**VNIT Nagpur**

**CPL Lab**

**Assignment 4**

**Course coordinator: U.A. Deshopande**

**Lab incharge: Mansi A. Radke, Swati Jaiswal**

**Maximum Marks: 10 (Could be scaled later)**

**Hard Deadline : Online submission 31st May, midnight**

An abstract data structure is a collection, or aggregate, of data. The data may be booleans, numbers, characters, strings or other data structures.

In computer science, a **set** is an abstract data type that can store unique values, without any particular order. Sets are of two types: Static and dynamic.

Typical operations that may be provided by a static set structure *S* are:

* is\_element\_of(*x*,*S*): checks whether the value *x* is in the set *S*.
* is\_empty(*S*): checks whether the set *S* is empty.
* size(*S*) or cardinality(*S*): returns the number of elements in *S*.
* enumerate(*S*): returns a list containing the elements of *S* in some arbitrary order.
* build(*x*1,*x*2,…,*xn*,): creates a set structure with values *x*1,*x*2,…,*xn*.

In addition to all above operations, dynamic set structures typically add:

* create(): creates a new, initially empty set structure.
* add(*S*,*x*): adds the element *x* to *S*, if it is not present already.
* remove(*S*, *x*): removes the element *x* from *S*, if it is present.

Following operations should be supported on two sets.

* union(*S*,*T*): returns the [union](https://en.wikipedia.org/wiki/Union_(set_theory)) of sets *S* and *T*.
* intersection(*S*,*T*): returns the [intersection](https://en.wikipedia.org/wiki/Intersection_(set_theory)) of sets *S* and *T*.
* difference(*S*,*T*): returns the [difference](https://en.wikipedia.org/wiki/Difference_(set_theory)) of sets *S* and *T*.
* subset(*S*,*T*): a predicate that tests whether the set *S* is a [subset](https://en.wikipedia.org/wiki/Subset) of set *T*.4

**Implement a set data structure using a hash table. For this assignment, you can consider a set of strings.**

**Programming language to be used: Any C, C++, Python, Java, or any other.**