LAB-8

11-12-2020

Stack and Queue Implementation

using Linked list

① stack Implementation:

```
void push(int ele)
{
  struct node *newnode;
  newnode = (struct node *) malloc (sizeof (struct node));
  newnode -> data = ele;
  newnode -> next = NULL;
  if (top == NULL)
   top = newnode;
   else
   {
    newnode -> next = top;
    top = newnode;
   }
   printf ("\n Element %d was inserted. \n", ele);
}

~~int~~
int pop()
{
   int ele;
   struct node *temp;
   if (top == NULL)
   {
      printf ("\n Stack underflow, The stack is empty! \n");
      stack();
   }
   ele = top -> data;
   temp = top;
   if (top -> next == NULL)
    top = NULL;
   else
    top = top -> next;
    free (temp);
}   return ele;
```

```
void display()
{
    struct node *i;
    if (top == NULL)
    {
      printf("\n Stack is Empty!\n");
      stack();
    }
    printf("\n Stack contains : \n Top -> ");
    for(i=top; i!=NULL; i=i->next)
       printf("\t%d", i->data);
}
```

⑤ Queue Implementation.

```
void insert(int ele)
{
    struct node *newnode;
    newnode = (struct node *) malloc(sizeof(struct node));
    newnode->data = ele;
    newnode->next = NULL;
    if( rear == NULL)
     {  front = newnode;
        rear = newnode;
     }
    else
    { rear->next = newnode;
      rear = newnode;
    }
    printf("\n Element %d was inserted!.\n", ele);
}

int delete()
 {
    int ele;
    struct node *temp;
    if(front == NULL)
    {  printf("\n Queue Underflow!\n");
       queue();
    }
```

```
    ele = front -> data;
    temp = front;
    if (front == rear)
    {
        front = NULL;
        rear = NULL;
    }
    else
        front = front -> next;
    free (temp);
    return ele;
}

void display()
{
    struct node *i;
    if (front == NULL && rear == NULL)
    {
        printf ("\n Queue Empty!\n");
        queue();
    }
    printf ("\n Queue contains : ");
    for (i = front; i != NULL; i = i -> next)
        print ("%d ", i -> data);
}
```