**Problem Statement:** Write a menu driven C program to show the insertion and deletion operation of array at different positions.

**Program Code:**

#include <stdio.h>

int main() {

int arr[100], n, choice, pos, val, i;

printf("Enter size of array (max 100): ");

scanf("%d", &n);

if (n < 1 || n > 100) {

printf("Invalid size!\n");

return 1;

}

printf("Enter elements: ");

for (i = 0; i < n; i++) {

scanf("%d", &arr[i]);

}

while (1) {

printf("\n1. Insert\n2. Delete\n3. Display\n4. Exit\nEnter choice: ");

scanf("%d", &choice);

if (choice == 1) {

if (n == 100) {

printf("Array is full. Cannot insert.\n");

continue;

}

printf("Enter position (1 to %d): ", n + 1);

scanf("%d", &pos);

if (pos < 1 || pos > n + 1) {

printf("Invalid position!\n");

continue;

}

printf("Enter value: ");

scanf("%d", &val);

for (i = n; i >= pos; i--) {

arr[i] = arr[i - 1];

}

arr[pos - 1] = val;

n++;

} else if (choice == 2) {

if (n == 0) {

printf("Array is empty. Nothing to delete.\n");

continue;

}

printf("Enter position (1 to %d): ", n);

scanf("%d", &pos);

if (pos < 1 || pos > n) {

printf("Invalid position!\n");

continue;

}

for (i = pos - 1; i < n - 1; i++) {

arr[i] = arr[i + 1];

}

n--;

} else if (choice == 3) {

if (n == 0) {

printf("Array is empty.\n");

} else {

printf("Array elements: ");

for (i = 0; i < n; i++) {

printf("%d ", arr[i]);

}

printf("\n");

}

} else if (choice == 4) {

printf("Exiting program.\n");

break;

} else {

printf("Invalid choice! Try again.\n");

}

}

return 0;

}

**Problem Statement:** Write a menu driven C program to show the insertion and deletion operation of Stack using Array.

**Program Code:**

#include <stdio.h>

#define MAX 10

int main() {

int stack[MAX], top=-1, choice, val, i;

while(1) {

printf("\n1.Push\n2.Pop\n3.Display\n4.Exit\nEnter choice: ");

scanf("%d",&choice);

if(choice==1) {

if(top==MAX-1) printf("Stack Overflow\n");

else {

printf("Enter value: ");

scanf("%d",&val);

stack[++top]=val;

}

} else if(choice==2) {

if(top==-1) printf("Stack Underflow\n");

else printf("Popped: %d\n",stack[top--]);

} else if(choice==3) {

if(top==-1) printf("Stack is empty\n");

else {

for(i=top;i>=0;i--) printf("%d ",stack[i]);

printf("\n");

}

} else if(choice==4) {

break;

} else {

printf("Invalid choice\n");

}

}

return 0;

}

**Problem Statement:** Write a menu driven C program to show the insertion and deletion operation of Linear Queue using Array.

**Program Code:**

#include <stdio.h>

#define MAX 10

int main() {

int queue[MAX], front=-1, rear=-1, choice, val, i;

while(1) {

printf("\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\nEnter choice: ");

scanf("%d",&choice);

if(choice==1) {

if(rear==MAX-1) printf("Queue Overflow\n");

else {

printf("Enter value: ");

scanf("%d",&val);

if(front==-1) front=0;

queue[++rear]=val;

}

} else if(choice==2) {

if(front==-1 || front>rear) printf("Queue Underflow\n");

else printf("Dequeued: %d\n",queue[front++]);

} else if(choice==3) {

if(front==-1 || front>rear) printf("Queue is empty\n");

else {

for(i=front;i<=rear;i++) printf("%d ",queue[i]);

printf("\n");

}

} else if(choice==4) {

break;

} else {

printf("Invalid choice\n");

}

}

return 0;

}

**Problem Statement:** Write a menu driven C program to show the insertion and deletion operation of Circular Queue using Array.

**Program Code:**

#include <stdio.h>

#define MAX 5

int main() {

int cq[MAX], front=-1, rear=-1, choice, val, i;

while(1) {

printf("\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\nEnter choice: ");

scanf("%d",&choice);

if(choice==1) {

if((front==0 && rear==MAX-1) || (front==rear+1))

printf("Queue Overflow\n");

else {

printf("Enter value: ");

scanf("%d",&val);

if(front==-1) front=0;

rear=(rear+1)%MAX;

cq[rear]=val;

}

} else if(choice==2) {

if(front==-1)

printf("Queue Underflow\n");

else {

printf("Dequeued: %d\n",cq[front]);

if(front==rear) front=rear=-1;

else front=(front+1)%MAX;

}

} else if(choice==3) {

if(front==-1)

printf("Queue is empty\n");

else {

i=front;

while(1) {

printf("%d ",cq[i]);

if(i==rear) break;

i=(i+1)%MAX;

}

printf("\n");

}

} else if(choice==4) {

break;

} else {

printf("Invalid choice\n");

}

}

return 0;

}

**Problem Statement:** Write a menu driven C program to show the insertion operation of Single Linked List at beginning, end and at any position.

**Program Code:**

#include <stdio.h>

#include <stdlib.h>

struct Node {

int data;

struct Node\* next;

};

struct Node\* head = NULL;

void insertBeginning(int val) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

newNode->data = val;

newNode->next = head;

head = newNode;

}

void insertEnd(int val) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

struct Node\* temp = head;

newNode->data = val;

newNode->next = NULL;

if(head==NULL) head = newNode;

else {

while(temp->next!=NULL) temp=temp->next;

temp->next=newNode;

}

}

void insertAtPos(int val, int pos) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

struct Node\* temp = head;

int i;

newNode->data = val;

if(pos==1) {

newNode->next=head;

head=newNode;

} else {

for(i=1;i<pos-1 && temp!=NULL;i++) temp=temp->next;

if(temp==NULL) {

printf("Invalid position\n");

free(newNode);

} else {

newNode->next=temp->next;

temp->next=newNode;

}

}

}

void display() {

struct Node\* temp=head;

if(temp==NULL) printf("List is empty\n");

else {

while(temp!=NULL) {

printf("%d ",temp->data);

temp=temp->next;

}

printf("\n");

}

}

int main() {

int choice, val, pos;

while(1) {

printf("\n1.Insert at Beginning\n2.Insert at End\n3.Insert at Position\n4.Display\n5.Exit\nEnter choice: ");

scanf("%d",&choice);

if(choice==1) {

printf("Enter value: ");

scanf("%d",&val);

insertBeginning(val);

} else if(choice==2) {

printf("Enter value: ");

scanf("%d",&val);

insertEnd(val);

} else if(choice==3) {

printf("Enter value: ");

scanf("%d",&val);

printf("Enter position: ");

scanf("%d",&pos);

insertAtPos(val,pos);

} else if(choice==4) {

display();

} else if(choice==5) {

break;

} else {

printf("Invalid choice\n");

}

}

return 0;

}