Avaignment - 9					
Till the state of					
Title - Bet Operations					
born was a distribution of the same and the					
Peroblem statement -					
To create ADT that implements SET concepts					
(i) Add new element					
(ii) Removes element					
(iii) Returns itrue if element is present.					
(iv) Return size of act.					
(V) Interaction					
(vi) Union					
(vii) Difference					
(viii) dubret					
Objective - To implement SET ADT and learn.					
det operations like interaction union.					
différence, auleret.					
H. Sulmy to E. B. A. A. tombers . H.					
Outcomes - we will have a program ready to					
fresjorm SET operations for various					
applications.					
of a day of the stands -					
H/W and S/W Requirements -					
Fedora 20					
Dell Optifelen 3020MT					
Keyboard 1 ( ) works in interest					
Monitar Julia Laure Caralina					
Erlikae to be land					
Scanned with CamScanner					

Theory -

sets are a lyke of abatract data type that allows you to alore a list of non-refreated values. Unlike an array, alls are unordered and unindered.

The two frogerties of sets are that the data is unordered and it is not dufsticated.

Let theory is waiful if you need to rollect data and do not race about their multiplicity or their order. Operations -

1. Union (A, 8)

returns AUB

- 2. Interaction (A, B) returns ANB
- 3. Difference (A,B) returns A-B
- 4. Queaet (A, B) thecks whether A is a auteast of B and returns o or 1 accordingly.

Class Structure class Node ? unt data; Node \* nest; 110 12 mildi. feublic: Node () { data = 0; next = NULL; } Node (int a) { idata = a; neset = NULL; 3 friend class set;

· winter (with

W Trierauckien

(viv) Dulpremcc.

(iii) author

dans det & Node \* etant; fublic: set () { start = NULL; 3 vaid deleter ; void deleter (int a); void indert (int x); void diaplay ( ); ( ) ( ) ( ) ( ) ( ) ( ) ( ) bool find (int a); det interxection ( det a); relet unionset ( set a); det differenceaet ( det a); void aubact (); Fire this wildelphon fire Precudo rodes void inacet (int se) & if (: find(a)) { if ( atant = = NULL) 1 - atant = new Node (a); return; some a soor 3 mars a grown to how in the Node \* 1 = start; while ( &-> neset! = NULL) t= 1-> neset; I -> nest = new Node (20); elae & rout << " Element already foresent!" << end!;

```
2. void display () {
    rout « " Set elements - " cc end;
     Node + I = start;
     if ( t == NULL) [
        rout « " EMPTY " « endl;
                           alle in
      nohile (t!=NULL) {
        rout << t -> data << " ";
        it = 1 -> neset; 10 halo ) with
    rout exend;
3. void deleten ( int 21) {
      Node * t = atant;
      if (d-> data == 2){
         wtart = it -> neset;
         delete t;
         return;
         while ( & -> next ! = NULL) {
     if while ( t -> next -> data = = 2) {
            Node * temps = + -> neset;
            d-> next = itemp -> next;
            delete temp;
             rout « ca « is deleted!" « end!;
```

	and the second s				
4.	find (int 2) {				
	Node * t = atart;				
	while ( t! = NULL) [ 11 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4				
	€ if ( t -> data = = >e)				
	relien true;				
	d= l-> nesit;				
	3 trusts in the				
	return falae;				
	3				
5.	void dize () {				
	Node * J = start;				
	int count = 0;				
	while (+) {				
	count ++;				
	d=d-> neset;				
	; is the				
	nout « " dize is - « « count « end!;				
	3				
	if i s. thind (t > date)				
6.	Let interaction ( Let a) {				
6.	Bet b;				
	node * t = start;				
	while (t's = NULL) {				
	if (a. find (t-> data))				
	of Ca. find ( )				
- 1	b: invert (t-> data);				
0.70					
1000	y towns to server				
-	d=t-> meret 3				
both	3 (Catt) <- 12 1 brinds 37 1.6				
	return b;				

```
7. det unionaet (det a) {
      Node * it = atant;
       det c;
        while (t!= NULL) {
           c. indext (t-> data);
           J= t-> nesit j
         3
        t = a. extant;
        while ( t! = NULL) {
           rif ( : c. find (t-> data))
               c. inacrt (t-> data);
         t= t-> nest;
  Let difference ( Let a) ?
8.
    set d;
      Node * t = atart;
      while (t) f
        if (! a. find (t -> data))
           ~ indert (t-> data);
       I= t-> neset;
               and the first of the
   tet word dubact ( idet aut) {
        Node * t = aut. atart;
        while ( f! = NULL) {
              if ( find (t-> data >)
                   0= t -> nest;
```

Scanned with CamScanner

			E. make and pro-		
	elae E		Mart 1 (100 42 %)		
	rout '<" Not a sulexet!" Zeendl;				
	return; 3				
	3 rout « " Is a aulonet! " « cendl;				
	23				
	Teat cases-				
	Naal	Extrected 01P	Actual OIP		
1.	Inacrt - 1	A = IX	<u>aduness</u>		
2.	Imacent - 2	A = 11 - 2 X	dunes		
3.	Indert - 3	[[-\2]-\3]X	Success		
<b>4</b> .	Delete - 2	11-3X	duccess		
5.	Inaert - 1	"Element already fresent!"	rduness		
		fresent!"	#L - 27		
	The State of the S				
6.	Union - AUB	C= {1, 3, 4, 53	duness		
	B- £1, 4,53				
	-	1. 4 Mar			
7.	Interaction -	B = {1}	Luces		
	B- 91,4,03				
		and the second s	d		
8.	Difference	D = 233	Success		
- 2					
g .	duliatet	It is not a	duces		
	B = 2, 3, 8, 73	Kulealt!	Control of the Contro		
	1		-		
	Conclusion -	expully implemented	SET ADT.		
	We our	exaline intermented			