:**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**

**PRACTICAL - 2**

**AIM:**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**

**PRACTICAL - 2**

**AIM:**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**

**PRACTICAL - 2**

**AIM:**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
|  |

**OUTPUT:**

**CONCLUSION:**