# Lab Task 1 & 2 Explanation

## Lab Task 1: Pointers

### Code Explanation:  
This program demonstrates the use of pointers in C++.  
- An integer variable `num` is initialized to `10`.  
- A pointer `ptr` is declared and assigned the address of `num`.  
- The value of `num` is accessed using the pointer.  
- The pointer is then used to modify `num`'s value to `20`.  
- The program prints the initial and modified values of `num` and confirms that the pointer reflects these changes.

### Code:

#include <iostream>  
using namespace std;  
int main() {  
 int num = 10;  
 int\* ptr = &num;  
  
 cout << "Initial value of num: " << num << endl;  
 cout << "Pointer pointing to value: " << \*ptr << endl;  
  
 \*ptr = 20;  
  
 cout << "Modified value of num: " << num << endl;  
 cout << "Pointer now points to value: " << \*ptr << endl;  
  
 return 0;  
}

### Output:

```  
Initial value of num: 10  
Pointer pointing to value: 10  
Modified value of num: 20  
Pointer now points to value: 20  
```

## Lab Task 2: Big O Notation (Loops and Arrays)

### Code Explanation:  
This program finds the maximum value in an integer array.  
- The function `findMax()` takes an array and its size as arguments.  
- It initializes `maxVal` to the first element.  
- A loop iterates through the array, updating `maxVal` if a larger element is found.  
- The function returns the maximum value.  
- The `main()` function initializes an array `{3, 5, 1, 8, 2}` and calls `findMax()` to find and print the largest value.

### Code:

#include <iostream>  
using namespace std;  
  
int findMax(int arr[], int n) {  
 int maxVal = arr[0];  
 for (int i = 1; i < n; i++) {  
 if (arr[i] > maxVal) {  
 maxVal = arr[i];  
 }  
 }  
 return maxVal;  
}  
  
int main() {  
 int arr[] = {3, 5, 1, 8, 2};  
 int n = sizeof(arr) / sizeof(arr[0]);  
 int maxValue = findMax(arr, n);  
   
 cout << "Maximum value: " << maxValue << endl;  
 return 0;  
}

### Output:

```  
Maximum value: 8  
```

### Time Complexity:

The `findMax()` function iterates through the array once, making its time complexity \*\*O(n)\*\*.