a2

Title :- To create ADT that implement the set concept

problem statement :- To create ADT that implement the 'set' concept. 1. Add (new element), place a value into set

2. Remove (element)
3. contains (element) Return value true if element is in collection
4. size() Return number of values in collection iterator
5. Intersection of two sets
6. union of two sets.
7. Difference between two sets
8. subset

objectives :- To understand the concept of creation of ADT that implement the set concepts and its operation on set.

Theory :- Sets are a type of abstract data type that allows you to store a list of non - repeated values their name derives from the mathematical concept of finite sets.
Unlike an array, sets are unordered and unindexed. you can think about sets as a room full of people you know They can move around the room changing order without altering the set of people in that room, and there are no duplicate people (unless you know someone who has cloned themselves)
These are the two properties of a set. The data is unordered and it is not duplicated.

sets have the most impact in mathematical set theory. These the ones are used in many kinds of proofs structures and abstract algebra creating relations from diffrent sets and codomains are also an important applications of sets.

In computer science, set theory is useful if you need to collect data and do not care about their multiplicity or their order. As we've seen on this page, hash tables and sets are very related. In databases, espacially for relational databases, sets are very useful There ~~are~~ e many commands that finds unions intersections and differences of diffrent tables and set of data.

The set has four basic operations

Function name.	Provided Functionality.
insert()	Adds it to the set
remove()	Removes : from the set
size()	Returns the size of the set
contains()	Returns whether or not the set contains.

sometimes operations are implemented that allow interactions between the two sets

Function Name.	Provided functionality.
union (S.T)	Returns the union of sets S & T
intersection (S-T)	Returns the intersection of sets & S T
diffrence (S-T)	Returns the diffrence of set S & T
subset (S.T)	Returns the whether or not sets is a subset of set T.

Function Name.	Provided functionality
union (S,T)	Returns the union of sets S & T
intersection (S,T)	Returns the intersection of sets S & T
difference (S.T)	Returns the difference of set S & T
Subset (S,T)	Returns the whether or not sets is a sebset of set T

Software requirement :- g++ /gcc compiler 64 bit
Fedora , eclipse IDE

Algorithm :-

```
1. Start
2. Define a set class with a private data member
3. Define a constructor to initialize the array to fixed.
   size and Set the size variable to zero.
4. Define the add() method . to add a new element to the set
5. Define. the remove () method
6. Define the contains () method - to check if the element
   exists or not
7. Define the size () method - to return no.of element
8. Define the iterator () method - to return an iterator
9. Define the intersection (set 1 and set2) method to return
   a new set that contains the elements that are in both
   Set 1 and set 2
10. Define. the union () method to return a new set
    that contains all set elements from set 1 & set 2
11. Define the difference (set 1, set 2) method to return
    a new set that contains the element that are in
    set 1 but not in set 2.
12. Define the subset (set 1, set 2) method to check if set 1
    is a subset of set 2
13. stop.
```

- we can ~~se~~ use. the function required by calling it whenever required.

~~collist~~

conclusion :- In conclusion, the problem statement calls for th
creation of a ADT that implement the "set" concept. A
set is a collection of distinct elements with no parti
order