```
Program-09
struct abde
unt data, degree;
Node * child, * Sibling, * parent;
Node* new Node (int Izey)
   Node * temp = new Node;
temp -> data = lzey;
   temp -> degree = 0;
temp -> child = temp -> parent = t;
return temp;
  Node* verge Binovial Trees (Node * b1, Node * b2)
     if (b1-)data) b2-)data)
         Swap(bl, b2);
        be-> parent - b!;
be-> sibling = bi->child;
bl-> child = be;
          61 -> degree++;
           essed setuan pl;
                                                      248146
      SURYA Gold
```

List (Node*) union Binovial Heap (list < Node*)	15.1
list < Node*	1
	595
Histanode* > yes;	
H wist (Node si iterator at = di orgini)	
Hist ( Node*): iterator at = de. Degine)	
dist < Node* : iterator at = ll. begin();  list < Node* : iterator at = ll. begin();  While (it!-diend() & 60t!=ll. end())	
111 5	
if (⁢) -> degree (= (*ot)->degree)	
3 COULD - Children C - Copper and act	
	-
-new.push-barlz (*it);	-
it++;	
3	
93.69	
<u> </u>	
-New onep parts (* of).	
-new.posh-back(*ot);	
5	
7	
5	
While (it 1= diends)	
<u> </u>	
- New. push-back (*;+).	
- New. punh-back (*it);	
}	
While (ot 1 = de. end)	
S COLOR COLO	
- NEM. bory-paris (*of);	
Ot+t;	
Jeturn - New;	
3	
Scanned with CamScanner	

```
List (Node*) adjust (list (Node*) - heap
        heap. Size () == 2
           = -heap.end();
                 heap. ende
              (*it1)-> degree ( (*it2)-> degree)
       it1++;
             31= heap. end()
```

else if (it31=heap.end() be	
che if (it31 = heap.eval) = (*it2) -> degree = (*it2) -> degree	0.
(*iti) -) degree == (*ite) -> degre (*iti) -> degree == (*ite) -> degre	e de
(* (tt)) sarges	1
itari.	
$\frac{1}{\sqrt{13}+1}$	
2	
close if ((*iti) > degree == (*ite) -> degree	
S S S S S S S S S S S S S S S S S S S	
Node * temp:	
#itt = verge Binovial Trees (*it1, *ite);	
ite - heap. exame(ite).	
if (it 3 1 = heap. end())	
it 3++:	
3	
Jeturn-heap;	
1:01/0101 * 1	
list (Node*) inscrtairee in Heap (dist (Node*)	heap
Node # free	
1:1+(0) 1 %.	
TELLO CIVAL LO CULTO	
temb - Minness (tage).	
temp = UnionBionomialHeap (-heap, temp);	
Just (temp).	

- 1. I Made to remove Min Francisce Return RHOOD
Ilist (Node *) reveoue Min Francisce Return BHeap (Node * tree
I I I ( a) A to a long:
list (Node* > heap; Node * temp = tree > child; Node * do;
Node * temp = + see -> chills,
Nocle * do;
While(temp)
\$
lo = feurp;
Laws a tema => siblim;
TOTAL CILITING COLUMN
10 -> SIBING - 10 OCC;
temp = temp -> sibling; lo -> sibling = NULL; heap.push-front (lo);
•
Deturn heap;
1 Clint ( ) adot ) lacad that lacus
distande *> insest (distande*) head, but lay
Node * teurp = new Node (12ey);
vode * teurp = hebroactings; return innertairee interp(head, teurp);
7
The same of the sa
Node* getrin (list (Node*) - heap)
Node* getrin (list Mode") = heap
Elist ( Mode* > :: isterator ist = - heap. begin();
list (Moder)
Node * temp - * it;
Node * frap = heap.end()
S
if ((*it)-)data (temp-)data)
temp xit;
?
eged sied Date Programme Control of the Control of
18 eturn temp;

	Clist (Node* > he	100
	dist a Node * Sextractrein Colist a Node * She	The same of the sa
	2	
	list (Node*) new heap, lo;	
4	Node * temp;	
1	terro = aetMin(-heap);	
H	teurp = getMin(-heap); dist (Node*): iterator it;	
	Lihile (it!= -heap.end())	
4	Libile ( at 1= -heap.end())	
+	if (* it 1=temp)	_
H		
	new-heap.push-back (*it);	
1	3	
+	do = revioue Minfrontisce Return 13 Heap (temp);	
-	ner heap = UnionBionovial Heap (New-heap)	(0).
1	New- Nead = adjust (New-Nead),	
+	return heu-heap;	
1	void printisee (Node *h)	
	\$	
	While (h)	
4	Cout < ch -> data cc";	
+	printigee (h-) child);	
	h-h-sibling;	
	3	
-	3	
1		

Void print Heap ( List ( Node\*). heap)

\* list ( Node\*): iterator it:

it = -heap.begin();

Lhile ( eit ! = -heap.end())

\* print isee (\*it);

at tt;

}