```
/*
Four functions for dynamic memory allocation:(Always include <stdlib.h>)
1.malloc():reserves a block of memory with a given amount of bytes. All the
values at allocated
memory are initialized to garbage values
Syntax:
ptr=(ptr-type*)malloc(size_in_bytes);
Example:
int *ptr;
ptr=(int*)malloc(3*sizeof(int))
2.calloc():reserves n blocks of memory with given amount of bytes. All the
values at allocated
memory are initialized to 0.
Syntax:
ptr=(ptr-type*)calloc(n,size_in_bytes);
3.realloc():If the dynamically allocated memory is insufficient we can
change the size of
previously allocated memory using realloc() function.
Syntax:
ptr=(ptr-type*)realloc(ptr,new_size_in_bytes);
```

```
4.free():Used to free the allocated memeory.

Syntax:

free(ptr);

*/

#include <stdio.h>

#include <stdlib.h>

int main()

{

// Use of malloc

// int *ptr;

// int n;

// printf("Enter the size of the array you want to create:");

// scanf("%d",&n);

// ptr = (int *)malloc(n * sizeof(int));

// for (int i = 0; i < n; i++)

// {

// printf("Enter value of element %d of the array:",i);

// scanf("%d",&ptr[i]);

// }

// for (int i = 0; i < n; i++)
// {

// printf("Value of element %d of the array is %d\n",i,ptr[i]);

// }




//Use of Calloc

int *ptr;

int n;

printf("Enter the size of the array you want to create:");

scanf("%d",&n);

ptr = (int *)calloc(n , sizeof(int));

for (int i = 0; i < n; i++)

{

printf("Enter value of element %d of the array:",i);

scanf("%d",&ptr[i]);

}

for (int i = 0; i < n; i++)

{

printf("Value of element %d of the array is %d\n",i,ptr[i]);
}

free(ptr);

return 0;

}
```