**STACK USING ARRAY IMPLEMENTATION**

**Step 1 - Include all the header files which are used in the program and define a constant 'SIZE' with specific value.**

**Step 2 - Declare all the functions used in stack implementation.**

**Step 3 - Create a one dimensional array with fixed size (int stack[SIZE])**

**Step 4 - Define a integer variable 'top' and initialize with '-1'. (int top = -1)**

**Step 5 - In main method, display menu with list of operations and make suitable function calls to perform operation selected by the user on the stack.**

**PROGRAM**

**#include<stdio.h>**

**#define SIZE 10**

**int stack[SIZE], top = -1;**

**int push(int value)**

**{**

**if(top == SIZE-1)**

**printf("\nStack is Full!!! Insertion is not possible!!!");**

**else**

**{**

**top++;**

**stack[top] = value;**

**printf("\nInsertion success!!!");**

**}**

**}**

**int pop()**

**{**

**if(top == -1)**

**printf("\nStack is Empty!!! Deletion is not possible!!!");**

**else**

**{**

**printf("\nDeleted : %d", stack[top]);**

**top--;**

**}**

**}**

**int display()**

**{**

**if(top == -1)**

**printf("\nStack is Empty!!!");**

**else**

**{**

**int i;**

**printf("\nStack elements are:\n");**

**for(i=top; i>=0; i--)**

**printf("%d\n",stack[i]);**

**}**

**}**

**int main()**

**{**

**int value, choice;**

**while(1)**

**{**

**printf("\n\n\*\*\*\*\* MENU \*\*\*\*\*\n");**

**printf("1. Push\n2. Pop\n3. Display\n4. Exit");**

**printf("\nEnter your choice: ");**

**scanf("%d",&choice);**

**switch(choice)**

**{**

**case 1: printf("Enter the value to be insert: ");**

**scanf("%d",&value);**

**push(value);**

**break;**

**case 2: pop();**

**break;**

**case 3: display();**

**break;**

**case 4: exit(0);**

**default: printf("\nWrong selection!!! Try again!!!");**

**}**

**}**

**}**

**OUTPUT**

**\*\*\*\*\* MENU \*\*\*\*\***

**1. Push**

**2. Pop**

**3. Display**

**4. Exit**

**Enter your choice: 1**

**Enter the value to be insert: 56**

**Insertion success!!!**

**Enter your choice: 1**

**Enter the value to be insert: 78**

**Insertion success!!!**

**Enter your choice: 3**

**Stack elements are:**

**78**

**56**

**Enter your choice: 2**

**Deleted : 78**

**Enter your choice: 3**

**Stack elements are:**

**56**

**STACK USING LINKED LIST**

**Step 1 - Include all the header files which are used in the program. And declare all the user defined functions.**

**Step 2 - Define a 'Node' structure with two members data and next.**

**Step 3 - Define a Node pointer 'top' and set it to NULL.**

**Step 4 - Implement the main method by displaying Menu with list of operations and make suitable function calls in the main method.**

**PROGRAM**

**#include<stdio.h>  
#include<stdlib.h>  
  
struct Node  
{  
   int data;  
   struct Node \*next;  
}\*top = NULL;  
  
void push(int);  
void pop();  
void display();  
  
void main()  
{  
   int choice, value;  
     
   printf("\n:: Stack using Linked List ::\n");  
   while(1)**

**{  
      printf("\n\*\*\*\*\*\* MENU \*\*\*\*\*\*\n");  
      printf("1. Push\n2. Pop\n3. Display\n4. Exit\n");  
      printf("Enter your choice: ");  
      scanf("%d",&choice);  
      switch(choice)**

**{**

**case 1: printf("Enter the value to be insert: ");  
scanf("%d", &value);  
push(value);  
break;  
case 2: pop(); break;  
case 3: display(); break;  
case 4: exit(0);  
default: printf("\nWrong selection!!! Please try again!!!\n");**

**}**

**}**

**}**

**void push(int value)  
{  
   struct Node \*newNode;  
   newNode = (struct Node\*)malloc(sizeof(struct Node));  
   newNode->data = value;  
   if(top == NULL)  
      newNode->next = NULL;  
   else  
      newNode->next = top;  
    top = newNode;  
    printf("\nInsertion is Success!!!\n");  
}**

**void pop()  
{  
   if(top == NULL)  
      printf("\nStack is Empty!!!\n");  
   else{  
      struct Node \*temp = top;  
      printf("\nDeleted element: %d", temp->data);  
      top = temp->next;  
      free(temp);  
   }  
}**

**void display()  
{  
   if(top == NULL)  
      printf("\nStack is Empty!!!\n");  
   else**

**{  
      struct Node \*temp = top;  
      while(temp->next != NULL)**

**{  
 printf("%d--->",temp->data);  
 temp = temp -> next;  
  }  
      printf("%d--->NULL",temp->data);**

**}**

**}**

**QUEUE USING ARRAY IMPLEMENTATION**

**Step 1 - Include all the header files which are used in the program and define a constant 'SIZE' with specific value.**

**Step 2 - Declare all the user defined functions which are used in queue implementation.**

**Step 3 - Create a one dimensional array with above defined SIZE (int queue[SIZE])**

**Step 4 - Define two integer variables 'front' and 'rear' and initialize both with '-1'. (int front = -1, rear = -1)**

**Step 5 - Then implement main method by displaying menu of operations list and make suitable function calls to perform operation selected by the user on queue.**

**PROGRAM**

**#include<stdio.h>**

**#define SIZE 10**

**int queue[SIZE], front = -1, rear = -1;**

**int enQueue(int value)**

**{**

**if(rear == SIZE-1)**

**printf("\nQueue is Full!!! Insertion is not possible!!!");**

**else**

**{**

**if(front == -1)**

**front = 0;**

**rear++;**

**queue[rear] = value;**

**printf("\nInsertion success!!!");**

**}**

**}**

**int deQueue()**

**{**

**if(front == rear)**

**printf("\nQueue is Empty!!! Deletion is not possible!!!");**

**else**

**{**

**printf("\nDeleted : %d", queue[front]);**

**front++;**

**if(front == rear)**

**front = rear = -1;**

**}**

**}**

**int display()**

**{**

**if(rear == -1)**

**printf("\nQueue is Empty!!!");**

**else{**

**int i;**

**printf("\nQueue elements are:\n");**

**for(i=front; i<=rear; i++)**

**printf("%d\t",queue[i]);**

**}**

**}**

**int main()**

**{**

**int value, choice;**

**while(1)**

**{**

**printf("\n\n\*\*\*\*\* MENU \*\*\*\*\*\n");**

**printf("1. Insertion\n2. Deletion\n3. Display\n4. Exit");**

**printf("\nEnter your choice: ");**

**scanf("%d",&choice);**

**switch(choice)**

**{**

**case 1: printf("Enter the value to be insert: ");**

**scanf("%d",&value);**

**enQueue(value);**

**break;**

**case 2: deQueue();**

**break;**

**case 3: display();**

**break;**

**case 4: exit(0);**

**default: printf("\nWrong selection!!! Try again!!!");**

**}**

**}**

**}**

**QUEUE USING LINKED LIST**

**Step 1 - Include all the header files which are used in the program. And declare all the user defined functions.**

**Step 2 - Define a 'Node' structure with two members data and next.**

**Step 3 - Define two Node pointers 'front' and 'rear' and set both to NULL.**

**Step 4 - Implement the main method by displaying Menu of list of operations and make suitable function calls in the main method to perform user selected operation.**

**#include<stdio.h>**

**struct Node**

**{**

**int data;**

**struct Node \*next;**

**}\*front = NULL,\*rear = NULL;**

**void insert(int);**

**void delete();**

**void display();**

**void main()**

**{**

**int choice, value;**

**printf("\n:: Queue Implementation using Linked List ::\n");**

**while(1)**

**{**

**printf("\n\*\*\*\*\*\* MENU \*\*\*\*\*\*\n");**

**printf("1. Insert\n2. Delete\n3. Display\n4. Exit\n");**

**printf("Enter your choice: ");**

**scanf("%d",&choice);**

**switch(choice)**

**{**

**case 1: printf("Enter the value to be insert: ");**

**scanf("%d", &value);**

**insert(value);**

**break;**

**case 2: delete(); break;**

**case 3: display(); break;**

**case 4: exit(0);**

**default: printf("\nWrong selection!!! Please try again!!!\n");**

**}**

**}**

**}**

**void insert(int value)**

**{**

**struct Node \*newNode;**

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

**newNode->data = value;**

**newNode -> next = NULL;**

**if(front == NULL)**

**front = rear = newNode;**

**else**

**{**

**rear -> next = newNode;**

**rear = newNode;**

**}**

**printf("\nInsertion is Success!!!\n");**

**}**

**void delete()**

**{**

**if(front == NULL)**

**printf("\nQueue is Empty!!!\n");**

**else**

**{**

**struct Node \*temp = front;**

**front = front -> next;**

**printf("\nDeleted element: %d\n", temp->data);**

**free(temp);**

**}**

**}**

**void display()**

**{**

**if(front == NULL)**

**printf("\nQueue is Empty!!!\n");**

**else**

**{**

**struct Node \*temp = front;**

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

**{**

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

**temp = temp -> next;**

**}**

**printf("%d--->NULL\n",temp->data);**

**}**

**}**