

OPERAZIONI CON LE LISTE



```
1  struct Node *addNode(int data){
2      struct Node * new = NULL;
3
4      if (head == NULL){
5          new = malloc(sizeof(struct Node));
6          if (new == NULL)
7              return NULL;
8          new->data = data;
9          head = new;
10         new->next = NULL;
11     } else{
12         new = malloc(sizeof(struct Node));
13         if (new == NULL)
14             return NULL;
15         new->data = data;
16         new->next = head;
17         head = new;
18     }
19
20     return new;
21 }
22
```



```
1  struct Node *insertNode(int pos, int data){
2      struct Node * current = head;
3
4      for (int i = 1; i < pos; i++)
5          current = current->next;
6
7      if (current == NULL)
8          return NULL;
9
10     struct Node * new = NULL;
11
12     new = malloc(sizeof(struct Node));
13     if (new == NULL)
14         return NULL;
15     new->data = data;
16     new->next = current->next;
17     current->next = new;
18
19     return new;
20 }
```



```
1  struct Node *removeNode(int data){
2      struct Node * current = head;
3      struct Node * prox = current->next;
4      if (current == NULL)
5          return NULL;
6
7      if (current->data == data) {
8          head = current->next;
9          current = head;
10         prox = current->next;
11     } else{
12         while (prox->data != data){
13             current = current->next;
14             prox = prox->next;
15         }
16         if (prox->data == data){
17             current->next = prox->next;
18             free(prox);
19         }
20     }
21 }
```



```
1  struct Node *sortList(){
2      struct Node * current = head;
3      struct Node * prox;
4      int temp;
5      int swapped;
6
7      do{
8          swapped = 0;
9          while (current->next != NULL){
10             prox = current->next;
11             if (prox->data < current->data){
12                 temp = current->data;
13                 current->data = prox->data;
14                 prox->data = temp;
15                 swapped = 1;
16             }
17             current = current->next;
18         }
19     } while (swapped);
20 }
```



```
1
2  struct Node * inserimentoOrdinato(int data){
3      struct Node *prev = NULL;
4      struct Node *curr = head;
5      struct Node *new = NULL;
6
7      new = malloc(sizeof(struct Node));
8      new->data = data;
9
10     while(curr != NULL && curr->data <= data){
11         prev = curr;
12         curr = curr->next;
13     }
14
15     if(prev == NULL){
16         new->next = head;
17         head = new;
18     }
19     else{
20         new->next = curr;
21         prev->next = new;
22     }
23
24     return head;
25 }
```