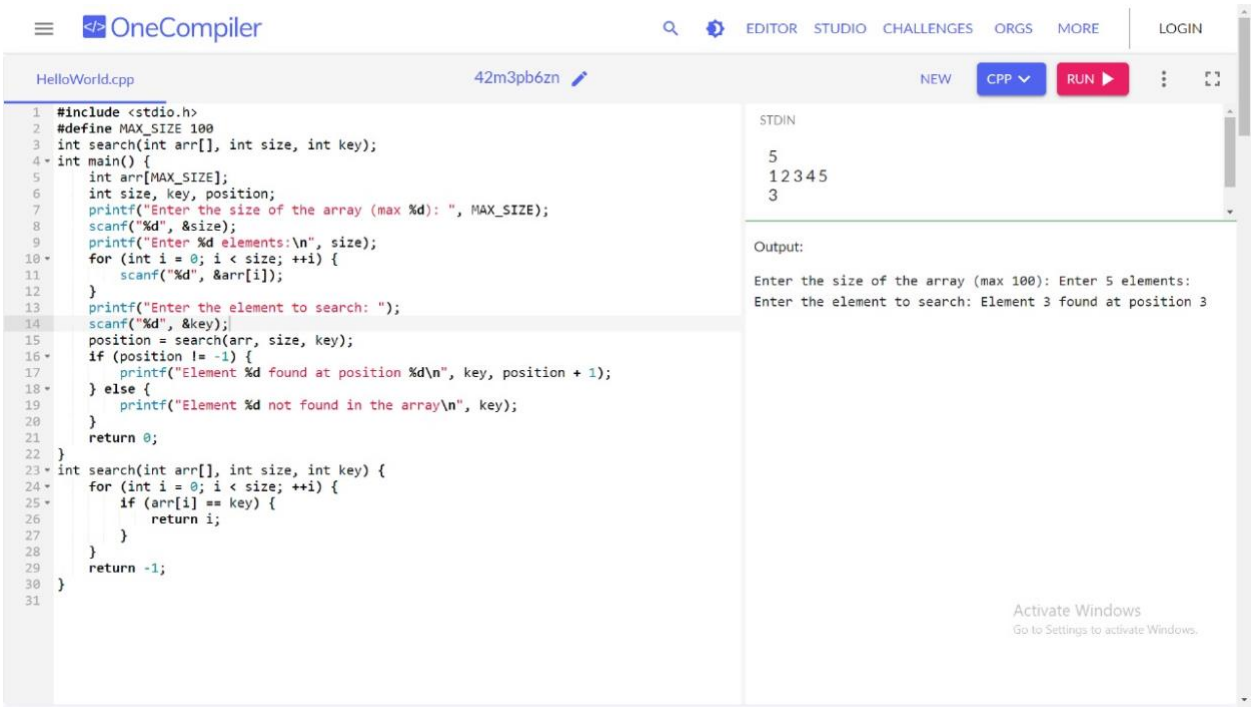
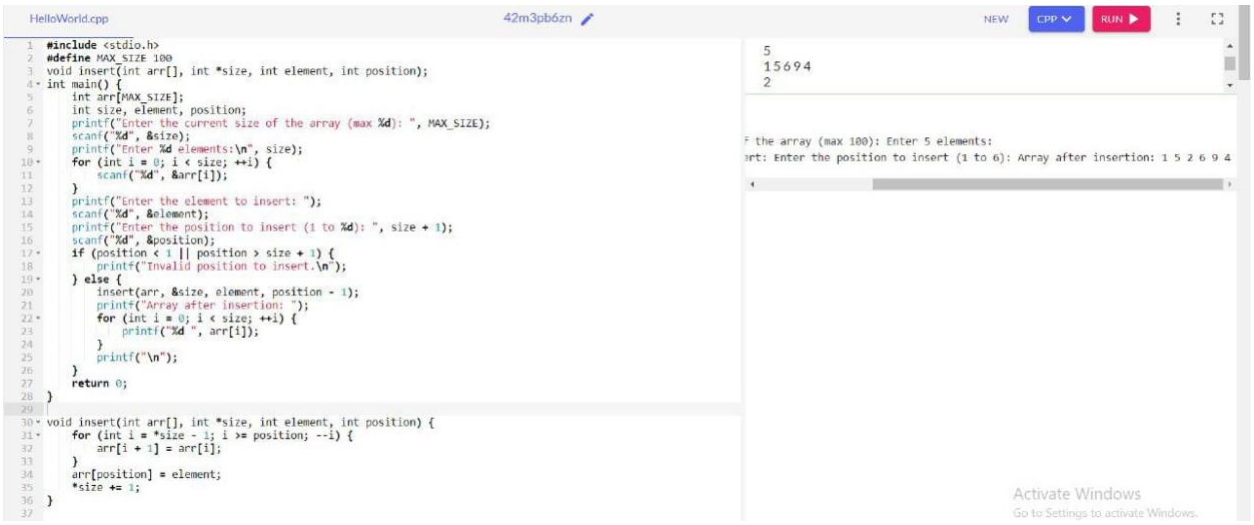
1. Write a C Program to implement following operations

a) traverse

b) search



c) insert


HelloWorld.cpp
42m3pb6zn
NEW
CPP
RUN
#include <stdio.h>
#define MAX_SIZE 100
void insert(int arr[], int *size, int element, int position);
int main() {
int arr[MAX_SIZE];
int size, element, position;
printf("Enter the current size of the array (max %d): ", MAX_SIZE);
scanf("%d", &size);
printf("Enter %d elements:\n", size);
for (int i = 0; i < size; ++i) {
scanf("%d", &arr[i]);
}
printf("Enter the element to insert: ");
scanf("%d", &element);
printf("Enter the position to insert (1 to %d): ", size + 1);
scanf("%d", &position);
if (position < 1 || position > size + 1) {
printf("Invalid position to insert.\n");
} else {
insert(arr, &size, element, position - 1);
printf("Array after insertion: ");
for (int i = 0; i < size; ++i) {
printf("%d ", arr[i]);
}
printf("\n");
}
return 0;
}
void insert(int arr[], int *size, int element, int position) {
for (int i = *size - 1; i >= position; --i) {
arr[i + 1] = arr[i];
}
arr[position] = element;
*size += 1;
}
5
15694
2
the array (max 100): Enter 5 elements:
rt: Enter the position to insert (1 to 6): Array after insertion: 1 5 2 6 9 4
Activate Windows
Go to Settings to activate Windows.

d) delete


HelloWorld.cpp
42m3pb6zn
NEW
CPP
RUN
#include <stdio.h>
#define MAX_SIZE 100
int main() {
int array[MAX_SIZE];
int size, i, pos;
printf("Enter size of the array: ");
scanf("%d", &size);
printf("Enter elements of the array:\n");
for (i = 0; i < size; i++) {
scanf("%d", &array[i]);
}
printf("Enter the position of the element to delete (0-indexed): ");
scanf("%d", &pos);
if (pos < 0 || pos >= size) {
printf("Invalid position!\n");
} else {
for (i = pos; i < size - 1; i++) {
array[i] = array[i + 1];
}
size--;
printf("Array after deletion:\n");
for (i = 0; i < size; i++) {
printf("%d ", array[i]);
}
printf("\n");
}
return 0;
}
5
10 20 30 40 50
2
Output:
Enter size of the array: Enter elements of the array:
Enter the position of the element to delete (0-indexed): Array
10 20 40 50
Activate Windows

e)update

HelloWorld.cpp 42m3pb6zn NEW CPP RUN

```c
#include <stdio.h>
#define MAX_SIZE 100
int main() {
    int array[MAX_SIZE];
    int size, i, pos, new_value;
printf("Enter size of the array: ");
    scanf("%d", &size);
printf("Enter elements of the array:\n");
    for (i = 0; i < size; i++) {
        scanf("%d", &array[i]);
    }
 printf("Enter the position of the element to update (0-indexed): ");
    scanf("%d", &pos);
if (pos < 0 || pos >= size) {
        printf("Invalid position!\n");
    } else {
        printf("Enter the new value: ");
        scanf("%d", &new_value);
array[pos] = new_value;
printf("Array after updating:\n");
        for (i = 0; i < size; i++) {
            printf("%d ", array[i]);
        }
        printf("\n");
    }

    return 0;
}
```

```
5
35789
4
```

Output:

```
Enter size of the array: Enter elements of the array:
Enter the position of the element to update (0-indexed): Enter
3 5 7 8 0
```

2. Writing a recursive function to calculate the factorial of a number.

main.c Share Run Output

```c
#include <stdio.h>
unsigned long long factorial(int n) {
    if (n == 0) {
        return 1;
    } else {
        return n * factorial(n - 1);
    }
}
int main() {
    int num;
    unsigned long long fact;
    printf("Enter a non-negative integer: ");
    scanf("%d", &num);
    fact = factorial(num);
    printf("Factorial of %d = %llu\n", num, fact);
    return 0;
}
```

```
/tmp/tkvZrHe1kX.o
Enter a non-negative integer: 5
Factorial of 5 = 120


=== Code Execution Successful ===
```

3. Write a C Program to find duplicate element in an array

main.c

Share Run Output

```c
#include <stdio.h>
int main() {
    int arr[] = {1, 2, 3, 4, 2, 7, 8, 8, 3};
    int n = sizeof(arr) / sizeof(arr[0]);
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] == arr[j]) {
                printf("Duplicate element: %d\n", arr[j]);
            }
        }
    }
    return 0;
}
```

```
/tmp/BOnWQqg52G.o
Duplicate element: 2
Duplicate element: 3
Duplicate element: 8


=== Code Execution Successful ===
```

4. Write a C Program to find Max and Min from an array elements

main.c

Share Run Output

```c
#include <stdio.h>
int main() {
    int arr[] = {3, 9, 2, 8, 5, 1};
    int n = sizeof(arr) / sizeof(arr[0]);
    int max = arr[0], min = arr[0];
    for (int i = 1; i < n; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
        if (arr[i] < min) {
            min = arr[i];
        }
    }
    printf("Maximum element in the array: %d\n", max);
    printf("Minimum element in the array: %d\n", min);
    return 0;
}
```

```
/tmp/zyfPmqff7d.o
Maximum element in the array: 9
Minimum element in the array: 1


=== Code Execution Successful ===
```

5. Given a number n. the task is to print the Fibonacci series and the sum of the series using

recursion.

input: n=10

output: Fibonacci series

0, 1, 1, 2, 3, 5, 8, 13, 21, 34

Sum: 88

main.c

Share | Run | Output | Clear

```c
#include <stdio.h>
int main() {
    int a = 0, b = 1, next, n = 10;
    for (int i = 0; i < n; i++) {
        printf("%d ", a);
        next = a + b;
        a = b;
        b = next;
    }
    printf("\n");
    return 0;
}
```

```
/tmp/MqpXn4Ya4N.o
0 1 1 2 3 5 8 13 21 34


=== Code Execution Successful ===
```

6. You are given an array arr in increasing order. Find the element x from arr using binary

search.

Example 1: arr={ 1,5,6,7,9,10},X=6

Output : Element found at location 2

Example 2: arr={ 1,5,6,7,9,10},X=11

Output : Element not found at location 2

HelloWorld.cpp | 42m3pb6zn | NEW | CPP | RUN

```c
#include <stdio.h>
int binarySearch(int arr[], int left, int right, int x);
int main() {
    int arr[] = {1, 5, 6, 7, 9, 10};
    int n = sizeof(arr) / sizeof(arr[0]);
    int x;

    printf("Enter the element to search: ");
    scanf("%d", &x);

    int result = binarySearch(arr, 0, n - 1, x);

    if (result == -1) {
        printf("Element not found\n");
    } else {
        printf("Element found at location %d\n", result + 1); // +1 because result is
    }
     return 0;
}
int binarySearch(int arr[], int left, int right, int x) {
    while (left <= right) {
        int mid = left + (right - left) / 2;
if (arr[mid] == x) {
            return mid;
        }
        if (arr[mid] < x) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1;
```

STDIN

```
5
```

Output:

```
Enter the element to search: Element found at location 2
```

6. You are given an array arr in increasing order. Find the element x from arr using linear search.

Example 1: arr={ 1,5,6,7,9,10},X=6

Output : Element found at location 2

Example 2: arr={ 1,5,6,7,9,10},X=11

Output : Element not found at location 2