ch 4 4.4 -> For binary search :-
base case : it's an array with one element
recursive case: it's dividing the array into two halfs and do the binary search in the half we want

4.5 -> O(n)   4.6 -> O(n)   4.7 -> O(1)   4.8 -> O(n^2)

ch 5 5.1 -> Consistent   5.2 -> not consistent
5.3 -> not consistent   5.4 -> Consistent

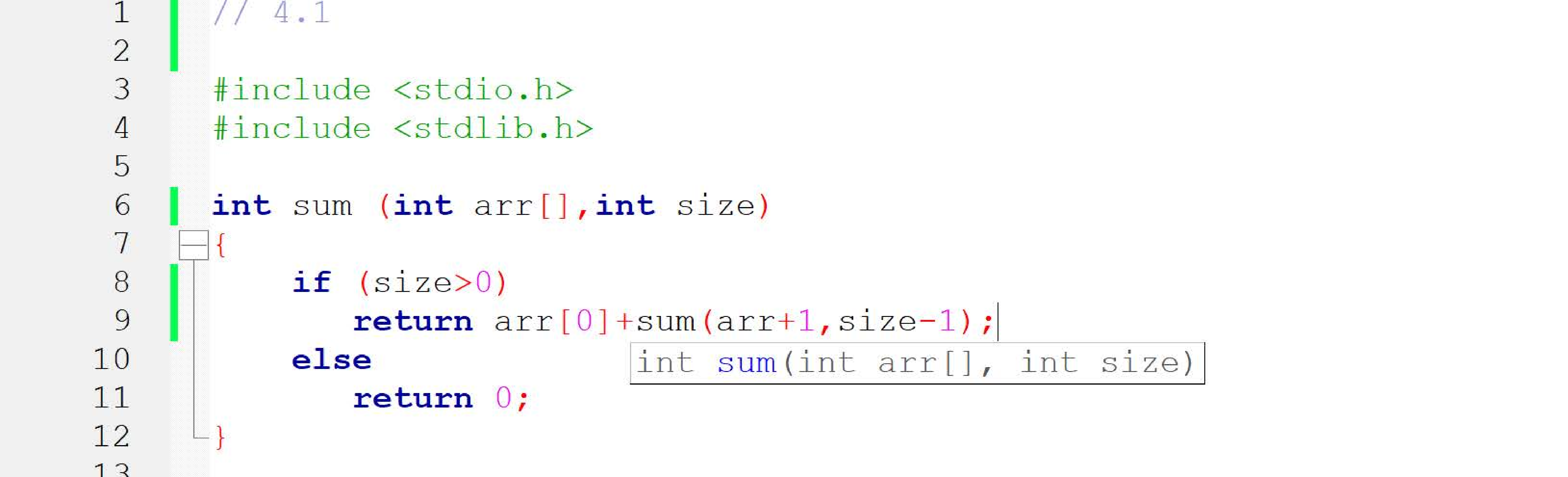
5.5 -> D   5.6 -> B   5.7 -> C, B, D

ch 6 6.1 -> 2 steps   6.2 -> 2 steps

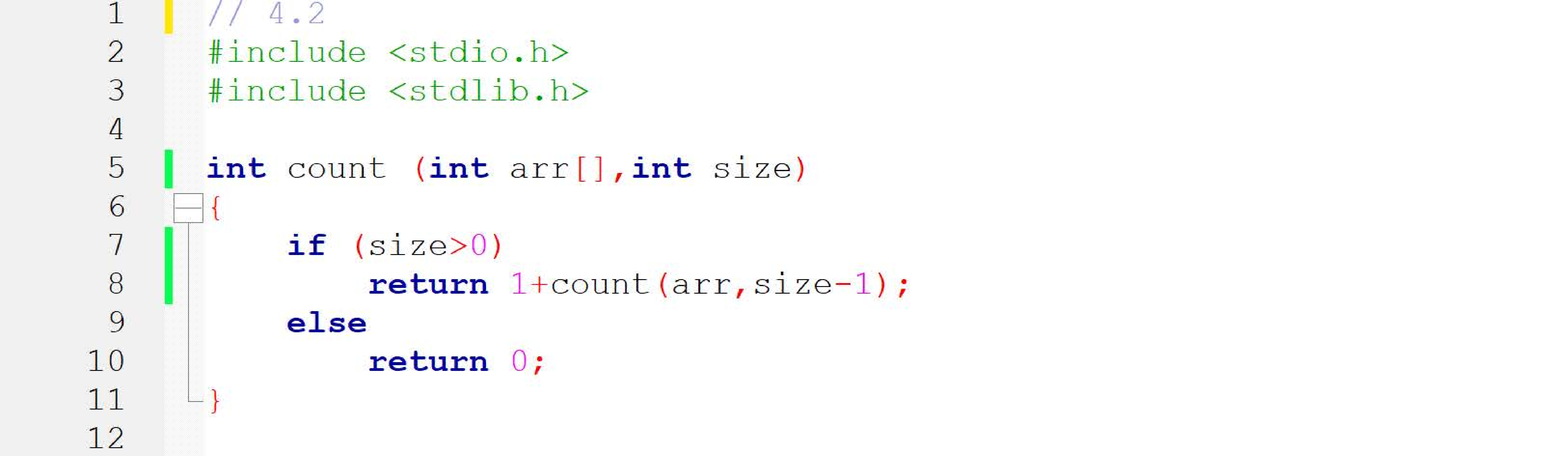
6.3 -> A: invalid   B: Valid   C: invalid

6.4 -> 1- Wake up   5- shower
2- Brush teeth   6- Get dressed
3- Eat breakfast   7- Pack lunch
4- Exercise

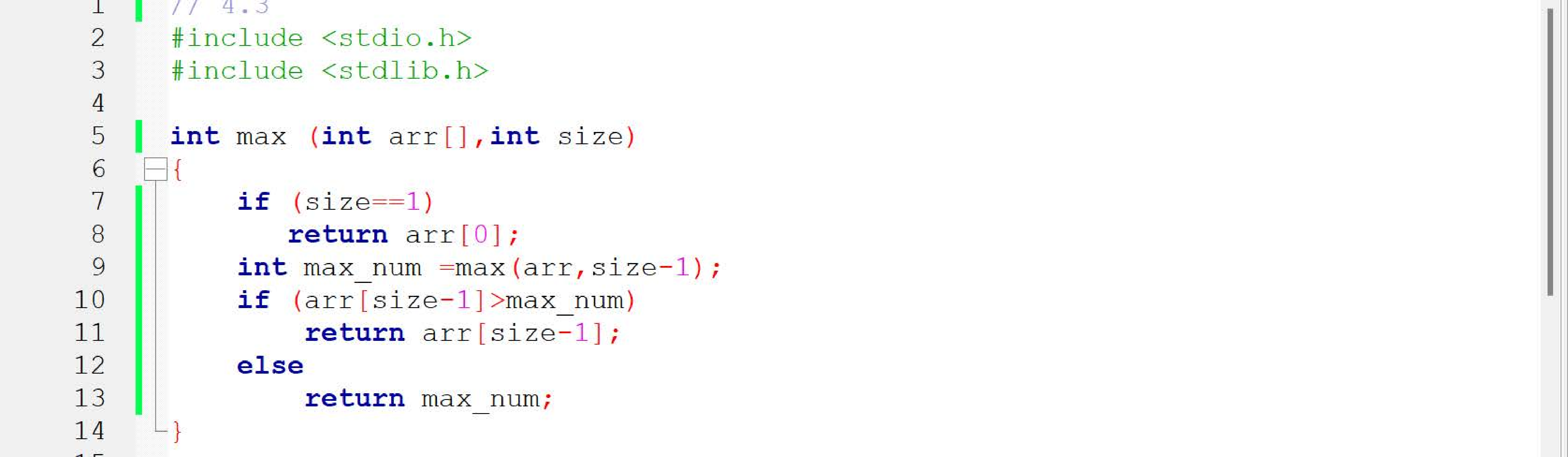
6.5 -> A, C



```c
// 4.1

#include <stdio.h>
#include <stdlib.h>

int sum (int arr[],int size)
{
    if (size>0)
       return arr[0]+sum(arr+1,size-1);
    else
       return 0;
}
```



```c
// 4.2
#include <stdio.h>
#include <stdlib.h>

int count (int arr[],int size)
{
    if (size>0)
        return 1+count(arr,size-1);
    else
        return 0;
}

```



```c
// 4.3
#include <stdio.h>
#include <stdlib.h>

int max (int arr[],int size)
{
    if (size==1)
       return arr[0];
    int max_num =max(arr,size-1);
    if (arr[size-1]>max_num)
        return arr[size-1];
    else
        return max_num;
}
```