18-11-20

Singly Linked List

```
#include <stdio.h>
#include <stdlib.h>
 struct node
  {
   int info;
   struct node * link;
  };
 typedef struct node * NODE;
  NODE getnode()
  {
   NODE x;
   x = (NODE)malloc(sizeof(struct node));
   if(x == NULL)
    {
    printf("memory full\n");
     exit(0);
    }
   return x;
  }
  Void freenode(NODE x) {
    {
     free(x);
   }
 NODE insert-front(NODE first, int item)
   {
     NODE temp;
      temp = getnode();
      temp->info = item;
      temp->link = NULL;
```

```
if (first == NULL)
   return temp;
temp -> link = first;
  first = temp;
  return first;
}
NODE ~~dele~~ insertrear (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;
        }
void display (NODE first)
    {NODE temp;
    if (first == NULL)
printf("List is empty cannot delete items\n");
for (temp = first; temp != NULL; temp = temp -> link)
    {printf("%d\n", temp -> info);
    }
}
```

```
int main()
{
  int item, choice;
  NODE first = NULL;
for(;;)
{
printf("\n1. Insert front \n2. Insert rear\n3. Display
       \n4. exit \n");
printf("Enter the choice\n");
scanf("%d", &choice);
switch(choice)
{
case 1 : printf("Enter the item at front-end\n");
         scanf("%d", &item);
         first = insert_front(first, item);
         break;
case 2: printf("Enter the item at the rear-end\n
         scanf("%d", &item);
         first = insert_rear(first, item);
         break;
  case 3: display(first);
          break;
  case 4:
  default: exit(0);
          break;
}}}
```