Date: 25/11/2020 Name: KusumomoR USN: 1BM19CSOFT Lab Program - 5 1) WAP to implement singly Linked List with following operations a) Create a linked list b) Insertion of a mode at first position, at any position and at end of list c) Display the contents of the linked list # include < stdie . h > # include < come . h7 struct node int info; struct node *link; typedet struct node *NODE; NODE getnode() x = (NODE) malloc (size of (struct node)); if (x == NULL) Printf ("mem full \n");
exit (0); neturn 2; roid free mode (NODE x) 2 pre (x); NODE insert-front (NODE first, int item) ? NODE temp; temp=getnode(); temp - info = item; temp - link = NULL; if (first == NULL) return temp;

```
first = stemp -> link = first;
     first = teng;
return first;
NODE insert-rear (NODE first, int item)
     NODE temp, cur;
    temp = getnode();
     temp - info = item;
     temp -> link = NULL;
     if (first == NULL)
      return temp;
    cur = first;
    while (cur -> link! = NULL)
       cur = cur - link;
    cur -> link = temp;
    return first;
NODE insert-pos (int item, int pos, NODE first)
     NODE temp;
     NODE previour;
     int count;
     temp= getrode ();
     temp - info = item;
      temp -> link = NULL;
     4 (first == NULL && pos==1)
        return temp;
     of (first == NULL)
      2 printf ("invalid pos/n");
        return first;
    if (pos == 1)
     à temp → link= first;
     return temp;
```

```
count =1;
  prev = NULL;
 while ( cur!= NULL &l count != pos)
  icu = first;
     prev = cue;
      cur = cur -> link;
   3 count ++;
if (count == pos)
3 prev - link = temp;
  temp - link = cur;
  return first;
printf (" IPIn");
retur first;
roid display (NODE first)
     NODE temps
      if (first == NULL)
        printf ("List is empty, cannot display items \n");
      else prints ("Contents of the list: m");
      for (temp = first; temp != NULL; temp = temp -> link)
       a printf (" % d \n", temp -> info);
     roid main ()
       int item, choice, pos;
NODE first = NULL;
       for (; ;)
         printf("\n1: Insert-front \n 2: Insert-lear \n3:
            Insert-pos \n 4° Display-list \n 5 : Frut \n");
       printf (" Enter the choice \n");
       scarf ( " %d", & item);
```

switch (choice) ease 1: printf ("Enter the item at front-end in");

scanf (" %d", & item);

first = insert-front (first, item);

break: case 2: printf (" Enter the item at rear-end (n");
scarf (" %d", & item);
first = insert-rear (first, item); case 3: printf ("Enter the position and item: \n"); searf (" %d", & pos); scanf ("%d", & item); first = insert-pos (item, pos, first); break; case 4: display (first); break; case 5: exit (0); défault: prints (" Invalid choice (n"); Lab Program - 6 20 WAP to implement Singly Linked List with following operations operations b) Deletión of first element, specified element and last element in the list. c) Display the contents of the linked list # include < stdie . h > # include 4 conio h7 struct mode struct nade *link; int info;

```
typedef struct node *NODE;
NODE getnode()
  X = (NODE) malloc ( size of (struct node));
  NODE x:
  4 (x == NULL)
     print (" men full \n");
      exit (0);
  return x;
roid freenode (NODE 2)
  free (x);
NODE insert front (NODE first, Int item)
  NODE temps
   temp = getnode ();
   temp - info = item;
    temp -> link = NULL;
    if (first == NULL)
       return tem;
     temp -> link = first;
    first = temp;
    return first;
 NODE delete-front (NODE first)
     NODE temp;
     if (first == NULL)
     2 printf ("List is empty rannot delete\n");
return first;
     temp = first;
     temp = temp - link;
     printf (" Item deleted at front-end is = %d\n", first-inf)
     free (first);
     return temp;
```

```
NODE insert-rear (NODE first, int item)
    NODE temp, cur;
      temp = getrode ();
      temp - info - item;
       temp - link = NVLL;
       if (first == NULL)
           return temp;
       cus = first;
      while ( cur -> link != NULL)
          rus = cur - link;
      cur → link = temp;
return first;
NODE delete-rear (NODE first)
   NODE cur, prev;
    of (foret == NULL)
    I printf ("List is empty cannot delete\n");
      return first;
    If ( first -> link == NULL)
      printf (" Item deleted is " ) d'm", first -> info);
      free (first);
     return NULL;
  prev = NULL;
   cu = first;
  while ( cus -> link != NULL)
    1 prev = cur;
    j'aur=aur→link;
   printf (" Item deleted at rear-end is %d", cur - info);
    free ( cue);
    prev -> link= NULL;
    return first;
```

```
NODE delete-pos(int pos, NODE first)
I NON prev, cur;
    int court;
    if (fist == NULL 11 posz=0)
     2 print (" Imalia position \n");
         return NULL;
    if (pos == 1)
       cus = first;
        first = first - link;
        print (" Item deleted is 1.d", wer > info);
        freewde (au);
     return first;
    prev = NULL;
   cur = first;
   count = 1;
   while (cur != NULL)
      if ( count == pos)
       2 break;
      prev= cur;
      cur = cur -) link;
    2 rount ++;
   if ( count != pos)
   printf (" Invalid position \n");
     return first;
   prer > link = cus -> link;
    print (" Item deleted is "/d", cur - info);
    freenode (cur);
    return first;
```

rold display (NODE first) NODE temp; 4 (first == NULL) printf (" List empty cannot display items (n'); else printy (" Contents of the list in"); for (temp first; temp!=NULL; temp= temp-> link) ? printf (" -/-d \n", temp -> info); roid main() int item, choice, pos; NODE first = NULL; for (;;) 2 printf ("\n1. Insert-front 2 : Delete-front \n 3: Insert-real In 4: Delete - real In 5: Delete - pos In 6: Display - list In 7: Exit (n); printy (" Enter the choice\n"); scarf ("%d", & choice); suitch (choice) 2 case 1: printf("Enter the item at front-end (n"); seary (" olad", & item); first = insert-front (first, item); case 2: first = delete-front (first); break; case 3° printf ("Enter the "item at rear-end \n"); scanf ("%d", & iten); first = insert-rear (first, item); break; case 4: first = delete-rear (first); rase 5: printf ("Enter the position:)n"); scarf (" "/.d", & pos); first = delete-post pos, first); break;

case 6: display (first); break; default: print ("Imalid choice In"); LAB PROGRAM-7 WAP to implement single Link list with following operations a) Sort the linked list 6) Reverse the linked list c) Concateration of two linked lists. # include 2 stdis. h> # include < stdlib . h 7 struct node int info; struct node *link; typedet struct node * NODE; NODE getrade()

int info;

struct node *link;

};

typedef struct node *NODE;

NODE getrode()

2 NODE x;

x = (NODE) malloc (size of (struct mode));

x = (NODE) malloc (size of (struct mode));

if (x == NULL)

2 printf ("mem full \n");

exit (0);

return x;

3

void freenode (NODE x)

2 free (z);

```
NODE insert-front (NODE first, int item)
   NODE temp;
   temp= getnode();
    temp - info = item;
    temp - link = NULL;
    4 (first == NULL)
    return temp;
temp > link = first;
     first = temp;
     return first;
NODE delete-front (NODE first)
   NODE & temp;
     if (first == NULL)
     I printf ("List is empty carnot delete \n");
       return first;
    temp = first;
    prints ('Item deleted at front-end is = "hd \n', first-info);
     free (first);
     return temp;
 NODE insert-rear (NODE first, int item)
    NODE temp, cur;
     temp = getnode ();
     temp - info = item;
     temp - link = NULL;
     if [first == NULL)
       return temp;
    cur = first;
    while (cur -> link != NULL)
       cus = cus - link;
       cur slink = terp;
    return first;
```

```
NODE delete- reas (NODE first)
  NODE eus, prer;
  printf ("list is empty cannot delete \n");
return first;
   if (first -> link == NULL)
    E printf (" Iten deleted is "bd \n", first - info);
       free (first);
      return NULL;
   prev = NULL;
    cur = first;
  print (" Item deleted at rear-end is "hd", our - info);
   free (cur);
   prev -> link = NULL;
  return first;
NODE order-list (int item, NODE first)
   NODE temp, prev, cur,
    temp = getnode();
    temp - info = item;
    temp - link = NULL;
    if (first == NULL) return temp;
    if ( item < first - info)
      temp -) link = first;
      return temp;
   prev = NULL;
   cur = first;
   while (cur!=NULL && item>cur -info)
    2 prev= cui;
      cur = cur - link;
```

```
prev - link temp;
   temp-slink= air;
   return first;
wid display (NODE first)
i NODE temp;
    printf ("list empty cannot display items \n");
    if (first == NUCL)
     printy ("Contents of the list: m");
    for (temp= first; temp!= NULL; temp= temp -> link)
        printf ("% od \n", temp >info);
NODE concate (NODE first, NODE second)
    NODE cur;
     if (first == NULL)
     return second;
      if (second == NULL)
       return first;
     reus = first;
    while ( cur > link != NULL)
       rur = aur -) link;
       un- link = second;
     return first;
NODE reverse (NODE first)
1 NODE cur, temps
   ru = NULL;
   while ( first != NULL)
   2 temp=first;
      first = first -> link;
      temp >> link = cur;
     cu = temp;
     return cur;
```

```
noid main ()
  int item, choice, key, n, i;
 NODE first = NULL, a, b;
 printf ("In1: Insert-front In 2: Delete-front In 3: Insert-May)
4: Delete rear n 5: Order-list (n 6: Display-list (n 7: Concat)
8: Reverseln 9: Exit (n");
printf ("Enter the choice \n");
scarf (" Tod", & choice);
2 case 1: printf ("Enter the item at front-end \n");
           scarf (" %d", & iten);
            first = insert-front (first, "item);
           break;
 case 2: first = delete-front (first);
 case 3: prints ("Enter the "item at rear-end m");
            scarf ("%d", & iten);
            first = insert-rear (first, item);
             break;
 rase 4: firet = delete-rear (firet);
 case 5: printf ("Enter the item to be inserted in ordered.
                 list (n");
          scarf ("%d", & item);
           first = order-list (item; first);
           break;
case 6: display (first);
            break;
       7: printf ("Enter the no of nodes in 1 \n");
           searf ("%d", &n);
            a = NULL;
            for (i=0;i×n; i++)
```

```
printy (" Enter the item In");
      aconf ("hd", & item);
      a - insett near (a, item);
  prints ("Enter the no. of nodes in 2 \n");
    Seanf ( 4.d", 8n);
     b = NULL;
    for (i=o; i=n; i++)
     E prints (" Enter the item \n");
        sear ( %d", & item);
        b="insert-rear (b, item);
       a = concat (a,b);
        display (a);
        break;
    case 8 % first = reverse (first);
              display (first);
     case 9: exit (0);
     default : printf (" Invalid choice \n");
333
```