1

Algo.cpp

```cpp
#include<iostream>
using namespace std;
void main(){
    void printNumbers(int n) {
    for (int i = 0; i < n; i++) {
        cout << i << " ";
    }
}
}

```

Time Complexity: _____

2

Algo.cpp

```cpp
#include<iostream>
using namespace std;
void main(){
    void printPairs(int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cout << "(" << i << "," << j << ")" << endl;
        }
    }
}

```

Time Complexity: _____

3

Algo.cpp

```cpp
#include<iostream>
using namespace std;
void main(){
    void printTwoLoops(int n) {
    for (int i = 0; i < n; i++) cout << i << " ";
    for (int j = 0; j < n; j++) cout << j << " ";
    }
}

```

Time Complexity: _____

4

[*] Algo.cpp

```cpp
#include<iostream>
using namespace std;
void main(){
    void halveTillZero(int n) {
    while (n > 0) {
        cout << n << " ";
        n /= 2;
        }
    }
}
```

Time Complexity: _____

5

Algo.cpp

```cpp
#include<iostream>
using namespace std;
void main(){
    void multiplyTillN(int n) {
    int i = 1;
    while (i < n) {
        cout << i << " ";
        i *= 2;
        }
    }
}
```

Time Complexity: _____

6

Algo.cpp

```cpp
#include<iostream>
using namespace std;
void main(){
    void sumOfNumbers(int n) {
    int sum = 0;
    for (int i = 1; i <= n; i++) {
        sum += i;
    }
    cout << "Sum: " << sum << endl;
}
}
```

Time Complexity: _____

7

Algo.cpp

```cpp
#include<iostream>
using namespace std;
void main(){
    void nestedLoopsExample(int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            for (int k = 0; k < n; k++) {
                cout << i << " " << j << " " << k << endl;
            }
        }
    }
}

}
```

Time Complexity: _____

8

Algo.cpp

```cpp
#include <iostream>
using namespace std;

void printTwoArrays(int arr1[], int n, int arr2[], int m) {
    for (int i = 0; i < n; i++) {  // Loop for first array
        cout << arr1[i] << " ";
    }
    for (int j = 0; j < m; j++) {  // Loop for second array
        cout << arr2[j] << " ";
    }
}

int main() {
    int arr1[] = {1, 2, 3};
    int arr2[] = {4, 5, 6, 7};
    printTwoArrays(arr1, 3, arr2, 4);  // Example with n = 3, m = 4
    return 0;
}
```

Time Complexity: _____

9

```
using namespace std;

void merge(int arr[], int l, int m, int r) {
    int n1 = m - l + 1, n2 = r - m;
    int left[n1], right[n2];

    for (int i = 0; i < n1; i++)
        left[i] = arr[l + i];
    for (int i = 0; i < n2; i++)
        right[i] = arr[m + 1 + i];

    int i = 0, j = 0, k = l;
    while (i < n1 && j < n2) {
        if (left[i] <= right[j]) arr[k++] = left[i++];
        else arr[k++] = right[j++];
    }

    while (i < n1) arr[k++] = left[i++];
    while (j < n2) arr[k++] = right[j++];
}

void mergeSort(int arr[], int l, int r) {
    if (l < r) {
        int m = (l + r) / 2;
        mergeSort(arr, l, m);
        mergeSort(arr, m + 1, r);
        merge(arr, l, m, r);
    }
}

int main() {
    int arr[] = {12, 11, 13, 5, 6, 7};
    mergeSort(arr, 0, 5);
    return 0;
}
```

Time Complexity: _____

10

[*] Untitled1

```cpp
#include <iostream>
using namespace std;

int binarySearch(int arr[], int n, int key) {
    int left = 0, right = n - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (arr[mid] == key) return mid;  // O(1) operation
        if (arr[mid] < key) left = mid + 1; // Adjust left bound
        else right = mid - 1;               // Adjust right bound
    }
    return -1;
}

int main() {
    int arr[] = {1, 2, 3, 4, 5, 6, 7, 8, 9};
    int result = binarySearch(arr, 9, 5);  // Example with n = 9
    cout << "Found at index: " << result;
    return 0;
}
```

Time Complexity: _____