practice:

```
#include<stdio.h>
#include<conio.h>
int main(){
int arr[5]={1,2,3,4,5};
printf("address of 0 is %d \n",&arr[0]);
printf("address of 0 is %d \n",arr);
printf("address of 1 is %d \n",&arr[1]);
printf("address of 1 is %d \n",arr+1);
printf("----------------\n");
printf("value of 0 is %d \n",*(&arr[0]));
printf("value of 0 is %d \n",*(arr));
printf("value of 1 is %d \n",*(&arr[1]));
printf("value of 1 is %d \n",*(arr+1));
return 0;
}
```

malloc

Program to calculate the sum of n numbers entered by the user using malloc

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

int main() {
  int n, i, *ptr, sum = 0;
  printf("Enter number of elements: ");
  scanf("%d", &n);
  ptr = (int*) malloc(n * sizeof(int));
  // if memory cannot be allocated
  if(ptr == NULL) {
          printf("Error! memory not allocated.");
          exit(0);
  }
  printf("Enter elements: ");
  for(i = 0; i < n; ++i) {
          scanf("%d", ptr + i);
          sum += *(ptr + i);
  }
  printf("Sum = %d", sum);
  // deallocating the memory
  free(ptr);
  return 0;
}
```

calloc

// Program to calculate the sum of n numbers entered by the user using calloc

```
#include <stdio.h>
#include <stdlib.h>
int main() {
  int n, i, *ptr, sum = 0;
  printf("Enter number of elements: ");
  scanf("%d", &n);
  ptr = (int*) calloc(n, sizeof(int));
  if(ptr == NULL) {
          printf("Error! memory not allocated.");
          exit(0);
  }
  printf("Enter elements: ");
  for(i = 0; i < n; ++i) {
          scanf("%d", ptr + i);
          sum += *(ptr + i);
  }
  printf("Sum = %d", sum);
  free(ptr);
  return 0;
}
```

realloc

```
#include <stdio.h>
#include <stdlib.h>
int main() {
  int *ptr, i , n1, n2;
  printf("Enter size: ");
  scanf("%d", &n1);
  ptr = (int*) malloc(n1 * sizeof(int));
  printf("Addresses of previously allocated memory:\n");
  for(i = 0; i < n1; ++i)
          printf("%d\n",ptr + i);
  printf("\nEnter the new size: ");
  scanf("%d", &n2);
  // rellocating the memory
  ptr = realloc(ptr, n2 * sizeof(int));
  printf("Addresses of newly allocated memory:\n");
  for(i = 0; i < n2; ++i)
          printf("%d\n", ptr + i);
  free(ptr);
  return 0;
}
```

realloc

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
	// This pointer will hold the
	// base address of the block created
	int* ptr;
	int n, i;
	// Get the number of elements for the array
	n = 5;
	printf("Enter number of elements: %d\n", n);
	// Dynamically allocate memory using calloc()
	ptr = (int*)calloc(n, sizeof(int));

	// Check if the memory has been successfully
	// allocated by malloc or not
	if (ptr == NULL) {
	printf("Memory not allocated.\n");
	exit(0);
	}
	else {
	// Memory has been successfully allocated
	printf("Memory successfully allocated using calloc.\n");
	// Get the elements of the array
	for (i = 0; i < n; ++i) {
	ptr[i] = i + 1;
	}
	// Print the elements of the array
	printf("The elements of the array are: ");
	for (i = 0; i < n; ++i) {
	printf("%d, ", ptr[i]);
	}
	// Get the new size for the array
	n = 10;
	printf("\n\nEnter the new size of the array: %d\n", n);
	// Dynamically re-allocate memory using realloc()
	ptr = realloc(ptr, n * sizeof(int));
	// Memory has been successfully allocated
	printf("Memory successfully re-allocated using realloc.\n");
	// Get the new elements of the array
	for (i = 5; i < n; ++i) {
```

```
        ptr[i] = i + 1;
        }
        // Print the elements of the array
        printf("The elements of the array are: ");
        for (i = 0; i < n; ++i) {
        printf("%d, ", ptr[i]);
        }
        free(ptr);
        }
        return 0;
}
```