```c
/*

Que : 1. Write a C program to to represent 1-D array using Dynamic Memory Allocation.
Owner: Rushikesh Sanjay Pokharkar
Batch: PPA9

*/

//                                        *********  Solution  *********


#include<stdio.h> // Include necessary header files
#include<stdlib.h>


void main() {

	int n, search, flag = 0; // Declaration of variables.

	int* p = NULL; // Initializing pointer to null value.

	printf("Enter How Many Elements Do You Want In Array : ");
	scanf_s("%d", &n); // Taking input - Number of elements in array.

	p = (int*)malloc(n * sizeof(int)); // Syntax for Dynamic memory allociation of 
array.

	printf("Enter Array Elements..\n");
	for (int i = 0; i < n; i++) // For loop to take input array elements.
	{
		scanf_s("%d", p + i);
	}

	printf("Array Elements are: ");
	for (int i = 0; i < n; i++) // For loop to print array elements.
	{
		printf("%d ", *(p + i));
	}

	printf("\nEnter a Searching Element: ");
	scanf_s("%d", &search); // Enter a Searching element to search in array.

	for (int i = 0; i < n; i++) // For loop to search element in array.
	{
		if (search == *(p + i))
		{
			flag = 1;
			break;
		}
	}

	if (flag == 1) {
		printf("Element Found..");
	}
	else {
		printf("Element Not Found..");
	}

	void free(p); // Free the memory which is given to pointer p Dynamically...
}
```

```c
/*

Que : 2. Write a C program to sort 1-D array in ascending order using Dynamic Memory
Allocation.
Owner: Rushikesh Sanjay Pokharkar
Batch: PPA9

*/

//                                      *********  Solution  *********


#include<stdio.h> // Include necessary header files
#include<stdlib.h>


void main() {

	int n, search, flag = 0; // Declaration of variables.

	int* p = NULL; // Initializing pointer to null value.

	printf("Enter How Many Elements Do You Want In Array : ");
	scanf_s("%d", &n); // Taking input - Number of elements in array.

	p = (int*)malloc(n * sizeof(int)); // Syntax for Dynamic memory allociation of
array.

	printf("Enter Array Elements..\n");
	for (int i = 0; i < n; i++) // For loop to take input array elements.
	{
		scanf_s("%d", p + i);
	}

	printf("Array Elements are: ");
	for (int i = 0; i < n; i++) // For loop to print array elements.
	{
		printf("%d ", *(p + i));
	}

	printf("\nThe sorted array in ascending order:  ");

	// Logic to sort array in ascending order
	for (int i = 0; i < n-1; i++)
	{
		for (int j = i+1; j < n; j++)
		{
			if (*(p + i) > *(p + j)) {
				int temp = *(p + i);
				*(p + i) = *(p + j);
				*(p + j) = temp;
			}
		}
	}

	for (int i = 0; i < n; i++) // For loop to print array elements.
	{
		printf("%d ", *(p + i));
	}
	void free(p); // Free the memory which is given to pointer p Dynamically...
}




/*

Que : 3. Write a C program to search given element in 1-D array using binary search
method
              (Use Dynamic Memory Allocation to represent an array).
Owner: Rushikesh Sanjay Pokharkar
Batch: PPA9

*/

//                                          *********  Solution  *********


#include<stdio.h> // Include necessary header files
#include<stdlib.h>


void main() {

	int n; // Declaration of variables.

	int* p = NULL; // Initializing pointer to null value.

	printf("Enter How Many Elements Do You Want In Array : ");
	scanf_s("%d", &n); // Taking input - Number of elements in array.

	p = (int*)malloc(n * sizeof(int)); // Syntax for Dynamic memory allociation of
array.

	printf("Enter Array Elements..\n");
	for (int i = 0; i < n; i++) // For loop to take input array elements.
	{
		scanf_s("%d", p + i);
	}

	printf("Array Elements are: ");
	for (int i = 0; i < n; i++) // For loop to print array elements.
	{
		printf("%d ", *(p + i));
	}

	//                     ..............Logic of Binary Search.................

	// TO use binary search method we need to sort given array.
	// Logic to sort array in ascending order
	for (int i = 0; i < n - 1; i++)
	{
		for (int j = i + 1; j < n; j++)
		{
			if (*(p + i) > *(p + j)) {
				int temp = *(p + i);
				*(p + i) = *(p + j);
				*(p + j) = temp;
			}
		}
	}
	printf("\nArray Elements After Sorting are: ");
	for (int i = 0; i < n; i++) // For loop to print array elements.
	{
		printf("%d ", *(p + i));
	}

	int search, flag = 0;
	printf("\nEnter a Searching Element: ");
	scanf_s("%d", &search);

	int first = 0;
	int last = n-1;
	int mid = n / 2;
label1:
	if (search == *(p + mid)) {
		flag = 1;
		goto label2;
	}
	else if (search > *(p + mid)) {
		first = mid + 1;
		last = n;
		mid = (first + last) / 2;
		if (first > last) {
			goto label2;
		}
		goto label1;
	}
	else if (search < *(p + mid)) {
		first = 0;
		last = mid - 1;
		mid = (first + last) / 2;
		if (first > last) {
			goto label2;
		}
		goto label1;
	}
	else {
		goto label2;
	}
label2:
	if (flag == 1) {
		printf("The Element %d is found...", search);
	}
	else {
		printf("The Element %d is not found...", search);
	}


	free(p); // Free the memory which is given to pointer p Dynamically...
}




/*

Que : 4. Write a C program to find second highest element in given 1-D array using
Dynamic Memory Allocation.
Owner: Rushikesh Sanjay Pokharkar
Batch: PPA9

*/
```

```c
//                                        *********  Solution  *********


#include<stdio.h> // Include necessary header files
#include<stdlib.h>


void main() {

       int n, search, flag = 0; // Declaration of variables.

       int* p = NULL; // Initializing pointer to null value.

       printf("Enter How Many Elements Do You Want In Array : ");
       scanf_s("%d", &n); // Taking input - Number of elements in array.

       p = (int*)malloc(n * sizeof(int)); // Syntax for Dynamic memory allociation of
array.

       printf("Enter Array Elements..\n");
       for (int i = 0; i < n; i++) // For loop to take input array elements.
       {
              scanf_s("%d", p + i);
       }

       printf("Array Elements are: ");
       for (int i = 0; i < n; i++) // For loop to print array elements.
       {
              printf("%d ", *(p + i));
       }

       // Logic to Find Second Highest Element in given array...
       int high = *(p + 0);
       int second_high = *(p + 0);

       for (int i = 0; i < n; i++)
       {
              if (*(p + i) > high) {
                     high = *(p + i);
              }
       }

       for (int i = 0; i < n; i++)
       {
              if (*(p + i) == high) {
                     continue;
              }
              else if (*(p + i) > second_high) {
                     second_high = *(p + i);
              }
       }

       printf("\nThe Second Highest Element In Given Array Is: %d", second_high);

       free(p); // Free the memory which is given to pointer p Dynamically...
}




/*
```

```c
Que : 5. Write a C program to reverse an given 1-D without using sorting algorithms.
(Use Dynamic Memory Allocation to represent an array).
Owner: Rushikesh Sanjay Pokharkar
Batch: PPA9

*/

//                                         *********  Solution  *********


#include<stdio.h> // Include necessary header files
#include<stdlib.h>


void main() {

	int n, search, flag = 0; // Declaration of variables.

	int* p = NULL; // Initializing pointer to null value.

	printf("Enter How Many Elements Do You Want In Array : ");
	scanf_s("%d", &n); // Taking input - Number of elements in array.

	p = (int*)malloc(n * sizeof(int)); // Syntax for Dynamic memory allociation of
array.

	printf("Enter Array Elements..\n");
	for (int i = 0; i < n; i++) // For loop to take input array elements.
	{
		scanf_s("%d", p + i);
	}

	printf("Array Elements are: ");
	for (int i = 0; i < n; i++) // For loop to print array elements.
	{
		printf("%d ", *(p + i));
	}

	printf("\nThe Reversed Array is: ");

	// Logic to reverse the array elements..
	int i = 0, j = n-1;
	while (i <= j) {
		int temp = *(p + i);
		*(p + i) = *(p + j);
		*(p + j) = temp;
		i++, j--;
	}

	for (int i = 0; i < n; i++) // For loop to print array elements.
	{
		printf("%d ", *(p + i));
	}

	free(p); // Free the memory which is given to pointer p Dynamically...
}
```