21/12/23

LAB 2

1. Swapping using pointers

```
#include <stdio.h>
void swap(int *, int *);
void main()
{
    int a,b;
    printf("Enter values of a and b:\n");
    scanf("%d%d", &a, &b);
    printf("Values before swapping : a=%d
            and b=%d", a,b);
    swap(&a, &b);
    printf("Values of a and b after swapping:
            a=%d and b=%d", a, b);
}

void swap(int *p, int *q)
{
    int temp;
    temp = *p;
    *p = *q;
    *q = temp;
}
```

OUTPUT:

```
Enter values of a and b:
5 4
Values before swapping : a=5 and b=4
Values of a and b after swapping : a=4 and b=5
```

2. Dynamic memory allocation

```
#include <stdio.h>
#include <stdlib.h>
void main()
{
    int *p, *q, *r;
    int i, m, n;
    printf(" Enter no. of elements for p: ");
    scanf(" Enter no %d", &m);
    printf(" Enter no. of elements for q: ");
    scanf("%d", &n);

    p = (int *) malloc (m * sizeof(int));
    q = (int *) calloc(n, sizeof(int));

    if(p==NULL && q==NULL)
        printf("Memory is not allocated");
    else
        printf("Memory allocated successfully");
    printf("Elements of p: \n");
    for(i=0; i<m; i++)
        printf("%d \t",i+1);

    printf("Elements of q: \n");
    for(i=0; i<n; i++)
        printf("%d \t", i);

    free(p);
    printf("\n Malloc memory successfully freed");
```

```
printf("\n Enter the new size of the
                array: ");
scanf("%d", &m);
r=(int *) realloc(q, n*sizeof(int));
if(r!=NULL)
    printf("Memory successfully reallocated
            using realloc");
else
    printf("Not allocated");
}
```

OUTPUT :-

```
Enter no. of elements for p: 5
Enter no. of elements for q: 4
Memory allocated successfully

Elements of p:
1  2  3  4  5
Elements of q:
0  1  2  3
Malloc memory successfully freed.

Enter the new size of array: 3
Memory successfully re-allocated using realloc
```

3. Stack Implementation.

```
#include <stdio.h>
int stack[100], i, j, ch, n, top;
void push();
void pop();
void display();
void main();
{
    int i;
    printf("Enter number of elements : ");
    scanf("%d", &n);
    while (ch != 4)
    {
        printf("Choose 1 - Push, 2 - Pop, 3 - Display
                4 - Exit : \n ");
        scanf("%d", &ch);
        printf("
        switch (ch)
        {
            case 1: push();
                    break;

            case 2:
                    pop();
                    break;
            case 3:
                    display();
                    break;
            case 4:
                    printf("Exited");
                    exit(0);
                    break();
        }
    }
}
```

```
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 :
1
Enter the value : 2
Choose 1 - Push , 2 - Pop , 3 - Display, 4 - Exit :
1
Enter the value : 3
Choose 1 - Push , 2 - Pop , 3 - Display , 4 - Exit :
1
Enter the value : 4
Choose 1 - Push , 2 - Pop , 3 - Display , 4 - Exit :
1
Enter the value : 5
Choose 1 - Push , 2 - Pop , 3 - Display , 4 - Exit :
3
Stack is : 2 3 4 5
Choose 1 - Push , 2 - Pop , 3 - Display , 4 - Exit :
2
Choose 1 - Push , 2 - Pop , 3 - Display , 4 - Exit :
3
Stack is : 2 3 4
Choose 1 - Push , 2 - Pop , 3 - Display , 4 - Exit :
4
Exited .
```

21/12/2023