SEARCHING

LINEAR SEARCH:

```
#include <stdio.h>

int linearSearch(int arr[], int n, int key) {
    for (int i = 0; i < n; i++) {
        if (arr[i] == key) {
            return i;
        }
    }
    return -1; // Element not found
}

int main() {
    int arr[100], n, key, index;

    printf("Enter the number of elements: ");
    scanf("%d", &n);

    printf("Enter elements:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

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

    index = linearSearch(arr, n, key);

    if (index != -1) {
        printf("Element found at index %d\n", index);
    } else {
        printf("Element not found\n");
    }

    return 0;
}
```

BINARY SEARCH

```
#include <stdio.h>

int binarySearch(int arr[], int low, int high, int key) {
    while (low <= high) {
        int mid = low + (high - low) / 2;

        if (arr[mid] == key) {
            return mid;
        } else if (arr[mid] < key) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }
```

```
    return -1; // Element not found
}

int main() {
    int arr[100], n, key, index;

    printf("Enter the number of elements: ");
    scanf("%d", &n);

    printf("Enter elements in sorted order:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

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

    index = binarySearch(arr, 0, n - 1, key);

    if (index != -1) {
        printf("Element found at index %d\n", index);
    } else {
        printf("Element not found\n");
    }

    return 0;
}
```

FIBONACCI SEARCH

```
#include <stdio.h>

int fibonacciSearch(int arr[], int n, int key) {
    int fib2 = 0; // (m-2)th Fibonacci number
    int fib1 = 1; // (m-1)th Fibonacci number
    int fib = fib2 + fib1; // mth Fibonacci number

    while (fib < n) {
        fib2 = fib1;
        fib1 = fib;
        fib = fib2 + fib1;
    }

    int offset = -1;

    while (fib > 1) {
        int i = (offset + fib2 < n - 1) ? offset + fib2 : n - 1;

        if (arr[i] == key) {
            return i;
        } else if (arr[i] < key) {
            fib = fib1;
            fib1 = fib2;
            fib2 = fib - fib1;
```

```
            offset = i;
        } else {
            fib = fib2;
            fib1 = fib1 - fib2;
            fib2 = fib - fib1;
        }
    }

    if (fib1 && arr[offset + 1] == key) {
        return offset + 1;
    }

    return -1; // Element not found
}

int main
```

INTERPOLATION SEARCH

```
#include <stdio.h>

int interpolationSearch(int arr[], int n, int key) {
    int low = 0;
    int high = n - 1;

    while (low <= high && key >= arr[low] && key <= arr[high]) {
        if (low == high) {
            if (arr[low] == key) {
                return low;
            }
            return -1; // Element not found
        }

        int pos = low + ((double)(key - arr[low]) / (arr[high] - arr[low])) * (high - low);

        if (arr[pos] == key) {
            return pos;
        } else if (arr[pos] < key) {
            low = pos + 1;
        } else {
            high = pos - 1;
        }
    }

    return -1; // Element not found
}

int main() {
    int arr[100], n, key, index;

    printf("Enter the number of elements: ");
    scanf("%d", &n);

    printf("Enter elements in sorted order:\n");
```

```
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

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

    index = interpolationSearch(arr, n, key);

    if (index != -1) {
        printf("Element found at index %d\n", index);
    } else {
        printf("Element not found\n");
    }

    return 0;
}
```

EXPONENTIAL SEARCH

```
#include <stdio.h>

int binarySearch(int arr[], int low, int high, int key) {
    while (low <= high) {
        int mid = low + (high - low) / 2;

        if (arr[mid] == key) {
            return mid;
        } else if (arr[mid] < key) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }
    return -1; // Element not found
}

int exponentialSearch(int arr[], int n, int key) {
    if (arr[0] == key) {
        return 0;
    }

    int i = 1;
    while (i < n && arr[i] <= key) {
        i *= 2;
    }

    return binarySearch(arr, i / 2, (i < n) ? i : n - 1, key);
}

int main() {
    int arr[100], n, key, index;

    printf("Enter the number of elements: ");
```

```
scanf("%d", &n);

printf("Enter elements in sorted order:\n");
for (int i = 0; i < n; i++) {
    scanf("%d", &arr[i]);
}

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

index = exponentialSearch(arr, n, key);

if (index != -1) {
    printf("Element found at index %d\n", index);
} else {
    printf("Element not found\n");
}
    return 0;
}
```

TERNARY SEARCH

```
#include <stdio.h>

int ternarySearch(int arr[], int left, int right, int key) {
    if (right >= left) {
        int mid1 = left + (right - left) / 3;
        int mid2 = right - (right - left) / 3;

        if (arr[mid1] == key) {
            return mid1;
        }
        if (arr[mid2] == key) {
            return mid2;
        }

        if (key < arr[mid1]) {
            return ternarySearch(arr, left, mid1 - 1, key);
        } else if (key > arr[mid2]) {
            return ternarySearch(arr, mid2 + 1, right, key);
        } else {
            return ternarySearch(arr, mid1 + 1, mid2 - 1, key);
        }
    }
    return -1; // Element not found
}

int main() {
    int arr[100], n, key, index;

    printf("Enter the number of elements: ");
    scanf("%d", &n);

    printf("Enter elements in sorted order:\n");
```

```
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

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

    index = ternarySearch(arr, 0, n - 1, key);

    if (index != -1) {
        printf("Element found at index %d\n", index);
    } else {
        printf("Element not found\n");
    }

    return 0;
}
```