1. Program to swap Numbers using pointers.

```c
#include<stdio.h>

void swap (int *a, int *b){
    int temp = *a;
    *a = *b;
    *b = temp;
}

int main(){
    int a, b;
    printf("Enter a and b: ");
    scanf("%d %d", &a, &b);
    swap(&a, &b);
    printf("After swapping a = %d and b = %d", a, b);
}
```

```
Enter a and b: 5 8
After swapping a = 8 and b = 5
```

2. Program to demonstrate dynamic memory allocation.

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

int main(){
    int *ptr1;
    int size = 5;
    ptr1 = (int *)malloc(size * sizeof(int));
    if(ptr1 == NULL){
        printf("Malloc failed");
        return 1;
    }
    printf("Malloc successful\n");

    int *ptr2;
    ptr2 = (int *)calloc(size, sizeof(int));
```

```
    if (ptr2 == NULL)
    {
        printf("Calloc failed");
        return 1;
    }
    printf("Calloc successful\n");

    int size2 = 10;
    ptr1 = (int *)realloc(ptr1, size2 * sizeof(int));
    if (ptr1 == NULL)
    {
        printf("Realloc failed");
        return 1;
    }
    printf("Realloc successful");

    free(ptr1);
    free(ptr2);
}
```

```
Malloc successful
Calloc successful
Realloc successful
```

3. Program to demonstrate stack.

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

#define MAX 10

struct Stack{
    int items[MAX];
    int top;
};

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

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

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

void push(struct  Stack *stack, int value){
    if(isFull(stack)){
        printf("Cant Push, Overflow\n");
    }
```

```
    else{
        stack->top++;
        stack->items[stack->top] = value;
        printf("Pushed %d\n", value);
    }
}

void pop(struct Stack *stack)
{
    if (isEmpty(stack))
    {
        printf("Cant Pop, Underflow\n");
    }
    else
    {
        stack->top--;
        printf("Poped the element\n");
    }
}

void display(struct Stack *stack){
    if (isEmpty(stack))
    {
        printf("Its empty\n");
    }
```

```
    }
    else{
        printf("Elements of Stack\n");
        for (int i = stack->top; i >= 0; i--){
            printf("%d\n", stack->items[i]);
        }
    }
}

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

    push(&stack, 10);
    push(&stack, 20);
    push(&stack, 30);

    display(&stack);

    pop(&stack);

    pop(&stack);
    display(&stack);

    return 0;
}
```

```
Pushed 10
Pushed 20
Pushed 30
Elements of Stack
30
20
10
Poped the element
Poped the element
Elements of Stack
10
```