1) Swapping using pointers;

```
#include <stdio.h>
void swap (int *, int *);
void main ()
{
    int a, b;
    int p, q;
    scanf ("%d %d", &a, &b);
    p = &a;
    q = &b;
    printf ("Before swapping a = %d and b = %d\n",
            a, b);
    swap (p, q);
    printf ("After swapping a = %d and b = %d",
            a, b);
}
void swap (int *p, int *q)
{
    int temp;
    temp = *p;
    *p = *q;
    *q = temp;
}
```

Output:

```
5 6
Before swapping a = 5 and b = 6
After swapping a = 6 and b = 5
```

Dynamic memory allocation

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

void main()
{
    int *p1, *p2;
    int n, i;

    printf("Enter the number of array elements\t");
    scanf("%d", &n);

    p1 = (int*) malloc (n* sizeof(int));
    p2 = (int*) calloc (n, sizeof(int));

    if (p1 == NULL && p2 == NULL)
    {
        printf("memory is not allocated");
        exit(0);
    }
    else
    {
        printf("memory is succesfully allocated");

        for (i=0; i<n; i++)
            p1[i] = i+1;
        printf("array elements in malloc are:\n");
        for (i=0; i<n; i++)
            printf("%d", p1[i]);

        for (i=0; i<n; i++)
            p2[i] = i+1;
```

```
printf ("array elements in calloc are: \n");
for (i=0; i<n; i++)
    printf ("%d", p2[i]);
}
printf ("Enter the new size of n");
scanf ("%d", &n);
p2 = (int*) realloc (p2, n*(sizeof(int)));
if (p2 == NULL)
{
    printf ("memory is not allocated");
    exit(0);
}
else
{
    printf ("memory is succesfully allocated");
    for (i=0; i<n; i++)
        p2[i] = i+1;
    printf ("array elements in realloc are: \n");
    for (i=0; i<n; i++)
        printf ("%d", p2[i]);
}
free (p1);
free (p2);
}
```

output

```
Enter the number of array elements 5
memory is succesfully allocated
array elements in malloc are:
1 2 3 4 5
array elements in calloc are:
1 2 3 4 5
```

Enter the new size of array
8
memory is succesfully allocated
array elements in realloc are:
1 2 3 4 5 6 7 8

3) Stack Implementation.

```
#include <stdio.h>

int stack[100], i, j, ch = 0, n, top = -1;

void push();
void pop();
void ~~show~~ display();
void main()
{
    printf("Enter the no. of elements in the stack");
    scanf("%d", &n);

    ~~printf~~ while (ch != 4)
    {
        printf("Choose one from the below options\n");
        printf("\n 1. Push \n 2. Pop \n 3. ~~Display~~ \n 4. Exit");
        printf("\n Enter your choice \n");
        scanf("%d", &ch);
        switch(ch)
        {
            case 1: push();
                    break;
            case 2; pop();
                    break;
```

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

        case 4: printf("Exit");
                break;

        default: printf("Please enter valid choice");
    }
}

void push()
{
    int val;
    if (top == n)
        printf("\n Overflow");
    else
    {
        printf("Enter the value:");
        scanf("%d", &val);
        top = top+1;
        stack[top] = val;
    }
}

Void pop()
{
    if (top == -1)
        printf("Underflow");
    else
        top = top-1
}
```

```
void display()
{
    if (top == -1)
        printf ("stack is empty");
    for (i = top ; i >= 0 ; i --)
    printf ("%d \n", stack[i]);
}
```

Output:

Enter number of elements : 5
choose 1 - Push
2 - Pop
3 - Display
4 - Exit

Enter your choice
1
Enter choose 1 - Push
2 - Pop
3 - Display
4 - Exit
Enter your choice
2
choose 1 - Push
2 - Pop
3 - Display
4 - Exit.
Enter your choice
3
choose 1 - Push
2 - Pop
3 - Display
4 - Exit

Sp.1
24/12/23