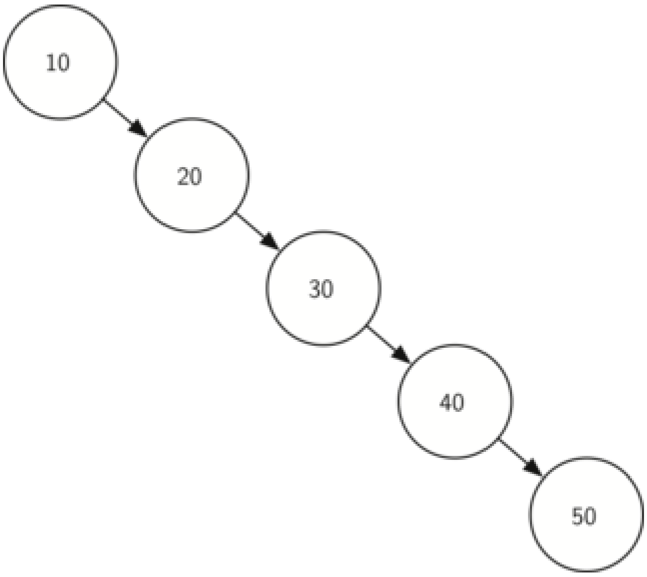
Coding MCQ

1. What is the worse-case time complexity for searching a key in a BST?

Ans. **O(n)**

Explanation :



For a skewed BST the time complexity to search a key is O(n).

2. Let's consider a binary heap having n elements already inserted into it. We are now given n more elements and we want to insert them into the heap(in any order). The time required to do the following operation is

Ans. **O(n)**

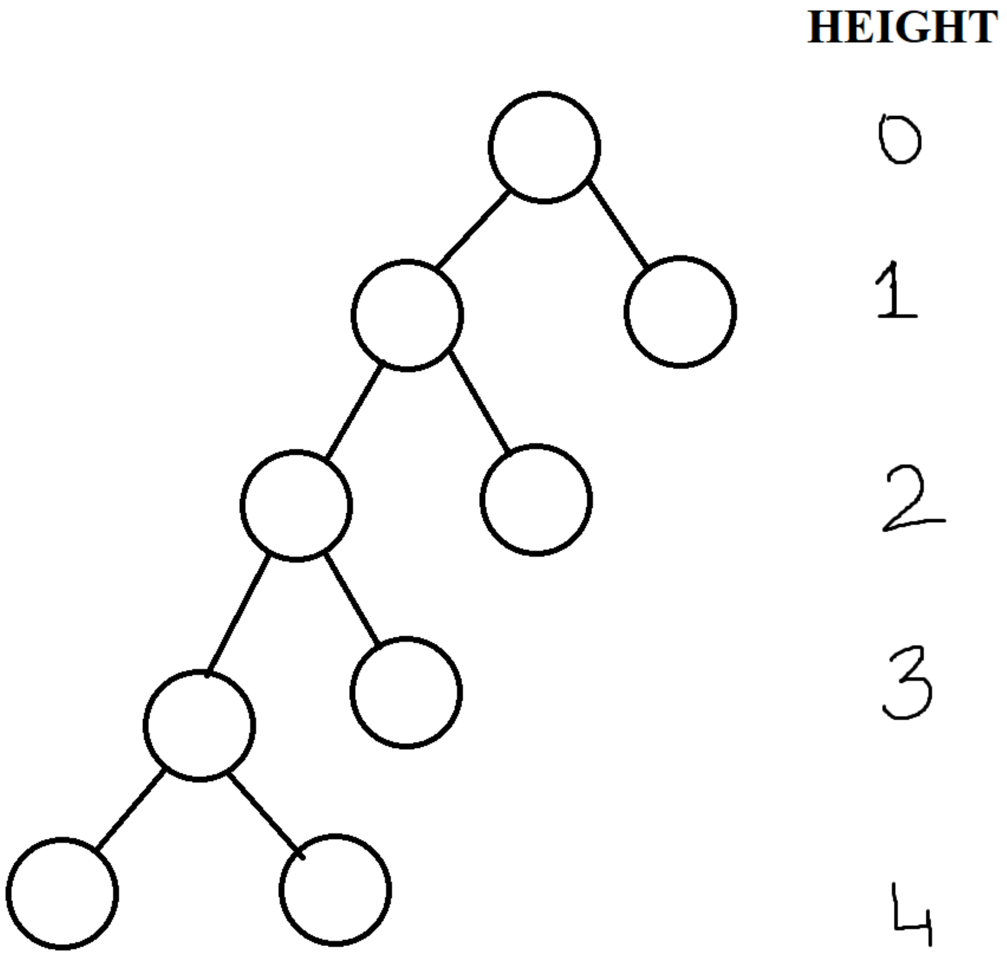
Explanation :

We can populate the array of n elements with n more elements and then call build heap on the array of 2n elements which will result in a complexity of O(n).

3. The height of the binary tree is the longest path(most number of edges) from root node to any leaf node in the tree. Let's construct a binary tree with n nodes. Each node has exactly 0 or 2 children. The maximum height which can be achieved is _______ .

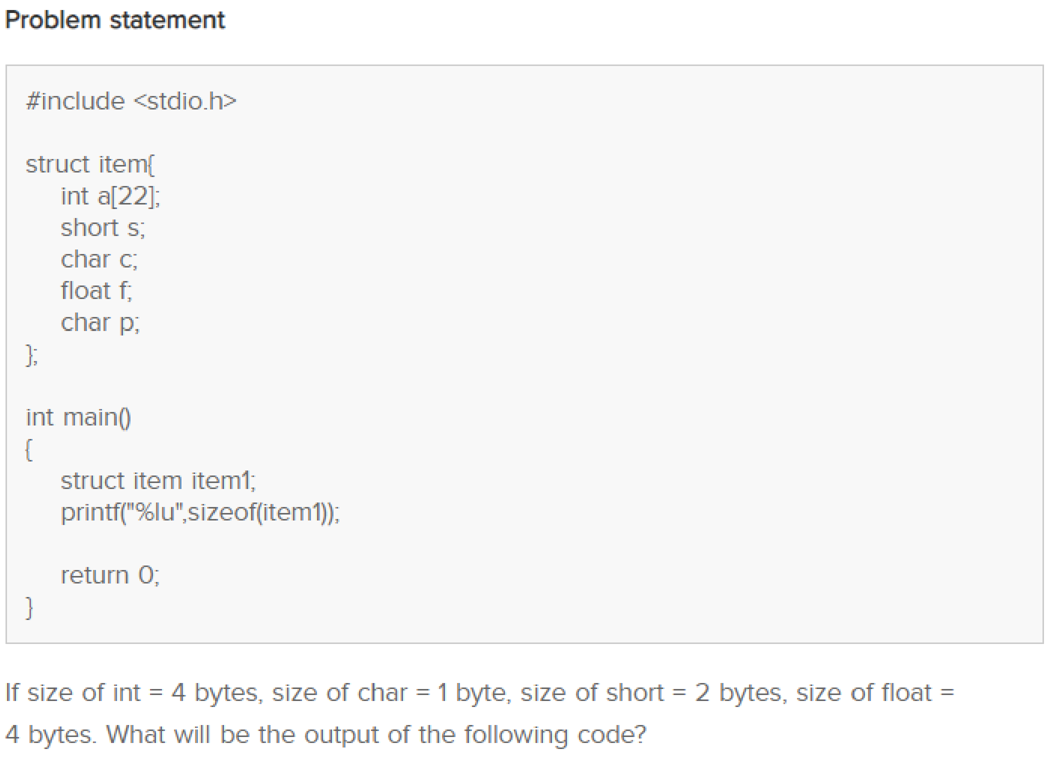
Ans. **(n - 1) / 2**

Explanation :



The above tree is an example of the required tree. In this tree every level has 2 elements except level 0.

4.



Problem statement

```
#include <stdio.h>

struct item{
    int a[22];
    short s;
    char c;
    float f;
    char p;
};

int main()
{
    struct item item1;
    printf("%lu",sizeof(item1));

    return 0;
}
```

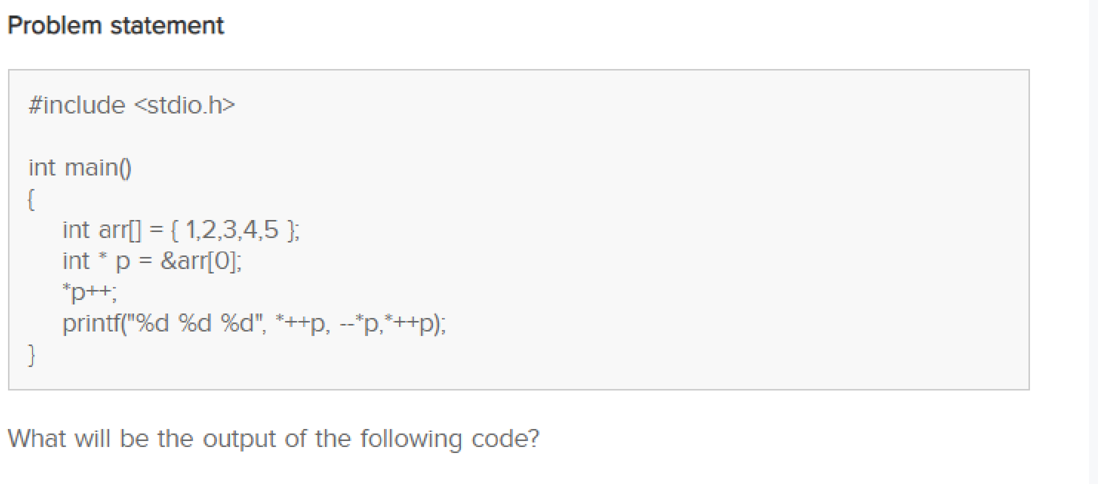
If size of int = 4 bytes, size of char = 1 byte, size of short = 2 bytes, size of float = 4 bytes. What will be the output of the following code?

Ans. 100

Explanation:-

[Structure Member Alignment, Padding and Data Packing - GeeksforGeeks]()

5.



Problem statement

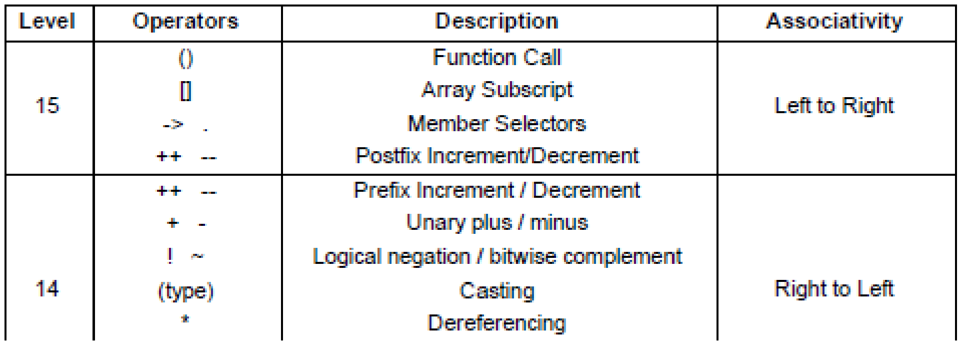
```
#include <stdio.h>

int main()
{
    int arr[] = { 1,2,3,4,5 };
    int * p = &arr[0];
    *p++;
    printf("%d %d %d", *++p, --*p,*++p);
}
```

What will be the output of the following code?

Ans. 4 2 3

Explanation :-



Level	Operators	Description	Associativity
15	() [] -> . ++ --	Function Call Array Subscript Member Selectors Postfix Increment/Decrement	Left to Right
14	++ -- + - ! ~ (type) *	Prefix Increment / Decrement Unary plus / minus Logical negation / bitwise complement Casting Dereferencing	Right to Left

By using the above precedence table we can solve the given problem. In printf the values are resolve from right to left. So the right most *++p will be evaluated first.

After *p++, p will point to index 1. So the rightmost *++p will be evaluated as follows:

a. ++p will make p point to index 2.
b. Dereference will give value 3.

Now --*p will be evaluated as follows :

a. *p will give value 3.
b. --3 gives 2.

Now *++p will be evaluated as follows:

a. P will increase that is it will point to index 3.
b. After dereferencing it will give value 4.