**Practical File**

of

**Intelligent Systems Lab**

**(PCC-CS-601)**

submitted in partial fulfillment of the requirement for the award of degree of

**Bachelor of Technology (B.Tech)**

in

**Computer Engineering**

By

**Name: Dhruv Mishra**

**(Roll No.: 21001003036)**

Under the guidance of

**Dr. Mamta Kathuria**

**Assistant Professor**

****

**Department of Computer Engineering**

**J. C. BOSE UNIVERSITY OF SCIENCE & TECHNOLOGY, YMCA**

**SECTOR-6 FARIDABAD**

**HARYANA-121006.**

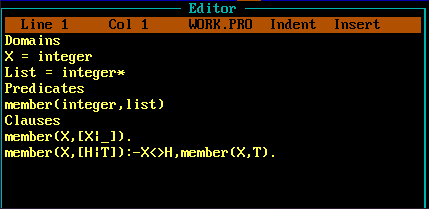
**INDEX:-**

**SNO NAME OF PROGRAM**

|  |  |
| --- | --- |
| **1.** | **Program to check whether an element is a member of a list or not.** |
| **2.** | **Program to check whether a list is a subset of another list or not.** |
| **3.** | **Program to count number of elements in a list.** |
| **4.** | **Program to find sum of all elements of a list.** |
| **5.** | **Program to implement Factorial of a number.** |
| **6.** | **Program to append an element to a list.** |
| **7.** | **Program to concatenate two lists.** |
| **8.** | **Program to delete an element from a list.** |
| **9.** | **Program to reverse a list.** |
| **10.** | **Program to delete all occurrences of an element from a list.** |
| **11.** | **Program to Replace an element by another element in a list.** |
| **12.** | **Program to Replace all occurrences of an element by another element in a list.** |
| **13.** | **Program to find union of two lists.** |
| **14.** | **Program to find intersection of two lists.** |
| **15.** | **Program to generate Fibonacci Series.** |
| **16.** | **Program to find last element of a list.** |
| **17.** | **Program to check if two lists are equal or not if in same order.** |
| **18.** | **Program to check if two lists are equal or not if in different order.** |
| **19.** | **Program to implement Quicksort.** |
| **20.** | **Program to implement Mergesort.** |
| **21.** | **Program to implement Bubblesort.** |
| **22.** | **Program to implement Selectionsort.** |
| **23.** | **Program to implement Insertionsort.** |
| **24.** | **Program to implement BFS.** |
| **25.** | **Program to implement DFS.** |
| **26.** | **Program to implement Family Relation Tree.** |

**Practical 1: Program to check whether an element is a member of a list or not.**

**Code:**

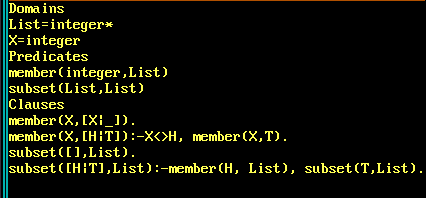
****

**Output:**

****

**Practical 2: Program to check whether a list is a subset of another list or not.**

**Code:**

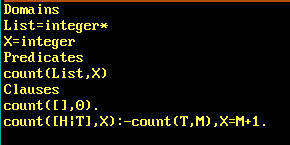
****

**Output:**

****

**Practical 3: Program to count number of elements in a list.**

**Code:**

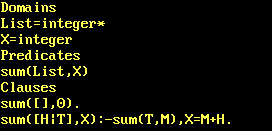
****

**Output:**

****

**Practical 4: Program to find sum of all elements of a list.**

**Code:**

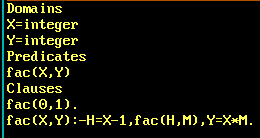
****

**Output:**

****

**Practical 5: Program to implement Factorial of a number.**

**Code:**

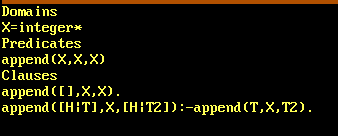
****

**Output:**

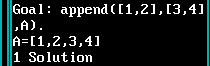
****

**Practical 6: Program to append an element to a list.**

**Code:**

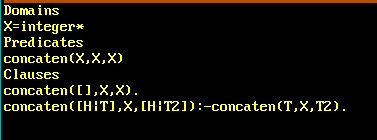
****

**Output:**

****

**Practical 7 : Program to concatenate two lists.**

**Code:**

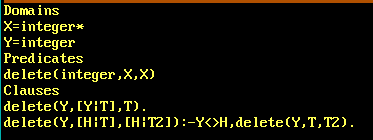
****

**Output:**

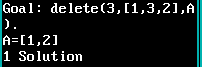
****

**Practical 8: Program to delete an element from a list.**

**Code:**

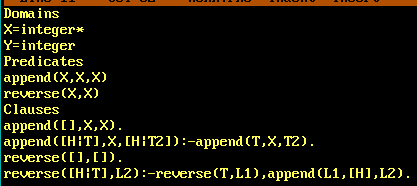
****

**Output:**

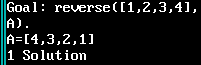
****

**Practical 9: Program to reverse a list.**

**Code:**

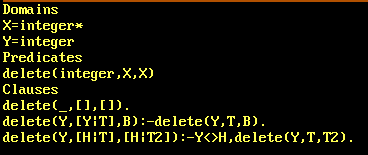
****

**Output:**

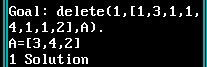
****

**Practical 10: Program to delete all occurrences of an element from a list.**

**Code:**

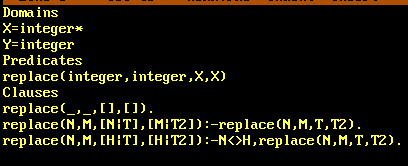
****

**Output:**

****

**Practical 11: Program to Replace an element by another element in a list.**

**Code:**

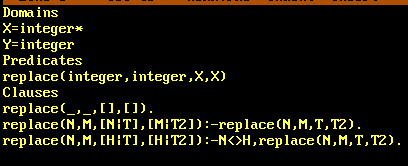
****

**Output:**

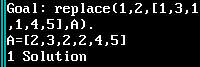
****

**Practical 12: Program to Replace all occurrences of an element by another element in a list.**

**Code:**

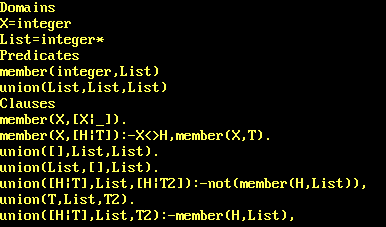
****

**Output:**

****

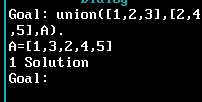
**Practical 13: Program to find union of two lists.**

**Code:**

****

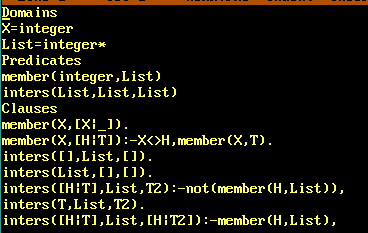
****

**Output:**

****

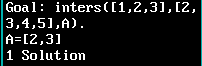
**Practical 14: Program to find intersection of two lists.**

**Code:**

****

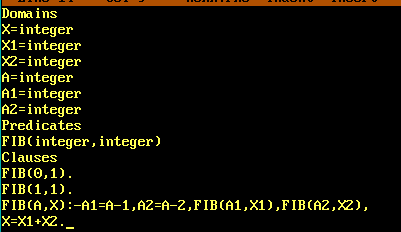
****

**Output:**

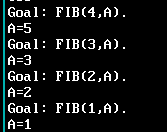
****

**Practical 15: Program to generate Fibonacci Series.**

**Code:**

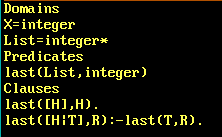
****

**Output:**

****

**Practical 16: Program to find last element of a list.**

**Code:**

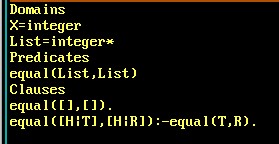
****

**Output:**

****

**Practical 17: Program to check if two lists are equal or not if in same order.**

**Code:**

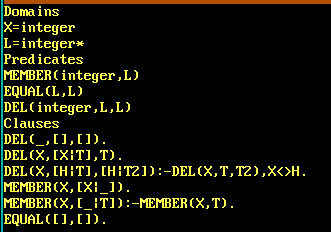


**Output:**

****

**Practical 18: Program to check if two lists are equal or not if in different order.**

**Code:**

****

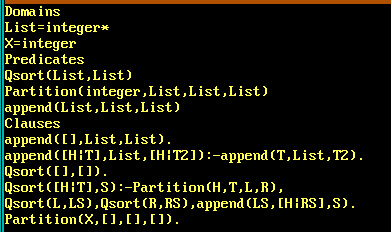
****

**Output:**

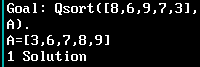
****

**Practical 19: Program to implement Quicksort.**

**Code:**

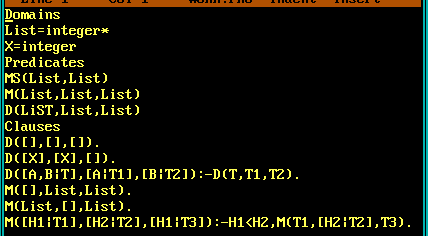
** **

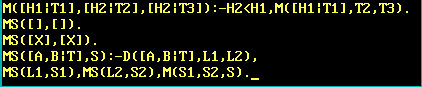
**Output:**

****

**Practical 20: Program to implement Mergesort.**

**Code:**

****

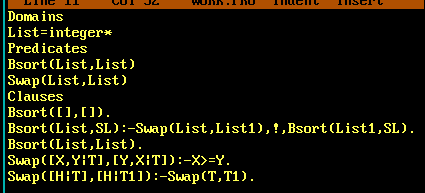
****

**Output:**

****

**Practical 21: Program to implement Bubblesort.**

**Code:**

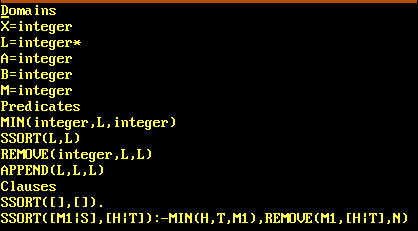
****

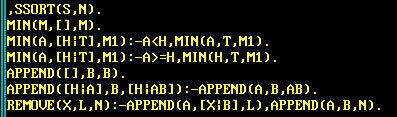
**Output:**

****

**Practical 22: Program to implement Selectionsort.**

**Code:**

****

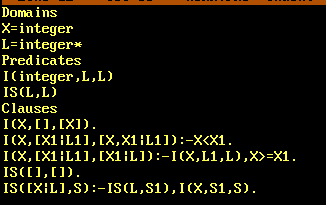
****

**Output:**

****

**Practical 23: Program to implement Insertionsort.**

**Code:**

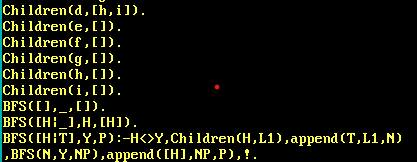
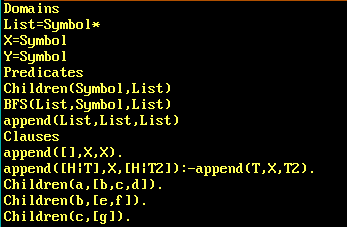
****

**Output:**

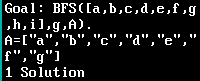
****

**Practical 24: Program to implement BFS.**

**Code:**

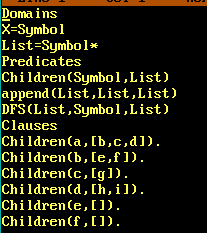
****

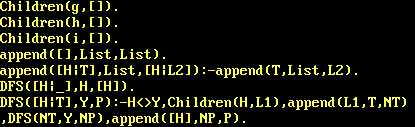
**Output:**

****

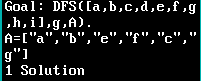
**Practical 25: Program to implement DFS.**

**Code:**

****

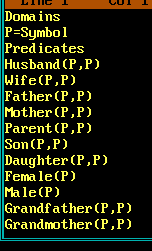
****

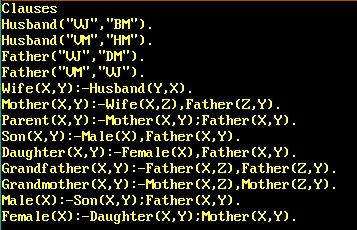
**Output:**

****

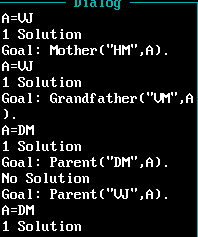
**Practical 26: Program to implement Family Relation Tree.**

**Code:**

****

****

**Output:**

****