SLL OPERATIONS

```
void push( struct Node *head, int new-data) {
    struct Node * ~~ptr~~ new-node = (struct Node *) malloc(sizeof (struct Node));
    new-node → data = new-data;
    new-node → next = NULL;
    if (head == NULL)
        head = new-node;
    else {
        new-node → next = head;
        head = new-node;
    }
}

void append( struct Node *head, int new-data) {
    struct Node *new-node = (struct Node *) malloc(sizeof (struct Node));
    struct Node * last = head;
    new-node → data = new-data;
    new-node → next = NULL;
    if (head == NULL) {
        head = new-node;
        return;
    }
    while (last → next != NULL)
        last = last → next;
        last → next = new-node;
}
void pop( struct Node * head) {
    struct Node * ptr = head;
    if (head == NULL)
        printf("Empty list");
    else {
        head = ptr → next;
        ptr → next = NULL;    free (ptr); } }
```

```
void reverse(struct Node *head){
    struct Node *next_ptr= NULL;
    struct Node *prev =NULL;
    struct Node * curr = head;
    while (curr != NULL) {
      next_ptr = curr →next;
      curr → next = prev;
      prev = curr;
      curr = next_ptr;
    }
    head = ~~ptr~~, prev;
}

struct Node * concat(struct Node *headref2, struct Node * headref3){
    struct Node *temp;
     if (headref2 == NULL)
        return ~~hed~~ headref3;
     else if (headref3 == NULL)
        return headref2;
     temp = headref2;
     while (temp → next != NULL) {
       temp = temp → next;
    }
     ptr → link = headref3;
     return headref2;
}
```