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,
num 2 = %d \n", num1, num 2 );
Swap (&num 1., &num 2) ;
Printf ("After Swapping : num1 = %d,
num 2 = %d \n", num1, num2);
return 0;
}
```

Output: Enter the first number : 10
Enter the Second number : 15

Before Swabbing : num 1 = 10, num 2 = 15
after Swabbing : 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);
}
```

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 ("1:1. 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
inp_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

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.

S.P.T
21/12/23

```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
10
enter the second number
15
before swapping num1=10, num2=15
after swapping num1=15, num2=10
```

```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");
```

```c
    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]);
}
printf("\n");

myFree(arr2);

return 0;
}
```

```
Enter the size of the array: 5
Enter elements of the array:
Element 1: 1
Element 2: 2
Element 3: 3
Element 4: 4
Element 5: 5
Elements of the array (malloc):
1 2 3 4 5
Enter additional elements of the array:
Element 6: 7
Element 7: 8
Element 8: 9
Element 9: 10
```

```c
#include<stdio.h>
int stack[4],choice,n,top,x,i;
void push(void);
void pop(void);
void display(void);
int main()
{
    top=-1;
    printf("\n Enter the size of STACK[MAX=100]:");
    scanf("%d",&n);
    printf("\n\t STACK OPERATIONS USING ARRAY");
    printf("\n\t 1.PUSH\n\t 2.POP\n\t 3.DISPLAY\n\t 4.EXIT");
    do
    {
        printf("\n Enter the Choice:");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            {
                push();
                break;
            }
            case 2:
            {
                pop();
                break;
            }
            case 3:
            {
                display();
                break;
            }
            case 4:
            {
```

```
            case 4:
            {
                printf("\n\t EXIT POINT ");
                break;
            }
            default:
            {
                printf ("\n\t Please Enter a Valid Choice(1/2/3/4)");
            }

        }
    }
    while(choice!=4);
    return 0;
}
void push()
{
    if(top>=n-1)
    {
        printf("\n\tSTACK is over flow");

    }
    else
    {
        printf(" Enter a value to be pushed:");
        scanf("%d",&x);
        top++;
        stack[top]=x;
    }
}
void pop()
{
    if(top<=-1)
    {
        printf("\n\t Stack is under flow");
```

```
    {
        printf(" Enter a value to be pushed:");
        scanf("%d",&x);
        top++;
        stack[top]=x;
    }
}
void pop()
{
    if(top<=-1)
    {
        printf("\n\t Stack is under flow");
    }
    else
    {
        printf("\n\t The popped elements is %d",stack[top]);
        top--;
    }
}
void display()
{
    if(top>=0)
    {
        printf("\n The elements in STACK \n");
        for(i=top; i>=0; i--)
            printf("\n%d",stack[i]);
        printf("\n Press Next Choice");
    }
    else
    {
        printf("\n The STACK is empty");
    }

}
```

```
Enter the size of STACK[MAX=100]:5

        STACK OPERATIONS USING ARRAY
        1.PUSH
        2.POP
        3.DISPLAY
        4.EXIT
Enter the Choice:3

The STACK is empty
Enter the Choice:1
Enter a value to be pushed:3

Enter the Choice:1
Enter a value to be pushed:5

Enter the Choice:3

The elements in STACK

5
3
Press Next Choice
Enter the Choice:2

        The popped elements is 5
Enter the Choice:3

The elements in STACK

3
Press Next Choice
Enter the Choice:
```