```
#include <stdio.h>
1
     void printArray(int* arr, int n)
 2
 3
4
         int i;
 5
         printf("Array: ");
         for (i = 0; i < n; i++) {
6
              printf("%d ", arr[i]);
7
 8
          printf("\n");
 9
10
     int main()
11
12
     1
         int arr[] = { 2, -1, 5, 6, 0, -3 };
13
          int n = sizeof(arr) / sizeof(arr[0]);
14
15
         printArray(arr, n);
16
         return 0;
17
18
19
```

```
[ ] onuuea i
     #include <stdio.h>
1
     int findElement(int arr[], int n, int key)
2
 3 -
 4
          int i;
          for (i = 0; i < n; i++)
5
              if (arr[i] == key)
 6
 7
                  return i;
 8
          return -1;
9
      int main()
10
11
     £
          int arr[] = { 12, 34, 10, 6, 40 };
12
13
          int n = sizeof(arr) / sizeof(arr[0]);
          int key = 40;
14
          int position = findElement(arr, n, key);
15
          if (position == -1)
16
17
              printf("Element not found");
          else
18
19
              printf("Element Found at Position: %d",
20
                     position + 1)
21
          return 0;
22
23
24
```

```
#include <stdio.h>
 1
     int main()
 2
 3 -
     1
         int arr[100] = { 0 };
 4
 5
          int i, x, pos, n = 10;
 6
          for (i = 0; i < 10; i++)
              arr[i] = i + 1;
 7
          for (i = 0; i < n; i++)
 8
              printf("%d ", arr[i]);
9
         printf("\n");
10
11
         x = 50;
12
         pos = 5;
13
          n++;
         for (i = n - 1; i > pos; i--)
14
              arr[i] = arr[i - 1];
15
16
         arr[pos - 1] = x;
         for (i = 0; i < n; i++)
17
              printf("%d ", arr[i]);
18
         printf("\n");
19
20
         return 0;
21
22
```

```
[^] Untitled I
     #include <stdio.h>
2
     int main()
 3 □ {
          if (remove("abc.txt") == 0)
 4
             printf("Deleted successfully");
5
6
          else
7
              printf("Unable to delete the file");
8
         return 0;
9
10
```

```
[*] Untitled]

1  #include <stdio.h>
2  int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    arr[2] = 10;
5  for (int i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}</pre>
```

```
#include <stdio.h>
2
     unsigned int factorial(unsigned int n)
3 🖃
     1
         if (n == 0)
4
5
             return 1;
         return n * factorial(n - 1);
6
7
8
     int main()
9
         int num = 5;
10
         printf("Factorial of %d is %d", num, factorial(num));
11
12
         return 0;
13
14
```

```
[*] Untitled1
     #include <stdio.h>
 1
 2
     int main()
 3
     1
          int arr[] = \{1, 2, 3, 4, 2, 7, 8, 8, 3\};
 4
 5
          int length = sizeof(arr)/sizeof(arr[0]);
 6
          printf("Duplicate elements in given array: \n");
 7
          for(int i = 0; i < length; i++) {
              for(int j = i + 1; j < length; j++) {
 8
                  if(arr[i] == arr[j])
 9
                      printf("%d\n", arr[j]);
10
11
12
13
          return 0;
14
```

```
[*] Untitled1
     #include <limits.h>
     #include <stdio.h>
2
     void findMinimumMaximum(int arr[], int N)
3
4 🗔 {
5
         int i;
6
         int minE = INT MAX, maxE = INT MIN;
         for (i = 0; i < N; i++) {
7
         if (arr[i] < minE) {
8
9
                  minE = arr[i];
10
             if (arr[i] > maxE) {
11
12
                  maxE = arr[i];
13
14
         printf("The minimum element is %d", minE);
15
         printf("\n");
16
17
          printf("The maximum element is %d", maxE);
18
19
         return;
20
     int main()
21
22 -
     1
         int arr[] = { 1, 2, 4, -1 };
23
         int N = sizeof(arr) / sizeof(arr[0]);
24
         findMinimumMaximum(arr, N);
25
26
         return 0;
```

27

28

```
[*] Untitled1
     #include <stdio.h>
2 int fibonacci(int n) {
 3
         if (n <= 1)
4
             return n;
 5
         else
 6
             return fibonacci(n-1) + fibonacci(n-2);
 7
8 - long long fibonacci_sum(int n) {
9
         if (n <= 0)
10
             return 0;
         return fibonacci(n) + fibonacci_sum(n-1);
11
12
13 ☐ int main() {
14
         int n;
15
         printf("Enter the number of terms in Fibonacci series: ");
         scanf("%d", &n);
16
17
         printf("Fibonacci Series up to %d terms:\n", n);
18 -
         for (int i = 0; i < n; ++i) {
             printf("%d ", fibonacci(i));
19
20
         printf("\n");
21
22
         long long sum = fibonacci_sum(n);
         printf("Sum of Fibonacci Series up to %d terms: %lld\n", n, sum);
23
24
         return 0;
25
26
```

```
[*] Untitled1
     #include <stdio.h>
2 int binarySearch(int arr[], int left, int right, int x) {
         while (left <= right) {
 3 -
4
             int mid = left + (right - left) / 2;
 5
             if (arr[mid] == x)
 6
                 return mid;
 7
             if (arr[mid] < x)
 8
                 left = mid + 1;
9
             else
10
                 right = mid - 1;
11
12
         return -1;
13
15
         int i, key, j;
16 -
         for (i = 1; i < n; i++) {
17
             key = arr[i];
18
             j = i - 1;
19 -
             while (j >= 0 && arr[j] > key) {
20
                 arr[j + 1] = arr[j];
21
                 j = j - 1;
22
23
             arr[j + 1] = key;
24
25
26 = int main() {
27
         int n, i, x;
         printf("Enter the number of elements in the array: ");
28
         scanf("%d", &n);
29
30
         int arr[n];
         printf("Enter %d elements:\n", n);
31
32 🖃
         for (i = 0; i < n; i++) {
             scanf("%d", &arr[i]);
33
34
35
         insertionSort(arr, n);
36
         printf("Sorted array in increasing order:\n");
27 🗀
         for /i - 0 · i / n · i .. \ /
```

```
printf("Sorted array in increasing order:\n");
for (i = 0; i < n; i++) {
    printf("%d ", arr[i]);
}
printf("\n");
printf("Enter the element to search: ");
scanf("%d", &x);
int result = binarySearch(arr, 0, n - 1, x);
if (result != -1) {
    printf("Element %d found at index %d.\n", x, result);
} else {
    printf("Element %d not found in the array.\n", x);
}
return 0;</pre>
```

36 | 37 |=

44 -

```
#include <stdio.h>
1
     void printArray(int* arr, int n)
 2
 3
4
         int i;
 5
         printf("Array: ");
         for (i = 0; i < n; i++) {
6
              printf("%d ", arr[i]);
7
 8
          printf("\n");
 9
10
     int main()
11
12
     1
         int arr[] = { 2, -1, 5, 6, 0, -3 };
13
          int n = sizeof(arr) / sizeof(arr[0]);
14
15
         printArray(arr, n);
16
         return 0;
17
18
19
```

```
[ ] onuuea i
     #include <stdio.h>
1
     int findElement(int arr[], int n, int key)
2
 3 -
 4
          int i;
          for (i = 0; i < n; i++)
5
              if (arr[i] == key)
 6
 7
                  return i;
 8
          return -1;
9
      int main()
10
11
     £
          int arr[] = { 12, 34, 10, 6, 40 };
12
13
          int n = sizeof(arr) / sizeof(arr[0]);
          int key = 40;
14
          int position = findElement(arr, n, key);
15
          if (position == -1)
16
17
              printf("Element not found");
          else
18
19
              printf("Element Found at Position: %d",
20
                     position + 1)
21
          return 0;
22
23
24
```

```
#include <stdio.h>
 1
     int main()
 2
 3 -
     1
         int arr[100] = { 0 };
 4
 5
          int i, x, pos, n = 10;
 6
          for (i = 0; i < 10; i++)
              arr[i] = i + 1;
 7
          for (i = 0; i < n; i++)
 8
              printf("%d ", arr[i]);
9
         printf("\n");
10
11
         x = 50;
12
         pos = 5;
13
          n++;
         for (i = n - 1; i > pos; i--)
14
              arr[i] = arr[i - 1];
15
16
         arr[pos - 1] = x;
         for (i = 0; i < n; i++)
17
              printf("%d ", arr[i]);
18
         printf("\n");
19
20
         return 0;
21
22
```

```
[^] Untitled I
     #include <stdio.h>
2
     int main()
 3 □ {
          if (remove("abc.txt") == 0)
 4
             printf("Deleted successfully");
5
6
          else
7
              printf("Unable to delete the file");
8
         return 0;
9
10
```

```
[*] Untitled]

1  #include <stdio.h>
2  int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    arr[2] = 10;
5  for (int i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}</pre>
```

```
#include <stdio.h>
2
     unsigned int factorial(unsigned int n)
3 🖃
     1
         if (n == 0)
4
5
             return 1;
         return n * factorial(n - 1);
6
7
8
     int main()
9
         int num = 5;
10
         printf("Factorial of %d is %d", num, factorial(num));
11
12
         return 0;
13
14
```

```
[*] Untitled1
     #include <stdio.h>
 1
 2
     int main()
 3
     1
          int arr[] = \{1, 2, 3, 4, 2, 7, 8, 8, 3\};
 4
 5
          int length = sizeof(arr)/sizeof(arr[0]);
 6
          printf("Duplicate elements in given array: \n");
 7
          for(int i = 0; i < length; i++) {
              for(int j = i + 1; j < length; j++) {
 8
                  if(arr[i] == arr[j])
 9
                      printf("%d\n", arr[j]);
10
11
12
13
          return 0;
14
```

```
[*] Untitled1
     #include <limits.h>
     #include <stdio.h>
2
     void findMinimumMaximum(int arr[], int N)
3
4 🗔 {
5
         int i;
6
         int minE = INT MAX, maxE = INT MIN;
         for (i = 0; i < N; i++) {
7
         if (arr[i] < minE) {
8
9
                  minE = arr[i];
10
             if (arr[i] > maxE) {
11
12
                  maxE = arr[i];
13
14
         printf("The minimum element is %d", minE);
15
         printf("\n");
16
17
          printf("The maximum element is %d", maxE);
18
19
         return;
20
     int main()
21
22 -
     1
         int arr[] = { 1, 2, 4, -1 };
23
         int N = sizeof(arr) / sizeof(arr[0]);
24
         findMinimumMaximum(arr, N);
25
26
         return 0;
```

27

28

```
[*] Untitled1
     #include <stdio.h>
2 int fibonacci(int n) {
 3
         if (n <= 1)
4
             return n;
 5
         else
 6
             return fibonacci(n-1) + fibonacci(n-2);
 7
8 - long long fibonacci_sum(int n) {
9
         if (n <= 0)
10
             return 0;
         return fibonacci(n) + fibonacci_sum(n-1);
11
12
13 ☐ int main() {
14
         int n;
15
         printf("Enter the number of terms in Fibonacci series: ");
         scanf("%d", &n);
16
17
         printf("Fibonacci Series up to %d terms:\n", n);
18 -
         for (int i = 0; i < n; ++i) {
             printf("%d ", fibonacci(i));
19
20
         printf("\n");
21
22
         long long sum = fibonacci_sum(n);
         printf("Sum of Fibonacci Series up to %d terms: %lld\n", n, sum);
23
24
         return 0;
25
26
```

```
[*] Untitled1
     #include <stdio.h>
2 int binarySearch(int arr[], int left, int right, int x) {
         while (left <= right) {
 3 -
4
             int mid = left + (right - left) / 2;
 5
             if (arr[mid] == x)
 6
                 return mid;
 7
             if (arr[mid] < x)
 8
                 left = mid + 1;
9
             else
10
                 right = mid - 1;
11
12
         return -1;
13
15
         int i, key, j;
16 -
         for (i = 1; i < n; i++) {
17
             key = arr[i];
18
             j = i - 1;
19 -
             while (j >= 0 && arr[j] > key) {
20
                 arr[j + 1] = arr[j];
21
                 j = j - 1;
22
23
             arr[j + 1] = key;
24
25
26 = int main() {
27
         int n, i, x;
         printf("Enter the number of elements in the array: ");
28
         scanf("%d", &n);
29
30
         int arr[n];
         printf("Enter %d elements:\n", n);
31
32 🖃
         for (i = 0; i < n; i++) {
             scanf("%d", &arr[i]);
33
34
35
         insertionSort(arr, n);
36
         printf("Sorted array in increasing order:\n");
27 🗀
         for /i - 0 · i / n · i .. \ /
```

```
printf("Sorted array in increasing order:\n");
for (i = 0; i < n; i++) {
    printf("%d ", arr[i]);
}
printf("\n");
printf("Enter the element to search: ");
scanf("%d", &x);
int result = binarySearch(arr, 0, n - 1, x);
if (result != -1) {
    printf("Element %d found at index %d.\n", x, result);
} else {
    printf("Element %d not found in the array.\n", x);
}
return 0;</pre>
```

36 | 37 |=

44 -

```
[*] Untitled1
     #include <stdio.h>
     int linearSearch(int* arr, int size, int key)
 2
3 🖵
 4 -
          for (int i = 0; i < size; i++) {
 5
              if (arr[i] == key) {
                  return i;
 6
 7
8
 9
          return -1;
10
11
     int main()
12 - {
13
          int arr[10] = { 3, 4, 1, 7, 5, 8, 11, 42, 3, 13 };
          int size = sizeof(arr) / sizeof(arr[0]);
14
          int key = 4;
15
          int index = linearSearch(arr, size, key);
16
17 -
          if (index == -1) {
18
              printf("The element is not present in the arr.");
19
20 -
          else {
             printf("The element is present at arr[%d].", index);
21
22
23
          return 0;
24
25
```