WEEK 9:

1) What is the worst case complexity of selection sort? **1 point**

- a) O(nlogn)
- b) O(logn)
- c) O(n)
- d) $O(n^2)$

Yes, the answer is correct.
Score: 1
Accepted Answers:
d) $O(n^2)$

2) What is the best case and worst case complexity of ordered linear search? **1 point**

- a) O(nlogn), O(logn)
- b) O(logn), O(nlogn)
- c) O(n), O(1)
- d) O(1), O(n)

Yes, the answer is correct.
Score: 1
Accepted Answers:
d) O(1), O(n)

3) Given an array arr = {12, 34, 47, 62, 85, 92, 95, 99,105} and key = 34; what are the mid values (corresponding array elements) generated in the first and second iterations? **1 point**

- a) 85 and 12
- b) 85 and 34
- c) 62 and 34
- d) 62 and 47

Yes, the answer is correct.
Score: 1
Accepted Answers:
b) 85 and 34

4) When the Binary search is best applied to an array? **1 point**

- a) For very large size array
- b) When the array is sorted
- c) When the array elements are mixed data type
- d) When the array is unsorted

Yes, the answer is correct.
Score: 1
Accepted Answers:
b) When the array is sorted

5) Consider the array A[ ]= {5,4,9,1,3} apply the insertion sort to sort the array. Consider the cost associated with each sort is 25 rupees, what is the total cost of the insertion sort for sorting the entire array? **1 point**

- a) 25
- b) 50
- c) 75
- d) 100

Yes, the answer is correct.
Score: 1

8) Consider an array of elements arr[5]= {5,4,3,2,1}, what are the steps of insertions done while doing insertion sort in the array. *1 point*

a) 4 5 3 2 1
3 4 5 2 1
2 3 4 5 1
1 2 3 4 5

b) 5 4 3 1 2
5 4 1 2 3
5 1 2 3 4
1 2 3 4 5

c) 4 3 2 1 5
3 2 1 5 4
2 1 5 4 3
1 5 4 3 2

d) 4 5 3 2 1
2 3 4 5 1
3 4 5 2 1
1 2 3 4 5

Yes, the answer is correct.
Score: 1

Accepted Answers:

a) 4 5 3 2 1
3 4 5 2 1
2 3 4 5 1
1 2 3 4 5

9) *1 point*

What will be the output of the following C code?

```
#include <stdio.h>
#if A == 1
   #define B 0
#else
   #define B 1
#endif
int main()
{
   printf("%d", B);
   return 0;
}
```

a) 0
b) 1
c) 01
d) None of the above

Yes, the answer is correct.
Score: 1

Accepted Answers:
b) 1

10) *1 point*

What will be the output?

```
#include <stdio.h>
#define a 10
int main()
{
 printf("%d ", a);
 int a=50;
 printf("%d ", a);
 return 0;
}
```

a) 10 10
b) 10 50
c) 50 50
d) Compilation error

Yes, the answer is correct.

Write a program to print all the locations at which a particular element (taken as input) is found in a list and also print the total number of times it occurs in the list. The location starts from 1.
For example if there are 4 elements in the array
5
6
5
7
If the element to search is 5 then the output will be
5 is present at location 1
5 is present at location 3
5 is present 2 times in the array.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	7 30 50 90 30 70 30 30 30	30 is present at location 1.\n 30 is present at location 4.\n 30 is present at location 6.\n 30 is present at location 7.\n 30 is present 4 times in the array.	30 is present at location 1.\n 30 is present at location 4.\n 30 is present at location 6.\n 30 is present at location 7.\n 30 is present 4 times in the array.\n	Passed
Test Case 2	4 50 60 20 10 80	80 is not present in the array.	80 is not present in the array.\n	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2023-09-27, 05:42 IST
Your last recorded submission was :

```c
#include <stdio.h>
int main()
{
   int array[100], search, n, count = 0;
   //"search" is the key element to search and 'n' is the total number of element of the array
   // "count" is to store total number of elements

 scanf("%d", &n); //Number of elements is taken from test case

 int c;
   for (c = 0; c < n; c++)
      scanf("%d", &array[c]);

   scanf("%d", &search); // The element to search is taken from test case

/* Use the printf statements as below:
"%d is present at location %d.\n"  for each locations
"%d is not present in the array.\n" if the element is not found in the list
"%d is present %d times in the array.\n"
*/
for(int i=0;i<n;i++)
{
  if(array[i]==search)
  {
    count++;
    printf("%d is present at location %d.\n",search,i+1);
  }
}
if(count)
{
  printf("%d is present %d times in the array.\n",search,count);
}
else
{
  printf("%d is not present in the array.\n",search);
}


}
```

Due on 2023-09-28, 23:59 IST

Write a C program to search a given element from a 1D array and display the position at which it is found by using linear search function.
The index location starts from 1.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	4 45 65 85 25 95	95 is not present in the array.	95 is not present in the array.	Passed
Test Case 2	5 6 9 5 4 7 6	6 is present at location 1.	6 is present at location 1.	Passed

The due date for submitting this assignment has passed.

2 out of 2 tests passed.

You scored 100.0/100.

Assignment submitted on 2023-09-27, 09:54 IST

Your last recorded submission was :

```c
#include <stdio.h>
int linear_search(int[], int, int);
int main()
{
   int array[100], search, c, n, position;
   /* search - element to search, c - counter, n - number of elements in array,
   position - The position in which the element is first found in the list. */

    scanf("%d", &n); // Number of elements in the array is read from the test case data

    for (c = 0; c < n; c++)
    scanf("%d", &array[c]); //Elements of array is read from the test data

    scanf("%d", &search);  //Element to search is read from the test case data

   /* Use the following in the printf statement to print the output
   printf("%d is not present in the array.", search);
   printf("%d is present at location %d.", search, position+1); //As array[0] has the position 1
   */
int f=0;
for(int position=0;position<n;position++)
{
  if(search==array[position])
  {
     printf("%d is present at location %d.", search, position+1);
    f=1;
  }
}
if(!f)
{
  printf("%d is not present in the array.", search);
}
}
```

Write a C program to search a given number from a sorted 1D array and display the position at which it is found using binary search algorithm. The index location starts from 1.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	6 1 2 3 4 5 6 2	2 found at location 2.	2 found at location 2.	Passed
Test Case 2	7 40 50 60 70 80 90 100 100	100 found at location 7.	100 found at location 7.	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2023-09-28, 09:59 IST
Your last recorded submission was :

```c
#include <stdio.h>
int main()
{
 int c, n, search,
 array[100];
 scanf("%d",&n); //number of elements in the array

 for (c = 0; c < n; c++)
 scanf("%d",&array[c]);

 scanf("%d", &search); //The element to search is read from test case.

/* Use the printf statements as below:
 printf("%d found at location %d.", search, variable_name);
 printf("Not found! %d isn't present in the list.", search);
*/
int f=0;

for(int i=0;i<n;i++)
{
  if(array[i]==search){
      printf("%d found at location %d.", search, i+1);
      f=1;
  }
}
if(!f)
{
 printf("Not found! %d isn't present in the list.", search);
}
}
```

Write a C program to reverse an array by swapping the elements and without using any new array.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	7 8 9 10 6 4 7 11	Reversed array elements are:\n 11\n 7\n 4\n 6\n 10\n 9\n 8	Reversed array elements are:\n 11\n 7\n 4\n 6\n 10\n 9\n 8\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2023-09-28, 10:00 IST
Your last recorded submission was :

```c
#include <stdio.h>
int main() {
  int array[100], n, c;
  scanf("%d", &n); // n is number of elements in the array.
  for (c = 0; c < n; c++) {
    scanf("%d", &array[c]);
  }
int temp;
for(int i=0,j=n-1;i<n/2;i++,j--)
{
  temp=array[i];
  array[i]=array[j];
  array[j]=temp;

}
printf("Reversed array elements are:\n");

  for (c = 0; c < n; c++) {
    printf("%d\n", array[c]);
  }
  return 0;
}
```