**Ques-1) Write a Program in C to implement insertion in 1-D Array.**

**Ans:**

#include<stdio.h>

int main()

{

int a[10], size, i;

printf("Enter the size of array: ");

scanf("%d", &size);

printf("Enter the element of array:\n ");

for(i=0; i<size; i++)

{

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

}

printf("Display Element :-\n ");

for(i=0; i<size; i++)

{

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

}

return 0;

}

**Output:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Ques-2) Write a Program in C to implement deletion in 1-D Array.**

**Ans:**

#include<stdio.h>

int main()

{

int a[10],size,i,pos;

printf("Enter the size of array: ");

scanf("%d",&size);

printf("Enter the element of array: \n");

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

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

}

printf("Enter the position to delete (1 to %d): ", size);

scanf("%d", &pos);

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

printf("Invalid position!\n");

}else{

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

a[i] = a[i + 1];

}

}

size--;

printf("Updated array: ");

for(i = 0; i < size; i++)

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

printf("\n");

return 0;

}

**Output:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Ques-3) Write a program in C to concatenate two arrays.**

**Ans:**

#include <stdio.h>

void concatenateArrays(int arr1[], int size1, int arr2[], int size2, int result[]) {

for (int i = 0; i < size1; i++) {

result[i] = arr1[i];

}

for (int i = 0; i < size2; i++) {

result[size1 + i] = arr2[i];

}

}

int main() {

int size1, size2;

printf("Enter the

size of the first array: ");

scanf("%d", &size1);

int arr1[size1];

printf("Enter %d elements for the first array: ", size1);

for (int i = 0; i < size1; i++) {

scanf("%d", &arr1[i]);

}

printf("Enter the size of the second array: ");

scanf("%d", &size2);

int arr2[size2];

printf("Enter %d elements for the second array: ", size2);

for (int i = 0; i < size2; i++) {

scanf("%d", &arr2[i]);

}

int result[size1 + size2];

concatenateArrays(arr1, size1, arr2, size2, result);

printf("Concatenated Array: ");

for (int i = 0; i < size1 + size2; i++) {

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

}

printf("\n");

return 0;

}

**Output:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Ques-4) Write a Program in C to implement the following operations on 2-D Array (Addition, Subtraction, Multiplication and Transpose).**

**Ans:**

|  |  |
| --- | --- |
| #include <stdio.h>  int main() {  int row, col;  printf("Enter the number of rows: ");  scanf("%d", &row);  printf("Enter the number of columns: ");  scanf("%d", &col);  int A[row][col], B[row][col], sum[row][col], diff[row][col],  product[row][col], transpose[row][col];  printf("Enter elements for first matrix:\n");  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  printf("Enter element [%d][%d]: ", i, j);  scanf("%d", &A[i][j]);  }  }  printf("\nEnter elements for second matrix:\n");  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  printf("Enter element [%d][%d]: ", i, j);  scanf("%d", &B[i][j]);  }  }  printf("\nFirst matrix:\n");  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  printf("%d ", A[i][j]);  }  printf("\n");  }  printf("\nSecond matrix:\n");  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  printf("%d ", B[i][j]);  }  printf("\n");  }  // Matrix operations  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  sum[i][j] = A[i][j] + B[i][j];  diff[i][j] = A[i][j] - B[i][j]; | product[i][j] = 0;  for (int k = 0; k < col; k++) {  product[i][j] += A[i][k] \* B[k][j];  }  }  }  // Transpose  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  transpose[j][i] = A[i][j];  }  }  // Results  printf("\nMatrix Addition Result:\n");  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  printf("%d\t", sum[i][j]);  }  printf("\n");  }  printf("\nMatrix Subtraction Result:\n");  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  printf("%d\t", diff[i][j]);  }  printf("\n");  }  printf("\nMatrix Multiplication Result:\n");  for (int i = 0; i < row; i++) {  for (int j = 0; j < col; j++) {  printf("%d\t", product[i][j]);  }  printf("\n");  }  printf("\nTranspose of First Matrix:\n");  for (int i = 0; i < col; i++) {  for (int j = 0; j < row; j++) {  printf("%d\t", transpose[i][j]);  }  printf("\n");  }  return 0;  } |

**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A white background with black and white clouds

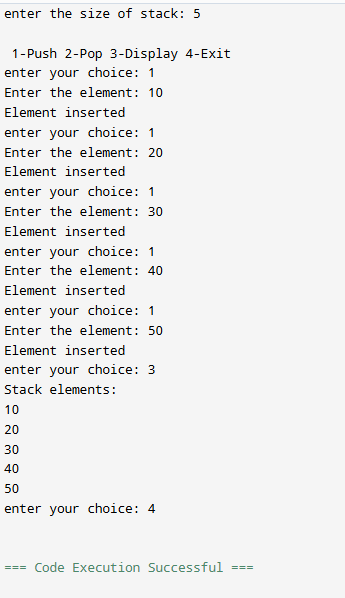
AI-generated content may be incorrect.**

**Ques-5) Write a Program in C to implement operations on stack using array.**

**Ans:**

|  |  |
| --- | --- |
| #include <stdio.h>  #include <stdlib.h>  int stack[10], maxstk, top = -1;  void push();  void pop();  void display();  void main() {  int ch;  printf("enter the size of stack: ");  scanf("%d", &maxstk);  printf("\n 1-Push 2-Pop 3-Display 4-Exit\n");  while(1) {  printf("enter your choice: ");  scanf("%d", &ch);  switch(ch) {  case 1:  push();  break;  case 2:  pop();  break;  case 3:  display();  break;  case 4:  exit(0);  default:  printf("Wrong choice\n");  }  }  }  void push() {  int ele;  if(top == maxstk - 1) {  printf("Overflow\n");  }  else {  printf("Enter the element: ");  scanf("%d", &ele);  top = top + 1;  stack[top] = ele;  printf("Element inserted\n");  }  }  void pop() {  if(top == -1)  printf("Underflow\n");  else { | printf("Element deleted: %d\n", stack[top]);  top = top - 1;  }  }  void display() {  if(top == -1)  printf("Underflow Condition\n");  else {  printf("Stack elements:\n");  for(int i = 0; i <= top; i++)  printf("%d\n", stack[i]);  }  } |

**Output:**

****

**Ques-6) Write a Program in C to implement operations on queue using array.**

**Ans:**

|  |  |
| --- | --- |
| #include <stdio.h>  #include <stdlib.h>  int queue[10], size, f = -1, r = -1;  void insert();  void delete();  void display();  int main() {  int ch;  printf("Enter the size of queue: ");  scanf("%d", &size);    printf("\n1-Insert 2-Delete 3-Display 4-Exit\n");  while (1) {  printf("Enter your choice: ");  scanf("%d", &ch);  switch (ch) {  case 1:  insert();  break;  case 2:  delete();  break;  case 3:  display();  break;  case 4:  exit(0);  default:  printf("Wrong choice\n");  }  }  return 0;  }  void insert() {  int ele;  if (r == size - 1) {  printf("Queue Overflow\n");  } else {  printf("Enter the element: ");  scanf("%d", &ele);  if (f == -1) {  f = r = 0;  } else {  r = r + 1;  }  queue[r] = ele;  printf("Element inserted\n");  }  } | void delete() {  if (f == -1) {  printf("Queue Underflow\n");  } else {  printf("Element deleted: %d\n", queue[f]);  if (f == r) {  f = r = -1;  } else {  f = f + 1;  }  }  }  void display() {  if (f == -1) {  printf("Queue is empty (Underflow Condition)\n");  } else {  printf("Elements in the queue are: ");  for (int i = f; i <= r; i++) {  printf("%d ", queue[i]);  }  printf("\n");  }  } |

**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Ques-7) Write a Program in C to implement operations on circular queue using array.**

**Ans:**

|  |  |
| --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #define MAX 10  int queue[MAX], front = -1, rear = -1, size;  void insert() {  int ele;  if ((rear + 1) % size == front) {  printf("Queue Overflow\n");  return;  }  printf("Enter the element: ");  scanf("%d", &ele);  if (front == -1) {  front = rear = 0;  } else {  rear = (rear + 1) % size;  }  queue[rear] = ele;  printf("Element inserted\n");  }  void delete() {  if (front == -1) {  printf("Queue Underflow\n");  return;  }  printf("Element deleted: %d\n", queue[front]);  if (front == rear) {  front = rear = -1;  } else {  front = (front + 1) % size;  }  }  void display() {  if (front == -1) {  printf("Queue is empty (Underflow Condition)\n");  return;  }  printf("Elements in the queue are: ");  int i = front;  while (1) {  printf("%d ", queue[i]);  if (i == rear) break;  i = (i + 1) % size;  } | printf("\n");  }  int main() {  int ch;  printf("Enter the size of the circular queue: ");  scanf("%d", &size);  printf("\n1-Insert 2-Delete 3-Display 4-Exit\n");  while (1) {  printf("Enter your choice: ");  scanf("%d", &ch);  switch (ch) {  case 1:  insert();  break;  case 2:  delete();  break;  case 3:  display();  break;  case 4:  exit(0);  default:  printf("Wrong choice\n");  }  }  return 0;  } |

**Output:**

A screenshot of a computer program

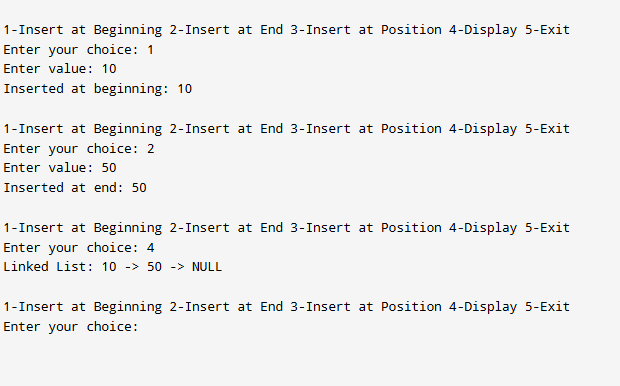
AI-generated content may be incorrect.

**Ques-8) Write a Program In C to implement insertion in linked list (beg; mid; end)?**

**Ans:**

|  |  |
| --- | --- |
| #include <stdio.h>  #include <stdlib.h>  struct Node {  int data;  struct Node\* next;  };  struct Node\* head = NULL;  void insertAtBeginning(int value) {  struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));  newNode->data = value;  newNode->next = head;  head = newNode;  printf("Inserted at beginning: %d\n", value);  }  void insertAtEnd(int value) {  struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));  newNode->data = value;  newNode->next = NULL;  if (head == NULL) {  head = newNode;  } else {  struct Node\* temp = head;  while (temp->next != NULL) {  temp = temp->next;  }  temp->next = newNode;  }  printf("Inserted at end: %d\n", value);  }  void insertAtPosition(int value, int pos) {  struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));  newNode->data = value;  if (pos == 1) {  newNode->next = head;  head = newNode;  } else {  struct Node\* temp = head;  for (int i = 1; temp != NULL && i < pos - 1; i++) {  temp = temp->next;  }  if (temp == NULL) {  printf("Invalid position!\n");  return;  }  newNode->next = temp->next; | temp->next = newNode;  }  printf("Inserted at position %d: %d\n", pos, value);  }  void display() {  struct Node\* temp = head;  if (temp == NULL) {  printf("List is empty\n");  return;  }  printf("Linked List: ");  while (temp != NULL) {  printf("%d -> ", temp->data);  temp = temp->next;  }  printf("NULL\n");  }  int main() {  int choice, value, position;  while (1) {  printf("\n1-Insert at Beginning 2-Insert at End 3-Insert at Position 4-Display 5-Exit\n");  printf("Enter your choice: ");  scanf("%d", &choice);  switch (choice) {  case 1:  printf("Enter value: ");  scanf("%d", &value);  insertAtBeginning(value);  break;  case 2:  printf("Enter value: ");  scanf("%d", &value);  insertAtEnd(value);  break;  case 3:  printf("Enter value and position: ");  scanf("%d %d", &value, &position);  insertAtPosition(value, position);  break;  case 4:  display();  break;  case 5:  exit(0);  default:  printf("Wrong choice!\n");  }  }  return 0;  } |

**Outputs:**

****