```
a Create a Doubly Linked List.
# anclude (Stdio. h)
# mclude (stillib.h)
Struct Nocle (
    ent data;
   Struct Node * next;
   Struct Node prev;
struct Node create Nock (Int data) {
   Struct Necle * new Node = (struct Necle *) malloc (street (struct Nede)):
   newNeode -> data = data;
   new Nede -> next = NVII;
   new vode -> prer = NULL
   return newNode;
void mscotNode (struct Nede* head, Ent data) f
     Struct Node * neurode = creale Node (data);
    newNode > next = * head;
     if ( * head! = NULL) of
          (* head) -> prev = newNode;
      *head = new Node;
void printle b (struct Node+ node) (
      while (node ! = NULL) f
          printf("/dala"; node-> data);
          node = node > next;
      percent conscessions
                                             output:
int main () 1
                                            Doubly Linked List: 500 200
    Struct Nools head = NULL;
    gnust Node (4 head, 100);
    onut Nede (4 head, 200);
   grout Node (4 head, 500);
   prantf ("Double Linked Lit's");
   Drintest ( head);
```

```
Insertion & Deletion
# anclude (stdio. h>
# anclude (stdlib.h)
struct Node of
     ant data;
    Struct Node prev;
    Struct Node * next;
void must node (struct node + exist node, int new Data) of
   Struct Node * new Nocle = (struct Node *) malloc (streeof (struct Node));
   new Nede -> data = new Data;
   newNode -> prev = exist Node -> prev;
  new Nocle -> noxt = exist Nocle;
   exist Node -> prev -> next= newAbole;
   crust Node > pre v - new Node;
void del Necle (struct Node * head, Fort value) of
   Struct Node * current = * head;
  while (current ! = NULL 34 current > data! = value) {
     current = current -> ruxt;
   if (current -> prev ! = NULL) f
      cornered
      prontf (value);
      ochurn;
    if convent -> prev! = NULL) {
      ament -> prev -> next = current -> next;
   Juse of
       * head = current -> next;
    of (current = next! = NULL) {
       current -> next -> prev = current -> prev;
   free Courrent;
```

```
void printlist (struct node + node) {
    while (node ! = NULL) {
       pront ("/d", node -> data);
       node = nocle -> next;
     brank ("10");
Ent main() f
    Struct Nede + head = NULL;
    ont seg Nocles;
    prosecution are not of parts may
    Start for
       Struct Nede + new Node = (Struct Node +) malloc (stroct Node)).
    for (mt i=1; i <= 10; i++) {
       newNede > data = i;
       newNocle > prev = NULL;
       newNode -> next = nigel;
       { (head! = NVIL) }
                             seed - Elife - Mich ( Test - Life, dala),
    head > prev = new Node;
       head = new Node;
   printf ("Original List:");
                                    output
   prontlist ( head);
   Insert Mode (head -> next, 11);
                                    Organal 181:432
   prontf ("After insciting: ").
                                   After Insortion 4 11 3-2
   printlist (head);
                                   Offer Deletrus:
   del Node (4 head, 5);
                                    oregenal List: 5 4 3 2 1
   printf ("After Deliting: ");
                                    After Inscring: 5 114 3 2
   pront List (head);
                                   After Deleting: 11 4 3 2
   return o
```

```
"C:\Users\User\Desktop\Data Structures\Stackss.exe"

10 20 30 40

10 30 40

Process returned 0 (0x0) execution time : 0.009 s

Press any key to continue.
```