```
#include <bits/stdc++.h>
using namespace std;

int Ternary_Search( int *A, int l, int r, int key){
	if (r >= l){
		int mid1 = l + (r - l) / 3;
		int mid2 = r - (r - l) / 3;
		if (A[mid1] == key){
			return mid1;
		}
		if (A[mid2] == key){
			return mid2;
		}
		if (key < A[mid1]){
			return Ternary_Search(A, l, mid1 - 1, key); // T(n/3)
		}
		else if (key > A[mid2]){
			return Ternary_Search(A, mid2 + 1, r, key); // T(n/3)
		}
		else{
			return Ternary_Search(A, mid1 + 1, mid2 - 1, key); // T(n/3)
		}
	}
	return -1; //not found
}

int main(){
	int l, r, p, key;
	int A[] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
	l = 0;
	r = 9;
	key = 5;
	p = Ternary_Search(A, l, r, key);
	cout << "Index of " << key << " is " << p << endl;
	key = 50;
	p = Ternary_Search(A, l, r, key);
	cout << "Index of " << key << " is " << p << endl;

	return 0;
}
```

n is the size of the list

Space complexity

陣列 A 需要 n 個空間，其他變數各 1 個空間，共 c 個空間。

O(n+c) = n

Time complexity

Best case:

當一次就找到，時間複雜度為 O(1)

Wosrt case

$log_3^n + log_3^n + log_3^n + C = 3log_3^n + C$

$O(3log_3^n + C) = log_3^n$