```c
#include <stdio.h>
#include <stdlib.h>

#define MAX_SIZE 10

struct Stack {
    int arr[MAX_SIZE];
    int top;
};

void initializeStack(struct Stack *stack) {
    stack->top = -1;
}

int isEmpty(struct Stack *stack) {
    return stack->top == -1;
}
int isFull(struct Stack *stack) {
    return stack->top == MAX_SIZE - 1;
}

void push(struct Stack *stack, int value) {
    if (isFull(stack)) {
        printf("Stack overflow. Cannot push element %d.\n", value);
        return;
    }

    stack->arr[++stack->top] = value;
    printf("Pushed %d onto the stack.\n", value);
}

void pop(struct Stack *stack) {
    if (isEmpty(stack)) {
        printf("Stack underflow. Cannot pop from an empty stack.\n");
        return;
    }
```

```c
    }

    printf("Popped %d from the stack.\n", stack->arr[stack->top--]);
}
void display(struct Stack *stack) {
    if (isEmpty(stack)) {
        printf("Stack is empty.\n");
        return;
    }

    printf("Stack elements: ");
    for (int i = 0; i <= stack->top; i++) {
        printf("%d ", stack->arr[i]);
    }
    printf("\n");
}

int main() {
    struct Stack myStack;
    initializeStack(&myStack);

    push(&myStack, 5);
    push(&myStack, 10);
    push(&myStack, 15);
    display(&myStack);

    pop(&myStack);
    display(&myStack);

    push(&myStack, 20);
    display(&myStack);

    return 0;
}

```

```
Pushed 5 onto the stack.
Pushed 10 onto the stack.
Pushed 15 onto the stack.
Stack elements: 5 10 15
Popped 15 from the stack.
Stack elements: 5 10
Pushed 20 onto the stack.
Stack elements: 5 10 20
```

```c
#include <stdio.h>
void swap(int *a,int *b)
{

    int temp=*a;
    *a=*b;
    *b=temp;
}
int main()
{
    int num1,num2;
    printf("enter the first number\n");
    scanf("%d",&num1);
    printf("enter the second number\n");
    scanf("%d",&num2);
    printf("before swapping num1=%d,num2=%d\n",num1,num2);
    swap(&num1,&num2);
    printf("after swapping num1=%d,num2=%d\n",num1,num2);
    return(0);
}
```

```
enter the first number
12
enter the second number
13
before swapping num1=12,num2=13
after swapping num1=13,num2=12
```

```c
#include <stdio.h>
#include <stdlib.h>

void* myMalloc(size_t size) {
    return malloc(size);
}

void* myRealloc(void* ptr, size_t size) {
    return realloc(ptr, size);
}

void* myCalloc(size_t num, size_t size) {
    return calloc(num, size);
}

void myFree(void* ptr) {
    free(ptr);
}

int main() {
    int *arr1, *arr2;
    size_t size;

    printf("Enter the size of the array: ");
    scanf("%zu", &size);


    arr1 = (int*)myMalloc(size * sizeof(int));

    if (arr1 == NULL) {
        printf("Memory allocation failed.\n");
        return 1;

    }
```

```
}

printf("Enter elements of the array:\n");
for (size_t i = 0; i < size; i++) {
    printf("Element %zu: ", i + 1);
    scanf("%d", &arr1[i]);
}

printf("Elements of the array (malloc):\n");
for (size_t i = 0; i < size; i++) {
    printf("%d ", arr1[i]);
}
printf("\n");


size *= 2;
arr2 = (int*)myRealloc(arr1, size * sizeof(int));


if (arr2 == NULL) {
    printf("Memory reallocation failed.\n");
    myFree(arr1);
    return 1;
}

printf("Enter additional elements of the array:\n");
for (size_t i = size / 2; i < size; i++) {
    printf("Element %zu: ", i + 1);
    scanf("%d", &arr2[i]);
}

printf("Elements of the array (realloc):\n");
for (size_t i = 0; i < size; i++) {
    printf("%d ", arr2[i]);
```

```c
if (arr2 == NULL) {
    printf("Memory reallocation failed.\n");
    myFree(arr1);
    return 1;
}

printf("Enter additional elements of the array:\n");
for (size_t i = size / 2; i < size; i++) {
    printf("Element %zu: ", i + 1);
    scanf("%d", &arr2[i]);
}

printf("Elements of the array (realloc):\n");
for (size_t i = 0; i < size; i++) {
    printf("%d ", arr2[i]);
}
printf("\n");

myFree(arr2);

return 0;
}
```

```
Enter the size of the array: 2
Enter elements of the array:
Element 1: 12
Element 2: 13
Elements of the array (malloc):
12 13
Enter additional elements of the array:
Element 3: 3
Element 4: 12
Elements of the array (realloc):
12 13 3 12
```

Lab Program - 1

Swapping using pointers.

```
#include <stdio.h>
void Swap (int *a, int * b) {
    int temp = *a;
    int *a = * b;
    *b = temp;
}
int main () {
    int num1, num2;
Printf("Enter the first number:");
Scanf ("%d", num1);
Printf ("Enter the Second number:");
Scanf ("%d", num2);

Printf ("Before swapping: num1 = %d,
num2 = %d \n", num1, num2);
Swap (&num1, &num2);
Printf ("After Swapping : num1 = %d,
num2 = %d \n", num1, num2);
return 0;
}
```

Output: Enter the first number : 10
Enter the second number : 15
Before Swapping : num 1 = 10, num 2 = 15
After Swapping : num 1 = 15, num 2 = 10

Prg - 2

dynamic memory allocation.

```
#include <stdio.h>
void MallocEx(int);
Void CallocEx (int);

void main ()
{
  int n ;
  Printf ("Enter the value of n \n");
  Scanf ("%d", &n);
  MallocEx (n);
  CallocEx (n);
}
Void MallocEx (int n)
{
  int *Ptr;
  int i;
  int arr[n];
  Ptr = (int *) malloc (n * sizeof(int));
  for (i=0; i<n; i++)
  {
    Ptr [i] = i+1;
  }
```

```
Printf("malloc dynamic memory allocation\n");
Printf("the Elements of array are :\n");
for(i=0;i<n;i++)
  {
    Printf("%d", Ptr[i]);
  }

  Printf("malloc dynamic memory
      allocation\n");
Printf("the Elements of array are:\n");
for  (i=0; i<n; i++)
    {
      Printf("%d", Ptr[i]);
    }
  Printf("\n");
  free(Ptr);
}
void CallocEx(int n)
{
  int *Ptr;
  int i;
  int arr[n];
  Ptr = (int*) calloc(n, sizeof(int));
  for(i=0;i<n;i++)
  {
    Ptr[i] = i+1;
  }
```

output:

```
Printf ("Calloc dynamic memory
        allocation \n");
Printf ("the elements of array are: \n");
for (i=0; i<n; i++)
  {
    Printf ("%d", Ptr[i]);
  }
Printf ("\n");
Printf ("\n");
Printf ("Realloc dynamic memory allocation \n");
Printf ("the elements of array are: \n");
n=15;
Ptr = (int *) realloc (Ptr, n * sizeof(int));
for (i=10; i<n; i++)
  {
    Ptr[i] = i+1;
  }
for (i=0; i<n; i++)
{
  Printf ("%d", Ptr[i]);
}
free (Ptr);
}
```

output: Enter number of Elements : 5

Memory number Successfully allocated using malloc

The Elements of array are : 1, 2, 3, 4, 5

memory Successfully allocated using Calloc

The Elements of array are : 1, 2, 3, 4, 5

Enter the new size of array : 10

memory Successfully re-allocated using realloc

The Elements of array are : 1, 2, 3, 4, 5, 6, 2, 8, 9, 10.

Program - 3

Stack Implementation

```
# include <stdio.h>
# include <stdlib.h>
# define SIZE 4
int top = -1;
int inp_array [SIZE];
void Push();
void POP();
void show();

void main()
{
  int ch;
  while (1)
  {
  Printf ("Operations on Stack: \n");
  Print ("1. Push the elements t \n2. POP the
          element \n3. Show \n4. End \n");
   Scanf ("%d", &ch);

  Switch (ch)
   {
    case 1;
      Push();
      break;
    case 2:
      POP()
    break;
```

```
        case 3:
            show()
            break;
        case 4:
            exit(0);
        default:
            printf("Invalid choice\n");
    }
  }
}
void push()
{
int x;
if (top == SIZE-1)
  {
    printf("overflow\n");
  }
  else
{
  printf("Enter the element to be
    added in stack:\n");
    scanf("%d", &x);
    top = top+1
    stk_array[top] = x;
  }
}
void pop()
    if (top == -1)
```

```
    {
      Printf ("Underflow \n");
    }
    Else
    {
      Printf ("Popped Element: %d \n",
              inp_array[top]);
      top = top - 1;
    }
}
void Show()
{ if (top == -1)
  {
  Printf ("Underflow \n");
  }
  Else
  {
  Printf ("Elements in stack are \n");
  {
  Printf ("%d \n", inp_array[i]);
  }
}
```

Output

Operations on Stack;

1) Push the Element
2) POP the Element
3) Show
4) End

Enter the choice;

3

Underflow

Operations on Stack;

1. Push the Element;
2) POP the Element
3) Show
4) End

Enter the choice

1

Enter the Element to be added in Stack

5

Operations on Stack.

1) Push the Element
2) POP the Element
3) Show
4) End

Enter the Choice

Enter the element to be added in Stack:

3

Operations on Stack:

1) Push the element

2) POP the element

3) Show

4) End

Enter the choice:

2

Enter the element to be removed.

3

POPPed Element- 3.