DYNAMIC MEMORY ALLOCATION

AIM:

To create a c program for demonnstrating the dynamic memory allocation of array memory usinf malloc ,calloc,realloc,free.

ALGORITHM:

· stdio.h and stdlib.h are included to perform input, output, and dynamic memory operations.

· A pointer variable is declared to store the base address of the dynamically allocated array.

· The user enters the size of the array to be allocated using malloc().

· Memory is allocated dynamically using malloc() based on the given size.

· The allocated memory is checked to ensure allocation is successful.

· The contents of the array before initialization are displayed.

· The user enters values to initialize the allocated memory.

· The initialized values of the array allocated using malloc() are displayed.

· The user enters a new size for the array.

· The previously allocated memory is resized using realloc().

· The contents of the array after reallocation are displayed.

· The memory allocated using malloc() and realloc() is released using free().

· The user enters the size of the array to be allocated using calloc().

· Memory is allocated and initialized to zero using calloc().

· The contents of the array before initialization are displayed.

· The user enters values to initialize the array allocated using calloc().

· The initialized values of the array allocated using calloc() are displayed.

· The memory allocated using calloc() is released using free().

· The program ends after completing all dynamic memory operations.

PROGRAM:

```
/*
* Program to demonstrate dynamic memory allocation
* Author: MUTHUGANESH S
* Date  : 16/01/2026
* Filename : DynamicMemory.c
```

```c
* retval : void
*/

#include <stdio.h>
#include <stdlib.h>
int main(void){
    int *Array;
    int Size, i;

    printf("Enter the size of the array for malloc: ");
    scanf("%d", &Size);

    // malloc is used to allocate memory allocation
    Array = (int *)malloc(Size * sizeof(int));
    if (Array == NULL){
        printf("Memory allocation failed\n");
        return 0;
    }

    printf("\nBefore initialization of malloc:\n");
    for(int i=0;i<Size;i++){
        printf("%d ",Array[i]);
    }
    printf("\n\nEnter %d integers to initialize the array:\n", Size);
    // Initializing the allocated memory
    for(i = 0; i < Size; i++){
        scanf("%d", &Array[i]);
    }

    printf("\nAfter initialization of malloc:\n");
    for(int i=0;i<Size;i++){
        printf("%d ",Array[i]);
    }

    //realloc is used to reallocate memory
    int NewSize;
    printf("\n\nEnter the new size of the array: ");
    scanf("%d", &NewSize);
    Array = (int *)realloc(Array, NewSize * sizeof(int));
    if (Array == NULL){
        printf("Memory reallocation failed\n");
        return 0;
    }

    printf("\nAfter reallocation of realloc:\n");
    for(int i=0;i<NewSize;i++){
        printf("%d ",Array[i]);
    }

    // Freeing the allocated memory
    free(Array);

    //calloc is used to allocate memory and initialize to zero
    printf("\n\nEnter the size of the array for calloc: ");
```

```
    scanf("%d", &Size);
    Array = (int *)calloc(Size, sizeof(int));

    printf("\nBefore initialization of calloc:\n");
    for(int i=0;i<Size;i++){
        printf("%d ",Array[i]);
    }
    printf("\n\nEnter %d integers to initialize the array:\n", Size);
    // Initializing the allocated memory
    for(i = 0; i < Size; i++){
        scanf("%d", &Array[i]);
    }

    printf("\nAfter initialization of calloc:\n");
    for(int i=0;i<Size;i++){
        printf("%d ",Array[i]);
    }

    // Freeing the allocated memory
    free(Array);

    return 0;
}
```

OUTPUT: